東京大学のスーパーコンピュータを用いた並列プログラミング教育（４）
― 工学部・工学系研究科共通科目「スパコンプログラミング１およびＩ」(2009 年度夏・冬学期)、および、全学ゼミ「スパコンプログラミング研究ゼミ」(夏学期）を通じて

片桐　孝洋

東京大学情報基盤センター　特任准教授

１．はじめに

計算科学・工学、ハードウェアの急速な発達を背景に、「第 3 の科学」としての大規模並列シミュレーションへの期待は、産学において一層高まっている。最新ハードウェアを駆使し、大規模並列シミュレーションプログラムを開発するための体系的な教育プログラムの事例は、世界的に見ても少ない。東京大学では 2008 年初頭から、「T2K オープンスパコン、次世代スーパーコンピュータ等を駆使した大規模シミュレーションを実施し、新しい科学を開拓する人材」を育成するための全学的な教育プログラムとして「学際計算科学・工学 人材育成プログラム」構想が検討されている[1][2]。

一方、東京大学情報基盤センター（以降、センター）では、スーパーコンピュータ（以降、スパコン）の潜在的な新規ユーザである東京大学工学部を主とする学部学生と大学院生に対して高性能計算（HPC）教育の支援を行ってきた。センターのスパコン啓発と、ユーザの研究分野(主として大規模数値シミュレーション分野)における高性能並列プログラム開発の長期的支援を行なうことが目的である。

以上の背景から、本講義は工学部および工学系研究科の共通科目「スパコンプログラミング（１）および（Ｉ)」を通年科目（夏学期、冬学期）として開講している。本講義は、工学部や工学系研究科以外の学生も受講可能である。一方、天才的なスパコンユーザを早期から養成し、高性能計算分野における優秀な人材を発掘する目的で、東京大学教養学部の学生に対し工学部と同様の講義を行う「スパコンプログラミング研究ゼミ」を開講している。2007 年度冬学期に「全学ゼミ」として開講して以来、2009 年度夏学期で 3 回目の開講となった。

双方の受講生に対し、2008 年 6 月稼働の T2K オープンスーパーコンピュータ（東大版）(HITACHI HA8000 クラスタシステム、以降 T2K マシンとよぶ）のアカウントを無料で発行し演習を行ってきている。これは、スーパーコンピューティング部門の「教育利用」サービスの一環である[3]。T2K マシンでは教育利用方針が強化され、4 ノード（64 コア）まで利用できるコア数（並列数）となった。これは、本講義の開始同時の 2007 年度に利用可能であった HITACHI SR11000/J2 の 1 ノード(16 コア)に対し、大幅に増加した。著者の知る限り 64 並列のスーパーコンピュータが教育利用で定常的に利用できる状況は数少なく、初等の並列処理教育環境としては日本有数の環境であるといえる。

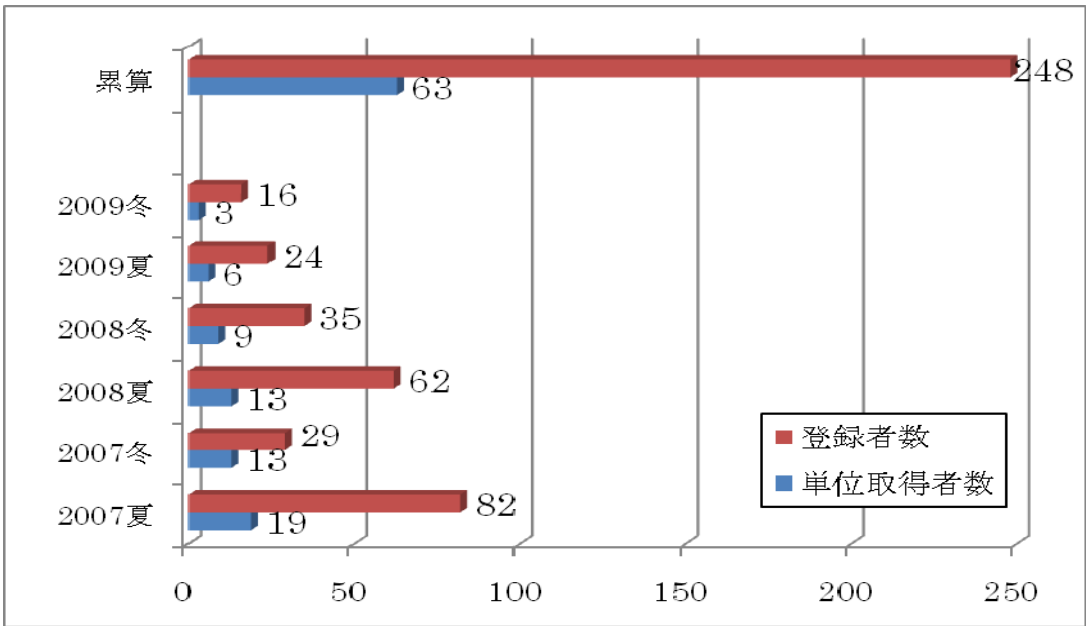
本報告は 2008 年度（2008 年 4 月～2009 年 3 月）の報告[4]に引き続き、2009 年度を通じての教育利用報告である。

２．受講者に対する統計データ

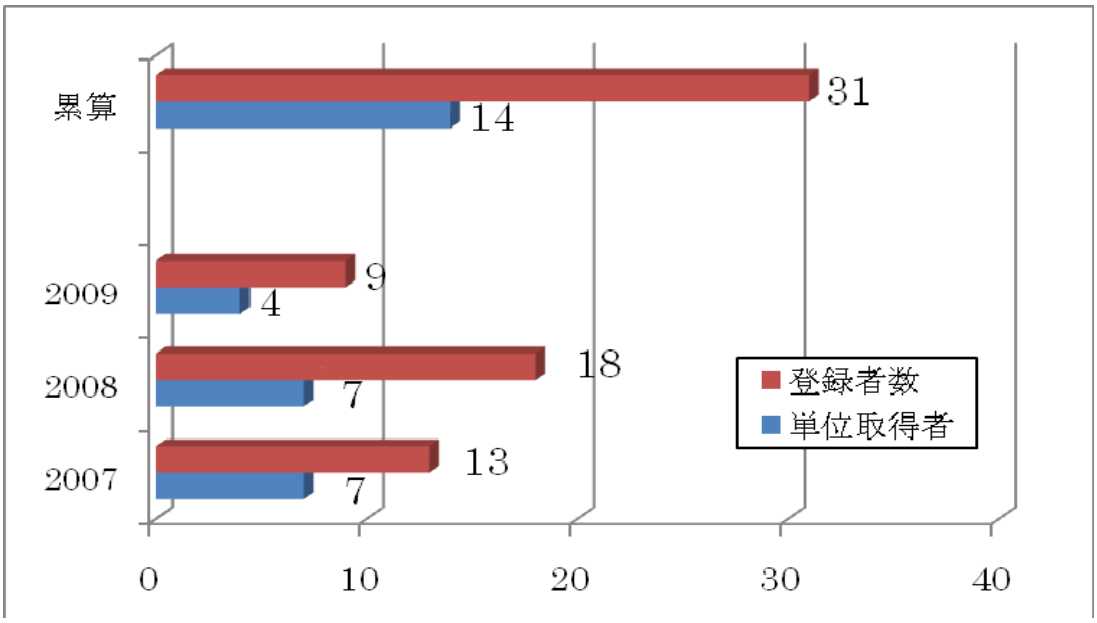
（１）受講者数

本講義は 2007 年度から夏学期、冬学期と年 2 回、連続して開講している。講義に事前登録した上でスパコンアカウントを発行した数（登録者数）と、単位を取得した人数（単位取得者数）

を図１にのせる。



(a) スパコンプログラミング（１）および（Ｉ）



(b) 全学ゼミ：スパコンプログラミング研究ゼミ

図１　講義登録者数および単位取得者数の累計

（２） 受講者の所属の累計

スパコンプログラミング（１）および（Ｉ）において、いままでアカウントを発行した学生の所属は、以下のとおりである（順不同）。

- 工学部
  - 機械情報工学科、機械工学科、精密工学科、電子情報工学科、航空宇宙工学科、計数工学科、マテリアル工学科、化学システム工学科、システム創成学科
- 理学部
  - 天文学科、地球惑星物理学科、化学科、数学科、物理学科

- 工学系研究科
  - システム創成学専攻、社会基盤学専攻、原子力国際専攻、精密機械工学専攻、物理工学専攻、航空宇宙工学専攻、機械工学専攻、電気系工学専攻、建築学専攻、海洋工学専攻、地球システム工学専攻、システム量子工学専攻、マテリアル工学専攻、化学システム工学専攻、人間環境学専攻
- 情報理工学系研究科
  - コンピュータ科学専攻、数理情報学専攻、電子情報学専攻、知能機械情報学専攻
- 新領域創成科学研究科
  - 基盤情報学専攻
- 理学系研究科
  - 地球惑星科学専攻、天文学専攻
- 数理科学研究科
  - 数理科学専攻
- 農学生命学研究科
  - 生産・環境生物学専攻

３．MPI を用いた並列プログラミング教育

（１）講義の方針

本講義を開講するに当たり、以下の指針を示している。

「高性能計算を学ぶためには、計算機アーキテクチャに始まり、コンパイラや OS といったシステムソフトウエア、さらに扱っているアプリケーションのアルゴリズムに至る広範な階層の知識が必要となる。講義でこれらすべてを扱うことはできない。そこで、厳選された実用的な課題について講義と演習を行う。本講義は、従来講義のように広い知識の獲得を目指すものではない。実際に高性能プログラムを基盤センターのスーパーコンピュータ上で開発できるという、実用的でかつ、研究者として生き残るために必須な技能の習得を目指すものである。この技能の習得により、受講者の研究を格段に進展させることを目標とする。」

MPI(Message Passing Interface)を用いた並列プログラミングにおいて、MPI の機能を網羅的に紹介する従来のテキストのような方針ではなく、センターユーザの多数を占める数値シミュレーション研究者に必要な最低限の実装知識と、センターのスパコンを利用するための最低限の技術の習得を目的にする。

（２）並列プログラミング教育の方針

並列プログラミング教育における最も重要な概念は、SIMD（ Single Instruction Multiple Data stream ）の概念であるという仮定のもとに教材を作成している。SIMD とは、並列計算機の分類の一つである。図２のように、同一の命令（たとえば、加算命令）がなされるが、加算する対象のデータは並列計算機の構成要素であるプロセッサ・エレメント(Processor Element, PE) で異なるというモデルである。

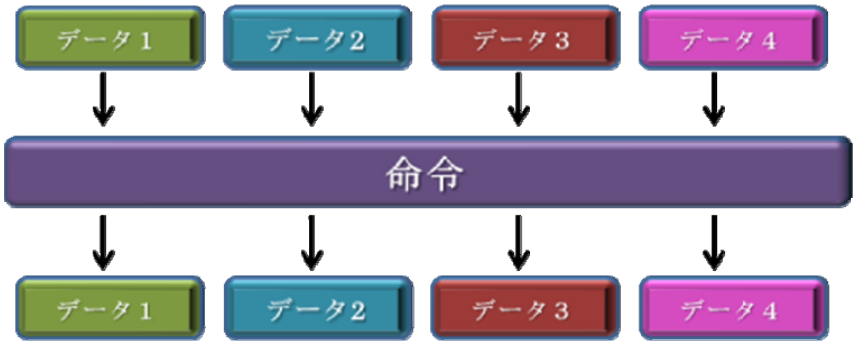


図 2　SIMD の概念

図2のSIMDの概念を、並列プログラミングの経験がない受講生に教えることは容易ではない。理由は、いままで扱ってきた逐次プログラムと受講生の頭にある「逐次」プログラミングモデルとの乖離があるからである。そこで、SIMD の概念が自然に身につくように、以下のような段階を経て並列化ができる方針を立てた。

- **段階１**　並列環境に慣れるため、簡単な並列プログラムを実行する。並列プログラムを眺める。
- **段階２**　逐次プログラムを実行する。逐次プログラムを解析する。
- **段階３**　逐次プログラムに存在するループに変更を施すだけで並列化ができる演習。
- **段階４**　逐次プログラムに存在するループに変更を施し、かつ単純な MPI 関数を記述することで並列化できる演習。
- **段階５**　データ構造の変更、多数の MPI 関数を記述することで並列化できる演習。

並列化のための段階 1 から 5 を通じて、逐次プログラムから並列プログラムを作成できる「方法論」を習得するというのが、本講義の最終的な目的である。この方法論は、一般的であり、特に行列計算処理の並列化においては効果的である。この方法論を習得することで、ソフトウエア工学的によい並列化の作法を身に付けさせる。

（３）内容

本講義で行った授業内容を表 1 に示す。この内容は初めて開講された 2007 年度夏学期以来、大幅な変更は無い。表 1 のように、本講義で用いたアプリケーションは、行列-ベクトル積、べき乗法（行列-ベクトル積が使われている固有値・固有ベクトルの初等的な数値計算法）、行列-行列積、LU 分解法の 4 種である。

表 1　授業内容

講義回数	内容概略
ガイダンス	初回ガイダンス、高性能計算の基礎
第1回講義	**並列数値処理の基本演算**：性能評価指標、基礎的な MPI 関数、データ分散方式、ベクトルどうしの演算、ベクトル-行列積、リダクション演算、数値計算ライブラリについて

第2回講義	**スーパーコンピュータを利用しよう**:スパコンを利用しよう、並列プログラミングの基礎、二分木総和演算
第3回講義	**高性能プログラミングの基礎(1)**:階層キャッシュメモリ、演算パイプライン、ループアンローリング、配列連続アクセス、キャッシュとキャッシュライン、キャッシュライン衝突、サンプルプログラムの実行、演習課題、レポート課題
第4回講義	**高性能プログラミングの基礎(2)**:ブロック化、その他の高速化技術、OpenMP 超入門、サンプルプログラム(OpenMP)の実行、演習課題、レポート課題
第5回講義	**行列-ベクトル積**:サンプルプログラム(行列-ベクトル積)の実行、並列化の注意点
第6回講義	**べき乗法**:べき乗法とは、サンプルプログラム(べき乗法)の実行、並列化の注意点
第7回講義	**行列-行列積(1)**:行列-行列積とは、ループ交換法、ブロック化(タイリング)法、Cannonのアルゴリズム、Foxのアルゴリズム、SUMMA、PUMMA、Strassenのアルゴリズム、サンプルプログラム(行列-行列積(1):簡単版)の実行、並列化の注意点
第8回講義	**行列-行列積(2)**:コンテスト課題発表、コンテストプログラムの実行、サンプルプログラム(行列-行列積(2):ちょっと難しい完全並列版)の実行、並列化の注意点、並列化のヒント
第9回講義	**LU 分解法(1)**:LU 分解法(ガウス・ジョルダン法、ガウス消去法、枢軸選択、LU 分解法(外積形式、内積形式、クラウト法、ブロック形式ガウス法、縦ブロックガウス法、前進・後退代入))、サンプルプログラム(LU 分解法)の実行、並列化のヒント、演習課題、レポート課題
第10回講義	**LU 分解法(2)**:LU 分解の逐次アルゴリズムの解説
第11回講義	**LU 分解法(3)**:レポート提出の注意、レポート課題採点基準、LU 分解の並列化のヒント(2)
第12回講義	**非同期通信**:1対1通信に関する MPI 用語、サンプルプログラム(非同期通信)の実行
第13回講義	**発展的話題**:ソフトウエア自動チューニング:背景、ソフトウエア自動チューニングとは、FIBER方式、自動チューニング記述言語ABCLibScript、ソフトウエアデモ、レポート課題

演習課題を与える一方、受講生が参加できる「プログラミングコンテスト」を講義の一環として開催した。コンテストの参加者、すなわちコンテストにおける出題をすべて解答する並列プログラムを提出した場合、レポートに加点を与えた。コンテストにおいて入賞(1 位～3 位)した場合、無条件で「優」を与えるという条件を付した。なお、2008 年度冬学期のコンテスト課題は、複数の右辺ベクトルをもつ「連立一次方程式の解法」である。このプログラムコンテストは、教養学部の全学ゼミの学生もハンデなしに参加を募った。

本講義の演習のために、表 2 に示すサンプルプログラム 9 本を教材として開発している。このサンプルプログラムは、受講生が講義中にダウンロードして実行確認をした上、演習で用いる。C 言語版と Fortran 言語版の 2 種を用意している。

表２　サンプルプログラム一覧

サンプルプログラム（並列化の段階）	サンプルプログラムの内容
#1. sp200Xsamples.tar（段階１、２）	並列版 Hello プログラム、並列円周率計算プログラム、逐次転送方式による並列総和演算プログラム、二分木通信方式による並列総和演算プログラム、時間計測方法の並列プログラム
#2. sp200XMat-Mat-noopt.tar（段階２）	行列-行列積の逐次プログラム（逐次チューニング用）
#3. sp200XMat-Mat-openmp.tar（段階２、３）	行列-行列積の逐次プログラム（OpenMP 並列化用）
#4. sp200XMat-vec.tar（段階２、３）	行列-ベクトル積の逐次プログラム
#5. sp200XPowM.tar（段階３、４）	べき乗法の逐次プログラム
#6. sp200XMat-Mat.tar（段階３）	行列-行列積の逐次プログラム（お手軽並列化用）
#7. sp200XMat-Mat-d.tar（段階３、４）	行列-行列積の逐次プログラム（完全分散並列化用）
#8. sp200XLU.tar（段階４、５）	LU 分解法による連立一次方程式の求解の逐次プログラム
#9. sp200XSP_Isend.tar（段階１、２）	非同期通信の並列プログラム

（４）スーパーコンピュータを用いた環境で特質すべき教育・演習事項

本講義の特筆すべき教授項目は、キャッシュ最適化技術を並列化の前に教えることである。計算機アーキテクチャのトレンドとして、今後もキャッシュを考慮したプログラミングができないと高性能化が実現できない。したがってその概念を、座学として教えるだけにとどまらず、サンプルプログラムを用いて速度を実感させる。

さらに実感させるだけではなく、最適化の課題を出すことで、実際にコードチューニングを行わせる。キャッシュ最適化の効果を実体験させることで、実学として身につけさせる工夫がなされている。

キャッシュ最適化の具体的なイメージを与えるために、行列-行列積の場合において、ブロック化する場合としない場合の違いを例として挙げている。この例を、図 3 に載せる。そのあと、具体的なコードを示し、説明を行っている。キャッシュブロック化のコードは、図 4 になる。

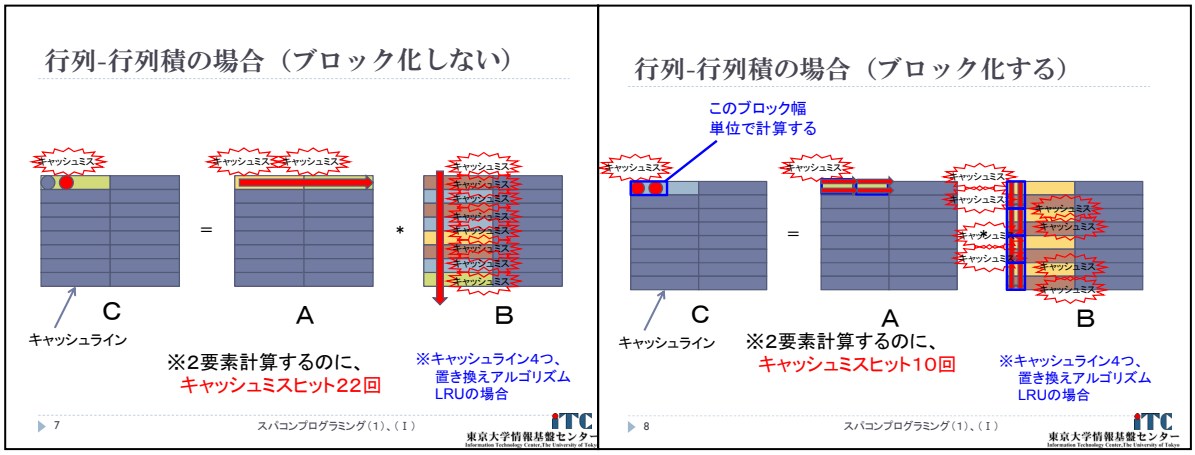


図 3　行列-行列積におけるブロック化の概念図

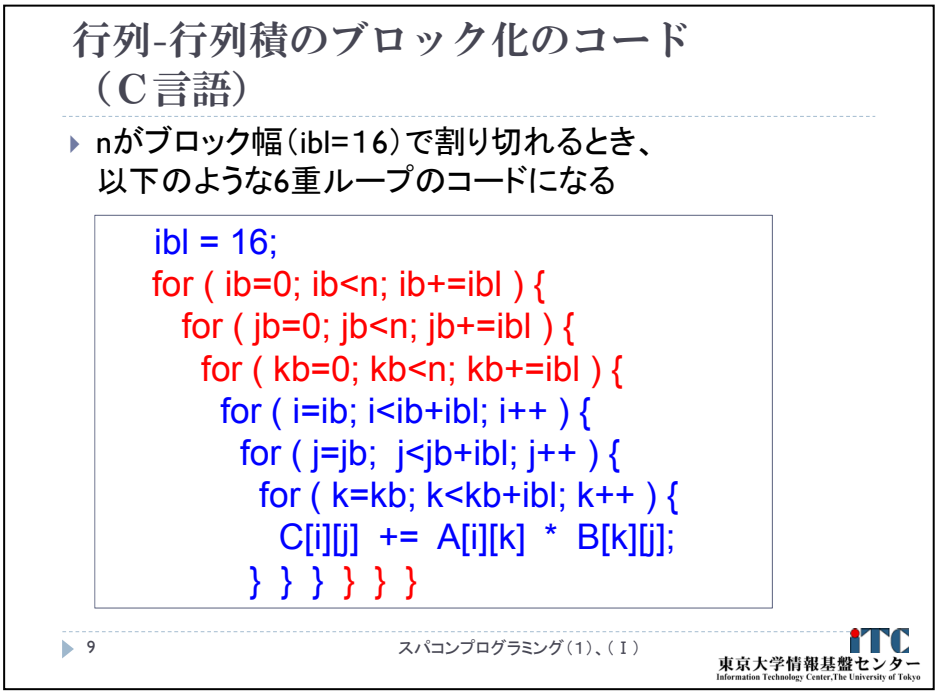


図 4　行列-行列積におけるブロック化コード（C言語）

（５）講義の経過と感想

今までの記事において講義の経緯を紹介しているので割愛する。

例年、コンテスト課題を提出できるのは３名程度である。コンテスト課題は、少なくとも中級以上のスキルを要求する。したがって、数名は本講義終了時において中級以上のレベルに達する。また上位入賞するプログラムは、アルゴリズム的に相当工夫がなされている。逆にいうならば、定常的に数名程度は天才的な技量をもつ学生が受講していることになる。このような＜センスのよい＞学生を応援し、HPC が必要とされる研究分野で育成していくことが、特に重要である。

以下、レポート提出者による本講義を受講した感想を列挙する。なお、改行以外の文章の修正は行っていない。

全学ゼミ受講者

- 並列化や、キャッシュの有効活用といった、アルゴリズムの改善とは違う方向からの高速化へのアプローチは、今まで経験したことがなかったので大変面白かった。将来研究などでスーパーコンピュータを使うことになった時に、この経験が活かせるといいなと思った。
- 基本的なことから説明されていて分かりやすかった。
- 講義時間中に充分な演習時間があってよかった。
- LU 分解にもうちょっと時間をかけて説明し、演習させてほしかった。
- 全般的に有意義だったと感じています。そんなにプログラミングの経験がない自分にも講義の流れにそってやることで比較的容易に理解を深めることができました。この講座で吸収できたことを少しでも今後にいかせたらと思います。

スパコンプログラミング（1）および（I）受講者

- 「コンピュータアーキテクチャの講義を履修してからこの講義を受けるべきだったな」と感じました。スパコンプログラミングと平行してコンピュータアーキテクチャの授業を履修していたのですが、向こうの授業でキャッシュやパイプラインを扱う前にこちらの授業でそれらのトピックに突入したので、当時はライトスルーやらダイレクトマップやらパイプラインなど、ほぼ全てチンプンカンプンでした。今スライドを読み直してみると、大体は理解できるので…
- 実習が授業時間に組み込まれていたのは大変良かった。実行できなかった場合にも丁寧に答えて頂き、またメールでも親切に対応してもらえましたのでその点は満足です。授業難易度は適切だと思います。行列積に関するプログラムは直感に優しく、慣れるには適切でした。LU 分解は、慣れていない段階ではやはり時間がかかりました。特にデバッグの面で前進代入や後退代入部分まで手が回らなかったので、LU 分解の時間を増やすか、デバッグを行わないよう、後退代入部分ははじめから正解を挙げておくなどして頂けるとたすかります。博士課程という立場上、授業時間以外に時間を取ることがなかなか出来ず、実際に講義時間内で手を動かせたということはやりやすかったです。
- 講義の難易度は想像していたほど高くはなかった。しかし、残念なことに体調不良のため半分程度しか出席できず、そのため途中からついていくのが難しくなってしまった。とはいえ資料がとてもよくできているので、自習である程度はカバーできて助かった。毎回の内容が独立していることが多かったので、前回の内容が理解しきれなくてもその次の講義を受けることができた。座学と演習を同時に進める形式だと、よりモチベーションが上がると思った。
- 並列プログラミングに関するわかりやすい授業であった。ただ原理だけでなく、計算量が多いアルゴリズムが複数用意して、それらがスパコンを利用して解く様々な方法について学ぶことで MPI に関する詳しい知識を得ることができたと思います。人数が少なかったため、ほとんど個人に教える感じでした。質問した時も、熱心に答えたと思います。
- 研究で MPI や OpenMP を使った並列化を行う必要があり、授業でやったことがそのまま役に立ちました。MPI は日本語の資料が得にくい上に実際に有用な関数がどれかわかりにくいので、授業で教えて頂いた実践的な内容は有用でした。並列化とは関係ないですが、コンパイラが最適化しやすくするためのコードの書き方（局所変数を使って再利用を明示す

るなど）も、普段から疑問に思っていた部分なので興味深かったです。自動チューニングの部分は実際にどれくらい効果があるのか分かりませんが、別の科目の講義にして各人の問題で実践できるくらいになったら更にいいのではないかと思いました。

- 朝早いのが一番大変でした。自分の研究では、行列計算を使うことはほとんどないのですが、ループアンローリングやキャッシュをうまく使っていく方法については、大変勉強になりました。また、MPI を実際に書いて並列化のきもが何処なのかがおおよそ分かってきました。この実習を生かして、今後の研究を大規模計算へと徐々に移行していこうと考えています。また、先生の公開されているスライドは、今後の MPI 実装におけるリファレンスとして多いに利用できるので、そういった意味でもスパコンの扱いを身に着けるための講義になったと思います。
- 東京大学のスパコンを使う経験ができて非常に面白かったと思います。私自身はまだ研究室が決めていないので(できれば流体の解析とシミュレーションを課題にしたいですが)実用が少ないですが、必ず将来の研究に役立つと思います。この前、LU 分解で連立方程式を解くプログラムと反復法による連立方程式の数値解法のプログラムを普通のパソコンで書いたことがありますが、その並列版のプログラムを見てまた印象を深めました。講義の内容としてもいろいろ勉強になりましたが、ちょっと LU 分解から飛躍的な感じがします。もっと前の段階でサンプルプログラムをいっぱいもらって『正しい並列プログラミング』の形を学ぶことになったら、もっと理解しやすくなると思います。
- 配布資料が非常に分かり易く勉強になりました．自分の研究のプログラムに応用することもでき(遺伝的アルゴリズムを用いた機械学習に MPI を利用し並列化を行いました)，修士論文を乗り切ることができました．非常に感謝しております．また，普段，あまり気にせずにプログラミングしている部分で実は高速化が可能なのだということを学べて非常に良い機会でした．学部でのプログラミングの講義は本で読めば分かるような内容で非常に退屈でしたが，この授業は新しい気づきがたくさんあり有益でした．プログラミングの高速化の楽しみに若干目覚めました．
- スーパーコンピュータについては漠然とすごく速い計算機程度の印象しか持っていなかったため、その構成や能力を生かす方法についての話が聞けたことはよかった。また、MPI を使うほどの規模の計算を行う機会はあまりないだろうが、最近は多コアの CPU も多いためスパコン以外でも授業で習ったことが役に立つことも増えると思う。実際に PC 用のコードの処理に時間がかかる繰り返し部分を単純に OpenMP で並列化しただけでも計算の高速化が行えた。そういった点で序盤の講義の計算機の高速化の基本的な話は興味を持って聞くことができた。MPI を用いた並列計算は、前述の通り使用する機会があまりないだろうし LU 分解のあたりからは理解することも大変だったが、その分うまくいった時はとても嬉しかったことを覚えている。シミュレーションのプログラムを書くことがあるので楽に速くできることは、とてもありがたい。そのため、ソフトウェア自動チューニングの話を、ちゃんとしてもらえるとよかった。

本講義の最終目標は、並列処理技術の習得により、個人の研究を各段に進めることにある。本講義終了後に実際に研究に適用できたという事例や、高速化の楽しさがわかったという感想があった。このことは、並列処理学の教育者冥利に尽きるものであり、大変うれしい反響である。

４．おわりに

本講義の事前登録者は、2009 年度まで累算し 279 名にものぼる。最終的な単位取得者数は、累算で 77 名である。本講義を継続していくことで、本センターのスーパーコンピュータが利用できる人材にとどまらず、高性能計算分野に貢献できる多数の人材を輩出できる。学界にとどまらず産業界にも人材が提供される。高性能計算分野の経済発展に貢献し、センターの社会貢献に資する礎となる教育を行っていく所存である。

本講義の授業資料は、平成 21 年度分から一般に公開している。以下のページを参照いただければ幸いである。

http://www.kata-lab.itc.u-tokyo.ac.jp/class-matr.htm

参　考　文　献

[1] 中島研吾、究極の「並列プログラミング教育」を目指して―地球惑星科学専攻での 4 年間と未来への提言―、スーパーコンピューティングニュース、Vol.11, No.4, pp.30-44,2009 年 7 月.

[2] 東京大学　学際計算科学・工学人材育成プログラム
http://nkl.cc.u-tokyo.ac.jp/CSEedu/

[3] 東京大学情報基盤センタースーパコンピューティング部門　教育利用
http://www.cc.u-tokyo.ac.jp/use_info/education/

[4] 片桐孝洋、東京大学のスーパーコンピュータを用いた並列プログラミング教育 (3) ― 工学部・工学系研究科共通科目「スパコンプログラミング１およびⅠ」(2008 年度夏・冬学期)、および、全学ゼミ 「スパコンプログラミング研究ゼミ」(夏学期) を通じて ―、スーパーコンピューティングニュース、Vol.10, No.5, pp.50-60,2008 年 9 月.