ISSN 1344—9567

スーパーコンピューティングニュース

Vol.23 No. 1, 2021.1

東京大学情報基盤センター
INFORMATION TECHNOLOGY CENTER, THE UNIVERSITY OF TOKYO

スーパーコンピュータシステム 利用負担金表

Oakbridge-CX スーパーコンピュータシステム 利用負担金表（2020 年 4 月 1 日）

<table>
<tr><th colspan="2" rowspan="2">コース</th><th colspan="2">負担金額（税込）</th><th rowspan="2">ディスク容量</th><th rowspan="2">備考</th></tr>
<tr><th>大学・公共機関等</th><th>企業</th></tr>
<tr><td colspan="2">パーソナルコース</td><td>申込 1 セット当り，
最大 3 セットまで
100,000 円
（8,640 ノード時間）</td><td></td><td>申込 1 セット当り
/work 4TB
利用者当り
/home 50GB</td><td>最大ノード数 256 ノード</td></tr>
<tr><td rowspan="2">グループコース</td><td>一般申込</td><td>申込 1 セット当り
100,000 円
（8,640 ノード時間）</td><td>申込 1 セット当り
120,000 円
（8,640 ノード時間）</td><td rowspan="2">グループ 1 セット当り
/work 4TB
利用者当り
/home 50GB</td><td rowspan="2">最大ノード数 256 ノード</td></tr>
<tr><td>ノード固定</td><td>申込 1 セット当り
150,000 円
（8,640 ノード時間）</td><td>申込 1 セット当り
180,000 円
（8,640 ノード時間）</td></tr>
<tr><td colspan="2">トークン追加</td><td>8,400 円
（720 ノード時間）</td><td>10,000 円
（720 ノード時間）</td><td></td><td></td></tr>
<tr><td colspan="2">ディスク追加</td><td colspan="2">6,480 円／（1TB*年）</td><td></td><td>1TB 単位で申込可
（/work のみ）</td></tr>
</table>

※トークン消費係数は 1.00 である。
※括弧内のノード時間は付与するトークン量。実行したジョブのノード時間積と消費係数に応じてトークンが消費される。
付与したトークンは，利用期間内に全量が使用できることを保証するものではない。
トークンは利用期間内に限り有効とし，利用終了後に残量がある場合でも繰越や利用負担金の返還は行わない。
トークンの他のシステムへの移行については，「トークン移行におけるノード時間積の換算表」を参照。
※ノード固定の申し込みには審査を要する。
※/home のディスク容量はパーソナルコースやグループコースに複数所属していても利用者当り 50GB 固定。

Oakforest-PACS スーパーコンピュータシステム 利用負担金表（2020 年 4 月 1 日）

<table>
<tr><th rowspan="2">コース</th><th colspan="2">負担金額（税込）</th><th rowspan="2">ディスク容量</th><th rowspan="2">備考</th></tr>
<tr><th>大学・公共機関等</th><th>企業</th></tr>
<tr><td>パーソナルコース</td><td>申込 1 セット当り，
最大 6 セットまで
50,000 円
（8,640 ノード時間）</td><td></td><td>申込 1 セット当り
/work 1TB
利用者当り
/home 50GB</td><td>最大ノード数 2,048 ノード</td></tr>
<tr><td>グループコース</td><td>申込 1 セット当り
50,000 円
（8,640 ノード時間）</td><td>申込 1 セット当り
60,000 円
（8,640 ノード時間）</td><td>グループ 1 セット当り
/work 1TB
利用者当り
/home 50GB</td><td>最大ノード数 2,048 ノード</td></tr>
<tr><td>トークン追加</td><td>4,200 円
（720 ノード時間）</td><td>5,000 円
（720 ノード時間）</td><td></td><td></td></tr>
<tr><td>ディスク追加</td><td colspan="2">6,480 円／（1TB*年）</td><td></td><td>1TB 単位で申込可
（/work のみ）</td></tr>
</table>

※トークン消費係数は 1.00 である。
※括弧内のノード時間は付与するトークン量。実行したジョブのノード時間積と消費係数に応じてトークンが消費される。
付与したトークンは，利用期間内に全量が使用できることを保証するものではない。
トークンは利用期間内に限り有効とし，利用終了後に残量がある場合でも繰越や利用負担金の返還は行わない。
トークンの他のシステムへの移行については，「トークン移行におけるノード時間積の換算表」を参照。
※/home のディスク容量はパーソナルコースやグループコースに複数所属していても利用者当り 50GB 固定。

Reedbush スーパーコンピュータシステム(Reedbush-H/L) 利用負担金表(2020 年 4 月 1 日)

コース		負担金額(税込)		ディスク容量	備考
		大学・公共機関等	企業		
パーソナルコース		申込 1 セット当り, 最大 2 セットまで 75,000 円 (8,640 ノード時間)		申込 1 セット当り /lustre 1TB 利用者当り /home 2GB	Reedbush-H 最大ノード数 32 ノード Reedbush-L 最大ノード数 16 ノード
グループコース	一般申込	申込 1 セット当り 75,000 円 (8,640 ノード時間)		グループ 1 セット当り /lustre 1TB 利用者当り /home 2GB	Reedbush-H 最大ノード数 32 ノード Reedbush-L 最大ノード数 16 ノード
	公募制度 Reedbush-H	申込 1 セット当り 公募制度 180,000 円 (21,600 ノード時間)	申込 1 セット当り 公募制度 216,000 円 (21,600 ノード時間)	グループ 1 セット当り /lustre 4TB 利用者当り /home 2GB	最大ノード数 32 ノード
	公募制度・ ノード固定 Reedbush-L	申込 1 セット当り 公募制度 300,000 円 ノード固定 450,000 円 (34,560 ノード時間)	申込 1 セット当り 公募制度 360,000 円 ノード固定 540,000 円 (34,560 ノード時間)	グループ 1 セット当り /lustre 4TB 利用者当り /home 2GB	最大ノード数 16 ノード
トークン追加		6,300 円 (720 ノード時間)	7,500 円 (720 ノード時間)		
ディスク追加		6,480 円／(1TB*年)			1TB 単位で申込可 (/lustre のみ)

※トークン消費係数は下記の通りである。

Reedbush-H: 2.50, Reedbush-L: 4.00

※括弧内のノード時間は付与するトークン量。実行したジョブのノード時間積と消費係数に応じてトークンが消費される。

付与したトークンは，利用期間内に全量が使用できることを保証するものではない。

トークンは利用期間内に限り有効とし，利用終了後に残量がある場合でも繰越や利用負担金の返還は行わない。

トークンの他のシステムへの移行については，「トークン移行におけるノード時間積の換算表」を参照。

※公募制度・ノード固定の申し込みには審査を要する。

※/home のディスク容量はパーソナルコースやグループコースに複数所属していても利用者当り 2GB 固定。

トークン移行におけるノード時間積の換算表

移行元＼移行先	Reedbush-H/L システム	Oakforest-PACS システム	Oakbridge-CX システム
Reedbush-H/L システム	―	1.5	0.75
Oakforest-PACS システム	0.6	―	0.5
Oakbridge-CX システム	1.3	2	―

移行先に追加されるトークン量（ノード時間）＝移行トークン量×係数

注意事項（Oakbridge-CX, Oakforest-PACS, Reedbush,スーパーコンピュータシステム 共通）

- 「大学・公共機関等」は大学，高等専門学校及び大学共同利用機関，文部科学省所管の独立行政法人，学術研究及び学術振興を目的とする国又は地方公共団体が所管する機関，並びに文部科学省科学研究費補助金の交付を受けて学術研究を行う者に適用する。
- 「企業」の申し込みには，企業利用申込書添付書類の提出および審査を要する。
- 利用期間は，利用開始月から終了月の末日またはサービス休止前までとする。利用期間内に計算機利用を中止した場合であっても利用負担金額の変更は行わない。年度の途中で利用開始または終了する場合の負担金額は月数別利用負担金表（Web ページ）を参照すること。
- 前掲の利用負担金表は基本セットの内容であり，最小セットについては Web ページを参照すること。
- パーソナルコース（ただし，本センターのスーパーコンピュータシステムに初めて登録された利用者）においてのみ，利用開始月の翌月末日までに利用を中止することができる。利用負担金はパーソナルコースの利用期間1ヶ月の金額を適用し，請求する。
- 利用負担金は，原則として利用開始月に応じ，以下の月の初旬に一括して請求する。
  - 利用開始月が 4 月から 7 月までは 10 月，8 月から 9 月までは 12 月，10 月から 12 月までは 3 月，1 月から 3 月までは 3 月末。
  - 前年度内に事前申込をした分については，利用開始月に関わらず，7 月初旬の請求となる。
- 利用負担金額が減額となる変更はできない。
- コース間の変更については，利用負担金が増額になる場合のみ別途相談に応じる。（ただし，利用者番号変更の場合がある。）
- グループコースのディスク量は，グループ全体の上限値である。

スーパーコンピュータシステム ジョブクラス制限値

Oakbridge-CX スーパーコンピュータシステム ジョブクラス制限値(2019 年 7 月 1 日)

キュー名※1	ノード数※2 (最大コア数)	制限時間 (経過時間)	メモリー容量 (GB) ※3	パーソナルコース	グループコース 申込 一般	グループコース 固定ノード
debug	1 ～ 16 (896)	30 分	168	○	○	○
short	1 ～ 8 (448)	8 時間	168	○	○	○
(regular)						
small	1 ～ 16 (896)	48 時間	168	○	○	○
medium	17 ～ 64 (3584)	〃	〃	○	○	○
large	65 ～ 128 (7168)	〃	〃	○	○	○
x-large	129 ～ 256 (14336)	24 時間	〃	○	○	○
challenge	1 ～ 1368 (76608)	24 時間	168	★	★	★
任意	申込数	任意 ※4	168	×	×	○
(interactive) ※5						
interactive_n1	1 (56)	2 時間	168	○	○	○
interactive_n8	2 ～ 8 (448)	10 分	〃	○	○	○

※1 キューの指定(〝#PJM -L“rscgrp=キュー名”〞)は, regular, debug, short を小文字で指定する
regular キューはノード数の指定(〝#PJM -L“node=ノード数”〞)でノード数別のキューに投入される
※2 トークンの消費係数は 1.00
※3 1ノード当りの利用者が利用可能なメモリー容量
※4 申込ノード数の合計以内ならば, キュー名・制限時間(原則 48 時間以内)は相談の上, 任意に設定可能
※5 インタラクティブジョブの起動は次のとおり (トークン消費なし)
pjsub --interact -g グループ名 -L“rscgrp=interactive,node=ノード数”

Oakforest-PACS スーパーコンピュータシステム ジョブクラス制限値(2018 年 4 月 1 日)

キュー名※1	ノード数※2 (最大コア数)	制限時間 (経過時間)	メモリー容量 (GB) ※3	パーソナルコース	グループコース
debug-cache	1 ～ 128 (8704)	30 分	82	○	○
debug-flat	1 ～ 128 (8704)	30 分	96	○	○
(regular-cache)					
small-cache	1 ～ 128 (8704)	48 時間	82	○	○
medium-cache	129 ～ 512 (34816)	〃	〃	○	○
large-cache	513 ～ 1024 (69632)	〃	〃	○	○
x-large-cache	1025 ～ 2048 (139264)	24 時間	〃	○	○
(regular-flat)					
small-flat	1 ～ 128 (8704)	48 時間	96	○	○
medium-flat	129 ～ 512 (34816)	〃	〃	○	○
large-flat	513 ～ 1024 (69632)	〃	〃	○	○
x-large-flat	1025 ～ 2048 (139264)	24 時間	〃	○	○
challenge	1 ～ 8208 (558144)	24 時間	82 / 96	★	★
(interactive-cache) ※4					
interactive_n1-cache	1 (68)	2 時間	82	○	○
interactive_n16-cache	2 ～ 16 (1088)	10 分	〃	○	○
(interactive-flat) ※4					
interactive_n1-flat	1 (68)	2 時間	96	○	○
interactive_n16-flat	2 ～ 16 (1088)	10 分	〃	○	○
prepost	1 (28)	6 時間	222	○	○

★ 審査による課題選定の上, 月1回の一定期間のみ利用可能(原則として月末処理日前日の朝～翌日朝)
※1 キューの指定(〝#PJM -L“rscgrp=キュー名”〞)は, regular-cache/flat, debug-cache/flat を小文字で指定する
regular-cache/flat キューはノード数の指定(〝#PJM -L“node=ノード数”〞)でノード数別のキューに投入される
※2 トークンの消費係数は 1.00
※3 1ノード当りの利用者が利用可能なメモリー容量
※4 インタラクティブジョブの起動は, pjsub --interact -g グループ名 -L“rscgrp=キュー名,node=ノード数”
(キュー名は interactive-cache/flat, トークン消費なし)

Reedbush スーパーコンピュータシステム（Reedbush-H） ジョブクラス制限値（2018 年 9 月 27 日）

キュー名※1	ノード数※2（最大 GPU 数）	制限時間（経過時間）	メモリー容量（GB）※3	パーソナルコース	グループコース	
					一般申込	公募制度
h-debug	1 ～ 4 (8)	30 分	244	○	○	○
h-short	1 ～ 4 (8)	2 時間	244	○	○	○
(h-regular)						
h-small	1 ～ 4 (8)	48 時間	244	○	○	○
h-medium	5 ～ 8 (16)	〃	〃	○	○	○
h-large	9 ～ 16 (32)	〃	〃	○	○	○
h-x-large	17 ～ 32 (64)	24 時間	〃	○	○	○
h-challenge	1 ～ 120 (240)	24 時間	244	★	★	★
(h-interactive) ※4						
h-interactive_1	1 (2)	2 時間	244	○	○	○
h-interactive_2	2 (4)	30 分	〃	○	○	○
(h-regular-low) ※5						
h-small-low	1 ～ 4 (8)	12 時間	244	▲	▲	▲
h-medium-low	5 ～ 8 (16)	〃	〃	×	▲	▲
h-large-low	9 ～ 16 (32)	〃	〃	×	▲	▲
h-x-large-low	17 ～ 32 (64)	6 時間	〃	×	▲	▲

▲　パーソナルコースは 最大 1 ノード，グループコースは申込ノード数の 4 分の 1 まで実行可（公募制度による申し込みの場合は申込ノード数まで実行可）
★　審査による課題選定の上，月1回の一定期間のみ利用可能（原則として月末処理日前日の朝～翌日朝）
※1 キューの指定（〝#PBS -q キュー名〞）は，h-short，h-regular，h-debug を小文字で指定する
h-regular キューはノード数の指定（〝#PBS -l select=ノード数〞）でノード数別のキューに投入される
※2 トークンの消費係数は 2.50
※3 1ノード当りの利用者が利用可能なメモリー容量
※4 インタラクティブジョブの起動は次のとおり（トークン消費なし）
qsub -I -q h-interactive -l select=ノード数 -l walltime=XX:XX -W group_list=グループ名
※5 非優先ジョブクラス（低プライオリティキュー）は，トークンの追加申込が締め切られ，ジョブ実行に必要なトークン残量が不足した場合のみ実行可

Reedbush スーパーコンピュータシステム（Reedbush-L） ジョブクラス制限値（2020 年 4 月 1 日）

キュー名※1	ノード数※2（最大 GPU 数）	制限時間（経過時間）	メモリー容量（GB）※3	パーソナルコース	グループコース		
					一般申込	公募制度	ノード固定
l-debug	1 ～ 4 (16)	30 分	244	○	○	○	○
(l-regular)							
l-small	1 ～ 4 (16)	168 時間	244	○	○	○	○
l-medium	5 ～ 8 (32)	〃	〃	○	○	○	○
l-large	9 ～ 16 (64)	〃	〃	○	○	○	○
任意	申込数	任意 ※4	244	×	×	×	○
(l-interactive) ※5							
l-interactive_1	1 (4)	24 時間	244	○	○	○	○
l-interactive_2	2 (8)	6 時間	〃	○	○	○	○
l-interactive_4	3 ～ 4 (16)	1 時間	〃	○	○	○	○
(l-regular-low) ※6							
l-small-low	1 ～ 4 (16)	12 時間	244	▲	▲	▲	▲
l-medium-low	5 ～ 8 (32)	〃	〃	×	▲	▲	▲
l-large-low	9 ～ 16 (64)	〃	〃	×	▲	▲	▲

▲　パーソナルコースは 最大 1 ノード，グループコースは申込ノード数の 4 分の 1 まで実行可（公募制度・ノード固定による申し込みの場合は申込ノード数まで実行可）
※1 キューの指定（〝#PBS -q キュー名〞）は，l-debug，l-regular を小文字で指定する
l-regular キューはノード数の指定（〝#PBS -l select=ノード数〞）でノード数別のキューに投入される
※2 トークンの消費係数は 4.00
※3 1ノード当りの利用者が利用可能なメモリー容量
※4 申込ノード数の合計以内ならば，キュー名・制限時間（原則 168 時間以内）は相談の上，任意に設定可能
※5 インタラクティブジョブの起動は次のとおり（トークン消費あり）
qsub -I -q l-interactive -l select=ノード数 -l walltime=XX:XX -W group_list=グループ名
※6 非優先ジョブクラス（低プライオリティキュー）は，トークンの追加申込が締め切られ，ジョブ実行に必要なトークン残量が不足した場合のみ実行可

巻頭言：スーパーコンピューティングは人類と地球を護る

中 島 研 吾
東京大学情報基盤センター

新しい年，2021 年の初めにあたって，皆さまとご家族，ご友人，周囲の皆さまのご健康を心からお祈り申し上げます。2020 年は人類がかつて経験したことのない一年でした。何よりも，健康であることのありがたさ，命の大切さを改めて実感した次第です。

2020 年 4 月 6 日に発表された政府の緊急事態宣言予告，および東京都の緊急事態措置案を受け，東京大学では「新型コロナウイルス感染拡大防止のための東京大学の活動制限指針[1]」（2020 年 4 月 3 日発表）が 4 月 8 日（水）からレベル 3（制限－大）に引き上げられ，当センターにおいても活動レベルをレベル 3 として，一部の保守サービスを縮退するなどして対応してまいりました[2]。その後，緊急事態宣言解除に応じて，段階的に活動レベル制限を緩和しましたが，2020 年 11 月以降の感染再拡大にともない，2021 年 1 月 1 日現在，レベル 2（制限－中）にてサービスを継続しております。当センター教職員，スパコン関連各社技術者との緊密な協力により，利用者の皆さまへの影響は最小限に留まっていると考えておりますが，細かいところでご不便をおかけしていること，この場をお借りしてお詫び申し上げます。

さて，新型コロナウイルス感染症 （COVID-19）に関連した問題解決に向けては「防疫」，「治療」，「創薬」など広範囲にわたり様々な手法による研究開発が急務であり，スーパーコンピュータの有する高速な計算能力，データ処理能力の貢献が期待されております。 このような状況の下，HPCI[3]（革新的ハイパフォーマンス・コンピューティング・インフラ ） においては，関係機関の協力のもと，関連する研究が必要とする計算資源を提供する臨時の課題募集「新型コロナウイルス感染症対応 HPCI 臨時公募課題[4]」がおこなわれています。当センターも HPCI 構成機関の一つとして積極的に参加しており，全 14 課題のうち 6 課題が当センターの Oakbridge-CX，Oakforest-PACS（JCAHPC）を使用しています。

一つ，明るいニュースとして，当センターでは新システム「Wisteria/BDEC-01（「計算・データ・学習」融合スーパーコンピュータシステム），ピーク性能 33.1 PFLOPS」の導入を決定いたしました[5]。同システムは東京大学柏IIキャンパスに建設中の総合研究棟（情報系）に設置され，2021 年 5 月 14 日に共同利用システムとして稼働開始します。

「Wisteria/BDEC-01」は「シミュレーションノード群（Odyssey）」と「データ・学習ノード群（Aquarius）」を有し，「計算・データ・学習」融合により，「Society 5.0[6]（サイバー空間（仮想空間）とフィジカル空間（現実空間）を高度に融合させたシステムにより，経済発展と社会的課題の解決を両立する，人間中心の社会（Society））」の実現に貢献するものです。シミュレーションノード群（Odyssey）は，世界最高性能を有するスーパーコンピュータ「富岳」と同じ富士通株式会社の「FUJITSU Processor A64FX」を 7,680 基搭載，ピーク性能は 25.9 PFLOPS で

---

1 https://www.u-tokyo.ac.jp/content/400137553.pdf
2 https://www.cc.u-tokyo.ac.jp/covid-19/
3 https://www.hpci-office.jp/
4 https://www.hpci-office.jp/pages/adoptionlist2020_25
5 https://www.cc.u-tokyo.ac.jp/public/pr/pr-wisteria.php
6 https://www8.cao.go.jp/cstp/society5_0/

す。データ・学習ノード群（Aquarius）にはインテル ディープラーニング・ブースト・テクノロジーを有するインテル社製「第3世代Xeonスケーラブルプロセッサ（開発コード名Ice Lake）」90基，NVIDIA社の最新GPUである「NVIDIA A100 Tensorコア」を360基搭載，ピーク性能は7.2 PFLOPSです。

「Wisteria/BDEC-01」は，シミュレーションノード群（Odyssey），データ・学習ノード群（Aquarius）を使用し，計算科学，データ科学，人工知能・機械学習の幅広いアプリケーションをカバーすることによって，最先端の科学技術計算を支える重要なインフラとなることは言うまでもありませんが，東京大学の関連各部局（生産技術研究所，地震研究所，大気海洋研究所，物性研究所），理化学研究所，及び利用者の皆さまとの協力のもと，ものづくり，地球科学分野（固体地球，大気・海洋），物質科学などの分野における「計算・データ・学習」融合により，Society 5.0実現に向けた重要なプラットフォームとなることが期待されます。

近年，当センターのシステムはYayoi，Oakleaf，Oakbridge，Oakforest，Reedbushなど設置キャンパスやシステムに由来する植物を名称として使用してきました。Wisteria（紫藤）は柏市にある手賀沼に伝わる「藤姫伝説」に因んでおり，藤の蔓の如く，「計算・データ・学習」融合のための各ノード群，ファイルシステム群が緊密に結合していく様を示しています。

シミュレーションノード群（Odyssey）とデータ・学習ノード群（Aquarius）は，それぞれアポロ13号の司令船（Command Module，CM）と月着陸船（Lunar Module，LM）の名称です。地球はいまCOVID-19により未曾有の危機に晒されています。OdysseyとAquariusがアポロ13号乗組員の地球への無事帰還をサポートしたごとく，Wisteria/BDEC-01も地球と人類を護り，救うことに貢献できれば，という願いが込められています。

遠くない将来に様々な問題が解決して，利用者の皆さまに直接お会い出来る日が来ることを心から願っております。そしてその問題の解決に当センターのシステム群（Reedbush-H/L，Oakforest-PACS，Oakbridge-CX，Wisteria/BDEC-01）が役立てられるよう，利用者の皆さまとは一層緊密にご協力させていただければと思います。是非お気軽にご相談ください。

それでは，本年もよろしくお願いいたします。

センターから

サービス休止等のお知らせ

2021 年 1 月下旬からの計算機サービス予定は以下のとおりです。

Oakbridge-CX スーパーコンピュータシステム

○ Oakbridge-CX スーパーコンピュータシステム サービス休止のお知らせ

日　付	利用者サービス	センター内作業
1 月 29 日（金）	9:00 ～ 20:00 までサービス休止	月末処理
2 月 26 日（金）	9:00 ～ 20:00 までサービス休止	月末処理
3 月 31 日（水）～ 4 月　1 日（木）	3/31 9:00 ～ 4/1 9:30 までサービス休止	年度末処理

・ Oakbridge-CX システムは，原則 24 時間サービスを行っています。
ただし，月末処理日 （原則として毎月最終金曜日） はサービスを停止します。

○ Oakbridge-CX スーパーコンピュータシステム 大規模 HPC チャレンジ のお知らせ (*)

大規模 HPC チャレンジ 実施期間
1 月 28 日（木）9:00 ～ 1 月 29 日（金）9:00 まで 2 月 25 日（木）9:00 ～ 2 月 26 日（金）9:00 まで 3 月 30 日（火）9:00 ～ 3 月 31 日（水）9:00 まで

・上記期間中，Oakbridge-CX の debug，short，regular，interactive，prepost，ノード固定 および 講義用キューのサービスを休止します。
ログインノードは通常どおり利用できます。

Oakforest-PACS スーパーコンピュータシステム

○ Oakforest-PACS スーパーコンピュータシステム サービス休止のお知らせ

日　付	利用者サービス	センター内作業
1 月 29 日（金）	9:00 ～ 22:00 までサービス休止	月末処理
2 月 26 日（金）	9:00 ～ 22:00 までサービス休止	月末処理
3 月 31 日（水）～ 4 月　1 日（木）	3/31 9:00 ～ 4/1 9:30 までサービス休止	年度末処理

・ Oakforest-PACS スーパーコンピュータシステムは，原則 24 時間サービスを行っています。
ただし，月末処理日（原則として毎月最終金曜日）はサービスを停止します。

○ Oakforest-PACS スーパーコンピュータシステム 大規模 HPC チャレンジ のお知らせ (*)

大規模 HPC チャレンジ 実施期間
1 月 28 日（木）9:00 ～ 1 月 29 日（金）9:00 まで 2 月 25 日（木）9:00 ～ 2 月 26 日（金）9:00 まで 3 月 30 日（火）9:00 ～ 3 月 31 日（水）9:00 まで

・上記期間中，Oakforest-PACS の regular-flat，regular-cache，debug-flat，debug-cache，interactive-flat，interactive-cache および 講義用キューのサービスを休止します。ログインノード，prepost キューは通常どおり利用できます。

Reedbush スーパーコンピュータシステム

○ Reedbush スーパーコンピュータシステム（Reedbush-H, Reedbush-L）サービス休止のお知らせ

日　付	利用者サービス	センター内作業
1 月 29 日（金）	9:00 ～ 17:00 までサービス休止	月末処理
2 月 26 日（金）	9:00 ～ 17:00 までサービス休止	月末処理
3 月 31 日（水）～ 4 月　1 日（木）	3/31 9:00 ～ 4/1 9:30 までサービス休止	年度末処理

・Reedbush-H, Reedbush-L スーパーコンピュータシステムは，原則 24 時間サービスを行っています。
ただし，月末処理日（原則として毎月最終金曜日）はサービスを停止します。

○ Reedbush スーパーコンピュータシステム（Reedbush-H）大規模 HPC チャレンジ のお知らせ（*）

大規模 HPC チャレンジ 実施期間
1 月 28 日（木）9:00　～　1 月 29 日（金）9:00 まで 2 月 25 日（木）9:00　～　2 月 26 日（金）9:00 まで 3 月 30 日（火）9:00　～　3 月 31 日（水）9:00 まで

・上記期間中，　Reedbush-H の h-debug, h-short, h-regular, h-interactive および 講義用キューのサービスを休止します。
Reedbush-L, ログインノード および prepost キューは利用可能です。

【注意事項】

・ サービス休止等の計画は原稿作成時の予定です。やむを得ずサービスを変更したり，休止したりする場合がありますので，最新の情報は　login　時のメッセージ及びスーパーコンピューティング部門の　Web　ページの運用スケジュール　(https://www.cc.u-tokyo.ac.jp/supercomputer/schedule.php) をご確認ください。
・ 平日の 9:00～17:00 以外，休日（土・日・祝日等）は，システム障害等でサービスが停止した場合，運転を継続できない場合があります。その場合は，その時間をもってサービスを中止しますのでご了承ください。
＊ 新型コロナにかかわる現状に鑑みて、当面の間大規模 HPC チャレンジを中止いたします。再開、再募集等の際は Web ページ (https://www.cc.u-tokyo.ac.jp/guide/hpc/) で適宜お知らせいたします。

システム変更等のお知らせ

(2020.11.5 － 2021.1.4 変更)

1. ハードウェア

1.1 Oakbridge-CX スーパーコンピュータシステム … なし

1.2 Oakforest-PACS スーパーコンピュータシステム … なし

1.3 Reedbush スーパーコンピュータシステム (Reedbush-H/L) … なし

2. ソフトウェア

2.1 Red Hat Enterprise Linux 7, CentOS 7 (Oakbridge-CX)

gfarm	2.7.17	(2020.11.27)
gfarm2fs	1.2.14	

インストールを実施しました。利用方法については、利用支援ポータルのお知らせ、または
ドキュメント閲覧より利用手引書をご覧ください。

2.2 RedHat Enterprise Linux 7, CentOS 7 (Oakforest-PACS)

Gromacs	2020.4	(2020.11.27)
OPA FM	10.10.3.1-2	(2020.11.27)
gfarm	2.7.17	(2020.11.27)
gfarm2fs	1.2.14	

インストールを実施しました。利用方法については、利用支援ポータルのお知らせ、または
ドキュメント閲覧より利用手引書をご覧ください。

Lustre ファイルシステム	(2020.12.18)

Lustre ファイルシステムの不良対策を実施しました (Lustre ファイルシステムのバージョン
アップを実施)。

2.3 RedHat Enterprise Linux 7 (Reedbush-H/L)

nVIDIA ドライバー	440.118.02	(2020.11.27)
Gromacs	2020.4	(2020.11.27)
gfarm	2.7.17	(2020.11.27)
gfarm2fs	1.2.14	

インストールを実施しました。利用方法については、利用支援ポータルのドキュメント閲覧より
利用手引書または各資料をご覧ください。

3. その他

3.1 Oakbridge-CX におけるバルクサブジョブ同時実行制限の変更について

・ジョブ管理ノード： バルクサブジョブ同時実行制限数をパーソナルコース（1 口：64 本、2 口：128 本、3 口：192 本）、 グループコース（256 本）ともに、16 本に変更（ユーザ登録ツールによる関連設定も変更）

・利用支援 Web ノード： 利用支援ポータルに Xcrypt 講習会の PDF を掲示

詳しくは、利用支援ポータルのドキュメント閲覧より利用手引書をご覧ください。

※Vol.22 No.6 2020.11 号に誤りがありました。お詫びして訂正いたします。

2.2 項 Oakforest-PACS のソフトウェアの変更等は以下のとおりです。
※LAMMPS はインストールされていません。Gromacs は 本号 2.2 項のとおり 2020.11.27 にインストールされています。

TeX Live	2020	(2020.09.25)
Intel 開発環境 2020 (update2)		(2020.09.25)
Intel Compiler	2020.2.254	
Intel MPI	2019.8.254	

2021 年度の利用申込（新規・継続）について

情報戦略課研究支援チーム
情報基盤センタースーパーコンピューティング部門

2021年度のスーパーコンピュータシステム利用申込（新規・継続）は下記のとおり取り扱います[1]。利用申込の内容によって、利用申込期限が異なりますのでご注意ください。

なお、この情報は本原稿作成時点での予定ですので、変更する場合があります。最新の情報については、スーパーコンピューティング部門Webページ（https://www.cc.u-tokyo.ac.jp/）でご確認ください。

また、**2021年4月からの利用に関する「利用登録のお知らせ」の送付については、2021年3月末を予定しております**ので、予めご了解ください。

１．新規利用申込

１－１．Oakbridge-CX、Oakforest-PACS、Reedbushのパーソナルコース

2021年2月上旬にスーパーコンピューティング部門Webページ（https://www.cc.u-tokyo.ac.jp/）に2021年度版の利用申込サイトを公開します。スーパーコンピュータシステム利用規程等をよくお読みになり、利用申込を行ってください。

申込は随時受け付けますが、**2021年4月初めからのご利用を希望される場合は、利用申込期限までに手続きを行ってください。**

■利用申込期限：2021年2月26日（金）（早めに手続きを行ってください）

１－２．Oakbridge-CX、Oakforest-PACS、Reedbush のグループコース

2021年1月下旬にスーパーコンピューティング部門Webページ（https://www.cc.u-tokyo.ac.jp/）に2021年度版の利用申込サイトを公開します。スーパーコンピュータシステム利用規程等をよくお読みになり、利用申込を行ってください。

申込は随時受け付けますが、**2021年4月初めからのご利用を希望される場合は、利用申込期限までに手続きを行ってください。**

なお、**申込状況により利用のお断りもしくは希望セット数どおりの提供ができない場合があります。**

■利用申込期限：2021年2月12日（金）（早めに手続きを行ってください）

---

[1] 企業、若手・女性、大規模 HPC チャレンジ、学際大規模情報基盤共同利用・共同研究拠点及び HPCI 等の公募制度による利用、講義・講習会等の教育利用及びトライアルユースを除きます。

２．継続利用申込

２－１．Oakbridge-CX、Oakforest-PACS、Reedbushのパーソナルコース

2021年2月上旬に2020年度の登録内容を記載した「継続手続きの案内」を利用者の方に送付します。2021年度も継続利用される場合は、案内に従い、利用申込期限までに手続きを行ってください。

■利用申込期限：2021年2月26日（金）（早めに手続きを行ってください）

２－２．Oakbridge-CX、Oakforest-PACS、Reedbushのグループコース

2021年1月下旬に代表者の方に2020年度の登録内容を記載した「継続手続きの案内」を送付します。2021年度も継続利用される場合は、案内に従い、利用申込期限までに手続きを行ってください。

なお、**申込状況により利用のお断りもしくは希望セット数どおりの提供ができない場合があります。**

また、提供セット数の決定にあたっては、2020年度利用実績、研究成果登録状況等を参考にさせていただく場合があります。

■利用申込期限：2021年2月12日（金）（早めに手続きを行ってください）

３．問い合わせ先

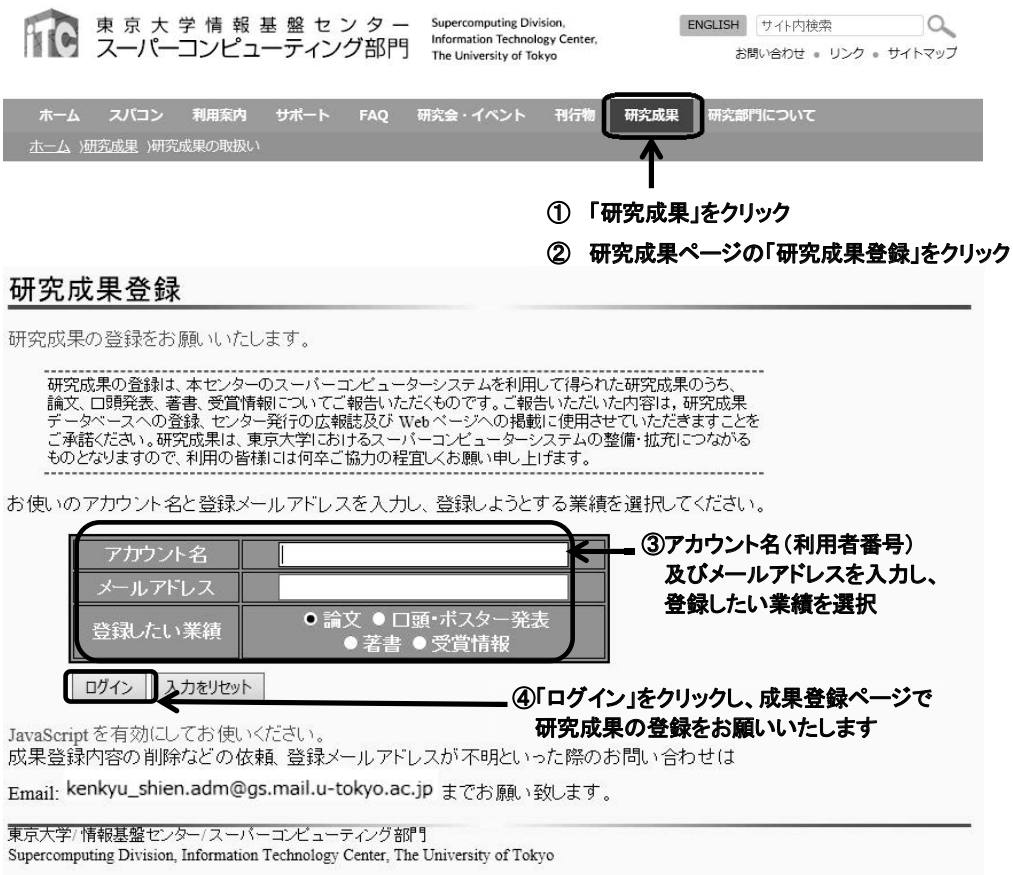
〒113-8658
東京都文京区弥生2-11-16（東京大学情報基盤センター内）
東京大学情報システム部
情報戦略課研究支援チーム
E-mail：uketsuke@cc.u-tokyo.ac.jp

研究成果の登録のお願い

情報戦略課研究支援チーム
情報基盤センタースーパーコンピューティング部門

研究成果の登録は、本センターのスーパーコンピューターシステムを利用して得られた研究成果のうち、論文、口頭発表、著書、受賞情報についてご報告いただくものです。研究成果の登録は、本センタースーパーコンピューティング部門の Web サイト（ https://www.cc.u-tokyo.ac.jp/ ）から「研究成果登録」に進んでください。なお、ご報告いただいた内容は、研究成果データベースへの登録、本センター発行の広報誌及び Web ページに掲載させていただきますので、ご了解ください。

研究成果は、東京大学におけるスーパーコンピューターシステムの整備・拡充につながるものとなりますので、利用者の皆様には何卒ご協力くださいますようお願い申し上げます。



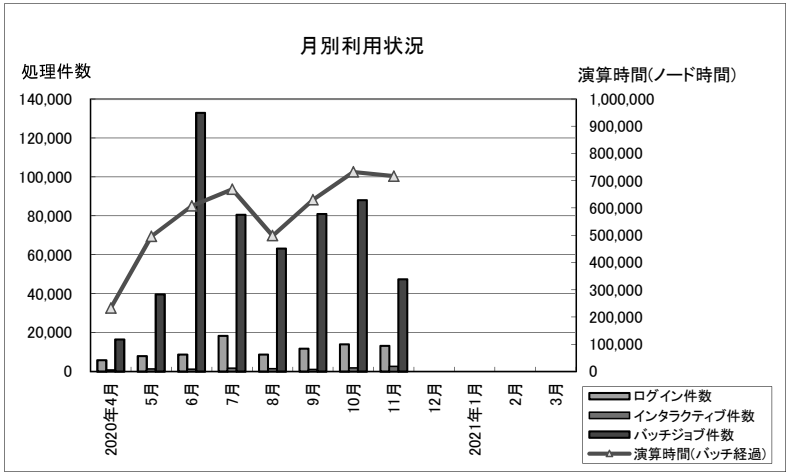
10・11月のジョブ統計

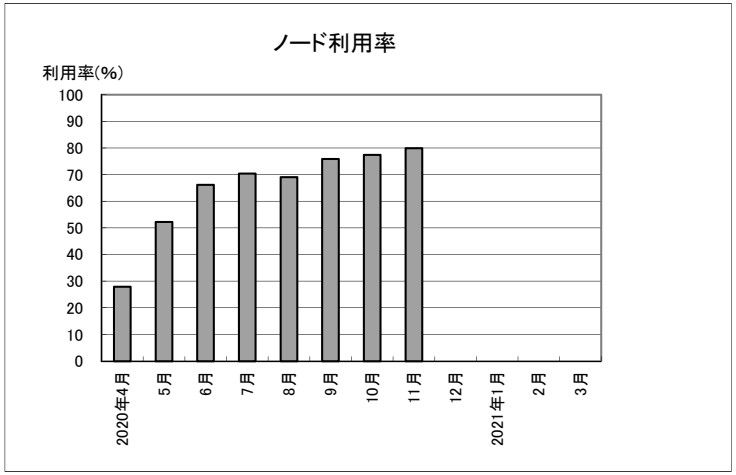
1. Oakbridge-CX スーパーコンピュータシステムジョブ処理状況（Red Hat Enterprise Linux 7、CentOS 7）

年月	登録者数	実利用者数	処理件数				接続時間[時間]	ファイル使用量 [GiB]		ログイン（実CPU）	演算時間 [ノード時間]（経過時間）			平均ノード利用数（ノード）	ノード利用率（%）
			ログイン	プリポスト	インタラクティブジョブ	バッチジョブ		/home	/lustre		プリポスト	インタラクティブジョブ	バッチジョブ		
2020年4月	1,092	147	5,802	20	789	16,434	24,146	647	16,881,212	1,745.43	20	354	233,086	379.9	27.9
5月	1,127	215	7,937	71	1,324	39,551	30,946	684	17,588,622	1,190.20	725	709	496,108	709.9	52.2
6月	1,298	217	8,669	156	1,242	132,825	39,950	1,000	24,529,271	2,673.80	627	936	607,896	898.9	66.1
7月	1,378	347	18,263	118	1,760	80,506	53,532	1,340	23,582,558	5,753.21	519	1,179	668,610	957.8	70.4
8月	1,371	295	8,780	67	1,450	63,090	27,508	1,350	25,301,247	9,448.79	136	949	498,413	938.4	69.0
9月	1,535	299	11,804	48	1,152	80,846	39,017	1,399	26,814,117	3,073.18	119	740	630,436	1,030.7	75.8
10月	1,250	254	13,972	102	1,841	87,992	72,162	1,468	28,961,173	6,384.71	519	829	732,192	1,052.0	77.4
11月	1,241	268	13,169	13	2,707	47,307	62,082	1,472	31,304,992	15,583.41	17	980	716,567	1,086.7	79.9
2019年11月	480	95	4,759	2	379	20,085	34,108	185	5,891,232	3,037.14	1	162	261,169	372.9	27.4
12月	529	114	5,019	1	276	26,914	35,531	393	14,788,166	2,489.42	0	138	371,203	511.4	37.6
2020年1月	562	117	5,831	0	377	32,698	43,594	495	15,790,227	3,237.89	0	141	294,689	410.2	30.2
2月	587	115	5,299	0	326	20,493	32,540	610	17,504,878	1,158.40	0	181	414,952	620.0	45.6
3月	588	98	4,785	1	464	26,064	28,456	586	16,978,662	1,903.42	11	317	680,984	961.4	70.7
合計			109,330	597	13,708	654,720	489,464			54,641.86	2,693	7,453	6,345,136		

・接続時間： ログイン時間の累計
・ログイン（実CPU）： コア時間単位
・2019年11分は合計に含まない

・ノード利用数： インタラクティブおよびバッチジョブの経過時間を1ノードが100%動作したと仮定した場合の利用ノード数。
計算式＝1ヶ月のインタラクティブおよびバッチジョブ経過時間合計÷1ヶ月の稼動時間
・ノード利用率： サービスノードに対する利用比率。 計算式＝ノード利用数÷サービスノード数×100



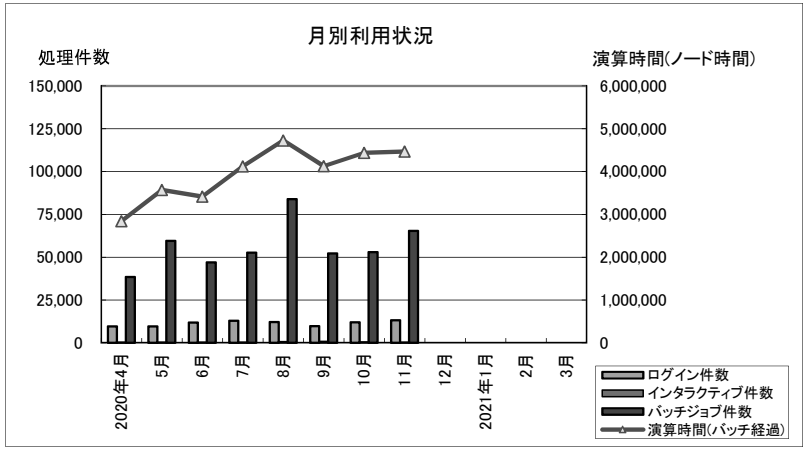


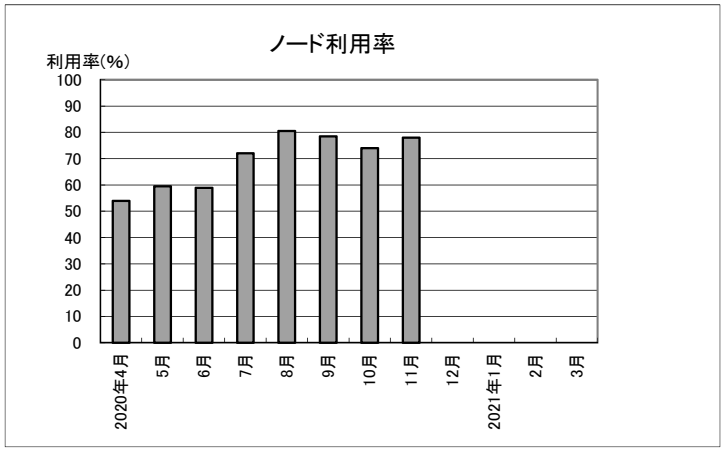
2. Oakforest-PACSスーパーコンピュータシステムジョブ処理状況 （RedHat Enterprise Linux 7、CentOS 7）

年月	登録者数	実利用者数	処理件数				接続時間［時間］	ファイル使用量［GiB］		ログイン（実CPU）	演算時間［ノード時間］（経過時間）			平均ノード利用数（ノード）	ノード利用率（%）
			ログイン	プリポスト	インタラクティブジョブ	バッチジョブ		/home	/work		プリポスト	インタラクティブジョブ	バッチジョブ		
2020年4月	1,885	464	9,576	461	253	38,464	54,218	3,927	6,326,936	1,829.07	709	214	2,840,983	4,423.7	53.9
5月	1,802	446	9,554	247	254	59,496	64,493	3,220	5,455,455	2,698.20	502	152	3,571,465	4,885.8	59.5
6月	1,611	455	11,840	151	358	47,010	67,932	3,361	5,858,067	1,957.58	342	161	3,415,672	4,831.4	58.9
7月	1,622	410	12,928	343	327	52,585	64,608	3,446	5,996,751	2,262.33	492	163	4,121,263	5,909.8	72.0
8月	1,617	401	12,139	315	618	83,908	70,019	3,854	6,332,050	11,191.93	685	514	4,726,977	6,611.0	80.5
9月	1,732	420	9,811	293	795	52,222	56,925	3,964	6,299,074	3,237.18	667	1,390	4,125,115	6,447.3	78.5
10月	1,766	522	12,023	191	464	52,938	82,796	4,179	6,492,672	6,460.30	398	710	4,437,924	6,071.9	74.0
11月	1,822	461	13,149	251	256	65,413	74,790	4,107	6,618,475	3,297.48	516	125	4,468,068	6,400.5	78.0
2019年11月	1,616	456	13,103	457	208	51,790	103,507	3,683	5,277,714	9,000.08	956	151	4,248,874	6,009.4	73.2
12月	1,618	454	14,180	470	437	72,460	89,295	3,659	5,615,226	5,920.77	976	214	4,913,167	6,721.0	81.9
2020年1月	1,598	445	11,965	408	378	48,545	118,573	3,725	6,034,310	9,427.65	666	217	5,198,931	7,111.9	86.6
2月	1,629	424	10,772	359	401	47,509	78,624	3,850	6,253,958	12,649.16	768	192	4,977,581	7,321.7	89.2
3月	1,598	401	11,791	536	533	64,010	80,219	3,761	6,408,220	9,302.12	1,062	196	5,581,379	7,713.3	94.0
合計			139,728	4,025	5,074	684,560	902,492			70,233.77	7,783	4,248	52,378,525		

・接続時間： ログイン時間の累計
・ログイン（実CPU）： コア時間単位
・2019年11分は合計に含まない

・ノード利用数： インタラクティブおよびバッチジョブの経過時間を1ノードが100%動作したと仮定した場合の利用ノード数。
計算式＝ノード利用率×総ノード数(8208)
・ノード利用率： サービスノードに対する利用比率。 計算式＝利用ノード時間÷サービスノード時間×100



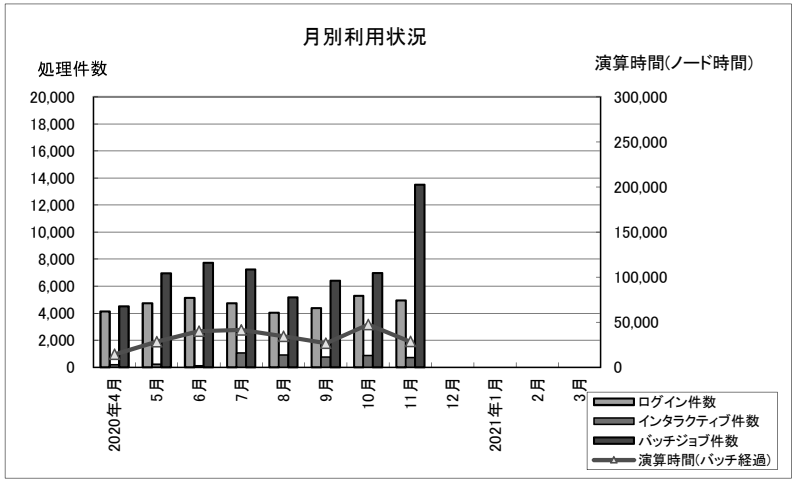


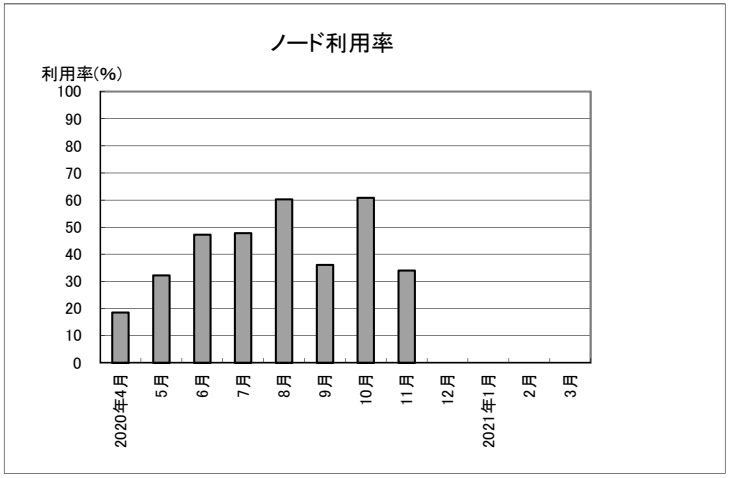
3. Reedbushスーパーコンピュータシステム（Reedbush-H）ジョブ処理状況 （RedHat Enterprise Linux 7）

22巻9月号よりReedbush-H/L/Uのジョブ処理状況の掲載順を変更しています。

年月	登録者数	実利用者数	処理件数				接続時間[時間]	ファイル使用量 [GiB]		ログイン（実CPU）	演算時間 [ノード時間]（経過時間）			平均ﾉｰﾄﾞ利用数（ノード）	ﾉｰﾄﾞ利用率（%）
			ログイン	プリポスト	インタラクティブジョブ	バッチジョブ		/home	/lustre		プリポスト	インタラクティブジョブ	バッチジョブ		
2020年4月	1,287	239	4,124	1	181	4,512	5,558	169	435,177	6.74	0	165	14,192	22.1	18.5
5月	1,287	204	4,724	1	230	6,950	7,699	133	300,576	9.43	0	293	28,184	38.6	32.2
6月	1,032	235	5,136	2	110	7,720	7,073	145	306,587	5.06	0	29	39,952	56.2	47.2
7月	1,069	188	4,736	8	1,053	7,234	7,653	148	299,291	6.06	5	1,024	41,261	57.4	47.8
8月	999	171	4,032	0	897	5,165	6,535	157	298,649	7	0	1,213	34,188	72.2	60.2
9月	986	161	4,364	0	742	6,403	8,054	159	321,098	12.18	0	1,077	26,655	43.3	36.1
10月	1,127	214	5,277	0	869	6,962	7,133	168	327,145	18.74	0	973	47,424	72.9	60.8
11月	1,070	188	4,943	0	717	13,502	6,705	168	337,922	7.48	0	675	28,402	40.8	34.0
2019年11月	1,294	333	8,566	21	589	9,721	11,653	168	460,973	17.58	17	247	51,108	72.1	60.1
12月	1,254	324	8,563	0	472	6,545	13,520	176	520,955	18.00	0	386	41,445	56.8	47.4
2020年1月	1,241	316	9,461	1	787	15,949	15,181	186	541,491	13.84	0	537	57,031	78.2	65.2
2月	1,237	273	5,826	0	515	25,272	9,731	183	616,806	11.83	0	323	42,821	62.7	52.2
3月	1,217	227	5,185	0	158	3,650	8,216	184	604,794	30.69	0	138	51,619	71.0	59.2
合計			66,371	13	6,731	109,864	103,058			146.76	5	6,833	453,174		

・接続時間： ログイン時間の累計
・ログイン（実CPU）： コア時間単位
・2019年11月分は合計に含まない
・ノード利用数： インタラクティブおよびバッチジョブの経過時間を1ノードが100%動作したと仮定した場合の利用ノード数。
計算式＝1ヶ月のインタラクティブおよびバッチジョブ経過時間合計÷1ヶ月の稼動時間
・ノード利用率： サービスノードに対する利用比率。 計算式＝ノード利用数÷サービスノード数×100



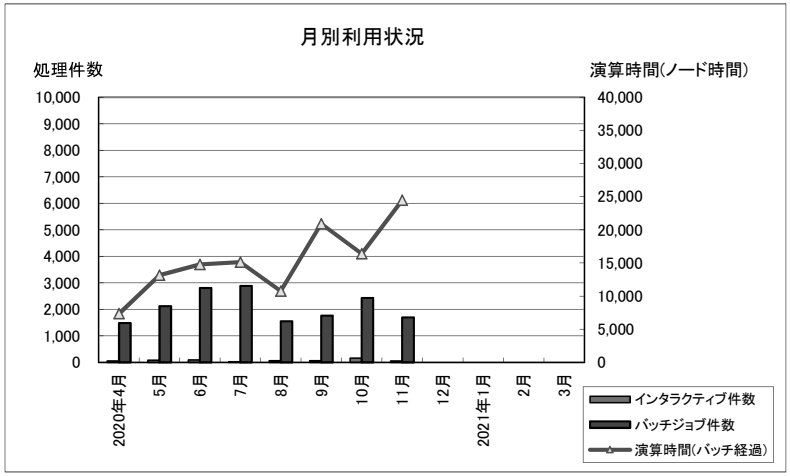


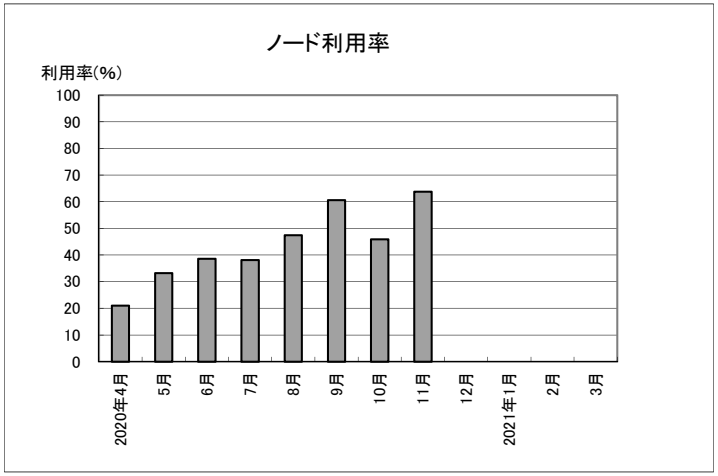
4. Reedbushスーパーコンピュータシステム（Reedbush-L）ジョブ処理状況 （RedHat Enterprise Linux 7）

年月	処理件数		演算時間 [ノード時間](経過時間)		平均ノード利用数（ノード）	ノード利用率（%）
	インタラクティブジョブ	バッチジョブ	インタラクティブジョブ	バッチジョブ		
2020年4月	54	1,484	37	7,332	11.4	21.0
5月	80	2,115	74	13,149	17.9	33.2
6月	91	2,807	79	14,764	20.8	38.6
7月	21	2,883	27	15,123	20.6	38.1
8月	58	1,551	64	10,739	25.6	47.4
9月	57	1,757	49	20,902	32.7	60.6
10月	156	2,427	53	16,400	24.8	45.9
11月	56	1,695	25	24,476	34.4	63.7
2019年11月	118	2,083	91	21,247	30.0	54.5
12月	93	2,313	80	26,748	36.5	66.3
2020年1月	133	2,560	158	32,196	44.0	79.9
2月	56	2,657	211	28,072	41.1	74.7
3月	16	727	24	34,927	47.9	87.2
合計	871	24,976	881	244,828		

・登録者数、実利用者数、ログイン件数、接続時間、ファイル使用量、ログイン（実CPU）はReedbush-Uと共通。
・2019年11月分は合計に含まない

・ノード利用数： インタラクティブおよびバッチジョブの経過時間を1ノードが100%動作したと仮定した場合の利用ノード数。
計算式＝1ヶ月のインタラクティブおよびバッチジョブ経過時間合計÷1ヶ月の稼動時間
・ノード利用率： サービスノードに対する利用比率。 計算式＝ノード利用数÷サービスノード数×100





Oakbridge-CX への計算物質科学ソフトウェアのインストール ー CP2K・LAMMPS

芝　隼 人

東京大学情報基盤センター

１．はじめに

現在、東京大学情報基盤センター（以下、当センター）では Reedbush-H/L, Oakforest-PACS, および Oakbridge-CX の３つのスーパーコンピューター(スパコン)システムが運用されている。これらの中で比較すると Oakbridge-CX （OBCX） がもっとも最近になって調達されたシステムであるが、以前のシステム上で利用されたプログラムからの高い移植性および汎用性を重視した設計となっており、幅広い研究分野における利用が可能な構成となっている。

私自身は 2020 年 2 月に当センターに着任して研究業務および運用業務にあたるようになったが、もとは物質科学・材料科学分野を出自とするアプリケーション屋である。この中で、OBCX において物質科学系のプログラムに関する相談案件が多いことが特に目に止まったため、将来導入するシステム構成検討も兼ねて OBCX システムにおける物質・材料系アプリケーション構成の見直しを行った。物質・材料系のソフトウェアは（工学・ものづくり分野のものとの比較においては）オープンソースライセンスによるものが多い。したがってセンターの立場からすると、サポートへの障壁も低い。物質・材料系分野の研究者による利用が多いアプリケーションとして CP2K および LAMMPS を選び、OBCX 上でビルドし、提供することにした。本稿は、利用者に資するべく、これらソフトウェアのビルドの手順を紹介するものである。

２．CP2K

CP2K は分子軌道法および第一原理密度汎関数法による分子の電子状態計算および各種アンサンブルによる分子力学計算を行うことができるオープンソースソフトウェアである。利用者の量子化学理論の背景知識を前提として、第一原理（ab initio）分子動力学法や QM/MM 法の大規模並列計算を可能にする。2000 年前後にプロトタイプコードの開発が開始、以降開発が現在まで続けられている。本稿執筆(2020 年 11 月末)時点での最新安定版は v7.1 であるが、今回は開発最新版である Trunk（v8.0, development）のインストールを行った。

以下、 OBCX における v8.0 のビルド方法を示す。なお、今後最新安定版として v8.1 がリリースされ次第、改めてインストールする予定である。

2.1　ソフトウェアのダウンロード

最初に、OBCX システムのログインノードにて、ソフトウェアをインストールする /work 以下の自身のディレクトリに移動するとともに、デフォルトコンパイラ Intel Parallel Studio XE 2019（19.0.5.281）がロードされていることを確認する。

```
$ module list
>   1) impi/2019.5.281    2) intel/2019.5.281
```

続いてソフトウェアをGitHubリポジトリからダウンロードする。

```
$ git clone --recursive https://github.com/cp2k/cp2k.git cp2k
```

作業ディレクトリ内にcp2kの名称のディレクトリが作られる（以下、${WORKDIR}と表記する）。その内部にソースコード、ライブラリ構築のためのtoolchainなど、必要なファイルが配置される。

2.2 toolchain を用いたライブラリの構築

CP2K のビルドに際して、必要な追加ライブラリのビルド作業を実施する。不足しているライブラリは CP2K に付属して提供される toolchain を用いて導入することができる。まずは、toolchain の配置されるディレクトリに移動するとともに、使用コンパイラの情報を環境変数により設定する。

```
$ cd ${WORKDIR}/tools/toolchain
$ export CC=icc
$ export CXX=icpc
$ export FC=ifort
$ export MPICC=mpiicc
$ export MPICXX=mpiicpc
$ export MPIFC=mpiifort
```

toolchain を使用して、OBCX システムに備わっていない一連のライブラリの構築を実施する。toolchain においては tools/toolchain/scripts ディレクトリ内にそれぞれのインストールスクリプトがあり、以下の書き換えを実施する（以下、行番号は開発版のアップデートによりずれることがあり得る）。

- tools/toolchain/scripts/install_libint.sh: 74行目付近に1行追加する
  - ✓ 自然科学研究機構（分子研）スパコンサイト[1]にあるCP2K v7.1のビルド方法の記載と同様、intel compiler でビルドが通らないパッケージを強制的に外している。

```
 L72: #cmake --build . > cmake.log 2>&1
 L73: #cmake --build . --target install > install.log 2>&1
+L74: sed -i -e "s/fortran_example check_test/libint_f.o \
      check_test/" fortran/Makefile.in
 L75: ./configure --prefix=${pkg_install_dir}  --with-cxx="$CXX \
        $LIBINT_CXXFLAGS"  --with-cxx-optflags="$LIBINT_CXXFLAGS \
      ……     （以下略）
```

[1] https://ccportal.ims.ac.jp/node/2639

- tools/toolchain/scripts/install_elpa.sh
  - ✓ Intel MKL をリンクさせるため。

L101: FCFLAGS="**-mkl=cluster** ${FCFLAGS} ${MATH_CFLAGS} …"

fftw, mpi-fftw, gsl, hdf5, superlu は、OBCX システムにプリインストールされたものを Environmental Module を呼び出すことで利用する[2]。

```
$ module load fftw mpi-fftw
$ module load gsl
$ module load hdf5
$ module load superlu
```

以上の準備ができたら、toolchain スクリプトを、必要なオプション指定を行った上で走らせる（cmake 3 系および python 3 系を要求されるので、こちらの Environmental module もロードしておく）。 CP2K v8.0 以降に附属の toolchain では、Intel MPI を使用したビルドのオプションが利用可能となって（--mpi-mode=intelmpi）、OBCX 環境でのビルドが容易となっている。なお現在のところ、SIRIUS および COSMA の両ライブラリの OBCX 上でのビルドについては、成功していない。

```
$ cd ..
$ module load cmake/3.14.5
$ module load python/3.7.3
$ ./install_cp2k_toolchain.sh --math-mode=mkl --mpi-mode=intelmpi \
--with-cmake=system --with-libxsmm=install --with-mkl=system \
--with-fftw=system --with-reflapack=no --with-scalapack=no \
--with-sirius=no  --with-cosma=no --with-plumed --with-gsl=system \
--with-hdf5=system --with-superlu=system
```

以上により、CP2K が利用するライブラリの構築が完了となる。

---

[2] システム側で FFTW3 のプリインストールがない場合には、toolchain を用いてインストールが可能である。この場合、Intel Compiler 使用時は次のような形で書き換える。toolchain インストールスクリプトの実行オプションでは--with-fftw=install とする。

- tools/toolchain/scripts/install_fftw.sh

```
 L49: grep '\bavx512f\b' /proc/cpuinfo 1>/dev/null && \
      FFTW_FLAGS="${FFTW_FLAGS} --enable-avx512"
+L50: sed -i -e "s/-no-gcc//g" configure
 L51: ./configure  --prefix=${pkg_install_dir} \
   --libdir="${pkg_install_dir}/lib" ${FFTW_FLAGS} > configure.log
```

2.3 ソフトウェアビルド

以上の準備が完了したら、ビルドに入ることが可能となる。OBCX システムではノード内に 28 コアの Intel CascadeLake プロセッサ 2 基が配され、ノードあたり合計 192 GiB のメモリを搭載している。CP2K はメモリ量に対する要求が非常に強いアプリケーションであるため、メモリ使用量の逓減のために CPU ソケット内部でのスレッド並列とするハイブリッド並列を行うことが合理的な選択となる。この理由から、今回は、psmp 版（MPI + OpenMP ハイブリッドにより並列化されたもの）をビルドすることとした。

CP2K パッケージの arch ディレクトリ内には、いくつかのアーキテクチャ向けのビルドオプション、コンパイルリンクを指定した arch ファイルが配置されている。ここでは、そのうちの Linux-x86-86-intel-regtest.psmp をリネームし、obcx.psmp として使用する。

```
$ cp Linux-x86-86-intel-regtest.psmp obcx.psmp
```

obcx.psmp の冒頭部分でコンパイラオプションおよびライブラリパスを次のように設定する。なお太字で示したのが変更部分であり、${WORKDIR}は CP2K のインストールディレクトリを示す。なお、Intel MKL のライブラリはシーケンシャル・モードを使用する。

obcx.psmp

```
CC          = mpiicc -std=c1x
FC          = mpiifort
LD          = mpiifort
AR          = ar -r

MPI_PATH    = ${WORKDIR}/tools/toolchain/install
INTEL_PATH = ${WORKDIR}/tools/toolchain/install

include  \
    $(MPI_PATH)/plumed-2.6.1/lib/plumed/src/lib/Plumed.inc.static

ELPA_VER    = 2020.05.001
ELPA_INC \
    = (MPI_PATH)/elpa-2020.05.001/include/elpa_openmp-$(ELPA_VER)
ELPA_LIB    = $(MPI_PATH)/elpa-2020.05.001/lib

LIBINT_INC  = $(INTEL_PATH)/libint-v2.6.0-cp2k-lmax-5/include
LIBINT_LIB  = $(INTEL_PATH)/libint-v2.6.0-cp2k-lmax-5/lib

LIBXC_INC   = $(INTEL_PATH)/libxc-4.3.4/include
LIBXC_LIB   = $(INTEL_PATH)/libxc-4.3.4/lib
```

（次頁へ続く）

```
LIBXSMM_INC = $(INTEL_PATH)/libxsmm-1.16.1/include
LIBXSMM_LIB = $(INTEL_PATH)/libxsmm-1.16.1/lib

SPGLIB_INC  = $(INTEL_PATH)/spglib-1.15.1/include
SPGLIB_LIB  = $(INTEL_PATH)/spglib-1.15.1/lib
```

また、同じ obcx.psmp ファイル中、次のように書き換えを実施する。

```
L58:  LDFLAGS = $(FCFLAGS) -static-intel -static_mpi -lstdc++
L63:  LIBS = $PLUMED_DEPENDENCIES -lgsl -lgslcblas
                                  (なお、libz.a のリンクは外す)
L76:  [LIBS += $(GCC_LIBRARY_DIR)/libstdc++.a] → 削除
```

以上の書き換えが終わったら、インストールディレクトリ中で次のコマンドを実行し、ビルドを行う。

```
$ make -j ${PARALLEL} ARCH=obcx VERSION=psmp
```

これにより実行バイナリ（obcx.psmp）が生成され、CP2K が実行できるようになる。ビルドは OBCX のログインノード上で 1 時間以内に終了する。また、CP2K をライブラリとして使用する場合には、次のコマンドにより libcp2k.a を./lib ディレクトリ内に作ることができる。

```
$ make -j ${PARALLEL} ARCH=obcx VERSION=psmp libcp2k
```

以上で CP2K のインストールは完了である。

今回のビルドでは FFTW3 はシステム側のものを利用する設定としたので、実行する際には

```
$ module load fftw mpi-fftw
```

として該当の Environmental Modules を呼び出されたい。また、使用するライブラリによっては hdf5, superlu も呼び出す必要がある。

また、Intel MKL を使用して MPI-OpenMP ハイブリッドジョブを実行する場合、プロセスあたり OpenMP スレッド数が 2 以上だとしばしばスタックサイズが上限を上回るため、プログラムがエラーメッセージとともに停止する。これを回避するためには環境変数を

```
export OMP_STACKSIZE = 100m
```

などと設定して、スタックサイズを拡張しておく必要がある。今回 OBCX システムに導入した Environmental Module (cp2k/v8.0dev)をロードすると、デフォルトでこの設定が反映される。

また、プロセスあたりスレッド数を 1、すなわち flat MPI とする場合であっても、明示的にスレッド数の指定（例えば export OMP_NUM_THREADS=1）が必要である。基底データファイルを読み出す先のディレクトリは CP2K_DATA_DIR 環境変数によって指定できる。

３．LAMMPS

LAMMPS（Large-scale Atomic/Molecular Massive Parallel Simulator）は、サンディア国立研究所の Steve Plimpton 博士らのグループが中心となって開発した並列古典分子動力学シミュレーション用のオープンソース汎用ソフトウェアである。LAMMPS においては、力場やアンサンブルをはじめとする各種の設定は、モジュール化されたプログラムを呼び出すことで利用する設計となっており、また逆に利用者が必要とする機能や最適化などをモジュールとして実装することも可能である（GPL ライセンスに依拠）。この特徴により、古典分子動力学法が関係する多彩なモデリングを広範にカバーする一大ソフトウェアに成長し、また利用にあたって必要なドキュメンテーションも充実している。

CP2K よりもビルドは容易で、ドキュメンテーションに従っていけば自然とインストールできる。下記に OBCX 上での一通りのセットアップ手順を記す。

3.1　ソフトウェアのダウンロード

最初に、OBCX システムのログインノードにて、ソフトウェアをインストールする /work 以下の自身のディレクトリに移動するとともに、デフォルトコンパイラ Intel Parallel Studio XE 2019（19.0.5.281）がロードされていることを確認する。

```
$ module list
>  1) impi/2019.5.281    2) intel/2019.5.281
```

LAMMPS ソフトウェアを GitHub リポジトリからダウンロード、ビルドを行う作業ディレクトリである src に移動する。

```
$ git clone -b stable https://github.com/lammps/lammps.git mylammps
$ cd  ./mylammps/src
```

“mylammps” はインストールディレクトリ名であり、以下では${INST_DIR}と表記する。これによりダウンロードされるの、2020 年 9 月時点では ver. 3Mar2020 であったが、本稿執筆（2020 年 11 月）の直前に最新安定版が ver. 29Oct2020 アップデートされた。このアップデートに伴う最大の変更点は、ソースコードが C++11 仕様に統一されたことであり、ビルドの Makefile において -std=c++11 のオプションを明示することが必要となった。また、パッケージ構成に一部変更があった。本稿では ver. 29Oct2020 のビルド方法を示す。

3.2 パッケージの選択

LAMMPS のソースコードは、機能に従って分類された複数のパッケージから構成され、パッケージごとにインストールを行うか否かを指定することができる。現在のインストールを指定しているパッケージ状況は、src ディレクトリにて次を実行することで確認できる。

```
$ make package-status
     Installed  NO: package ASPHERE
     Installed  NO: package BODY
     Installed  NO: package CLASS2
     Installed  NO: package COLLOID
                ⋮
```

インストールしたいパッケージについて、次のコマンドを実行する。

```
$ make yes-${PACKAGENAME}
```

${PACKAGENAME}にはパッケージ名を入れるが、LAMMPS にあっては全て大文字で記述する。今回 OBCX システムの Environmental module (lammps/3mar2020)においては、外部ライブラリへの依存性の強い GPU, KIM, KOKKOS, USER-ADIOS, USER-QUIP, USER-SCAFACOS, USER-VTK を除く全てのパッケージをインストールした(この一部にはインストール可能とみられるものもあるが、私自身の時間の制約上、導入を見送った)。

また、LAMMPS Ver. 29Oct2020 では機械学習原子間相互作用ポテンシャルの計算を行う MLIAP パッケージが正規のパッケージとして追加されたが、OBCX における lammps/3mar2020 では提供していない。

3.3 ライブラリのビルド

同じ src ディレクトリ上から、ライブラリ(実体は ${INST_DIR}/lib 上にある)をビルドできる。本節では少し長くなるが、traditional make による各インストール方法を紹介する。HDF5, NetCDF, GSL は OBCX システム上のものを利用できるため、改めてインストールの必要はない。

・LATTE

まず、Intel MKL を利用するようにライブラリビルド用の makefile の設定を実施する。OBCX 上では、現状マニュアル通りのインストール方法のままではビルドが通らないため、ややアドホックではあるが下記のように対処した[3]。

[3] 次のウェブサイトの記述を参考にした:
Compile LATTE package for LAMMPS using Intel Compilers
https://thelinuxcluster.com/2019/06/10/compile-latte-package-for-lammps-using-intel-compilers/

```
$ cd ${INST_DIR}/src
$ make lib-latte args="-b -v 1.2.2"
```

(-b はパッケージのダウンロードおよびインストールを指示するオプションではあるが、一発ではビルドは通らない)

```
$ vi ${INST_DIR}/lib/latte/LATTE-1.2.2/makefile.CHOICES
    L46: #For intel compiler
    L47: FC = ifort
    L48: FCL = $(FC)
    L49: FFLAGS = -O3 -fpp -qopenmp
    L50: LINKFLAG = -qopenmp

    L52: #GNU BLAS/LAPACK libraries:
    L53: #LIB = -llapack -lblas        (コメントアウト)
    L54: #Intel MKL BLAS/LAPACK libraries:
    L55: LIB = -Wl,--no-as-needed -L${MKLROOT}/lib/intel64 \
         -lmkl_lapack95_lp64 -lmkl_gf_lp64 -lmkl_gnu_thread \
         -lmkl_core -lmkl_gnu_thread -lmkl_core -ldl -lpthread -lm

$ cd ${INST_DIR}/lib/latte
$ mv Makefile.lammps Makefile.lammps.past
$ cp Makefile.lammps.ifort Makefile.lammps

$ vi ${INST_DIR}/lib/latte/Makefile.lammps
     latte_SYSINC =
     latte_SYSLIB = ../../lib/latte/filelink.o \
           -llatte -lifport -lifcore -lsvml \ #-lompstub -limf 【削除】
           -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core \
           -lmkl_intel_thread -lpthread -qopenmp  【書換】
     latte_SYSPATH = -qopenmp -L${MKLROOT}/lib/intel64 \    【書換】
                     -lmkl_lapack95_lp64
$ cd ${INST_DIR}/lib/latte
```

・MESSAGE

LAMMPS の現バージョン（ver. 29Oct2020）では、言語仕様として C++11 のみをサポートとなったため、次の通り Makefile を書き換えることが必要となる（以下、他のパッケージでも同様の書き換えを行なっている）

```
$ vi ${INST_DIR}/lib/message/cslib/src/Makefile
   L43: #CC=mpigcc     (次の書き換えを実施する)
  +L43: CC=mpigxx -std=c++11
$ cd ${INST_DIR}/src
$ make lib-message args="-m"
```

・MSCG

```
$ module load gsl
$ vi ${INST_DIR}/lib/mscg/Makefile.mpi
   L26: #CC=mpicc   (次の書き換えを実施する)
  +L27: CC=mpicc -std=c++11
$ cd ${INST_DIR}/src
$ make lib-message args="-b -m mpi"
```

・POEMS

```
$ cd ${INST_DIR}/lib/poems
$ vi Makefile.mpi   (次の書き換えを実施する)
  -L70: #CC =  mpicxx
  -L71: #CCFLAGS =   -O3 -g -fPIC  -Wall
  +L72: CC = mpiicpc -std=c++11
  +L73: CCFLAGS = -O2 -g -fPIC  -Wall
$ cd ${INST_DIR}/src
$ make lib-poems args="-m mpi"
```

・VORONOI

```
$ cd ${INST_DIR}/src
$ make lib-voronoi args="-b"
```

・USER-ATC

```
$ cd ${INST_DIR}/lib/atc
$ vi Makefile.mpi    (次の書き換えを実施する)
  -L72: #CC =  mpicxx
  -L73: #CCFLAGS =      -O3 -Wall -g -fPIC
  +L72: CC = mpiicpc -std=c++11
  +L73: CCFLAGS = -O2 -fp-model fast=2 -no-prec-div \
                  -qoverride-limits -qopt-zmm-usage=high
$ cd ${INST_DIR}/src
$ make lib-atc args="-m mpi"     (← ATC をコンパイル)
```

・USER-AWPMD

```
$ cd ${INST_DIR}/lib/awpmd
$ vi Makefile.mpi    （次の書き換えを実施する）
  -L39: #CC = mpicxx
  -L40: #CCFLAGS =  -O3 -fPIC -Isystems …
  +L39: CC = mpiicpc
  +L40: CCFLAGS = -O2 -g -fPIC -Isystems …（以下そのまま）
$ cd ${INST_DIR}/lib/linalg
$ make -f Makefile.mpi
$ vi Makefile.mpi    （次の書き換えを実施する）
  -L21: #CC = mpifort
  -L21: #CCFLAGS =  -O3 -fPIC
  +L21: CC = mpiifort
  +L21: CCFLAGS = -O2 -fPIC
$ cd ${INST_DIR}/src
$ make lib-linalg args="-m mpi"
$ make lib-awpmd args="-m mpi"
```

・USER-COLVARS

```
$ cd ${INST_DIR}/lib/colvars
$ vi Makefile.mpi    （次の書き換えを実施する）
  -L8 : #CXX =  mpicxx
  -L9 : #CXXFLAGS =  -O3 -g -Wall -fPIC -funroll-loops
  +L10: CXX = mpiicpc -std=c++11
  +L11: CXXFLAGS = -O2 -fp-model fast=2 -no-prec-div \
$ cd ${INST_DIR}/src
$ make lib-colvars args="-m mpi"    （← COLVARS をコンパイル）
```

・USER-H5MD

```
$ cd ${INST_DIR}/src
$ make lib-h5md args="-m h5cc"    （← H5MD をコンパイル）
```

・USER-MESONT

```
$ cd ${INST_DIR}/src
$ make lib-mesont args="-m ifort"    （←MESONT をコンパイル）
```

・USER-PLUMED

```
$ cd ${INST_DIR}/src
$ make lib-plumed args="-b"    （← PLUMED をダウンロード、コンパイル）
```

・USER-QMMM

```
$ cd ${INST_DIR}/src
$ make lib-qmmm args="-m mpi"     (← Quantum Espresso をコンパイル)
```

・USER-SMD

```
$ cd ${INST_DIR}/src
$ make lib-smd args="-b"          (← Eigen をダウンロード)
```

3.4 ソフトウェアビルド

必要なライブラリのインストールができたら、最後にソフトウェアをビルドする。LAMMPS では src/MAKE 以下のディレクトリに、各種のアーキテクチャに対応した Makefile が用意されている。src/MAKE/Makefile.mpi が標準の MPI 並列用の Makefile である。ここでは、OBCX システムにおいて最適な Intel MKL + Intel MPI ライブラリを利用するもので上書きしておく。また、一部パッケージにおいて使用するシステム側のライブラリの Environmental modules をロードする。

```
$ cd MAKE
$ mv Makefile.mpi  Makefile.mpi.past
$ cp ./OPTIONS/Makefile.intel_cpu_intelmpi ./Makefile.mpi
$ module load hdf5/1.10.5  netcfg/4.7.0  gsl/2.5
$ cd ..
```

こうして作成した Makefile.mpi によって、バイナリ名 lmp_mpi という Intel プロセッサ向け最適化済みのバイナリを作成することができる。ビルドは次のコマンドで実行できる。

```
$ make -j ${PARALLEL} mpi
```

また、静的ライブラリオブジェクト liblammps_mpi.a が必要であれば、次のコマンドで生成できる。

```
$ make mode=static mpi
```

ソフトウェアの実行については、src ディレクトリに生成されたバイナリ lmp_mpi を実行することとなるので、src ディレクトリに対して PATH を通しておく。

一部機能（USER-OMP, USER-INTEL パッケージ内）においては MPI-OpenMP ハイブリッド並列を利用できる場合がある。OBCX システムにおけるハイブリッド並列の際のコアバインディングのデフォルト指定は KMP_AFFINITY=balanced となっているが、これは LAMMPS におけるハイブリッド並列では非推奨である。代替として KMP_AFFINITY=scattered などを使用いただきたい。

より具体的な実行インプットファイルの作成方法については、LAMMPS 開発者によって更新されている公式マニュアル（ https://lammps.sandia.gov/doc/Manual.html ）を参照いただきたい。

４．OBCX システムにおけるビルド済 CP2K, LAMMPS の利用方法

OBCX 上にインストールした CP2K および LAMMPS は、OBCX システム上 Environmental Module を呼び出すことにより利用することが可能である。

```
$ module load cp2k/v8.0dev        (CP2K v8.0 development)
$ module load lammps/3mar2020     (LAMMPS ver. 3Mar2020)
```

これらの Environmental Modules の簡単な実行方法については

```
$ module help cp2k/v8.0dev
$ module help lammps/3mar2020
```

Environental Modules の設定内容については

```
$ module show cp2k/v8.0dev
$ module show lammps/3mar2020
```

によりコマンドライン上に表示できるので、参照されたい。

５．結語にかえて

物質・材料研究分野のソフトウェアは単一のランの内部で呼び出すワークロードが比較的多岐に及び、ユーザーが汎用性の高い CPU を搭載したシステムとして OBCX を積極的に選んでいると考えられる。OBCX はこの要件を満たす国内指折りのスパコンシステムであり、物質・材料系のワークロードは当面高い割合を示し続けると予想している。量子化学計算と古典分子動力学計算という物質・材料系を代表する２手法をカバーする代表的ソフトウェアである CP2K, LAMMPS の利用が可能になったことで、これら周辺分野を含めた研究が活性化することを願っている。

CP2K, LAMMPS とも一部機能は GPU による計算が可能である。2021 年 5 月に稼働開始予定の Wisteria/BDEC-01 システムにおいても、CPU 演算ノード群である Odyssey、GPU 搭載ノード群である Aquarius の双方でインストールされる予定である。利用いただければ幸いである。

最後になって付け加えるが、もう一つのソフトウェアとして VASP (The Vienna Ab initio Simulation Package) に言及したい。VASP は PAW 擬ポテンシャル法による平面波基底第一原理計算を行うことができ、第一原理電子状態計算のソフトウェアとして最も普及度が高いものであるが、実は OBCX において最も顕著に多く利用されているソフトウェアであることがわかってきた。2020 年 6 月以来、利用動向調査を OBCX 納入ベンダーである富士通株式会社システムエンジニア各位のご協力をいただき実施継続しているが、ここ半年では、全ジョブのトークン消費量中、

約 15% が VASP 利用と見られる。VASP は利用者グループが直接ライセンスを購入する必要がある有償ソフトウェアであり、センターから積極的なサポートすることは予算面の制約から容易ではないが、引き続き利用動向を注視していきたい。

謝辞

CP2K のビルドに際しては、石井良樹博士（兵庫県立大学大学院シミュレーション学研究科）、大戸達彦博士（大阪大学大学院基礎工学研究科）に助言、意見、およびインプットスクリプトの提供をいただいた。Bo Thomsen 博士（日本原子力研究開発機構システム計算科学センター）には、CP2K の Environmental Module に対して有益なフィードバック情報を戴いた。また、これらソフトウェアを OBCX における Environmental Modules として提供する際に富士通株式会社のエンジニアの方々にお世話になった。ここにこれらの方々に謝意を表する。

南極周極流域の乱流混合過程を想定した３次元 ray-tracing simulation

髙 橋　杏

University of Washington

１．はじめに

深層海洋の子午面循環の構造は，主に海洋内部波の砕波によってもたらされる微細な乱流混合，およびそれに伴う鉛直拡散係数の空間分布に強く依存すると考えられている[1,2]。しかしながら，乱流混合強度や鉛直拡散係数の測定には特殊な測器を必要とし，また膨大な時間を要する。そこで，それらの空間分布を効率よく把握するための代替的な方法として，より観測が容易なファインスケール（鉛直波長 $10-100$ m）の流速鉛直シアーや等密度面のストレインから乱流運動エネルギー散逸率 ε を推定するパラメタリゼーション[3,4,5]が用いられてきた。このパラメタリゼーションでは，ε を「背景内部波間の非線形相互作用により乱流スケールへとカスケードダウンするエネルギーフラックス」として推定する。

本研究で着目する南極周極流は，南極大陸の周りを陸地に阻まれることなく周回する，世界最大級の海流である。ここでは，上空を吹く偏西風の変動に伴い上層に励起される近慣性内部波と，南極周極流が急峻な海底地形と衝突することに伴い深層に励起される内部風下波が，それぞれ砕波することにより，強い乱流混合が生じると考えられている[6,7]。

近年実施された複数の乱流直接観測では，乱流パラメタリゼーションから推定した ε が実際の値よりも最大 10 倍程度大きくなる傾向が報告されている[7,8]。Takahashi (2019)[9] では，流速・密度場と乱流混合強度の同時観測を実施し，その解析を通じて，南極周極流域の多くの場所でシアーやストレインの鉛直波数スペクトルが低波数側に歪んだ形状（図１参照）をしており，そのことによってパラメタリゼーションによる ε の過大評価傾向が生じていると指摘した。乱流パラメタリゼーションの定式化には，シアーやストレインスペクトルの標準模型である Garrett-Munk (GM; Garrett and Munk 1975[10]) の背景内部波スペクトルモデルが用いられており，平坦な鉛直波数スペクトル形状が暗黙に仮定されているからである。

本研究では，内部波エネルギーの周波数-波数空間内におけるカスケード過程を直接評価することができる ray-tracing simulation[3,11] を用いて，鉛直波数スペクトルの歪みが乱流パラメタリゼーションの推定精度にどのような影響を与えるのかを定量的に検証した。

２．Ray-tracing simulation

Ray-tracing simulation では，背景内部波場を構成する内部波パケットのひとつひとつが背景内部波場の流速シアーによる屈折を受け，周波数 σ と波数 $\boldsymbol{k}=(k,l,m)$ を変化させながら伝播していく過程を追跡する。

内部波パケットの鉛直波数 m，周波数 σ，鉛直座標 z の初期値は，

$$m_{i,j,n,\theta}^{init}=\pm\, i/400\ [\mathrm{cpm}]\ \ (i=1,2,\ldots,40), \tag{1}$$

$$\sigma_{i,j,n,\theta}^{init}=\pm 1.01\times 2^{0.5(j-1)} f\ [s^{-1}]\ \ (j=1,2,\ldots,13), \tag{2}$$

$$z_{i,j,n,\theta}^{init}=-200\,n+100\ [\mathrm{m}]\ \ (n=1,2,\ldots,10) \tag{3}$$

から与えられ，水平波数 $\kappa=(k,l)$ の初期値は線形内部重力波の分散関係式（4）を用いて

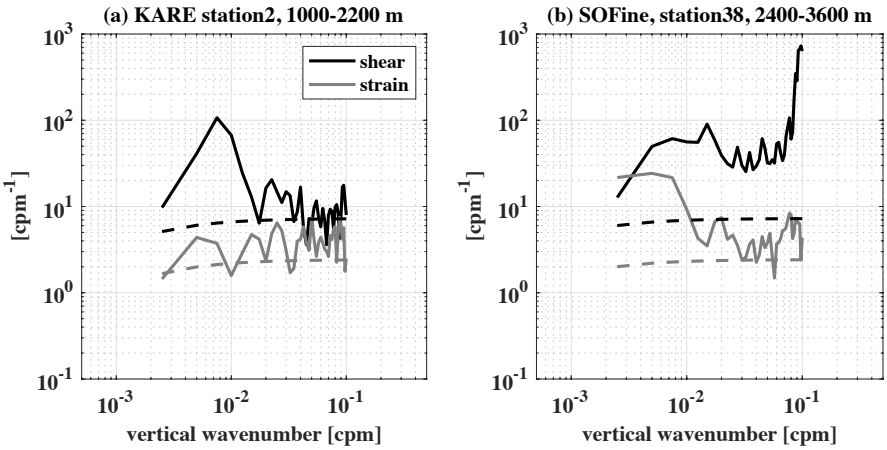


図１： 南極周極流域における現場観測で得られた鉛直波数スペクトル

実線は観測で得られたシアー(黒)，ストレイン(灰色)の鉛直波数スペクトル。観測プロジェクト KARE[8]，SOFine[7] のラベルに示された観測点/水深のデータから計算した。点線は GM スペクトルモデルを示している。

$$\kappa^2 = \frac{\sigma^2 - f^2}{N^2 - \sigma^2} m^2, \tag{4}$$

$$k_{i,j,n,\theta}^{init} = \kappa^{init} \cos\theta_{n'}, \tag{5}$$

$$l_{i,j,n,\theta}^{init} = \kappa^{init} \sin\theta_{n'}, \tag{6}$$

$$\theta_{n'} = \frac{n'\pi}{6}, \qquad (n' = 1, 2, \ldots, 12) \tag{7}$$

から与えられる。ここで f はコリオリ周波数，N は浮力周波数である。

内部波パケットは砕波するまでの間，波活動量 $A = E/\sigma$ （E はエネルギー)を保存しながら，周波数と波数を変化させていく [12]。波数と位置の時間発展方程式は，

$$\frac{d\boldsymbol{k}}{dt} = -\boldsymbol{k} \cdot \nabla \boldsymbol{u_{BG}}, \tag{8}$$

$$\frac{d\boldsymbol{x}}{dt} = \frac{d\sigma}{d\boldsymbol{k}} + \boldsymbol{u_{BG}} \tag{9}$$

のように表され，周波数は線形内部重力波の分散関係式(4)を満たすように変化する。ここで $\boldsymbol{u_{BG}}\,(x, y, z, t)$ は背景内部波場の流速である。

内部波パケットの鉛直波数が砕波限界 $m_{break} = 0.2$ cpm より大きくなったときに「砕波した」とみなし，そのとき内部波パケットが持っていたエネルギー $E^{fin} = A\sigma^{fin}$ と砕波までに要した時間 τ から乱流運動エネルギー散逸率 ε を計算する。

$$\varepsilon = \sum_{i,j,n,\theta}^{\lceil m^{fin} \rceil > m_{break}} \frac{E_{i,j,n,\theta}^{fin}}{\tau_{i,j,n,\theta}} \tag{10}$$

ここで，上付添字 fin は砕波時の値を示す。

１回の ray-tracing simulation を行うのに $124800 (= 40 \times 2 \times 13 \times 10 \times 12)$ 個もの内部波パ

ケットの時間発展を計算する必要があるが，今回 Oakforest-PACS を使用し，この部分を並列化させることで，計算時間を短縮させることができた。

３．実験設定

本研究では，背景内部波場として与えるスペクトルモデルを変更することにより，５種類の ray-tracing simulation を行った（表１）。standard 実験では，背景内部波場として GM 内部波場を与えた。具体的には，乱雑な位相を持つ線形内部重力波が GM スペクトルモデルのエネルギースペクトル $E_{GM}(\sigma, m)$ を満たすように重なり合っている状況を考えた。hump1 から hump4 実験では， GM スペクトルモデルに鉛直低波数の内部波束を重ね合わせることで，実際に南極周極流域で観測されたような歪んだ鉛直波数スペクトル形状を再現したものを背景内部場として与えた。hump とは「こぶ」という意味であり，鉛直低波数側に「こぶ」を持つスペクトルの形状からこのように名付けた。５種類の実験で与えた背景内部波場の鉛直波数スペクトルを図２に示す。

表１： ray-tracing simulation の実験設定

実験名	背景内部波場
standard	GM スペクトルモデル
hump1	GM ＋ 中振幅の近慣性内部波束（σ~$1.01f$）
hump2	GM ＋ 高振幅の近慣性内部波束（σ~$1.01f$）
hump3	GM ＋ 中振幅の内部波束（σ~$2f$）
hump4	GM ＋ 中振幅の内部波束（σ~$2f$）

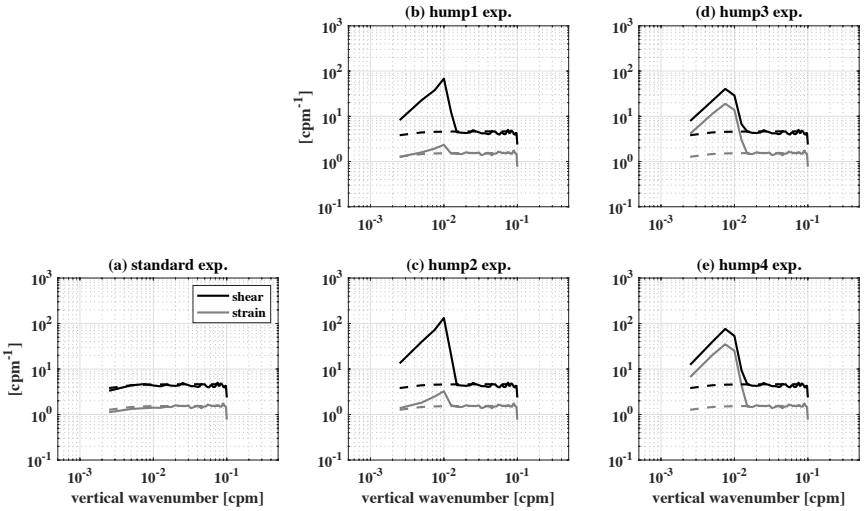


図２： ray-tracing simulation で与えた背景内部波場の鉛直波数スペクトル

実線は各実験で与えた背景内部波場のシアー(黒)，ストレイン(灰色)の鉛直波数スペクトル。点線は GM スペクトルモデルを示している。

４．結果

図３には，３種類の ray-tracing simulation（standard，hump2，hump4 実験）における，内部波パケットが初期状態に持つ波活動量 A，内部波パケットが砕波するまでに要した時間 τ，乱流運動エネルギー散逸率 ε の周波数-鉛直波数分布を示す。τ および ε の図において左下が空白になっているのは，初期に低周波数かつ鉛直低波数の内部波パケットは計算時間内（40 慣性周期）に砕波まで至らなかったことを示している。

２つの hump 実験では，特に鉛直高波数領域で standard 実験よりも ε の値が大きくなっている一方，鉛直低波数の内部波パケットは（standard 実験と同様に）計算時間内に砕波していないことがわかる。これは，スペクトルの hump に対応する鉛直低波数の内部波パケットは，高波数の内部波を砕波させやすくする「背景シアー」としての役割を果たすものの，自ら砕波スケールまでカスケードダウンすることはないため，ε に直接的には寄与しないことを示している。

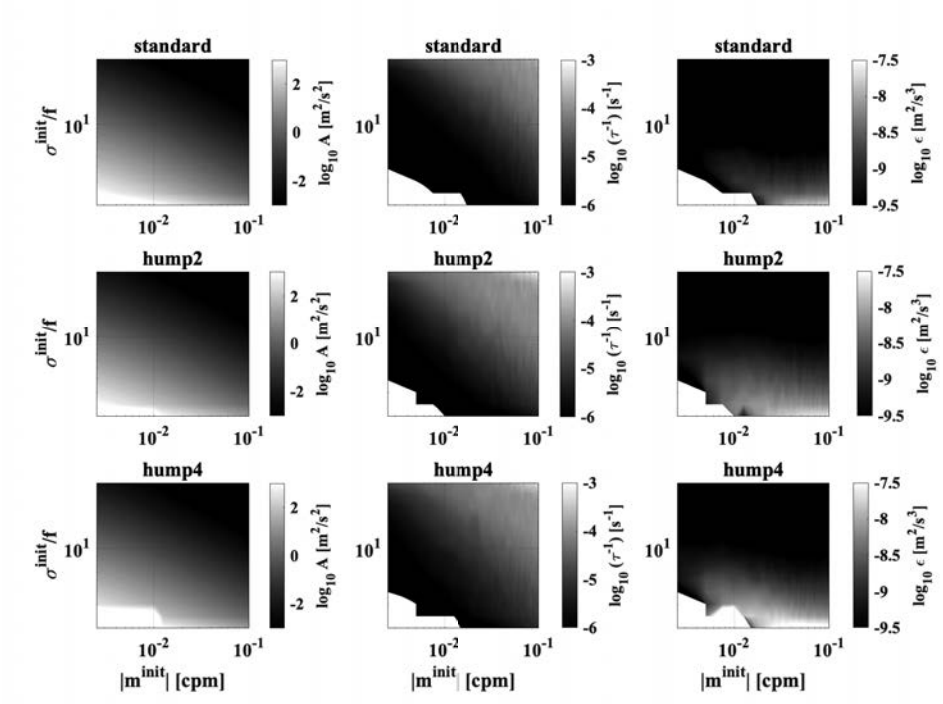


図３： standard，hump2，hump4 実験における内部波パケットの砕波の比較

内部波パケットの波活動量 A（左列），砕波に要した時間 τ の逆数（中列），砕波に伴う乱流運動エネルギー散逸率 ε の分布を，内部波パケットが初期に持つ鉛直波数 m^{init} および周波数 σ^{init} 平面上に示している。

乱流パラメタリゼーションは，GM モデルのような平坦な形状の鉛直波数スペクトルに基づいて定式化されているため，観測機器のノイズの影響を受けにくい鉛直低波数帯の情報を用いて ε を推定することが一般的である。ε に直接的には寄与しない（hump に対応する）低波数帯から推定されたスペクトルレベルは，砕波限界に近い高波数帯のスペクトルレベルよりも大きくなるため，乱流パラメタリゼーションは真の ε を過大評価してしまうと考えられる。

各実験で与えた鉛直波数スペクトルに乱流パラメタリゼーションを適用することで推定した乱流運動エネルギー散逸率 ε_{fine} と式(10)から算出した ε を比較したところ，hump実験では $\varepsilon_{fine} > \varepsilon$ となることが確認された（図４）。さらに乱流パラメタリゼーションにおける過大評価傾向の程度は，スペクトルのエネルギーレベル E_{IW} が大きく，シアーとストレインの比 R_ω が小さいときに顕著となり，このような内部波パラメータに対する依存性は乱流直接観測[8,9]の結果とも整合的であった。

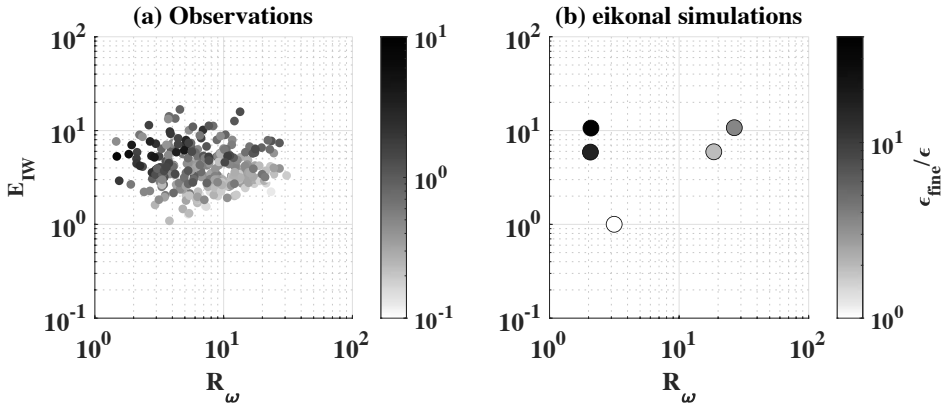


図４：(a)観測データと(b)eikonal simulationにおける乱流パラメタリゼーションの過大評価傾向と内部波パラメータ(E_{IW}, R_ω)の関係

ドットの色は(a)乱流直接観測または(b)eikonal simulationにおいて計算された乱流運動エネルギー散逸率 ε と乱流パラメタリゼーションによる推定値 ε_{fine} の比($\varepsilon_{fine}/\varepsilon$)を示している。

５．まとめ

本研究では，３次元のray-tracing simulationを用いて，低波数側に歪んだ鉛直波数スペクトルを持つ内部波場における乱流スケールへのエネルギーカスケード過程の計算を行い，南極周極流域で実施した現場観測の結果[8,9]を再現することに成功した。これまで，南極周極流域における乱流混合過程は，近慣性内部波や内部風下波の砕波によって生じると考えられてきた[6,7]。これに対して本研究で行ったray-tracing simulationの結果は，実際に観測で捉えられているような鉛直低波数（鉛直波長~100 m）の内部波パケットは「背景シアー」として鉛直高波数の内部波の砕波を促進するが，自ら砕波することはないため，乱流混合への寄与は間接的であることを示した。本研究の結果をまとめたTakahashi et al.(in revision)[13]では，既存の乱流パラメタリゼーションの改良に向けた議論も行っている。

謝　辞

本研究は東京大学情報基盤センターの若手・女性利用者推薦制度（インターン）平成30年度採択課題として行われた（採択時には筆者は東京大学大学院に在学していた）。数値計算には東京大学情報基盤センターのOakforest-PACSを使用した。また日比谷紀之教授（東京大学理学系研究科地球惑星科学専攻）およびAlberto Naveira Garabato教授（National Oceanography Centre, University of Southampton）には研究を進める上で多くの助言をいただいた。

参 考 文 献

1. Munk, W., and Wunsch, C. (1998), Abyssal recipes II: energetics of tidal and wind mixing, *Deep Sea Research Part I*, *45(12)*, 1977-2010.
2. Waterhouse, A. F., et al. (2014), Global Patterns of Diapycnal Mixing from Measurements of the Turbulent Dissipation Rate, *Journal of Physical Oceanography*, *44*(7), 1854-1872.
3. Henyey, F. S., Wright, J., and Flatté, S. M. (1986), Energy and action flow through the internal wave field: An eikonal approach, *Journal of Geophysical Research*, 91(C7), 8487– 8495.
4. Polzin, K. L., Toole, J. M., and Schmitt, R. W. (1995), Finescale Parameterizations of Turbulent Dissipation, *Journal of Physical Oceanography*, *25*(3), 306-328.
5. Ijichi, T., and Hibiya, T. (2015), Frequency-Based Correction of Finescale Parameterization of Turbulent Dissipation in the Deep Ocean, *Journal of Atmospheric and Oceanic Technology*, *32*(8), 1526-1535.
6. Waterman, S., Naveira Garabato, A. C., and Polzin, K. L. (2013), Internal Waves and Turbulence in the Antarctic Circumpolar Current, *Journal of Physical Oceanography*, *43*(2), 259-282.
7. Sheen, K. L., et al. (2013), Rates and mechanisms of turbulent dissipation and mixing in the Southern Ocean: Results from the Diapycnal and Isopycnal Mixing Experiment in the Southern Ocean (DIMES), *Journal of Geophysical Research: Oceans*, 118, 2774-2792.
8. Takahashi, A., and Hibiya, T. (2019), Assessment of finescale parameterizations of deep ocean mixing in the presence of geostrophic current shear: Results of microstructure measurements in the Antarctic Circumpolar Current region, *Journal of Geophysical Research: Oceans*, 124, 135–153.
9. Takahashi, A. (2019), Assessment of the finescale parameterizations of deep ocean mixing in the Antarctic Circumpolar Current region based on microstructure measurements and eikonal calculations, Ph.D. thesis, The University of Tokyo.
10. Garrett, C., and Munk, W. (1975), Space-time scales of internal waves: A progress report, *Journal of Geophysical Research*, 80(3), 291– 297.
11. Ijichi, T., and Hibiya, T. (2017), Eikonal Calculations for Energy Transfer in the Deep-Ocean Internal Wave Field near Mixing Hotspots, *Journal of Physical Oceanography*, *47*(1), 199-210.
12. Bretherton, F. P., and Garrett C. (1968), Wavetrains in inhomogeneous moving media, *Proceedings of the Royal Society A,* **302,** 529–554.
13. Takahashi, A., Hibiya, T., and Naveira Garabato, A. C. Influence of the Distortion of Vertical Wavenumber Spectrum on Estimates of Turbulent Mixing using the Finescale Parameterization, submitted to *Journal of Physical Oceanography (in revision).*

SC20 参加報告

星野 哲也

東京大学情報基盤センター

スーパーコンピューティング部門は、2020年11月16日から19日までの間、オンラインイベントとして開催されたSC20（The International Conference for High Performance Computing, Networking, Storage, and Analysis）に参加し、ウェブ上での研究展示を行いました。本稿では、SC20に参加した東京大学情報基盤センター教職員が参加した中から気になったことを記します。

SC20 について

SC20 は、高性能計算(HPC)分野に於いて最大級の国際会議であるとともに、様々な情報技術関連企業や研究所・大学等の技術展示会としても知られています。今回の開催地は、ジョージア州アトランタを予定しておりましたが、新型コロナウィルスの感染拡大状況を鑑み、オンラインイベントとしての開催となりました。SC の目玉イベントでもある、大ブースでの技術展示が見られず、寂しい印象がありました。また時差の都合上、日本からのリアルタイムでの参加はなかなか難しいものでしたが、発表の様子がムービー形式で後から視聴できるよう配慮されていたため、普段の SC なら並列進行故に見ることのできない同時間帯の発表も見ることができました。

各種ランキングについて

毎年のSCではスーパーコンピュータの性能に関する様々なランキングが更新されます。最も有名なスパコンランキングであるTop500、スパコンの電力性能を競うGreen500、実際のアプリケーションに近いと言われるHPCGや、スパコンのIO性能を競うIO-500などがあります。本稿ではTop500から見える全体的な傾向について述べます。

表 1　Top500 の上位 10 のスーパーコンピュータ (http://www.top500.org/ より抜粋、一部編集)

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Rmax / Rpeak	Power (kW)	Rmax / Power
1	Supercomputer Fugaku	7,630,848	442,010.0	537,212.0	0.82	29,899	14.8
2	Summit	2,414,592	148,600.0	200,794.9	0.74	10,096	14.7
3	Sierra	1,572,480	94,640.0	125,712.0	0.75	7,438	12.7
4	Sunway TaihuLight	10,649,600	93,014.6	125,435.9	0.74	15,371	6.05
5	Selene	555,520	63,460.0	79,215.0	0.80	2,646	24.0
6	Tianhe-2A	4,981,760	61,444.5	100,678.7	0.61	18,482	3.32
7	JUWELS Booster Module	449,280	44,120.0	70,980.0	0.62	1,764	25.0
8	HPC5	669,760	35,450.0	51,720.8	0.69	2,252	15.7
9	Frontera	448,448	23,516.0	38,745.9	0.61	N/A	N/A
10	Dammam-7	672,520	22,400.0	55,423.6	0.40	N/A	N/A

Top500 (http://www.top500.org/)は世界のスーパーコンピュータの性能を LINPACK という係数行列が密の連立一次元方程式を解くベンチマークの処理速度によって競うものです。1993 年の開始以来、6 月にヨーロッパで行われる会議である ISC と、本会議 SC にて年 2 回の更新を続けています。

今回のランキングにおいては、6 月のランキングで初登場した、理化学研究所の計算科学研究センターが保有する富岳スーパーコンピュータが引き続きトップとなりました。富岳は A64FX と呼ばれる富士通が設計した ARM アーキテクチャのプロセッサを 158,976 ノード搭載したスーパーコンピュータです。A64FX に関しては日本語の情報が Web 上にあふれているためそちらに任せるとして、我々として着目したいのが表 1 の Rmax/Rpeak です。これは実行効率と呼ばれる指標で、実際に得られた性能を理論性能で割ったものであり、オリジナルの Top500 から勝手に追加した指標です。Rmax/Rpeak の値は高い程良いのですが、この値を高めるのはコア数が増加するほど難しくなります。富岳は 763 万ものコアを搭載し、Top500 の一位に輝く程の大規模なスパコンでありながら、82%の高い実行効率を誇っているという点で驚異的です。これは A64FX 自体の実行効率の高さに加え、Tofu インターコネクトを用いたノード間の高次元接続技術によるところが大きいと考えられます。当センターでも今年 5 月より A64FX・Tofu インターコネクトを用いたスパコン Wisteria/BDEC-01 の運用を開始することが決まっており、今から楽しみです。

Top10 にランクインした他のシステムを見ると、GPU を搭載したシステムが目につきます。Top10 のスパコンでは富岳に次ぐ実行効率(Rmax/Rpeak)を誇る Selena は、NVIDIA 社が所有するスパコンであり、NVIDIA の最新の GPU である A100 GPU を搭載しています。Selena の特徴は高い電力効率であり、23.983 GFlops/watts の電力効率を誇ります。実行効率で劣るもののさらに高い電力効率を誇る JUWELS Booster Module（Forschungszentrum Juelich 独）も A100 GPU を搭載したスパコンです。なお当センターが運用開始予定の Wisteria/BDEC-01 は、A64FX を搭載したシミュレーションノード

群(Odyssey)に加え、A100 GPU を搭載したデータ学習ノード群(Aquarius)も備えております。

本センターが運用するスパコンでは、Oakforest-PACSがTop500の22位となり、Oakbridge-CXが68位となりました。

東京大学情報基盤センターによる展示

東京大学情報基盤センターは昨年同様、「ITC/JCAHPC, The University of Tokyo」の名義によるブース展示を行いました。筑波大学計算科学研究センターと共同で設立した最先端共同HPC基盤施設をJCAHPCと呼び、2016年12月より共同でOakforest-PACSスーパーコンピュータの運用を行なっています。例年は研究事例を紹介するポスター展示や、ブースでのプレゼンテーションが恒例となっておりました、今回はオンラインによる開催であったため、PDFや動画による研究事例の紹介を行いました。

- Supercomputers and Applications: ITC/JCAHPC The University of Tokyo
- mdx: Infrastructure for leveraging data
- h3-Open-BDEC: Innovative Software Platform for Scientific Computing in the Exascale Era by Integrations of (Simulation + Data + Learning)
- SW/HW Optimizations for Next-Generation Supercomputing
- UT-Helper: Support for HPC and DA Utilizing Unused cores
- Design of Parallel BEM Analysis Framework for SIMD Processors
- Fast Surrogate for Approximating Steady Flow Simulations
- Extending molecular dynamics for liquid / soft matter
- Exploration of dark matter sub-halos by using N-body simulations
- Multiplicative Schwartz type Block Multi-color Gauss-Seidel Smoother for Multigrid
- $\mathcal{H}$ACApK library with low-rank structured matrices
- Low Precision Computing in Sparse Linear Solvers
- Accuracy Verification of Sparse Linear Solvers with FP64/FP32 Arithmetic

詳細は以下のリンクからご覧ください https://www.cc.u-tokyo.ac.jp/public/sc20.php
次回、SC21は2021年11月14日から19日の日程でミズーリ州セントルイスにて開催される予定です。

2020 ACM ゴードン・ベル賞 研究紹介

“Pushing the Limit of Molecular Dynamics with Ab Initio Accuracy to 100 Million Atoms with Machine Learning”

芝　隼 人

東京大学情報基盤センター

１．はじめに

ゴードン・ベル賞は、高い並列実行性能を発揮した HPC アプリケーションに対して毎年 Association for Computing Machinery (ACM) より授与される賞である。例年 4 月に会議録論文の投稿締切が設けられており、査読を経てファイナリストとして選ばれた著者が 11 月に開催される Supercomputing Conference (SC) において発表を行い、これに基づき選考が行われる。2020 年度の SC20 は、当初は米国・アトランタで開催予定であったものの、コロナ禍の影響により 11 月 9〜19 日の期間のオンライン開催となったためファイナリストによる研究発表および授賞もオンラインで視聴することになった。今年度のファイナリストには、以下の 6 件が残った（発表順での記載）。２件は富岳、４件は Summit（米・オークリッジ国立研究所）を利用したものである。

1. 320 億グリッド数有限要素法による壁面近傍微細乱流渦の大規模シミュレーションによる数値曳航水槽試験（流体工学、富岳）
2. 機械学習の利用による第一原理分子動力学の 1 億原子数規模への拡張（物性科学、Summit）
3. 最先端 HPC システムによる励起状態 GW 計算の加速（物性科学、Summit）
4. 3.5km メッシュ全地球気象シミュレーションからの 1024 個アンサンブルデータ同化（気象学、富岳）
5. スクエア・キロメートル・アレイ電波望遠鏡のフルスケールデータ処理（天文観測、Summit）
6. 136 ペタフロップスのスケーラブルグラフ解析（データ科学、Summit）

全講演が終了後、上記のうちプリンストン大学のメンバーを中心とするグループによる 2. の論文に対して本賞が与えられることが発表された。本グループの成果を要約すれば、「量子力学に基づく計算を用いたのと同等の高い精度の力計算を用いた分子動力学＝第一原理分子動力学の計算を、深層学習手法の利用によって 10^5 以上のオーダーで加速し、1 億原子規模まで計算可能となった」というものである。ほんの 10 年前だと第一原理分子動力学は 1000 原子もできなかったことを考えると、隔世の感がある結果である。

2020 年ゴードン・ベル賞グループの一人、北京応用物理与計算数学研究所の Han Wang（王涵）氏はシミュレーションモデリングの分担で中心的な役割を果たしていたようである。個人的な話になり恐縮であるが、2012 年夏にオランダ・ライデンのローレンツ・センターで開催された１ヶ月間の小規模ワークショップに一緒に出ていて、オランダで音楽祭めぐりとか一緒にしたくらいには仲良く過ごした思い出がある。もともと氏はソフトマターモデリングや長距離相互作用計算に関わる古典系の手法開発をされていたと思っていたので、ずいぶんと毛色の違うことで大きな貢献をされているのを思いがけず見て（講演・質疑応答はグループの別の方々が分担された）大

変驚いた。こういう経緯で、今回の授賞対象の研究内容に私も興味を持ち少し調べてみたので、以下に簡単に紹介したい。なお、論文上の表面的な内容しか追跡しておらず不正確な記述があるかもしれないことを、予めご了承いただきたい。

2020・2021 年度のゴードン・ベル賞には、地球規模の危機としての COVID-19 問題の解決に向けた HPC 利用に対する特別賞が設置され、４件の論文（全て Summit を利用）がファイナリストとなって本賞とは別に選考が実施されたことを合わせて付記する。

２．第一原理分子動力学

分子動力学法は多数の原子・分子からなる体系に対して、原子・分子の従う運動方程式を数値的に解くことにより熱力学的・機械的・動力学的性質を明らかにできるシミュレーション手法である。分子動力学法による現象再現の正確性の鍵となるのは、分子間相互作用を記述する力場の精度である。分子間の相対位置の関数として定まっている古典力場の計算はニュートン方程式の単純な実装であり、高速に解くことができるが精度には限りがある。高精度の記述を行うには、量子力学に基づく手法（バンド計算）から定めた有限温度の電子状態密度に基づき各原子分子に加わる力を逐次計算し、時間発展をシミュレーションするべきという一般的な考え方があるが、このような手法は「第一原理分子動力学法」（ab initio molecular dynamics）と総称される。第一原理分子動力学計算に用いる効率的なバンド計算手法のルーツとなっているのは、2020 年ゴードン・ベル賞論文の著者の一人である Roberto Car らによって提案された Car-Parinello 法である。現在では第一原理分子動力学法の手法は当時から大きく変化しているものの、多くは広い意味で Car-Parinello 法に属すると言えるであろう。

第一原理分子動力学計算はヴァリエーションがいくつもあり、その紹介は本稿では控える。VASP, Quantum Espresso, CP2K, OpenMX, RSDFT, CONQUEST などのソフトウェアで、それぞれの特性に応じた第一原理分子動力学の実行が可能と思われる。2020 年ゴードン・ベル賞論文中には明確な記載がないが、Quantum Espresso (水) および VASP (銅) における Car-Parinello 分子動力学ルーチンを使用して、力場の訓練データセットの生成が行われたと思われる。

３．深層ポテンシャル分子動力学 (DPMD)

今回のゴードン・ベル賞論文 (Jia, 2020) に先行して、同じく著者である Han Wang, Weinan E らは、2018 年 4 月に深層ポテンシャル分子動力学（deep potential molecular dynamics, DPMD）の手法をアメリカ物理学会の専門誌 Physical Review Letters に新たに発表した (Zhang, 2018)。DPMD は、原子配置とエネルギーの対応関係を深層ニューラルネットワークによって学習し、実際のシミュレーション計算では学習結果を利用することで分子動力学シミュレーションを飛躍的に加速する手法である。

DPMD は、Embedded Atom Method (EAM)という経験論的な多体力ポテシャルに近い考え方で力場ポテンシャルを構成しており、原子 i のエネルギー E_i をカットオフ径内に位置する近傍原子の状態から決める。すなわち、近傍粒子 j の位置を原子 i の位置を表現する関数$\{\mathbf{D}_{ij}\}$として

(A) $\mathbf{D}_{ij} = \{1/R_{ij}\}$ （原子間距離の逆数）

(B) $\mathbf{D}_{ij} = \{1/R_{ij}, x_{ij}/R_{ij}, y_{ij}/R_{ij}, z_{ij}/R_{ij}\}$ （角度を含む情報）

のいずれかを考慮、この関数$\{\mathbf{D}_{ij}\}$を入力層、原子 i のエネルギー E_i を出力層とする深層ニューラルネットワークを構築する。入力層・出力層間には隠れ層が非線形変換 $\varphi(x) = \tanh x$ を介し

$$d_k^{\mathrm{out}} = \varphi\left(\sum_l w_{kl} d_l^{\mathrm{in}} + b_k\right) \quad (1)$$

という形で入るが、論文中では隠れ層の数は数層でもかなりの精度が確保できるとされている。コスト関数としては、粒子あたりエネルギー・粒子 i への力・ビリアルテンソルにおける訓練データと DPMD の結果との間の差を考慮して

$$L(p_\epsilon, p_f, p_\xi) = p_\epsilon \Delta\epsilon^2 + \frac{p_f}{3N}\sum_i \left|\Delta \boldsymbol{F}_i\right|^2 + \frac{p_\xi}{9}\left\|\Delta\xi\right\|^2 \quad (2)$$

を用い、それぞれの係数p_ϵ, p_f, p_ξを訓練時に漸次調整しながら Adam 法による隠れ層の最適化を行っている。学習元となるデータは第一原理分子動力学計算から与えられる。

DPMD においては、分子動力学実行フェーズでは計算量が問題サイズに対して線形にスケールする。力場を小規模の第一原理分子動力学からアレイジョブ的に大量に訓練したあとに行う大規模系の並列分子動力学シミュレーションの実装そのものは技術的に難しくない。今回論文では並列分子シミュレータ LAMMPS の改良コードが使用されたように見受けられる。ただし、局所範囲内の短距離多体相互作用のみが考慮され、クーロン長距離相互作用はカットオフ距離で切断された取り扱いとなっている。クーロン力の正当な取り扱いこそが第一原理分子動力学法の要諦であるという観点からは、依然大きな課題が残っていると言えるだろう。

４．高速化技術

2020 年ゴードン・ベル賞論文 (Jia, 2020) においては、Summit スーパーコンピューター 最大 4,560 ノードを用いた DPMD の計算が報告された。物理系として水および銅の計算が実施されている。水の分子科学・材料科学における重要性は言うまでもないが、銅についてはその経験論的ポテンシャルが外力変形時の歪み-応力関係を比較的よく再現することから、ベンチマーク計算によく使用されるという背景がある。

論文 (Jia, 2020) 中では、主として次の技術改良が報告されている。

A) 計算粒度向上と隣接リストの工夫

DPMD のモデルにおいて最も計算負荷が大きいのは、原子間距離の距離など局所環境の情報を深層ニューラルネットに埋め込む行列 “environmental matrix”の構成である。一方、局所環境は隣接粒子 j に関わる情報から決まるため、その情報を保持する隣接リストのデータ構造と類似した性質を持つ。本論文では、隣接リストの順序を工夫し、原子種ごとに整列した上で距離順にソートをかけて構築することによって、条件分岐を避けて隣接リストから environmental matrix を構成できるように工夫をしている。

また隣接リストは（A）粒子種（B）距離（C）粒子番号、の 3 つの情報を保持するものとなっているが、これらを 64 ビットで表現するデータ構造の記述改良も合わせて行われている。

B) ニューラルネットワーク実装の改良

Environmental matrix が構成できてしまえば、深層学習の計算そのものについて実行すること自体は TensorFlow の標準機能において可能であるが、今回のモデルでは限界性能に近い速度が発揮されない。理由として、隠れ層の計算においては $x \cdot W + b$ の形の行列計算が多数行われ律速となるが、DPMD のモデルの性質上、行列 x が場合によっては 100 万行というようなサイズに及ぶ縦長のものだからである（行列Wのサイズは非常に小さい）。このため通常の深層学習と比較すると、行列行列積（MUTMUL）よりも最適化の不十分な和演算(SUM)の

方に大幅に計算負荷がかかる。これらの演算を CUBLAS GEMM でまとめて処理することができるように TensorFlow の書き換えが実施されている。

また、隠れ層間の順伝搬に際しては活性化関数である TANH の演算が繰り返し行われるが、力を計算する際にエネルギーを変位で微分をとるため出力に対する微分関数 TANHGrad も必要となる。両者を一括して順伝搬の際に同時計算するべく対応する CUDA カーネルを追加することによって、計算の高速化が実現されている。

C) 混合精度演算の実装

DPMD はポテンシャルの微分計算の負荷のかかる部分を深層学習によって置き換えるため、混合精度演算による演算を実装しやすくなる。今回の計算では、２種類の混合精度演算 MIX-32, MIX-16 について十分な安定性および精度が確保できることが確認されている。MIX-32 は隠れ層の自由パラメターを単精度（32-bit）で実装しており、MIX-16 は自由パラメターの大半を半精度(16-bit)とするが、3 層目以降の fitting や TANH は単精度を保持するものである。これにより、倍精度/MIX-32 および MIX-32/MIX-16 の比において、ほとんどの場合 1.7 倍以上という理想的な加速を得ることに成功している。

以上を中心とする技術改良の結果として、Summit 全系 4,560 ノードを使用した計算では、銅 15,925,248 原子に対して倍精度 91.0 PFlops という計算性能を記録した。対理論ピーク性能比で 45% にも及ぶ実行効率である。ストロングスケーリングによるベンチマークについても大変良好であり、数十ナノ秒という長時間のシミュレーションに耐えるものであると言える。第一原理計算の精度でのシミュレーションとしては前人未到の時空規模の計算といって良いであろう。

５．最後に

以上、2020 年ゴードン・ベル賞の授賞対象研究について簡単に紹介した。相互作用の局所性を利用してシミュレーションを加速するという考え方は古典分子動力学計算の延長線上にあるもので、物理モデルそのものには革新的なアイデアがあるわけではないように思う。深層学習に精度の補償を肩代わりさせることで 10^5 倍ものオーダーの計算の加速をさせることに成功しているという実用的観点からみた飛躍、それが可能な問題を見つける目の付け所が瞠目に値するものと思われる。物質科学モデリングと応用数理の双方の研究者の共同研究ならではの成果を達成したグループの実行力に感嘆するとともに、日本国内の高性能計算分野の研究者に対しても示唆的なところが多いのではなかろうかと考え、ここに記事を執筆した次第である。

本稿の最終稿に対してコメントをくださった似鳥啓吾博士（理研 R-CCS）に感謝する。

参　考　文　献

Linfeng Zhang, Jiequn Han, Han Wang, Roberto Car, and Weinan E, “Deep Potential Molecular Dynamics: A Scalable Model with the Accuracy of Quantum Mechanics”, *Physical Review Letters* Vol. 120, pp. 143001/1-6, 2018.

Weile Jia, Han Wang, Mohan Chen, Denghui Lu, Lin Lin, Roberto Car, Weinan E, and Linfeng Zhang, “Pushing the limit of molecular dynamics with *ab initio* accuracy to 100 million atoms with machine learning”, in Proceedings of SC’20: International Conference for High Performance Computing, Networking, Storage, and Analysis. IEEE, November 2020, No. 5, pp. 1-14.

教育利用報告

「実践的シミュレーションソフトウェア開発演習」

高橋英男

東京大学大学院工学系研究科非常勤講師

１．はじめに

本稿では、大学院工学系研究科機械工学専攻向けの演習科目「実践的シミュレーションソフトウェア開発演習」についての、2020 年度の内容について紹介する。本科目は 2009 年より S1・S2 学期（旧夏学期）に実施しているものであり、スパコン向けに並列化された物理シミュレーションプログラムを対象として、ソフトウェア工学の成果や知見を踏まえてシステマチックに開発するスキルを与えることを目的としている。本演習で利用した実行環境は Oakbridge-CX である。

本科目の指導においては、シミュレーションの理論や HPC(High performance computing)の知識については大学教員が、ソフトウェアの開発プロセスや開発ツールについては、企業在籍ないし出身のスタッフが担当している。

シミュレーションの題材としては２つのテーマから受講生が選択する。一つは、分子動力学であり、もう一つは流体力学である。開発は小グループによるプロジェクト形式をとり、やるべきことを漏らさず管理するために、プロジェクト管理サービスの Trello を利用する。プログラムの設計を検討するためにクラス図やシーケンス図などの図を用いて、全体を俯瞰しつつ設計を進める。グループ開発を実現するためにバージョン管理ツールの git と gitlab を使う。

本年度は新型コロナウィルス感染拡大の防止のため、初のリモート講義となった。これに対応するために進め方や教材をアレンジする必要に迫られたので、その点についてまず紹介する。コロナとは独立に行った変更点についても紹介する。

本年度の科目登録者数は、例年より少な目の 8 人であった。受講生の成果としては例年よりもむしろ充実したものとなった。

２．リモート講義に対応した変更点

リモート講義を実現するために、他の多くの科目同様にビデオ会議サービスの Zoom を利用した。本科目の後半はチーム開発になるので、チーム別の作業時間においては Zoom のブレイクアウトルーム機能を活用した。科目の時間外においても、受講生同士で決めた時間で Zoom は利用されており、大きな問題なく利用できた。

Zoom によるリモート講義となったことで、例年使っている演習室の iMac の代わりに、受講生個人の Windows PC や Mac を使うことが必要となった。例年はソースコードの編集と、ある程度のコンパイルとデバッグまでは iMac で進めておいて、並列処理がからむデバッグや性能評価をスパコン上で実施していた。それに対して、今年は学生の手元の環境がまちまちなので、ソースの編集に始まる一通りの作業をスパコンのログインノード上で実施することとした。スパコンに特有の ssh(secure shell)経由のリモート作業は煩雑なので、これは受講生に負担を掛けるかと心配したが、結果的には杞憂であった。開発ツールの IDE(Integrated Development Environment)

のいくつかで、近年リモート開発のサポートが進んでおり、ローカルマシンでの開発に近い感覚でリモート開発ができるようになってきている。便利な IDE の存在は学生の間でも広く知られているようで、講師から紹介した訳ではないのに、全員がマイクロソフト社の無償 IDE である Visual Studio Code と、そのリモート開発用プラグインをインストールして使っていた。

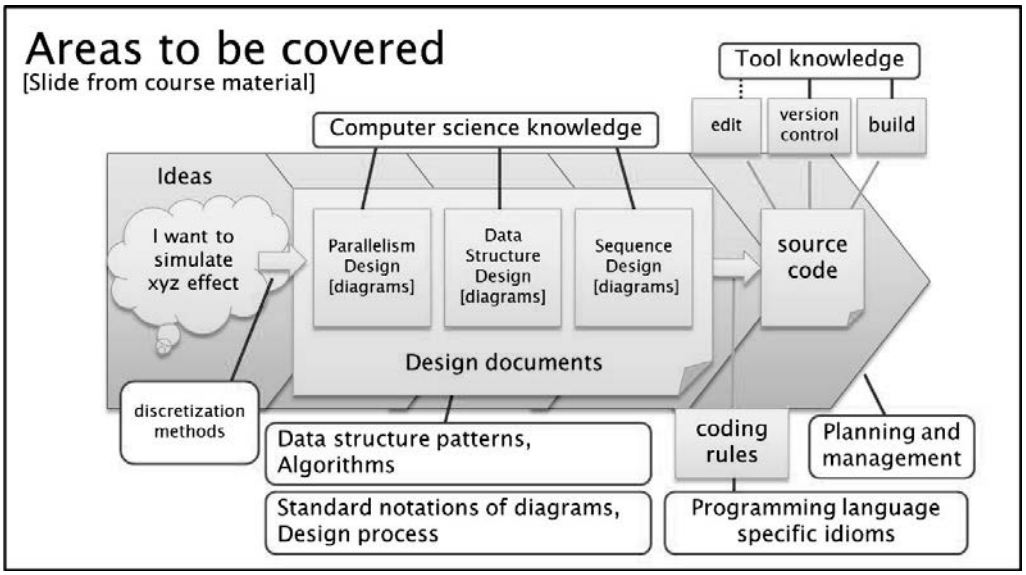


図１ 本科目で扱う領域

本科目ではアイデアから始まってソースコードに至る過程で作成する図表や、プログラム開発のプロセスやツールも教えているため資料が多く、講義時間の占める割合も高くなっている。

３．リモート講義とは独立な変更点

本科目は教えようとしている内容が多く、演習科目としては講義時間の占める割合が高い。図１に、本科目で扱うスキルの領域を示す。この中でも、図の右上の角に示す”Tool knowledge”に関する事項は教材を整備すれば受講生のペースで自習が可能になると以前から考えており、前年度の報告においても課題認識として述べたが[i]、今年度は具体的に以下の取り組みをした。

まず、linux の初歩的なコマンド類についての早見表を web 上で用意した[ii]。これは、情報基盤センターが教員向けに提供されている web サーバに本科目用のディレクトリを作成いただいて掲載したものである。本資料はすぐに必要となるコマンドの使用例を示し、疑問が湧いたら少し踏み込んだことも調べられるコンテンツとなることを目指している。

次に、UI が複雑で早見表を簡単には書けない２つのツール向けに、動画によるチュートリアルを作成した。２つのツールとは、先ほどの Visual Studio Code と、作図ツールであるチェンジビジョン社の astah UML である。動画は YouTube の非公開動画[iii]として登録してあり、URL は授業内で紹介した。どちらのツールも、ツール本来の能力が発揮される正しい使い方の説明を心掛けたところ、各々1 時間程度の長い動画となった。同等な内容の説明を図入りのドキュメントとして用意しようとすると、大量のスクリーンショットを撮らねばならず、動画作成よりも多くの時間がかかる。動画作成は資料を用意する側としては負担が少ない方法だと感じた。

早見表と動画のそれぞれの効果については、今年度はまだ評価できていない。来年度はアン

ケートなどで受講生からのフィードバックを得たいと思う。

４．既存コンテンツの活用促進

既存の教材や資料のうちで、受講生が見落としがちになることが分かってきた事項に対して、目を向けてもらう試みを３つほど行った。

一つ目は、OBCX の利用手引書である。本科目は配布資料がとても多いこともあり、手引書を紹介しても受講生にあまり読まれない状況にあった。これを講義の中で度々参照するようにした。例えばジョブスクリプトの内容に関する質問が出た場合に、手引書の目次を開いて該当箇所を辿って、質問された事項を説明している箇所を示して説明するようにした。これにより、受講生が手引書の利用価値をよく認識し、活用につながったと考える。

二つ目は本科目の基礎演習に関するものである。本科目では物理シミュレーションプログラムに取り組む前段階として、大きな行列同士の行列積を求める並列処理プログラムを作成する。並列化の手段として MPI と OpenMP の双方を講義では説明しており、それぞれを用いた並列化を出題するのだが、例年 MPI の方のみ実装して提出する学生がほとんどであった。基礎演習は出題から締め切りまでに数週間の時間が与えられるので、時間不足なのではなく見落とされていると考え、今年は締め切り前の週に出題範囲について強調するアナウンスをした。その結果、ほとんどの学生が MPI と OpenMP 双方による解を提出した。

三つ目は実際に物理シミュレーションに取り組む応用演習に関してのものである。応用演習の最終的な成果物は、小グループごとに開発したシミュレーションプログラムについてのプレゼンである。プレゼンでは開発プロジェクトの評価、プログラムの出来栄えの評価、シミュレーション結果の評価について発表を求める。例年、シミュレーションをなんとか動かすところまでで時間切れになるグループが多い。この状況を改善すべく、今年は評価のための時間が残るように、プログラムを動作させる締め切りを１週間早めることにした。その結果、最終日のプレゼンにおいては、３グループ中２グループがシミュレーション結果の評価まで含んだ発表をした。

５．おわりに

本科目の主要教材は書籍として出版している[iv]こともあり、web サイトでの教材公開には踏み出していなかった。本科目開設当初の暗黙の理解では、linux のシェルやコマンド類などの既存ツールの説明は各自調べてもらうものとしていたが、実際には初歩的な操作で躓く学生が跡を絶たなかった。「早見表みたいなものはないのですか」と要望されたこともあった。linux コマンドの入門書を紹介したこともあるが買われることは稀である。そこでようやく、科目で使う範囲の機能については科目自身でリファレンスを提供するか、外部の書籍や資料を具体的に指定する必要があると考えるに至り、今年は教材 web サイトの開設に踏み切った。作るからには、他の科目や研究現場にも知られ、活用されるものを目指したい。

---

[i] https://www.cc.u-tokyo.ac.jp/public/VOL21/No5/15.lec201909-satof.pdf
[ii] https://lecture.ecc.u-tokyo.ac.jp/~hideo-t/
[iii] URL を直接指定すれば誰でも見られるが、検索やレコメンデーションには出てこない動画
[iv] http://www.utp.or.jp/book/b306662.html http://www.utp.or.jp/book/b307123.html

東京大学大学院工学系研究科「材料量子モデリング入門」

渡邉聡

東京大学大学院工学系研究科マテリアル工学専攻

１．はじめに

「材料量子モデリング入門」（「日韓遠隔講義Ⅷ」）は、ソウル国立大学（以下、「ソウル大」と記す）材料系専攻の HAN Seungwu 教授と私が共同で担当して 2018 年度 S1S2 タームにはじめて開講された。東京大学大学院工学系研究科はソウル大工科大学との間で 2007 年度からテレビ会議システムを利用した遠隔講義を実施していたが、この枠組みの中での従来の遠隔講義は、ソウル大提供科目については東大生はテレビ会議システムを通じて聴講し、逆に東大提供科目の場合はソウル大生がテレビ会議システムを通じて聴講する、というものであった。これに対し、共同で担当し、半分ずつ分担することにより、全部を一人で担当するより教員の負荷を軽くするとともに、海外大学教員による講義という従来の遠隔講義の魅力は残しつつ、半分は目の前で受講することで受講生のモチベーションを高めることを「材料量子モデリング入門」ではねらった。東京大学情報基盤センターの Reedbush-U を利用した実習も含めた形で実施し、その報告は本誌に掲載された[1]。

報告記事にも書いた通り、2018 年度の講義終了時点では隔年開講でこの科目を続けたいということで HAN 教授と意見が一致していた。そして、2019 年秋から具体的な相談を始め、2020 年度 S1S2 タームに 2 回目を実施した。以下にその内容を報告する。

２．講義概要

本講義は、量子力学に基づいて材料の構造や性質を予測する「密度汎関数法」という理論計算手法の概要の理解を目指しており、理解を深めるために計算機シミュレーションの実習も含めている。2018 年度同様、東大側ではマテリアル工学専攻「材料量子モデリング入門」および工学系研究科共通「日韓遠隔講義Ⅷ」の２つの科目名が付与され、2 単位の科目として開講した。両大学の学事歴等の違いを考慮し、4 月～6 月の間に計 8 回、水曜日 15 時～18 時（途中に休憩 15 分）という変則的な日程となった。

講義日程と各回の内容を表 1 に示す。学事暦の違いや両国の祝日のため、残念ながら第 2 回と第 3 回との間がずいぶん長く空いてしまった。なお、ソウル大側では 3 単位の科目として開講しており、表 1 の内容に加えて 3 月に古典分子動力学法という計算法に関する講義と実習を行っている。東大生がこれを履修せず表 1 の部分だけを履修しても問題ないように講義・実習内容を調整するとともに、興味ある東大側受講生は古典分子動力学法部分の講義資料を入手したり、録画を視聴したりできるようにした。なお、私が主に理論の枠組みと研究事例の紹介を、HAN 教授が実習とそのための基礎的な講義を担当したが、2018 年度は私の担当分を先に実施したのに対し、それでは具体的なイメージを把握しにくいとの受講生の声があったことから、今回は順序を逆にして実施してみた。

計画時には自校の受講生に対して対面で講義する予定であったが、新型コロナ禍のため、全

[1] 渡邉聡、スーパーコンピューティングニュース Vol. 20、No. 6、pp. 55-58 (2018)。

表１：講義日程および内容。

回数	日程	担当教員	講義内容
1	4/8	HAN	講義と実習（基礎的な概念の解説、ソフトウェアの使い方）
2	4/22	HAN	講義と実習（エネルギーバンド構造）
3	5/13	HAN	講義と実習（結晶中の欠陥の電子状態）
4	5/20	HAN	実習（結晶中の欠陥の電子状態）
5	5/27	渡邉	講義（序論、量子力学の復習、分子軌道法の概要）
6	6/3	渡邉	講義（密度汎関数法の概要）
7	6/10	渡邉	講義（様々な系のモデリング法、原子ダイナミクスの計算法）
8	6/17	渡邉	講義（前回までの補足、量子モデリングの将来展望）

ての講義・実習をオンラインで実施した。HAN 教授担当分は 2018 年度の講義の録画を配信し、実習課題のみ若干改訂したが、私の担当分は、改訂したい部分が多かったこともあって Zoom で行った。Zoom での講義は録画して Google Drive にアップロードし、受講登録者が後日視聴できるようにした。

なお、講義は実習の説明を含めてすべて英語で行い、講義資料もすべて英語で作成したが、実習に関する質問は日本語でも受け付けた。

３．実習実施内容

2018 年度の実習には Reedbush-U を利用したが、このシステムの利用期間が 2020 年 6 月末までであったのに対し、レポート課題に時間をかけて取り組みたい受講生もいる可能性を考慮して課題提出の最終締め切りを 7 月末に設定したため、今回は Reedbush-H を使用した。なお、ソウル大生も本講義で Reedbush-H を利用可能であることを事前に情報基盤センターに確認し、HAN 教授にその旨連絡したが、使用するソフトウェアのライセンスや実行環境を有していることを履修条件として既にアナウンスしたとのことで、東大生のみ Reedbush-H を利用した。実習は受講生各自が自宅等から Reedbush-H にアクセスして行う形となり、実習に対するサポートはすべてオンラインで行った。具体的には、指定の講義時間中（15 時～18 時）は Zoom セッションをずっと開いておいて chat での質問を受け付けた他、Google Form と ITC-LMS でも質問を受け付けた。多くの質問はティーチングアシスタント 2 名が分担して答えてくれた。

密度汎関数法計算の実習に使用したプログラムは、前回に引き続き世界的に広く使われている VASP（Vienna Ab initio simulation package）[2]である。VASP はライセンス料が有償のプログラムであるが、担当教員がライセンスを所有しており、受講生がプログラムをコピーできないようにして使用するのであれば、開発元に概要を通知して教育目的で使用する場合ライセンス料は不要である。この科目用の Reedbush-H アカウントを東大側担当教員・講義補助者（特任助教およびティーチングアシスタント）・履修者に発行してもらい、特任助教のアカウントのディレクトリに VASP のバイナリーファイルを置き、履修者にはこのファイルの実行権限のみ与える形（すなわち自分のディレクトリへのコピーはできない形）で実習を実施した。

メインのプログラムは VASP であるが、実習において前処理・後処理、ジョブ投入等を行うた

[2] https://www.vasp.at/

め、学生各自のノートPCには以下のプログラムをインストールしてもらうようにした。

（１）原子配置等の可視化プログラムVESTA[3]

（２）SSHクライアントプログラムPuTTy[4]（Windowsユーザーのみ）

（３）ファイル転送プログラムWinSCP[5]（Windowsユーザーのみ）

2018 年度に実施した際にはソウル大側が作成した前処理・後処理用の Python コードがReedbush-U上でうまく動作しないことが直前に判明して対応に苦労したが、今回は事前に改めてチェックし、ライブラリを１つ用意し直すことでReedbush-H上で動作することを確かめた。また前回はX-windowsシステムを各自のノートPCにインストールすることも推奨したが、今回はX-windowsシステムを各自のPCにインストールせずに実習を進められるようにした。

４．受講者に関する統計等

2018年度は履修登録者10名、単位取得者6名だったが、今回は履修登録者22名、単位取得者8名だった。履修登録者の内訳は工学系研究科マテリアル工学専攻16名（修士15名、博士1名)、同機械工学専攻3名（修士2名、博士1名)、同化学システム工学専攻2名（修士・博士各1名)、同応用化学専攻1名（修士)、単位取得者の内訳はマテリアル工学専攻修士6名、機械工学専攻修士1名、化学システム工学専攻修士1名であった。他に聴講を許可した学部学生2名（工学部マテリアル工学科)、研究員2名（新領域創成科学研究科物質系専攻）が受講した。変則的な日程であることに加えて実習もオンラインで行うことになったため、何名受講してくれるか心配だったが、履修登録者数、単位取得者数ともに2018年度を上回り、ホッとしている。なお、ソウル大側は当初30名程度、最終回も20名が出席しており、この数は2018年度とほぼ同じである。

履修登録者数に比べて単位取得者数が少ないのは、課題に（実際には難しくないのだが）時間がかかりそうと思われたことや修了要件を既に満たしているので聴講だけした学生が相当数いたことが主因と推測しているが、実習のサポートにも改善すべき点があったかもしれないと反省している。Google Form 等を利用したサポートシステムはうまく機能し、質問してきた受講生の実習時のトラブルはすべて解決できたのだが、Reedbush-Hの利用状況から、質問する前にあきらめてしまった受講生が相当数いたと推測されたからである。例えば、SSH を利用したリモートアクセスのための鍵の生成や登録がうまくいかなかった受講生がいたのではないかと思われる。対面形式での実施の場合、このような問題は実習時間中にティーチングアシスタントに気軽に質問して解決できたと思われるが、オンラインでの質問は、入力の手間や解答までのタイムラグ等の点で、対面時に比べると少しハードルが高かったかもしれない。また、実習を始める前にソフトウェアのインストールやSSHの鍵生成・登録等について時間をかけて説明すればよかったのだが、今回はスケジュールの都合もあって説明は短時間で済ませ、あとは受講生に配布資料を読んで進めてもらう形にしてしまい、この点も反省している。

５．終わりに

初めての点が多かった2018年度の講義に比べ、今回はその反省点を踏まえて準備した甲斐あ

---

[3] https://jp-minerals.org/vesta/jp/

[4] https://www.chiark.greenend.org.uk/~sgtatham/putty/index.html

[5] https://winscp.net/eng/docs/lang:jp

って、新型コロナ禍にもかかわらず全体としては前回より良い形で実施できたと思っている。しかし、前節にも述べた通り改善すべき点はあり、一方、新型コロナ禍は本講義に限らずオンライン教育のやり方については大きな進展をもたらした。HAN 教授とは 2 年後にまた開講しようと話をしているが、開講する場合にはこれらを踏まえてさらに改善した形で実施したい。

謝辞

本講義の実施に際しては、HAN Seungwu 教授に多大な協力をいただいた他、東京大学情報基盤センターの職員の皆様、工学系研究科国際工学教育推進機構関口圭子特任専門員（2019 年度中の準備）および杉浦仁美学術支援専門職員（2020 年度）、同研究科マテリアル工学専攻清水康司助教、およびティーチングアシスタントの大野雅央さん（東大）、MANO Poobodin さん（東大）、LEE Kyeongpung さん（ソウル大）、HONG Changho さん（ソウル大）に大変お世話になった。記して感謝の意を表する。

大規模数値計算論

石　原　　卓

岡山大学大学院環境生命科学研究科

１．はじめに

岡山大学環境理工学部環境数理学科[1]では 1 年次に「計算機リテラシー」を履修した後、２年次は「プログラミング言語 A」と「プログラミング言語 B」で Python と C 言語を用いたプログラミングを各々履修する。また、２年後期は「計算解析」で基礎的な数値計算法、３年次には「数値シミュレーション I」と「数値シミュレーション II」で常微分方程式や偏微分方程式の解を数値的に求める方法について計算機実習を併用して学ぶ。その他、統計データ解析系の講義では R を用いたプログラミングやデータ解析も履修する。したがって、計算機実習については結構手厚い教育を受けており、学生もパソコンを活用することについては特に何も躊躇することなく取り組めるようになっている。ただ、その先のさらに進んだ教育がないのが現状であった。

私は 2017 年に岡山大学に着任する前、名古屋大学において 21 世紀 COE プログラム「計算科学フロンティア」にメンバーとして参加させていただいた経験がある。その COE の教育プログラムの目玉の一つが「スパコン実機を用いた演習」であった。COE に関係する研究室には並列計算の技術が学生を通じて導入されていくような感じがあった。この演習は好評で COE 終了後も計算科学連携教育研究センターが名古屋大学情報基盤センターと連携し、「大規模並列数値計算特論」という講義で継続実施された。学生は特に躊躇することなく並列計算できるようになっていったという印象がある。スパコンは敷居が高いが、使ってみるとそれほどでもなく、「スパコンを使ったことがある」とひとこと言えるようになる体験こそが重要であると私は思うようになった。

岡山大学大学院環境生命科学研究科にてどんな講義をしようかと考え、無謀にも「スパコン実習ができる講義」を行ってみようと思って計画したのが「大規模数値計算論」であった。自分自身が応用計算側の研究者であり、並列計算の講義もまともに受けたことがないのに教えるというのはかなり無理があった。また、当初は使用予定であった名古屋大学のスパコンがリプレイスの関係で 2020 年度の前半は使えないことを知って途方に暮れていた。そんな時見つけたのが、東京大学情報基盤センターの利用案内にあったスーパーコンピュータの「教育利用」であった。こうして、とにかく実施することにした「大規模数値計算論」の実経験について簡単に報告する。

２．大規模数値計算論

本講義の目的は、ただ一つ「『スパコンを使ったことがある』とひとこと言えるようにする」であったので、学期のはじめのガイダンスでもそのように宣伝した。履修者は少なくて良い、自分の研究室の学生が並列計算できるようになれば良いと思っていたところ、11 人と想定以上に多くの履修者がいた。さて、『スパコンを使ったことがある』と言えるためには、確かに「速い！」と実感できることが重要であると思うと、実はこれが大変であった。そう言えば、名古屋大学にいた頃、山本有作先生（現電気通信大学）が並列計算の講義を大学院生向けにされていて、行列・

[1] 2021 年 4 月より岡山大学工学部数理データサイエンスコース

行列積がいくつかの並列計算手法の導入で速くなっていく様子を演習入りで講義されていたが、最終的にはマシンに特化したライブラリを使うのが最も速いという結論であったことを思い出した。また、GRAPE を開発されゴードンベル賞の常連という印象のあった牧野淳一郎先生（現神戸大学）とお話しする機会があった時、N 体シミュレーションで高速計算しやすい主な理由は「データの load/store に比べ演算が多い問題であるから」という趣旨の説明をされていたことも思い出した。その他、私自身の研究では実際に大規模な流体計算を行っており、そもそも領域を分割（並列計算）しなければ解けない問題があることも教えるべきであると思った。また、自分の学生がパラメータスタディを多数の逐次計算で行っていたので、一気に並列計算する手法も早く導入しなければいけないと感じていた。以上が、当初思いついた「題材」であった。

実際の講義は、名古屋大学での「大規模並列数値計算特論」を参考に、スパコンの歴史、並列計算の必要性、スパコン使用環境の構築、スパコンへのログイン、スレッド並列（自動並列と OpenMP）、MPI とハイブリッド計算という順番で実施することにした。新型コロナウィルスの感染防止の関係で講義をオンラインでする必要があった。学生の多くは windows の PC を持っていたので、VirtualBox を install させ、そこにインポートした ubuntu の環境からスパコンにログインして使うことにした。実際のオンライン講義で、ubuntu と zoom の併用はメモリの少ない PC ではうまくいかない場合もあったが、zoom をスマートフォンに切り替えてもらって対処した。

スレッド並列（自動並列と OpenMP）では N 体シミュレーションのプログラムを例題に用いた。自動並列と OpenMP の速度の差が出る理由についてはうまく説明できなかったが、確かにスレッド並列計算による高速化を体験できた。学期の前半で Oakforest-PACS を用いた並列計算の講習会があったので参加した。Batch job の shell の書き方についての説明が勉強になった。1 node の中のコアとメモリの使い方について質問してみると何が最良かは試してみないとわからないとのことであった。この情報は実際に有益であった。MPI では、既存のプログラムを少し書き換えて、パラメータスタディが並列でできるようなプログラムをまず扱った。次に、MPI の文献を参考にして、偏微分方程式を差分法で数値積分する問題において領域を分割して並列計算する実習を行なった。扱ったのは小さい問題サイズであったが、領域を分割して通信しながら解くとは「こういうこと」という体験は学生にとって新鮮だったようである。

３．まとめと感想

2020 年度の前期、東京大学情報基盤センターのホームページで見つけた「スーパーコンピュータの教育利用」を活用して、岡山大学大学院環境生命科学研究科の修士課程の学生対象に「大規模数値計算論」という講義でスパコン実機を用いた演習を行ってみた。新型コロナウィルスの感染防止の関係で、講義も途中に受講した講習会もオンラインであった。オンライン講義やオンライン講習会が普通に行われるようになってくると、身近にスパコンがないことは教育や学習において全くハンディにはならないと実感するようになった。私の講義の評判がどうであったかはさておき、少なくとも現在、研究室の学生が並列計算を普通に実行できるようになっていることを考えると「スパコン活用のカルチャー」の導入には成功したのではないかと思えた。今後も講習会や教育目的のためのスパコン利用サービスは活用していきたいと考えている。

参　考　文　献

青木幸也　並列プログラミング虎の巻 MPI 版 平成二十八年八月一日版

教育活動報告：計算科学プログラミングI

（大学院情報理工学系研究科コンピュータ科学専攻開講科目）

松 本 正 晴

東京大学大学院情報理工学系研究科コンピュータ科学専攻

本稿では，2020 年度 S セメスターに実施した，計算科学プログラミング I（大学院情報理工学系研究科コンピュータ科学専攻, 金曜 1 限@オンライン講義）について紹介する。東京大学では，計算物理学などの計算科学・工学から情報科学まで様々な学問領域の英知を結集した学際的研究教育プログラム「計算科学アライアンス[1]」が 2016 年 4 月より開始されているが，本講義は，計算科学アライアンスの認定講義として 2017 年度より開講されているものである。

計算科学とは，科学技術上の様々な問題に対して数値的なモデル化を施し，計算機シミュレーションによる評価を行う学問分野であるが，本講義は，座学とスーパーコンピュータ（Oakbridge-CX）による実習を通じて計算科学についての実践的な知識やプログラミング技術を身につけることを目的とする。具体的な学習内容は，計算科学を行う上で必要となる数理物理学的な知識として，物理現象の偏微分方程式による記述とその特性や種類，偏微分方程式の数値解法の一種である差分法を基にした支配方程式の離散化手法とその計算精度, 計算スキームアルゴリズムと数値安定性，などの他，マルチコア CPU を搭載する計算機上で並列計算を行うために必要となる知識として，計算機アーキテクチャの特性，並列計算のための MPI/OpenMP プログラミング手法, 逐次計算/並列計算の高速化, 数値計算ライブラリの利用法, などに関する座学・実習である。

昨年度まで本講義は情報基盤センターの大演習室において対面で行われており, 実習に用いるローカル端末として教育用計算機システム（ECCS）を利用させていただいていたが，今年度は新型コロナウイルスの影響により対面講義が禁止となったことにより，Zoom を用いたオンライン講義となり, 実習には個人所有の PC を用いることとした。講義で扱うプログラミング言語は，ハイパフォーマンスコンピューティング分野でよく用いられる C 言語と Fortran の 2 種類とし，それぞれの言語に対応する教材を用意し, 履修者がどちらか一方を選んで実習を行うこととした。また，本講義は大学院向けであることから，履修する上で，UNIX の基本的な知識（ファイル構造，コマンド，ログイン方法，エディタの操作など）と経験，C 言語，または Fortran の基本的な知識と経験があること，を前提条件とした。

表 1 に講義日程と内容を示す。5 回目の授業から Oakbridge-CX の利用を開始し，8 回目までは差分法や並列計算に関する基礎的な内容となっている。9 回目以降は，実践的な科学技術解析用アプリケーションの開発へ向けて，電磁波解析や流体解析など，具体的な物理計算のための計算アルゴリズムの学習とプログラム実装等を行う，という内容であった。一方で，Oakbridge-CX を含む情報基盤センターのスーパーコンピュータは各月の最終週の金曜日に月末処理のためのメンテナンスを行うことが多く，本講義も毎週金曜 1 限に開講していた関係で，第 11 回はスーパーコンピュータが利用できない座学のみの回となってしまい, 講義の組み立てに多少の苦労もあ

[1] https://www.compsci-alliance.jp/

ったが，基本的には講義は毎回，座学に 30 分～1 時間程度の時間を割き，残りの時間を実習時間とすることが多かった。履修登録者数は 2017 年度が 34 名，2018 年度は 37 名，2019 年度は 75 名と年々増加傾向にあり，今年度は 80 名となった。内訳は，本講義の開講所属である大学院情報理工学系研究科コンピュータ科学専攻の学生が 15 名，情報理工学系研究科の他専攻から 27 名，工学系研究科から 26 名，理学系研究科から 6 名，その他研究科から 6 名と，他専攻・研究科のさまざまな学生が履修している点が本講義の特徴の一つと言える。しかし，実際に講義に出席して単位を取得したのは 69 名であった。

一般に，学生がスーパーコンピュータを自由に利用できるという機会は限られているのが普通であるが，本講義で（情報基盤センターのスーパーコンピュータの教育利用を通じて），学生に並列プログラミングの基礎知識から，実践的な科学技術アプリケーションへ向けた並列プログラムの実装までを（計算資源は限られているが，）経験させたことは，非常に貴重な体験で有意義なものであったと期待すると同時に，今後，本講義を履修した学生が，自分の研究テーマでスーパーコンピュータを利用し，研究を進展させていくことを願っている。

表 1　講義日程・内容（2020 年度 S セメスター@オンライン講義）

回数	日付	時間	内容
1	04 月 03 日（金）	08:30 – 10:15	**ガイダンス** 講義の進め方や Zoom の使い方等に関する説明
2	04 月 10 日（金）	08:30 – 10:15	**イントロダクション** 計算科学とは？スーパーコンピュータとは？
3	04 月 17 日（金）	08:30 – 10:15	**並列プログラミングの基礎** アムダールの法則，台数効果
4	04 月 24 日（金）	08:30 – 10:15	**MPI の基礎** プロセス並列，集団通信，1 対 1 通信
5	05 月 01 日（金）	08:30 – 10:15	**スーパーコンピュータ（Oakbridge-CX）の利用** アカウント発行，ログイン方法，ジョブ投入方法
6	05 月 08 日（金）	08:30 – 10:15	**スーパーコンピュータ（Oakbridge-CX）の利用** （第 5 回に引き続き）
7	05 月 15 日（金）	08:30 – 10:15	**Poisson 方程式の差分解法と並列化** 偏微分方程式の種類や特徴，領域分割
8	06 月 05 日（金）	08:30 – 10:15	**OpenMP 入門**（座学のみ） スレッド並列化，ハイブリッド並列化
9	06 月 12 日（金）	08:30 – 10:15	**FDTD 法による電磁波解析**
10	06 月 19 日（金）	08:30 – 10:15	**非圧縮性流体（MAC 法系）の数値解析**
11	06 月 26 日（金）	08:30 – 10:15	**高性能プログラミングの基礎**（座学のみ） SIMD，メモリキャッシュ
12	07 月 03 日（金）	08:30 – 10:15	**差分法における一般座標変換** メトリックス，ヤコビアン，格子生成法
13	07 月 10 日（金）	08:30 – 10:15	**粒子法（SPH 法，MPS 法）による流体解析**

教育活動報告：計算科学概論

（工学部物理工学科・理学部物理学科共通開講科目）

松 本 正 晴

東京大学大学院情報理工学系研究科コンピュータ科学専攻

大 久 保 毅

東京大学大学院理学系研究科物理学専攻

本稿では，2020 年度 S セメスターに実施した，計算科学概論（工学部物理工学科・理学部物理学科 4 年生対象，月曜 3 限@オンライン講義）について紹介する。東京大学では，計算物理学などの計算科学・工学から情報科学まで様々な学問領域の英知を結集した学際的研究教育プログラム「計算科学アライアンス[1]」が 2016 年 4 月より開始されているが，本講義は，計算科学アライアンスの認定講義として 2017 年度より開講されているものである。

本講義では，計算科学の様々な分野で行なわれている研究やシミュレーション手法の概要を複数の教員によるオムニバス形式で紹介し，並行して計算機を使った実習を行うことで，計算科学の現状を俯瞰し，最先端の研究とその手法についての知識を得ることを目的としている。表 1 に講義日程と各回の担当者，内容を示す。計算科学の幅広い分野に加えて，計算機科学分野も扱い，合計 7 テーマのオムニバス形式で，各テーマについて，基本的に座学 1 回と実習 1 回の計 2 回を基本とした講義を行った。1 つ目のテーマ（第 1 回）だけは，その内容から実習を行うことが難しいため，座学 1 回のみで実習は無く，また講義と各教員のスケジュールを勘案した結果，山地先生と奥田先生にお願いしたテーマでは連続した日程ではなくなってしまった。

昨年度まで本講義では，座学パートは工学部 6 号館セミナー室 A，実習パートは情報基盤センターの大演習室においてそれぞれ対面で行われ，実習に用いるローカル端末として教育用計算機システム（ECCS）を利用させていただいていたが，今年度は新型コロナウイルスの影響により対面講義が禁止となったことにより，全て Zoom を用いたオンライン講義となり，実習には個人所有の PC を用いることとした。また本講義では，計算科学の現状を俯瞰し，最先端の研究とその手法についての知識を得ることを目的とする以上，スーパーコンピュータの利用は欠かせないであろうという判断の下，スーパーコンピュータ（Oakbridge-CX）を教育利用させて頂いていた。これは，第 3 回目の講義から利用を開始することを前提に実習内容を組んでおり，ほとんどの学生にとってスーパーコンピュータの利用自体が始めてであったので，それを踏まえて第 1 回に近年の高性能計算機のアーキテクチャについて，第 2 回にスーパーコンピュータの特性や，並列プログラミングの基礎的な内容について講義し，第 3 回は Oakbridge-CX の基本的な利用方法や，ソースコードのコンパイル・実行方法，MPI プログラミングに関する基礎について，ハンズオンによる実習を行った。第 3 回以降は，ECCS 端末と Oakbridge-CX の併用を前提に，第 4，8 回には量子多体問題とオープンソースアプリケーション HΦの利用，第 5，9 回には，並列化構造解

[1] http://www.compsci-alliance.jp/

析アプリケーションの利用, 第 6, 7 回では, 高性能プログラミングの実践ということで，intrinsics 関数による演算の SIMD 化や，OpenMP によるマルチコア並列化に関する実習，第 10，11 回には有限要素解析のための大規模疎行列ソルバー，そして第 12，13 回は，格子スピン模型の計算科学に関する講義と実習を行った。

講義の成績は，各テーマの担当者の先生方に，講義・実習内容に関連したレポート課題を設定して頂き，学生はそれら合計 7 つの課題のうちから 3 つを選んで提出することで評価を行った。また，4 つ以上の課題を提出した場合はその内容に応じて加点することとした。毎回の講義参加者数は約 15 名程度であったが，3 つ以上の課題を提出し，最終的に単位を取得した者は 6 名のみであった。今年度は新型コロナウイルスの影響により対面講義が禁止となり，オンライン講義への対応のために講義開始日が 4 月 20 日までずれ込んだことや， 各教員のスケジュール調整が難しく，テーマの順番について課題を残す結果となってしまった。テーマの順番や，各テーマの繋がりを意識するなど，より体系的な構成を考慮していく必要がある。受講者のうち，単位取得者が少なかったのは, このテーマの順番が受講者の興味を削いでしまった原因の一部になった可能性もあり，今後は with コロナを見据えて，オンライン講義を前提とした講義構成を考えていく必要がある。

本講義を通じて，計算科学・計算機科学の最先端の研究に関する知識を得るだけでなく，実際にアプリケーションを動かしたり，プログラミングをしたりといった，計算科学の手法に関する実践的な実習ができたことが学生にとって貴重な経験であったとすれば，大変うれしく思う。

表 1　講義日程・担当者・内容

回数	日付	担当者	内容
1	04 月 20 日（月）	中村 宏 [1]	高性能計算機のアーキテクチャ
2	04 月 27 日（月）	松本 正晴	スーパーコンピュータと並列プログラミング
3	05 月 11 日（月）		Oakbridge-CX の利用，MPI プログラミング
4	05 月 18 日（月）	山地 洋平 [2]	大規模疎行列固有値問題と量子多体問題
5	05 月 25 日（月）	奥田 洋司 [3]	連続体の並列有限要素法解析入門
6	06 月 08 日（月）	田浦健次朗 [4]	高性能プログラミングと性能測定
7	06 月 15 日（月）		SIMD プログラミング，OpenMP 並列化
8	06 月 22 日（月）	山地 洋平	オープンソースアプリ HΦ による実習
9	06 月 29 日（月）	奥田 洋司	構造解析アプリケーションによる CAE 実践
10	07 月 06 日（月）	市村 強 [5]	大規模疎行列ソルバー入門
11	07 月 13 日（月）		有限要素解析の高速化
12	07 月 14 日（火）	大久保 毅	格子スピン模型の計算科学
13	07 月 15 日（水）		モンテカルロ法，テンソルネットワーク法

1. 大学院情報理工学系研究科システム情報学専攻
2. 大学院工学系研究科物理工学専攻
3. 大学院新領域創成科学研究科人間環境学専攻
4. 大学院情報理工学系研究科電子情報学専攻
5. 地震研究所巨大地震津波災害予測研究センター

國家理論中心數學組『高效能計算』短期課程（2020 NCTS Summer Course）
Advanced Course on Multi-Threaded Parallel Programming using OpenMP/OpenACC for Multicore/Manycore Systems

中島研吾

東京大学情報基盤センター

本稿は，「國家理論中心數學組『高效能計算』短期課程（2020 NCTS Summer Course）」の一環として，2020 年 8 月 22 日（土）・29 日（土），9 月 5 日（土）にオンラインで開催された，集中講義「Advanced Course on Multi-Threaded Parallel Programming using OpenMP/OpenACC for Multicore/Manycore Systems[1]」（共催：國家理論科學研究中心（National Center for Theoretical Sciences，NCTS）[2]，東京大学情報基盤センター他）について紹介したものである。

國家理論科學研究中心（NCTS）は 1997 年に台湾の National Science Council （NSC，行政院国家科学委員会）によって設立された横断的研究組織で，物理・数学の 2 部門がある。数学部門（數學組）の本部は 2015 年から國立臺灣大學（台北）に置かれている。8 つの重点分野があり，国際交流も盛んに行っているほか，会議，チュートリアル等様々なイベントを主催，サポートしている。中核となっている研究者は，国立台湾大学，国立清華大学，国立中央大学，中央研究院（Academia Sinica）等の台湾におけるトップクラスの大学・研究機関を本務としており，参加している学生もこれらの大学の学部生，大学院生である。当センターと NCTS は 2018 年 1 月に研究交流協定覚書（Memorandum of Understanding，MOU）を締結している。本集中講義は例年 7 月に 4 日間で國立臺灣大學（National Taiwan University, NTU）[3]で開催してきたが，新型コロナウイルス感染症対策のため，3 週間に分けて，オンラインで実施したものである。受講者は，Oakbridge-CX（OBCX）を使用したプログラミング実習もオンラインで受講することができる。また，講義はビデオ録画されるため，オンデマンドの受講も可能である。

今回は合計 19 名の受講申込があった。実施した講義内容と資料は東大側で準備したホームページ[4]で見ることができる。本講義の内容は，2020 年夏学期に情報理工学系研究科，工学系研究科の講義として実施した「科学技術計算I，コンピュータ科学特別講義I，スレッド並列コンピューティング」[5]の教材を使用している。2020 年夏学期は既にオンラインで講義を実施したので，その際の教材をそのまま利用することができた。普段の講義では Zoom を使用しているが，台湾では公的機関では Zoom を利用できないため，Webex を利用した。

本集中講義実施に当たって，多大なご協力を頂いた，王偉仲教授（國立臺灣大學）と NCTS のスタッフに対してこの場を借りて深甚なる謝意を表したい。

---

1 https://sites.google.com/site/school4scicomp/2020-b-nk
2 http://www.cts.nthu.edu.tw/main.php
3 http://www.ntu.edu.tw/
4 http://nkl.cc.u-tokyo.ac.jp/NTU2020SummerOnline/
5 http://nkl.cc.u-tokyo.ac.jp/20s

大学院工学系研究科電気系工学専攻修士実験

MPI による並列プログラミング入門

下川辺隆史・塙敏博・中島研吾

東京大学情報基盤センター

本稿は 2020 年 S1 タームに実施された大学院工学系研究科電気系工学専攻修士実験「MPI による並列プログラミング入門 」について紹介する。修士実験は同専攻の修士課程（1 年）の学生が 2 人または 3 人が 1 組となって，1 ターム（約二ヶ月）のうちに，各教員が提供する課題に基づき実習を行うもので，原則として受講者の専門とは異なる分野の課題を選択することとなっている。

科学技術シミュレーションにおいて大規模並列システムが広く使用されるようになったが，そのためには，並列計算プログラミングに関する知識と経験が必須である。本実験では，分散並列システムにおいて広く使用されている MPI（Message Passing Interface）による並列プログラミングについて講義，実習を行う。実習では情報基盤センターの Oakforest-PACS スーパーコンピュータを使用する。

本年度は 3 組，6 名の受講者があり，新型コロナウイルス感染症拡大防止のため，すべてオンラインで実施した。下記内容について座学及び演習を行なった：

- Oakforest-PACS ログイン
- MPI 並列プログラミング
- SPMD（Single Program Multiple Data）型パラダイムの習得
- MPI プログラムによる数値積分

教育利用報告：工学院大学３年次講義「並列･分散システム」

藤　井　昭　宏

工学院大学情報学部

教育利用制度により， Oakbridge-CX システムを工学院大学の情報学部 3 年生の講義で利用させていただいた。今年度で本制度を利用させていただけて，7 年目となった。2020 年度の講義と Oakbridge-CX システムを利用した効果について報告する。

まず今年度はコロナ禍により授業の行い方に制約があり，対面では行えないことになった。前半を座学とし，後半を演習としているが，前半の授業は，音声付き資料を配付し，質問を受けられる時間を設定しチャットなどで対応した。後半の演習の授業では，画面キャプチャを多く入れた資料を作成し，学生が迷うことがないように心がけ，簡単なプログラムの実行や作成をさせた。対面授業であれば，よくできる学生が授業内容を理解していない学生をフォローしてくれるのだが，オンラインではそのようなことができず，質問の時間を作っても学生が声を出さない限りこちらからは気づけないので，演習としての適切な運営は難しい状態であった。

授業内容は 例年通り 1 CPU と並列性，2 並列システム，3 分散メモリと共有メモリでのプログラム，4 Oakbridge-CX での演習とした。1，2 が主に並列システムの知識に関するもので，3，4 が並列システム上でのプログラムに関する講義と演習としている。最終課題は昨年度と同様に，オイラー法による重力についての多体問題の逐次コードの OpenMP，MPI を用いた並列化レポートとした。OMP 化では相互の星の作用を計算するループにプラグマをいれるだけであり，MPI 化では，各プロセスに計算を担当する星集合を分割し，担当の星集合のデータを更新したのち全プロセスで allgather をするものである。

今年度は期末試験を設定できなかったため,毎回のオンライン授業に含まれる簡単な課題と,期末のプログラミングを伴うレポート課題を提出させ評価した。オンラインの制約もあり，単位を取得出来なかった学生の割合が例年より増え，20%程度になっていた。今年度の履修人数は 66 人であり,実際に単位を取得したのは 52 人だった。また講義の日程と簡単な内容のリストは表１のようになっている。本講義は並列計算の導入として位置づけており,はじめに基礎知識を講義し，最後の４回の講義時間(表１の 4-1～4-4)のみ Oakbridge-CX を利用してプログラム演習を行った。上に述べたようなレポート課題もだしており,授業時間外にも各自に自習させ,今年度の前期末である，8 月末までアカウントを利用させて頂いた。表 1 でプログラム演習のところのみオンデマンドにしている理由は，本学の演習室の仮想環境に一度ログインしてからスパコンへアクセスするように説明しており，受講生が一斉に本学の演習室の仮想環境にアクセスしないようにするためである。学生対応のコストも抑えるため，説明上でのスパコン接続の環境は本学の仮想環境のみとした。

工学院大学では，卒業研究を含め，研究用に Oakbridge-CX を利用できるようにトークンを購入している。今年度も過去のこの授業を受け,スパコンを使い卒論を進めている学生や研究会発表を行う学生もいた。学部 3 年生のうちから本制度の支援を受け，この環境の認証に関する手続きやジョブ投入の方法，さらに並列システムの基礎知識と合わせて簡単な並列プログラムの実装まで経験させておけたことはこれまでと同様に非常に有意義なものになったと考えている。

表１：講義日程，内容

日付	時間	内容
5 月 12 日	15:35-17:20	1-1　CPU の仕組み プロセス，キャッシュ
5 月 19 日	15:35-17:20	1-2 並列性の分類 {命令，スレッド，プロセス}レベル並列性
5 月 26 日	15:35-17:20	2-1 並列システム 共有メモリや分散メモリでの相互結合網，キャッシュの一貫性，α β モデル，計算と通信のコスト
6 月 2 日	15:35-17:20	2-2 共有メモリ型と分散メモリ型の並列処理 SPMD，共有･分散メモリ，性能計測，アムダールの法則
6 月 9 日	15:35-17:20	3-1 分散メモリ型の並列処理 MPI 基礎，集団通信
6 月 16 日	15:35-17:20	3-2 MPI のプログラム MPI の簡単なプログラム 内積など
6 月 30 日	15:35-17:20	3-3 マルチスレッドと排他制御 mutex_lock，semaphore，デッドロックとその検知
7 月 7 日	15:35-17:20	3-4 OpenMP の書き方と例題 文法と簡単なプログラム例.　False sharing
7 月 14 日	オンデマンド	4-1 並列プログラムの実践 1 計算環境説明，数値積分による円周率の計算(OMP 化)
7 月 21 日	オンデマンド	4-2 並列プログラムの実践 2 数値積分による円周率の計算の MPI 化
7 月 28 日	オンデマンド	4-3 並列プログラムの実践 3 逐次の多体問題のプログラムの OMP 化にむけて (依存関係のない for 文の説明)
8 月 4 日	オンデマンド	4-4 並列プログラムの実践 4 逐次の多体問題のプログラムの MPI 化にむけて (データを全プロセスでコピーして持ち，自分の担当データのみ更新し，allgather で共有するモデル)
8 月 11 日	オンデマンド	レポート課題 ヒント

※オンデマンドの部分は，時間を限定せずに，この日程のどこかの時間で演習を行うように指示した.

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。

2020 クライオ EM 講習会

包 明 久　柳 澤 春 明　吉 川 雅 英

東京大学大学院医学系研究科生体構造学教室

１．目的

クライオ電子顕微鏡による生体分子の構造解析は、電子顕微鏡の高精度な自動制御、単粒子解析法の発展や電子直接検出器の実用化によって大きく進展することになった。これにより従来のX線解析法に頼っていた構造解析が、結晶化の過程が不要となり、分散性の良好な凍結試料を作製することにより、構造解析を行うことが可能になった。

しかしながらクライオ電子顕微鏡の利用においては、試料作製のノウハウや高度な構造解析手法が必要とされるため、一般の研究者に普及させるには、一連の基本的な知識や実習体験が必要と思われる。

そこで、電子顕微鏡の構造、原理や試料作製法、単粒子解析法などの基本的なところから講義と実習を行い、クライオ電子顕微鏡を用いた構造解析手法の概観を把握してもらうことを目的に講習会を開催することにした。参加者には、講習会への参加を機に、それぞれの手法を習熟、発展させることを期待したい。

２．参加者およびプログラム

参加者：12 名

（内訳）　東京大学：4 名　他大学：5 名　公的研究機関：2 名　企業：1 名

プログラム(初日は Zoom による遠隔講義、2 日目以降は実験室およびセミナー室で実施)

11 月 16 日　10:00 電子顕微鏡の構造
　　11:30 試料作製法
　　12:20　昼　食
　　13:30 単粒子解析の基礎
　　14:30 トモグラフィーの原理と応用（15:30 終了）

11 月 17 日　10:00 ネガティブ染色試料のロード、軸合わせ、基本操作
　　12:00 昼　食
　　13:00 画像収集（17:00 終了）

11 月 18 日　10:00 クライオ試料のロード、軸合わせ、基本操作
　　12:00 昼　食
　　13:00 各種試料の観察（17:00 終了）

11 月 19 日　10:00 三次元再構成の基礎(RELION)
　　Tutorial data(apoferritin)
　　12:00 昼　食
　　13:00 単粒子実習（17:30 終了）

３．Reedbush を活用したデータ解析実習

透過型電子顕微鏡の画像として得られるのは、三次元の生体分子の二次元投影像である。単粒子解析では、各粒子の投影像の向きを計算機上で求めることで、大量の投影像から三次元構造を再構成する。この計算をするためのプログラムはいくつか公開されているが、今回は最も広く使われているオープンソースのソフトウェアである RELION(ver3.1)を使用し、チュートリアルデータとして配布されている β-galactosidase の解析処理を行うことで、単粒子構造解析のフローチャートを学んでもらった。

RELION の動作のためには GPU 搭載の計算機を用意することが望ましい。現在では比較的安価(100-200 万円）で購入することが可能であるが、必要な構成を理解したりプログラムをインストールするのが難しいというユーザーも多いであろう。今回の実習では東京大学が提供する Reedbush を使用して、あらかじめこれらの設定を整えた環境を提供することで、各自の事前準備を最低限にして実習を行うことができた。参加者はほぼ全員が RELION に触れたことのない未経験者であり、Linux の使用経験も少ないという方も多かったが、一日の講習で一連の処理をおこない、電顕像から三次元構造が得られる過程を体験してもらえたと思う。今回使用したチュートリアルデータは 20 枚程度の電顕像から 3 Å 程度の分解能の立体構造まで到達できるものであり、1 日程度の実習の中ですべておこなうことが可能である。しかし、実際の構造解析ではこのようなデータが得られることは稀であり、通常は数百から数千枚程度の電顕像を処理することが必要となる。単粒子解析では「このフローチャートに従えばどのような構造でも解析できる」といった解法はなく、得られたデータセットに対して常に各処理の順番や回数、パラメータなどを最適化する必要がある。参加者には今後、今回のチュートリアルデータの処理を基本として、より難しいデータの解析を試行錯誤してみてほしい。

４．利用者へのアンケート結果

12 名中 10 名から回答が得られた。

各講義・実習の難易度と意見

講義・実習	難易度 1:簡単→5:難しい	代表的意見
電子顕微鏡の構造	3.2	非常にわかりやすかった。
試料作製法	2.9	講義で使用する資料が予め手元にあれば良かった。
単粒子構造解析の基礎	3.5	予備知識がなく、理解が難しいところもあった。
トモグラフィーの原理と応用	3.2	発展途上の技術であることが分かった。
試料作製実習	3.0	クライオ電顕サンプルの作製も体験できればよかった。
ネガティブ染色試料観察	3.6	電顕操作の流れは理解できた。
クライオ試料観察	3.8	もっと時間が欲しかった。
データ解析実習	3.7	フローチャートで一連の流れを示してほしかった。

講習会の満足度（10 点満点）：9.5

［クライオ電子顕微鏡 Krios の見学］

［データ解析実習の様子］

第 140 回お試しアカウント付き並列プログラミング講習会「科学技術計算の効率化入門」実施報告

伊田 明弘

スーパーコンピューティングチーム

2020 年 10 月 7 日（水）、第 140 回お試しアカウント付き並列プログラミング講習会「科学技術計算の効率化入門」が、オンラインにて開催されました。

本講習会は、東京大学内および学外における当センターのスーパーコンピュータの利用を考えているユーザに加え、社会貢献の一環として、高性能計算や並列処理の技術習得を目的にした企業に所属する研究者、技術者の方が参加可能になっております。

受講者は、大学院学生(修士)：10 名、大学院学生(博士)：4 名、助教：1 名、講師 1 名、准教授：1 名、教授 1 名、研究機関研究員 3 名、企業の方：4 名の参加者合計：25 名でした。

1 カ月有効なお試しアカウントが与えられ、Oakbridge-CX スーパーコンピュータシステムの利用方法、科学技術計算ライブラリ利用に関する演習、シミュレーションの効率化に関する講習が、終日の日程で行われました。

当日のプログラムを、以下に記します。

10 月 7 日（水）

13:00-13:15	システム紹介
13:15-14:15	スパコンと線形計算ライブラリ(BLAS, LAPACK)
14:15-14:30	休憩＆質問
14:30-15:45	Xcrypt を用いたジョブ並列処理
15:45-16:30	実習＆質問

講習会終了後にアンケートを実施しました。参加者の内 17 名から、講習会に関するアンケートをご提出いただきました。表 1 は質問項目と回答（5 段階評価）の人数分布です。今回は、プログラミング経験が無い方から 46 年の経験豊富な方まで、様々な方々が参加されました。並列化プログラミングについては、17 名中 11 名の方が経験ありと回答されており、その内 9 名は５年以下と回答されています。普段使用するプログラミング言語を訪ねた質問には、Basic, Fortran, C, C++,C#,Python, Java,R と様々な言語が回答として得られました。難易度については、適切という回答が大半を占めました。全体的な満足度としては、概ね高評価でした（17 名中 9 人が 4 以上、平均値は 3.76）。

以下のご意見を頂きました。

■Zoomによるオンライン講習会で良かったこと

・移動がない

・東京から遠方にいる者でも気軽に参加できる

・会場に行かなくてよい分、時間を節約できるため空いている時間を利用して気軽に参加することができる

・チャットで気軽に質問ができる

・we can just join the course at home and no need to spend time in transportation.

■Zoomによるオンライン講習会で悪かったこと

・講師や受講者の顔が見えない

・似たような作業をしている他の組織の方々と知り合いになることができない

・It is a little bit difficult to solve problems when you face it.

■講習会全般に対する意見・要望など

・hoping sensei can give more examples to let students try and ask students that if they can keep with the class or not and slightly slow down the speed of the class.

・前半は比較的、易しかったのですが、後半のXcryptは当方にとってはスピードが速く、ついていけませんでした。そのため、後でじっくり読んで実習をすることにします。

表1　アンケート集計結果

	評点	1	2	3	4	5
(a) 講習会時間	短い⇔長い		1	16		
(b) 講習会講義内容（プレゼン）	簡単⇔難		3	12	2	
(c) 配布資料内容	簡単⇔難		4	11	2	
(d) サンプルプログラム内容	簡単⇔難		1	14	2	
(e) 満足度（平均4.0）	不満⇔満足	1		7	3	6

以上

第141回お試しアカウント付き並列プログラミング講習会「MPI基礎：並列プログラミング入門」実施報告

三木 洋平
東京大学情報基盤センター

2020年10月13日（火）、第141回お試しアカウント付き並列プログラミング講習会「MPI基礎：並列プログラミング入門」が開催されました。例年は東京大学情報基盤センターにおいて開催されている本講習会ですが、今回は新型コロナウイルス感染症対策のためにZoomおよびSlackを用いたオンライン講習会として実施されました。

本講習会は、東京大学内および学外における当センターのスーパーコンピュータの利用を考えているユーザに加え、社会貢献の一環として、高性能計算や並列処理の技術習得を目的にした企業に所属する研究者、技術者の方が参加可能になっております。

受講者の内訳は、大学学部生：1名、大学院学生（修士）：6名、大学院学生（博士）：3名、研究機関研究員：3名、企業：5名、その他：1名であり、合計19名でした。

1ヶ月間有効となるお試しアカウントが与えられ、Oakforest-PACSスーパーコンピュータシステムの利用方法、MPI（Message Passing Interface）を用いたプログラミングに関する基礎演習が、1日終日の日程で行われました。

当日のプログラムを、以下に載せます。

- 10月13日（火）
  - 10：00 - 11：20　テストプログラムの実行など（演習）
  - 11：30 - 12：30　並列プログラミングの基本（座学）
  - 13：40 - 14：40　MPIプログラム実習I(演習)
  - 14：50 - 15：50　MPIプログラミング実習II（演習）
  - 16：00 - 17：00　MPIプログラミング実習III（演習）

対面での講習会として開催してきた本講習会では例年スパコンへのログイン支援を当日行ってきましたが、今回はオンライン開催ということもあり、事前に配布した資料を参考に事前にログインしてもらうこととしました。これは2020年4月30日（木）に開催した第132回講習会と同様の対応です。さらに、この時のアンケート結果をもとにオンライン講習会としての弱点を克服するべく、Slackの導入によって質疑応答をしやすくする、より基礎的な演習課題を増やして参加者間の進捗具合を同期しやすくするといった工夫を施しました。また、座学パートについてはZoom講習会を録画したものを、東京大学情報基盤センタースーパーコンピューティング部門のYouTubeチャンネル（https://www.youtube.com/channel/UC2CHaGp1AO-vqRlV7wmUO-w/videos）にアップロー

ドした（https://www.youtube.com/watch?v=i4ViD7Nk5m0）のでいつでも視聴できるようになっています。

19名中14名の参加者について、講習会に関するアンケートをご提出いただきました。主要な項目の集計結果を以下に掲載します。

プログラミング経験については1年から最長で20年間と幅広く分布していました(図1)が、並列プログラミング入門ということもあり、並列プログラミング経験についてはまったくない方が11名と多数派でした。普段使用しているプログラミング言語については、Pythonが最多でFortranおよびC/C++が続きました（図2)。回を重ねる毎にPythonユーザが増えていっている印象です。

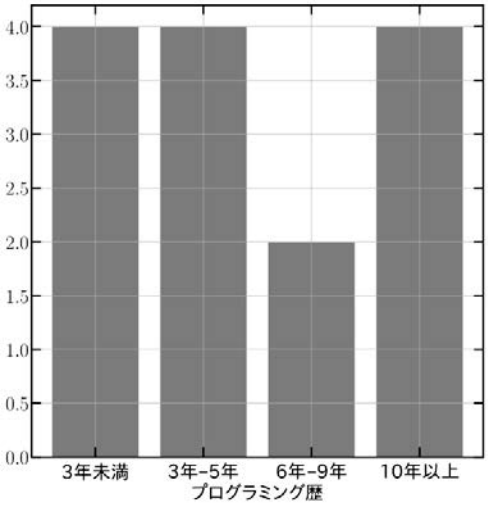


図1：参加者のプログラミング経験

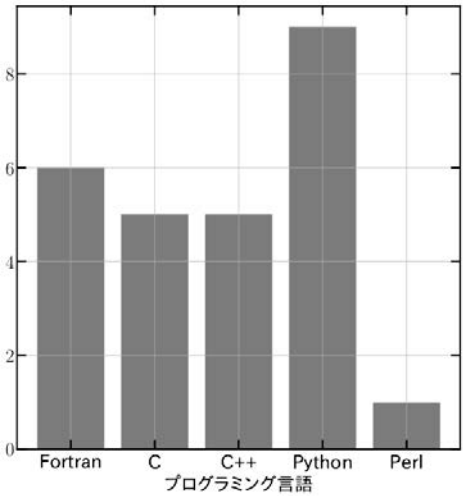


図 2：普段使用しているプログラミング言語

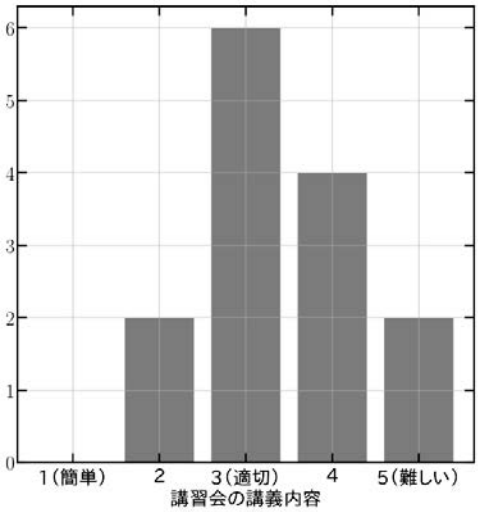


図 3：講習会の講義内容（プレゼン）

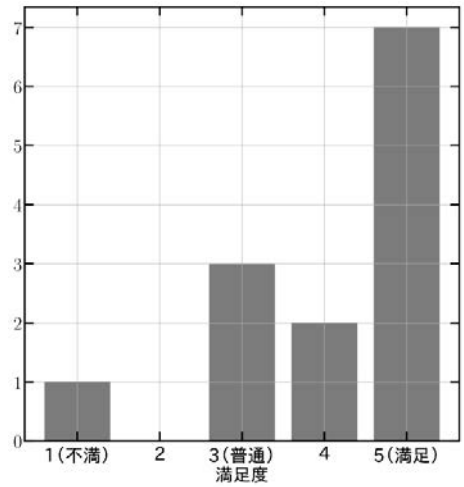


図 4：講習会に参加した満足度

講習会の難易度設定については、多くの参加者が適切と回答していましたが、例年の現地開催での講習会よりは難しく感じた方が多かったようです（図 3)。また、満足度の平均値は 4.0 であり、図 4 に示すようにおおむね満足度が高かったようです。

今回は Zoom を用いての完全オンライン開催であったので、オンライン開催に関する回答をいただきました。オンライン開催で良かったことについての回答は

- わざわざ会場へ行かなくてよかったこと
- 実習ができたのが良かったです
- マルチディスプレイで確認しながら受講できた
- 移動がない
- 情報共有しやすい
- 講師の方の画面と自分の画面を並べながら作業をできた
- 参加しやすくなったと思う
- 現地に出向かなくて良くて参加が楽なこと
- 画面共有が楽であること
- 地方からでも気軽に参加できる

で、悪かったことについての回答は

- 画面を zoom に使ってしまうので、手元の資料、ターミナル、zoom を同時に見ることが難しかった
- 説明者がプロンプト画面を出したり、プレゼン画面に変えたりしていると、とても見づらかった。プレゼン画面固定でお願いしたい。
- 途中でノイズがのった時間あり(午前中の一部)。それ以外は特に問題なし。
- 職場にいると雑用が飛び込んできて集中できない。

というものでした。会場まで移動する必要がなくなったため遠隔地からの参加が容易になったというメリットを回答される方が多かったです。また、特にディスプレイの数などの参加者側の環境によって感想が大きく変わるようであり、参加者の環境をある程度揃えられるオンサイト講習会とはまた違った困難があります。

また、自由記述欄においては以下の感想をいただきました。

- お忙しいところ、丁寧な講演をしていただきまして大変ありがとうございました。
- MPI は経験がなかったのですが、まずはさわりの部分だけでも知ることができて大変有意義でした。
- 説明が非常にわかりやすかったです。ありがとうございました。
- 演習の初めはどのように書けば良いのかわからなかったので、穴埋め形式の簡単な例があるとよかったです。
- 全体的に説明も分かりやすく得たかった知識を得ることができました。

- とりあえず簡単なプログラムを動かさせてから座学に入るという方式がとても良かったように思います。
- また課題について、講習中に終わる可能な簡単なものと講義後もじっくりとできる課題の二つがあるのがと良いと思いました。
- 講師の方がどんな質問にもものすごく親切に回答していたのが印象的でした。私も大変、良い勉強になりました。

演習課題については、第 132 回講習会の際のアンケートを受けて基礎的な問題を増やしました。感想からも、簡単な演習課題があることが歓迎されていることが伺えますので、今後もアップデートしていく予定です。

同様の講習会があれば、「また受けたい」という回答が 9 名、「どちらともいえない」が 4 名で、その他の講習会にも期待されていることが伺えます。

以上

第142回お試しアカウント付き並列プログラミング講習会「MPI上級」実施報告

塙　敏博
東京大学情報基盤センター

2020年10月26日（月）、第142回お試しアカウント付き並列プログラミング講習会「MPI上級」が開催されました。MPI上級は、毎年1回ずつの開催で今回が4回目となりました。例年は東京大学情報基盤センターにおいて開催されている本講習会ですが、今回は新型コロナウイルス感染症対策のためにZoomを用いたオンライン講習会として実施されました。

本講習会は、東京大学内および学外における当センターのスーパーコンピュータの利用を考えているユーザに加え、社会貢献の一環として、高性能計算や並列処理の技術習得を目的にした企業に所属する研究者、技術者の方が参加可能になっております。

受講者は、大学院生(修士)：3名、大学院学生(博士)：2名、教授：1名、研究機関研究員：1名、企業の方：4名、参加者合計：11名、でした。

1ヶ月有効となるお試しアカウントが与えられ、Oakforest-PACSスーパーコンピュータシステムの利用方法、MPI(Message Passing Interface)の高度な機能を用いたプログラミングに関する講習会を1日間で実施しました。

当日のプログラムを、以下に掲載します。

- 10月26日（月）
  - 10：00 - 11：20　MPI概要、Oakforest-PACSで使えるMPI実装
  - 11：30 - 12：30　ノンブロッキング通信（演習）
  - 13：30 - 14：30　派生データ型、MPI-IO（演習）
  - 14：40 - 16：10　コミュニケータ、マルチスレッドとMultiple-Endpoint（演習）
  - 16：20 - 17：30　片側通信（演習）

9名の参加者について、講習会に関するアンケートをご提出いただきました。

MPI上級だけあって、プログラミング経験については、5年未満が約半数いたものの、20年を超える方も2名いらっしゃいました。使用しているプログラミング言語については、Pythonがトップで、次がCとC++（複数回答可）となり、Pythonのユーザが明らかに増加しています。

主要な項目の集計結果を以下に示します。

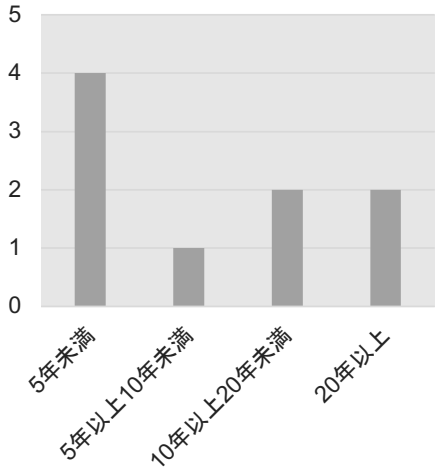


図 1 プログラミング経験

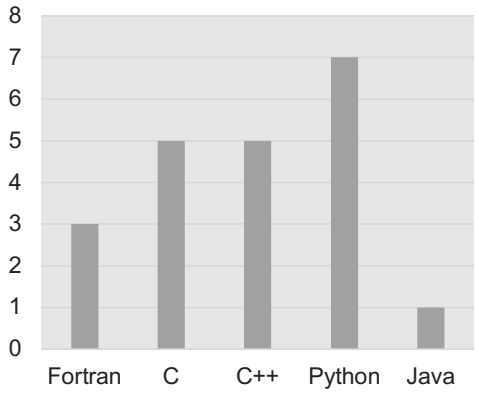


図 2 普段使用するプログラミング言語

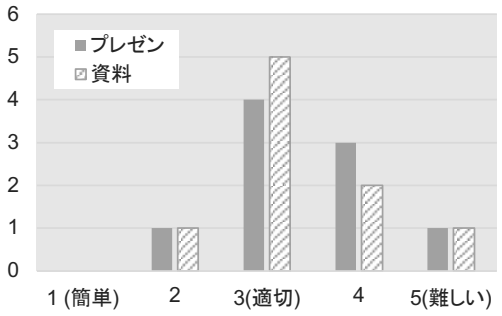


図 3 講習会の内容

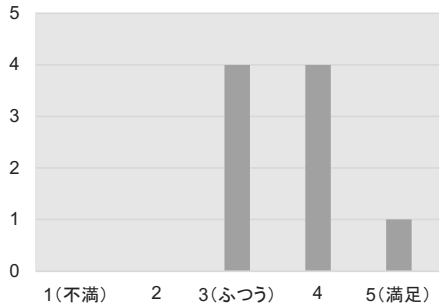


図 4 講習会参加の満足度

図 1 より、顧客満足度の平均値は 3.7 でした。

また、以下の感想をいただきました。

- もう少し演習量が増えたら良かったと思う.
- 本編以外の説明に時間が割かれ、後半の演習時間などが逼迫する印象があります。何度か講習を受講する場合はシステム部分が重複するため、システム部分は別立てにして本編だけにして欲しい。
- MPI に関する最近の動向について、勉強になりました。丁寧に説明していただいた点が良かったです。
- 大変かもしれませんが、参加者自身が課題を持ち込むという形でも良いかもしれないと思いました。

- ありがとうございました。一部 Fortran 版は実装が難しいという部分があり、サンプルプログラムが C 言語のみという演習がありましたが、Fortran 版がなぜ難しいのか詳しく聞けるとなお良かったです。

Zoom と Slack を用いた講習会形式について良かった点として、

- 興味のある講習会が自分の大学でなく，他の大学の情報基盤センターで開講される場合がしばしばある．例年だと，このような場合に講習会には興味はあるけど，出張費(航空券など)が高くて中々受講し難かった．しかし，今年のように，ZOOM で開講されると，その負担がなくなるし，さらに出張にかかる時間のコストなども非常に低減できるので非常に良かったと思う．また，ZOOM での開講により，受講する場所を自分で自由に選べることができるようになったので，自分がより集中できる場所で受講できたことも非常に良かったと思う．しかも ZOOM と SLACK を用いたところ，質疑応答にも何の不便なく受講できたと思う．今後，コロナなどに関係なく，他のセミナーもこのような形式で行われると，非常に参加しやすくなると思うし，可能であれば，ぜひそうして頂きたい．(東京大学情報基盤センターでは，非常に興味深い講習会が多数開かれているようなので，他の講習会もぜひ受講したいと思う．)
- 手元を直接見られる、質問がしやすい。
- 移動がない，講師の操作を見ることができる。
- オンサイトで受講した時よりも、聞き取りやすく、コードも見やすかった。
- 遠隔地からでも、時間的な制約が無く参加し易い点が良かったです。
- 遠隔地からでも、時間的な制約が無く参加し易い点が良かったです。
- 講師の方の作業画面が大きく見え、かつ自分のターミナルの隣なので見比べやすかった。東京に行かなくとも参加できる。

など、大変好評でした。

悪かった点としては、

- 画面の切り替えによって前まで見えていた画面が見えなくなってしまう。
- 集中力が切れる。
- 画面が見切れたりした。
- 使用しているドキュメントに修正があっても反映されるまでにタイムラグがあり、すぐに見たいときに不便に感じました。インタラクティブの操作に慣れていないため、質問するタイミングが難しい。

などがありました。どうしても一方通行になってしまうため、難しい部分があることはわかりました。

内容が盛り沢山で１日で実施している関係で、演習時間については不足気味でした。今後も改善を続けていきたいと思います。同時に、ホームページで資料を公開しております

し、YouTube で講習会録画も公開していますので、関心のある方はごらんください。

同様の講習会があれば、「また受けたい」という回答が 6 名、「どちらともいえない」が 3 名で、その他の講習会にも期待されていることが伺えます。

平成 24 年 4 月から、当センターのスーパーコンピュータシステムを利用する企業利用者向けトライアルユース制度（パーソナルコース相当）では、お試しアカウント付き講習会の受講が義務づけられています。企業の方でトライアルユース制度（パーソナルコース相当）をご利用の方は、本講習会の日程について事前にご確認ください。

詳細および講習会への申込みは、以下のホームページでご確認ください。

https://www.cc.u-tokyo.ac.jp/events/lectures/

以上

第 143 回お試しアカウント付き並列プログラミング講習会 OpenMP によるマルチコア・メニィコア並列プログラミング入門（オンライン）

中 島 研 吾

東京大学情報基盤センター

本稿では，2020 年 11 月 2 日（月）にオンライン開催した第 143 回お試しアカウント付き並列プログラミング講習会「OpenMP によるマルチコア・メニィコア並列プログラミング入門[1]」（共催 : 東京大学情報基盤センター、PC クラスタコンソーシアム（実用アプリケーション部会・HPC オープンソースソフトウェア普及部会））について紹介する。

東京大学情報基盤センター（以下，本センター）では 2007 年からスーパーコンピュータを使用した「お試しアカウント付き並列プログラミング講習会」を開催しているが，新型コロナウイルス感染症対策のため，2020 年 4 月からは全ての講習会を Zoom 使用による「オンライン」講習会として実施している。本講習会は第 131 回（2020 年 4 月 27 日開催）と同じ内容である。

近年マイクロプロセッサのマルチコア化が進み，様々なプログラミングモデルが提案されている。中でも OpenMP は指示行（ディレクティヴ）を挿入するだけで手軽に「並列化」ができるため，広く使用されており，様々な解説書も出版されている。メモリへの書き込みと参照が同時に起こるような「データ依存性（data dependency）」が生じる場合に並列化を実施するには，適切なデータの並べ替えを施す必要があるが，このような対策は OpenMP 向けの解説書でも詳しく取り上げられることは余り無い。本講習会では，「有限体積法から導かれる疎行列を対象とした ICCG 法」を題材として，科学技術計算のためのマルチコアプログラミングにおいて重要なデータ配置, reordering などのアルゴリズムについての講義, スパコン（Oakbridge-CX（OBCX）[2]）を使用した実習を実施した。本講習会は筆者が本学大学院工学系研究科・情報理工学系研究科の講義として実施している「計算科学アライアンス特別講義 I」，「科学技術計算 I」，「スレッド並列コンピューティング」の内容に準拠している[3]。

今回も前回に引き続き，スパコンログインのための手順（PC へのソフトウェアのインストール，SSH 鍵認証の原理）についての詳細な資料[4]を準備し，前以て受講者には利用者 ID，初期パスワード等を送付することによって，可能な限り予めログインまでを済ませて講習会に臨んでもらうこととした。特にトラブルもなく，無事にログインまで済ませた状況で参加された模様である。画面にスライドを投影してのオンライン講義で重要なのは，**可能な限り詳細な情報を予め資料に書き込んでおく**，ということである（当たり前のことではあるが）。対面の講義では，受講者の表情を見ながら，説明不足と思われる点を更に詳しく，というようなことを臨機応変にできるのだが，オンラインではそれができない。前回の講習会，2020 年度夏学期の講義などで得られた知見を元に講習会資料は更にわかりやすく改良した。本来であれば 105 分×10 回程度の講義の内容を 1 日（正味 7.5 時間（450 分）程度）に圧縮しての内容となった（表 1）。

---

1 https://www.cc.u-tokyo.ac.jp/events/lectures/143/
2 https://www.cc.u-tokyo.ac.jp/supercomputer/obcx/service/
3 http://nkl.cc.u-tokyo.ac.jp/20s/
4 http://nkl.cc.u-tokyo.ac.jp/seminars/multicore/OnlineClass.pdf

題材とする有限体積法コードの説明に従来は 3 コマ，3 時間程度をかけるのであるが，教材[5]を予め提供することにより予習してもらい，90 分程度で済ませることができた。また，表 1 にある「Oakbridge-CX へのログイン」時間も，上述の事前準備により大幅に短縮することができた。

表 1　本講習会の概要

09：00-10：30	有限体積法
10：30-11：00	Oakbridge-CX へのログイン
11：00-12：30	OpenMP 入門
13：30-16：00	オーダリング
16：00-17：30	OpenMP 並列化
17：30-18：00	質疑

講義内容の詳細については，ウェブページから資料及び録画ビデオをダウンロードできるのでそちらを参照いただきたい（https://www.cc.u-tokyo.ac.jp/events/lectures/143/）。本講義では，受講者の多様なバックグラウンドを考慮して，全講義内容について Fortran，C 両方による教材を準備している。受講者は Oakbridge-CX の最大 8 ノード（最大 448 コア，実行時間上限 15 分）を利用できる（実際に使用したのは 1 ノード）。アカウントは講習会終了後 1 ヶ月間有効であり，復習に利用することができる。

事前登録者 11 名，出席者 9 名（学生：3 名，企業：6 名）と，第 131 回（出席者 18 名）と比較して少なかった。講習会終了後にアンケートを実施した（回収本数：7）。表 2 は質問項目と回答（5 段階評価）の人数分布である。全体的な満足度の平均値は 5 点満点で 4.57 と第 131 回の 4.00 と比較して上昇した。アンケートの自由記述欄については，「現地に行かなくて良い分，気軽にまた感染も気にせずに受講できる」，「他の参加者も含めた周囲の人とコミュニケーションをとりづらいのが難点」，「休憩時間が少ない」というコメントは第 131 回と変わらないが，一方で「手を動かせる時間があり理解が深まった」というコメントもあった。講義や教材の難易度については「3：普通」が多く適切なレベルであったようだ。

本稿を執筆している 2021 年 1 月初頭の段階で，東京を中心に COVID-19 感染者数が急激に増加している。今後も「オンライン講習会」を中心に考えていく必要があり，本センターとしても講習会，プログラミング教育の今後のあり方を継続して検討していく予定である

表 2　アンケート集計結果

	評点	1	2	3	4	5
(a) 講習会時間	短い⇔長い			3	4	
(b) 講習会講義内容（プレゼン）	簡単⇔難			6		1
(c) 配布資料内容	簡単⇔難			6	1	
(d) サンプルプログラム内容	簡単⇔難		1	5		1
(e) 満足度（平均 4.57）	不満⇔満足				3	4

[5] http://nkl.cc.u-tokyo.ac.jp/seminars/multicore/

第 144 回お試しアカウント付き並列プログラミング講習会
一日速習：三次元並列有限要素法とハイブリッド並列プログラミング（オンライン）

中 島 研 吾

東京大学情報基盤センター

本稿は，2020 年 11 月 6 日（金）にオンライン開催した「第 144 回お試しアカウント付き並列プログラミング講習会　一日速習：三次元並列有限要素法とハイブリッド並列プログラミング[1]」（共催：東京大学情報基盤センター、PC クラスタコンソーシアム（実用アプリケーション部会・HPC オープンソースソフトウェア普及部会））の開催報告である。

本センターでは様々な並列プログラミング講習会を実施しており[2]，本センターの利用者に限定せず，また大学教職員，学部・大学院学生，研究機関研究者のみならず，企業の技術者・研究者にも門戸を開き，本センターのスパコンを使用した実習も実施して，並列プログラミング技術の普及に貢献して来た。新型コロナウイルス感染症対策のため，2020 年 4 月からは全ての講習会を Zoom 使用による「オンライン」講習会として実施している。

本講習会は，有限要素法による熱伝導解析プログラムを MPI 及び OpenMP を使用して並列化するための手順、特に並列分散データ構造に関する考え方を中心に説明するもので，手元のプログラムを，「並列化」したいと考えている受講者を対象としている。有限要素法についてはプログラムの簡単な解説を実施するが，予備知識の無い方は事前に公開する資料での予習の他，別途実施したオンライン講習会「一日速習：有限要素法プログラミング徹底入門[3]」のサイトで講義を録画したビデオを使った予習も可能である。

スケジュールを表 1 に示す。講義内容の詳細については，ウェブページ[4]から資料をダウンロードできるのでそちらを参照いただきたい。Fortran と C の両者の教材が準備されている。今回は Fortran を普段使用している受講者が多かったため，Fortran 向け教材による説明を実施した。受講者は Oakbridge-CX の最大 8 ノード（最大 448 コア，実行時間上限 15 分）を利用できる（実際に使用したのは 1 ノード）。アカウントは講習会終了後 1 ヶ月間有効であり，復習に利用することができる。

事前登録者 12 名，出席者 10 名（学生：3 名，大学教員 3 名，企業：4 名）であった。講習会終了後にアンケートを実施した（回収本数：8）。表 2 は質問項目と回答（5 段階評価）の人数分布である。全体的な満足度の平均値は 5 点満点で 4.375 と高かった。講義や教材の難易度については「3：普通」が多く適切かやや難しいレベルであったようだ。

アンケートの自由記述欄については，「現地に行かなくて良い分，気軽にまた感染も気にせずに受講できる」，「他の参加者も含めた周囲の人とコミュニケーションをとりづらいのが難点」，「休憩時間が少ない」というコメントは他のオンライン講習会と同様であった。本講習会については以前から「一日では短い」という指摘があったが，今回も複数の受講者から，一日では

---

1 https://www.cc.u-tokyo.ac.jp/events/lectures/144/
2 https://www.cc.u-tokyo.ac.jp/events/lectures/
3 http://nkl.cc.u-tokyo.ac.jp/FEM/
4 http://nkl.cc.u-tokyo.ac.jp/pFEM/

短く，二日にして，演習を増やしてはどうかというコメントがあった。

本稿を執筆している 2021 年 1 月初頭の段階で，東京を中心に COVID-19 感染者数が急激に増加している。今後も「オンライン講習会」を中心に考えていく必要があり，本センターとしても講習会，プログラミング教育の今後のあり方を継続して検討していく予定である。また，オンライン化により，必ずしも 2 日間連続で設定する必要もなくなったため，今後は 2 日にわたって実施することも検討したい。

表 1　一日速習：三次元並列有限要素法とハイブリッド並列プログラミング　スケジュール
講師：中島研吾（東京大学情報基盤センター）

09:00～09:30	有限要素法プログラムの概要
09:30～10:15	並列有限要素法への道
10:15～10:45	OBCX ログイン
10:45～12:00	並列データ構造・一般化された通信テーブル
13:00～14:00	一次元並列有限要素法
14:00～15:30	三次元並列有限要素法（データ構造）
15:30～17:00	三次元並列有限要素法（計算本体，並列可視化）
17:10～18:00	OpenMP 超入門，ハイブリッド並列プログラミング

表 2　アンケート集計結果

	評点	1	2	3	4	5
(a) 講習会時間	短い⇔長い	1	1	5	1	
(b) 講習会講義内容（プレゼン）	簡単⇔難	1		2	5	
(c) 配布資料内容	簡単⇔難	1		3	4	
(d) サンプルプログラム内容	簡単⇔難		1	5	2	
(e) 満足度（平均 4.375）	不満⇔満足				5	3

第 145 回お試しアカウント付き並列プログラミング講習会
一日速習：有限要素法プログラミング徹底入門（オンライン）

中 島 研 吾

東京大学情報基盤センター

本稿では，2020 年 11 月 10 日（火）にオンライン開催した第 145 回お試しアカウント付き並列プログラミング講習会「一日速習：有限要素法プログラミング徹底入門[1]」（共催：東京大学情報基盤センター、PC クラスタコンソーシアム（実用アプリケーション部会・HPC オープンソースソフトウェア普及部会））について紹介する。

有限要素法（Finite Element Method，FEM）は偏微分方程式の数値解法として，様々な科学技術計算に使用されている。基礎的な研究開発の他，NASTRAN に代表される有限要素法による商用コードも既に半世紀近く産業利用も含む様々な分野で設計・安全評価などに使用されている。有限要素法は要素単位のローカルな演算から構成されているところから，並列化が比較的容易であることが知られている。

本センターでは，本センターのスーパーコンピュータの利用促進と並列プログラミング技術の普及を目的として，様々な並列有限要素法の講習会[2] を実施してきた。

有限要素法の理論，手法は大学教養程度の数学，物理の知識があれば充分に理解できるものであるが，並列化を実施するためには，単に個々の事項の理解に留まらず，数学的背景，数値アルゴリズムとプログラミングを結びつけて理解している必要がある。有限要素法そのものと並列化を 1 日で全て解説するのは，スケジュール的にも困難で，細かいところは省略せざるを得ないため，受講者にとっても消化不良となる傾向があった。

本センターでは，2020 年 6 月 12 日に第 134 回講習会で，初めての試みとして，並列化には立ち入らず，**スパコンを使用したハンズオンも実施せず**，有限要素法を構成する基礎的な理論，数値アルゴリズムとその実装に主眼を置いた講習会を 1 日で実施したところ，比較的好評であり需要が高いことも確認できたため，今回，同様の内容で講習会を実施した。今回は，前回（第 134 回）のフィードバックから得られた知見も踏まえ，更に改良した詳細な教材を提供した。

本講義の内容（表 1）は 2019 年度冬学期に講義として実施した，科学技術計算 II（大学院情報理工学系研究科数理情報学専攻）／コンピュータ科学アライアンス特別講義 II（同 コンピュータ科学専攻）／ハイブリッド分散並列コンピューティング（大学院工学系研究科電気系工学専攻）「並列有限要素法入門」[3] の導入部と同じ内容を日本語で実施したものであり，最初の「有限要素法入門[4]」については，時間の都合もあり，受講者に教材を予習してもらい，当日は簡単な説明に留めた。

今回は，前述のようにスパコンは利用せず，各自の PC に一次元・三次元有限要素法プログラム（Fortran・C）をダウンロードしてもらい，数学的・理論的な背景から，実問題（定常熱伝導）まで，実習も交え，大規模連立一次方程式の反復解法（前処理付き共役勾配法）も含め

---

[1] https://www.cc.u-tokyo.ac.jp/events/lectures/145
[2] https://www.cc.u-tokyo.ac.jp/events/lectures/116
[3] http://nkl.cc.u-tokyo.ac.jp/19w
[4] http://nkl.cc.u-tokyo.ac.jp/FEM/01-FEMintro.pdf

て解説した。詳細はホームページ（https://www.cc.u-tokyo.ac.jp/events/lectures/145/）を参照されたい。当日利用した資料，サンプルプログラムの他，録画したビデオを見ることもできる。

表 1　本講習会の概要

09：00-09：45	有限要素法入門
09：45-12：30	一次元有限要素法
13：30-16：00	三次元有限要素法
16：00-17：00	並列有限要素法への道
17：00-17：15	討論・質疑等

事前登録者は 14 名，出席者は 10 名（学生：5 名，大学教職員 1 名，研究機関：1 名，企業：3 名）となった。講習会終了後にアンケートを実施した（回収本数：7）。表 2 は質問項目と回答（5 段階評価）の人数分布である。全体的な満足度の平均値は 5 点満点で 4.57 と第 134 回の 4.32 と比較して上昇した。講義や教材の難易度については「3：普通」が多く適切なレベルであったようだ。アンケートの自由記述欄には，多くの参加者が：

- 現地に行かなくて良い分，気軽にまた感染も気にせずに受講できる
- 有限要素法について，基礎的な理論からプログラム実装，連立一次方程式の反復解法も含めてコンパクトにまとまっていて有用であった
- 教材が詳細でわかりやすい

というようなコメントを記載していた。一日で 1D と 3D を両方やったため，1D のみ一日かけてじっくりやる講習会があっても良いのでは，というコメントもあった。

本稿を執筆している 2021 年 1 月初頭の段階で，東京を中心に COVID-19 感染者数が急激に増加している。今後も「オンライン講習会」を中心に考えていく必要があり，本センターとしても講習会，プログラミング教育の今後のあり方を継続して検討していく予定である

表 2　アンケート集計結果

	評点	1	2	3	4	5
(a) 講習会時間	短い⇔長い		1	5	1	
(b) 講習会講義内容（プレゼン）	簡単⇔難		1	6		
(c) 配布資料内容	簡単⇔難			6	1	
(d) サンプルプログラム内容	簡単⇔難		1	5	1	
(e) 満足度（平均 4.57）	不満⇔満足			1	1	5

第 146 回お試しアカウント付き並列プログラミング講習会
有限要素法で学ぶ並列プログラミングの基礎（オンライン）

中 島 研 吾

東京大学情報基盤センター

本稿は，2020 年 12 月 1 日（火）にオンライン開催した「第 146 回お試しアカウント付き並列プログラミング講習会　有限要素法で学ぶ並列プログラミングの基礎[1]」（共催：東京大学情報基盤センター、PC クラスタコンソーシアム（実用アプリケーション部会・HPC オープンソースソフトウェア普及部会））の開催報告である。

本センターでは様々な並列プログラミング講習会を実施しており[2]，本センターの利用者に限定せず，また大学教職員，学部・大学院学生，研究機関研究者のみならず，企業の技術者・研究者にも門戸を開き，本センターのスパコンを使用した実習も実施して，並列プログラミング技術の普及に貢献して来た。新型コロナウイルス感染症対策のため，2020 年 4 月からは全ての講習会を Zoom 使用による「オンライン」講習会として実施している。

本講習会は，有限要素法，MPI の基礎についてじっくり学びたいという要望に応えるべく，有限要素法による「一次元」熱伝導解析プログラムを，MPI を使用して並列化するための手順について解説，実習を実施するものであり，2016 年 5 月に「有限要素法で学ぶ並列プログラミングの基礎」として新たに開講した[3]。MPI（Message Passing Interface）は SPMD（Single Program Multiple Data）型と呼ばれるパラダイムを実現するのに適している。有限要素法は要素単位のローカルな処理に基づいているため，SPMD 型パラダイムの適用が容易であり，MPI を使用した並列化とは非常に相性が良いことが知られている。本講習会では：

- 有限要素法のプログラミング
- MPI による並列プログラミングの基礎
- 前処理付き反復法による連立一次方程式解法のアルゴリズム

など，大規模シミュレーションに必須の数値アルゴリズムから，並列プログラミングまで幅広い知識を 2 日間で効率的に身につけることができるものであった。今回は，オンライン化にあたって従来 2 日間にわたって実施していた内容を 1 日間に圧縮して実施した。受講者は Oakbridge-CX の最大 8 ノード（最大 448 コア，実行時間上限 15 分）を利用できる（実際に使用したのは 1 ノード）。アカウントは講習会終了後 1 ヶ月間有効であり，復習に利用することができる。講義資料等はホームページ[4]を参照されたい。

事前登録者 18 名，出席者 12 名（学生：3 名，大学教員 3 名，研究機関：2 名，企業：4 名）であった。講習会終了後にアンケートを実施した（回収本数：9）。表 2 は質問項目と回答（5 段階評価）の人数分布である。全体的な満足度の平均値は 5 点満点で 4.33 と高かった。講義や

---

1 https://www.cc.u-tokyo.ac.jp/events/lectures/146/
2 https://www.cc.u-tokyo.ac.jp/events/lectures/
3 https://www.cc.u-tokyo.ac.jp/events/lectures/56/
4 http://nkl.cc.u-tokyo.ac.jp/FEMintro/

教材の難易度については「3：普通」が多く適切なレベルであったようだ。

アンケートの自由記述欄については，「現地に行かなくて良い分，気軽にまた感染も気にせずに受講できる」，「他の参加者も含めた周囲の人とコミュニケーションをとりづらいのが難点」，「休憩時間が少ない」というコメントは他のオンライン講習会と同様であった。また，今回は回線トラブルのためか，初期のうちに音声が途絶えてしまったことがあり，受講者にご不便をおかけしたこと，この場を借りてお詫びしたい。教材については詳細でわかりやすいという評価が多かったものの，やはり 2 日間の内容を 1 日に押し込めたために，演習時間も少なくなり，「従来通り 2 日間実施した方が良い」という意見が複数寄せられた。

本稿を執筆している 2021 年 1 月初頭の段階で，東京を中心に COVID-19 感染者数が急激に増加している。今後も「オンライン講習会」を中心に考えていく必要があり，本センターとしても講習会，プログラミング教育の今後のあり方を継続して検討していく予定である。また，オンライン化により，必ずしも 2 日間連続で設定する必要もなくなったため，今後は 2 日にわたって実施することも検討したい。

表 1　有限要素法で学ぶ並列プログラミングの基礎　スケジュール

09：00–09：30	重み付き残差法
09：30–11：30	一次元有限要素法
11：30–12：00	並列有限要素法への道
13：30–16：30	MPI 並列プログラミング
16：30–18：00	一次元並列有限要素法

表 2　アンケート集計結果

	評点	1	2	3	4	5
(a) 講習会時間	短い⇔長い	1	1	4	1	2
(b) 講習会講義内容（プレゼン）	簡単⇔難	1		3	4	1
(c) 配布資料内容	簡単⇔難	1		5	2	1
(d) サンプルプログラム内容	簡単⇔難	1		5	2	1
(e) 満足度（平均 4.33）	不満⇔満足			2	2	5

第 9 回 JCAHPC セミナー（第 4 回 OFP 利活用報告会）
「人類と地球を護るスーパーコンピューティング」（オンライン）

中 島 研 吾

東京大学情報基盤センター

2020 年 10 月 15 日（木）にオンラインで開催された第 9 回 JCAHPC セミナー）[1]の概要を報告する。

現在，人類と地球は新型コロナウイルス感染症 (COVID-19) という未曾有の危機に直面している。問題解決に向けては「防疫」,「治療」,「創薬」など広範囲にわたり様々な手法による研究開発が急務であり，スーパーコンピュータの有する高速な計算能力，データ処理能力の貢献が期待されている。このような状況の下，HPCI[2]（革新的ハイパフォーマンス・コンピューティング・インフラ ）においては，関係機関の協力のもと，関連する研究が必要とする計算資源を提供する臨時の課題募集「新型コロナウイルス感染症対応 HPCI 臨時公募課題[3]」がおこなわれている。

最先端共同 HPC 基盤施設（JCAHPC: Joint Center for Advanced High Performance Computing）[4]は筑波大学計算科学研究センターと東京大学情報基盤センターとが共同で設立した組織であり，国内最高クラスの性能を有する Oakforest-PACS システム（OFP）[5]を設計，導入，運用している。両センターは本施設を連携・協力して運営することにより，最先端の計算科学を推進し，我が国の学術及び科学技術の振興に寄与してきた。

筑波大学・東京大学の両センターと JCAHPC は，HPCI システム構成機関として「新型コロナウイルス感染症対応 HPCI 臨時公募課題」に計算資源を提供し、新型コロナウイルス感染症に関する研究を支援している。 2021 年 1 月 1 日現在，合計 14 課題が採択されているが，そのうち 3 課題が最先端共同 HPC 基盤施設（JCAHPC）の Oakforest-PACS を使用したものであり，2 課題が筑波大学計算科学研究センターの Cygnus，3 課題が東大情報基盤センターの Oakbridge-CX（OBCX）となっており，合計 8 課題，全体の 6 割近くが筑波大・東大関連のシステムを利用して実施されている。

JCAHPC では，2017 年から毎年 10 月に「OFP 利活用報告会」として利用者，JCAHPC 教員により，OFP における研究開発事例の紹介を実施してきた。第 4 回目となる今回は「人類と地球を護るスーパーコンピューティング」として,「新型コロナウイルス感染症対応 HPCI 臨時公募課題」の事例の他，OFP によるゲリラ豪雨予測リアルタイム実証実験について紹介した。 また,「新型コロナウイルス感染症対応 HPCI 臨時公募課題」については OFP だけでなく，Cygnus（筑波大，1 件），OBCX（東大，2 件）を利用した課題についても紹介した。

今回は完全オンラインで開催され, 合計 97 名の登録者があり, これまでで最大人数であった。なお，通常，メイン会場としている東大柏キャンパス第 2 総合研究棟の会議室は，最大収容人

---

1 https://www.cc.u-tokyo.ac.jp/events/jcahpc/09.php
2 https://www.hpci-office.jp/
3 https://www.hpci-office.jp/pages/adoptionlist2020_25
4 http://jcahpc.jp/
5 http://www.cc.u-tokyo.ac.jp/system/ofp/

数が 70 名程度である。

表 1：第 9 回 JCAHPC セミナー　プログラム

時間帯	講演者・講演題目	利用システム	座長
13:00-13:15	中島研吾（JCAHPC/東京大学：Opening		
13:15-13:45	三好建正（理化学研究所）：ゲリラ豪雨予測のリアルタイム実証実験	OFP	下川辺隆史（JCAHPC/東京大学）
13:45-14:15	Marco Edoardo Rosti（OIST）: Spreading of polydisperse droplets in a turbulent puff of saturated exhaled air	OBCX	
14:15-14:45	岡田純一（UT Heart 研究所）：COVID-19 治療の候補薬: chloroquine、hydroxychloroquine、azithromycin の催不整脈リスクの評価ならびにその低減策に関する研究	OFP	
14:45-15:00	（休憩）		
15:00-15:30	杉田有治（理化学研究所）：新型コロナウイルス表面のタンパク質動的構造予測	OFP	高橋大介（JCAHPC/筑波大学）
15:30-16:00	望月祐志（立教大学）：新型コロナウイルスの主要プロテアーゼに関するフラグメント分子軌道計算	OFP	
16:00-16:30	重田育照（筑波大学）: Covid-19 関連タンパクに対する統合的インシリコリポジショニング	Cygnus	
16:30-17:00	星野忠次（千葉大学）：計算機解析による SARS-CoV-2 増殖阻害化合物の探索	OBCX	
17:00-17:10	朴泰祐（JCAHPC/筑波大学）：Closing		

RIKEN International HPC Summer School 2020
- Toward Society 5.0 -

中 島 研 吾

東京大学情報基盤センター
理化学研究所計算科学研究センター

本稿は, 2020 年 9 月 28 日（月）～30　日（水）にオンラインで実施された,「RIKEN International HPC Summer School 2020 - Toward Society 5.0 -（主催：理化学研究所計算科学研究センター，共催：東京大学情報基盤センター，京都大学学術情報メディアセンター，大阪大学サイバーメディアセンター，神戸大学計算科学教育センター)」の概要を報告するものである。詳細はホームページ[1]を参照されたい。

理化学研究所計算科学研究センター（R-CCS）では，Society5.0 実現に向けて，次代を担う国際的な視野を持った計算科学技術分野の若手研究者等の育成のため，並列計算機を使いこなすためのプログラミング手法を習得できるスクールを開催している。R-CCS の研究者が講義を実施している。今回は完全オンラインで実施され，Oakbridge-CX システム（東京大学情報基盤センター，OBCX）を使用した。表 1 に示すように，OpenMP，MPI による並列プログラミングの基礎，差分法による 2 次元 CFD コードの並列化，行列演算ライブラリ，Deep Neural Network 入門など多岐にわたった内容であり,「富岳」バーチャルツァーも実施された。講義・実習は全て英語で実施された。

出席者は 37 名であり，そのうち 30 名は日本の大学・研究機関・企業に所属する研究者，学生であり，うち留学生が 17 名である。また，R-CCS に 2020 年 4 月から雇用されているポスドクで，COVID-19 のために入国できていない研究者も海外からリモート参加した。また，海外機関に所属する 7 名のうち 5 名はシンガポールの大学の学部生・大学院生で，理研 R-CCS と National Supercomputing Center, Singapore（NSCS）間の共同研究協定に基づき受け入れたものである。

Zoom，Slack を通じて，非常に活発な議論が行われ，並列性能のグラフを Slack に貼り付けて比べ合うなど，オンラインならではの光景も見られた。受講者には概して好評であったが，OBCX が混雑していることもあり，ややジョブが流れにくい傾向にあった。

[1] https://www.r-ccs.riken.jp/en/events/200928.html

表 1：「RIKEN International HPC Summer School 2020 - Toward Society 5.0 -」プログラム

September 28 (Mon), 2020

10:00-10:10	Welcome
10:20-12:00	Basics of parallel programming and execution ✓ Parallel execution models ✓ Basics of MPI and OpenMP
13:30-14:50	Introduction of the example application ✓ 2D CFD
15:00-16:10	Setup of parallel execution environments
16:10-17:10	Hands-on Practice ✓ How to use MPI (send, receive and reduce) ✓ Simple example of OpenMP

September 29 (Tue), 2020

09:00-10:20	Hands-on Practice (cont.) ✓ How to use MPI (send, receive and reduce) ✓ Simple example of OpenMP
10:30-12:00	Lecture→ Hands-on Practice ✓ How to parallelize the example program
13:30-14:50	Hands-on Practice ✓ Parallelization of the example program
16:10-17:10	Hands-on Practice (cont.) ✓ Parallelization of the example program
17:20	Group Photo

September 30 (Wed), 2020

09:00-12:00	Lecture ✓ Matrix Computation (Basics & Library)
13:30-14:00	Fugaku Virtual Computer Tour
14:10-15:30	Hands-on Practice ✓ How to Use Numerical Libraries
15:40-17:10	Lectures and Hands-On ✓ Introduction to Deep Neural Network (DNN)
17:10-17:20	Closing

原　稿　募　集

本誌では利用者の皆様からの原稿を募集しています。以下の執筆要項に基づいて投稿してください。

執　筆　要　項

1　内容は，本センターのスーパーコンピューターシステムの利用者にとって有意義な情報の提供となる原稿とします。

2　掲載可否については当編集委員会で決定させていただきます。

3　掲載可とした投稿原稿に対して，加除訂正を行うことがあります。

4　原稿枚数には特に制限はありませんが，シリーズに分割することもあります。

5　プログラムの実例が大量になる場合（概ね１頁を超える）は，本文には一部のみを記述し，投稿者の Web ページ等に全体を掲載し，その URL を引用するようにしてください。

6　原稿は横書きにしてください。

7　原稿は，印刷出来上がり寸法がＢ５判で文字の大きさ９ポイントを標準とし，印字部分は必ず左端を 2.5cm 以上空けて，縦 21cm，横 14cm になるようにしてください。Ａ４サイズの場合はＢ５サイズに縮小した場合に上記のサイズになるようにしてください。

　併せて，PDF 形式（フォント埋め込み）の完全原稿を電子メールにて uketsuke@cc.u-tokyo.ac.jp まで提出願います。

8　投稿原稿は返却しません。

9　採用された原稿は，本センターの Web ページ上でも掲載させていただきます。

希望がある場合は，１タイトルにつき 50 部の別刷を差し上げます。

【スーパーコンピュータシステム利用案内】

お知らせ	Web ページ
サービス案内、運転状況など	https://www.cc.u-tokyo.ac.jp/
公開鍵登録、マニュアル閲覧など	https://obcx-www.cc.u-tokyo.ac.jp/ (Oakbridge-CX) https://ofp-www.jcahpc.jp/ (Oakforest-PACS) https://reedbush-www.cc.u-tokyo.ac.jp/ (Reedbush)

お問い合わせ内容	お問い合わせ先
利用申込関係	電子メール： uketsuke@cc.u-tokyo.ac.jp スーパーコンピュータシステム利用申込書提出先 〒113-8658 東京都文京区弥生 2-11-16 東京大学情報システム部 情報戦略課研究支援チーム
プログラム相談・システム利用に関する質問	https://www.cc.u-tokyo.ac.jp/supports/contact/#SOUDAN
システムに関する要望・提案	voice@cc.u-tokyo.ac.jp

【IP ネットワーク経由時のホスト名】

シ ス テ ム	ホ ス ト 名
Oakbridge-CX スーパーコンピュータシステム	obcx.cc.u-tokyo.ac.jp 以下のホストの何れかに接続します※ obcx0{1-6}.cc.u-tokyo.ac.jp
Oakforest-PACS スーパーコンピュータシステム	ofp.jcahpc.jp 以下のホストの何れかに接続します※ ofp0{1-6}.jcahpc.jp
Reedbush スーパーコンピュータシステム (Reedbush-H/L)	reedbush.cc.u-tokyo.ac.jp 以下のホストの何れかに接続します※ reedbush-{u{1-4},h1}.cc.u-tokyo.ac.jp

※どのホストに接続しても同じです。

【編集】
東京大学情報基盤センタースーパーコンピューティング研究部門
東京大学情報システム部情報基盤課スーパーコンピューティングチーム
〃 情報戦略課研究支援チーム
【発行】
東京大学情報基盤センター
〒113-8658 東京都文京区弥生2-11-16
（電話） 03-5841-2717（ダイヤルイン）
（FAX） 03-5841-2708

東京大学情報基盤センター ・ スーパーコンピューティングニュース
Vol.23 No.1(2021.1)

目　　次