（オンライン） ISSN 2436—4940

スーパーコンピューティングニュース

Vol.23 No.6, 2021.11

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

Wisteria/BDEC-01 スーパーコンピュータシステム 利用負担金表（2021 年 5 月 14 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		
一般申込	【Wisteria-O/A】 申込 1 セット当り 60,000 円 （8,640 トークン）		グループ 1 セット当り /work 2TB 利用者当り /home 50GB	【Wisteria-O】 最大ノード数 2,304 ノード 【Wisteria-A】 最大 GPU 数 64
公募制度による申込 （Wisteria-A における「GPU 専有申込」,「ノード固定」も可能）	【Wisteria-O】 申込 1 セット当り 60,000 円 （8,640 トークン）	【Wisteria-O】 申込 1 セット当り 72,000 円 （8,640 トークン）	グループ 1 セット当り /work 2TB 利用者当り /home 50GB	
	【Wisteria-A】 申込 1 セット当り 180,000 円 （25,920 トークン）	【Wisteria-A】 申込 1 セット当り 216,000 円 （25,920 トークン）	グループ 1 セット当り /work 6TB 利用者当り /home 50GB	
GPU 専有申込 （Wisteria-A のみ，公募制度による申込可能）	【Wisteria-A】 申込 1 セット当り 270,000 円 （25,920 トークン）	【Wisteria-A】 申込 1 セット当り 324,000 円 （25,920 トークン）	グループ 1 セット当り /work 6TB 利用者当り /home 50GB	1, 2, 4GPU のみ申込可，申込単位は下表参照，利用期間は 1 ヶ月単位で設定可
ノード固定 （Wisteria-A のみ，公募制度による申込可能）	【Wisteria-A】 申込 1 セット当り 2,160,000 円 （207,360 トークン）	【Wisteria-A】 申込 1 セット当り 2,592,000 円 （207,360 トークン）	グループ 1 セット当り /work 48TB 利用者当り /home 50GB	1 ノード 8GPU 1 年分，1 セットのみ申込可，利用期間は 1 ヶ月単位で設定可
トークン追加	5,000 円 （720 トークン）	6,000 円 （720 トークン）		
ディスク追加	6,480 円／（1TB*年）			1TB 単位で申込可 （/work のみ）

※Wisteria/BDEC-01 においてはパーソナルコースとグループコースの区分を廃止し、これまでのパーソナルコースは一般申込に統合した。

※Wisteria-O のトークン消費係数は 1.00（1 ノード当り），Wisteria-A のトークン消費係数は 3.00（1GPU 当り）である。
Wisteria-O にトークン消費係数 1.50 のノード群(優先利用向け)を全体の 15%程度設ける。

※括弧（ ）内は付与するトークン量。実行したジョブのノード時間積または GPU 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。

※公募制度による申し込み，ノード固定の申し込みには審査を要する。

※/home のディスク容量は複数のグループに所属している場合でも利用者当り 50GB 固定。

※GPU 専有申込の申込単位

GPU 数	トークン量	大学・公共機関等	企業
1	25,920	270,000 円	324,000 円
2	51,840	540,000 円	648,000 円
4	103,680	1,080,000 円	1,296,000 円

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）

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 3 セットまで 100,000 円 （8,640 ノード時間）		申込 1 セット当り /work 4TB 利用者当り /home 50GB	最大ノード数 256 ノード
グループコース	一般申込	申込 1 セット当り 100,000 円 （8,640 ノード時間）	申込 1 セット当り 120,000 円 （8,640 ノード時間）	グループ 1 セット当り /work 4TB 利用者当り /home 50GB	最大ノード数 256 ノード
	ノード固定	申込 1 セット当り 150,000 円 （8,640 ノード時間）	申込 1 セット当り 180,000 円 （8,640 ノード時間）		
トークン追加		8,400 円 （720 ノード時間）	10,000 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 （/work のみ）

※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）

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 6 セットまで 50,000 円 （8,640 ノード時間）		申込 1 セット当り /work 1TB 利用者当り /home 50GB	最大ノード数 2,048 ノード
グループコース	申込 1 セット当り 50,000 円 （8,640 ノード時間）	申込 1 セット当り 60,000 円 （8,640 ノード時間）	グループ 1 セット当り /work 1TB 利用者当り /home 50GB	最大ノード数 2,048 ノード
トークン追加	4,200 円 （720 ノード時間）	5,000 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 （/work のみ）

※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	Wisteria/BDEC-01 システム
Reedbush-H/L システム	ー	1.5	0.75	1.2
Oakforest-PACS システム	0.6	ー	0.5	0.8
Oakbridge-CX システム	1.3	2	ー	1.6
Wisteria/BDEC-01 システム	0.8	1.2	0.6	ー

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（Wisteria/BDEC-01, 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

Wisteria/BDEC-01 スーパーコンピュータシステム（Wisteria-O） ジョブクラス制限値（2021 年 5 月 14 日）

キュー名※1	ノード数※2（最大コア数）	制限時間（経過時間） 正式運用期間	試験運用期間	メモリー容量（GiB）※3	一般申込	公募制度による申込
debug-o	1 ～ 144 (6,912)	30 分	30 分	28	○	○
short-o	1 ～ 72 (3,456)	8 時間	4 時間	28	○	○
(regular-o)						
small-o	1 ～ 144 (6,912)	48 時間	12 時間	28	○	○
medium-o	145 ～ 576 (27,648)	〃	〃	〃	○	○
large-o	577 ～ 1,152 (55,296)	〃	〃	〃	○	○
x-large-o	1,153 ～ 2,304 (110,592)	24 時間	6 時間	〃	○	○
priority-o	1 ～ 288 (13,824)	48 時間	12 時間	28	○	○
challenge-o	1 ～ 7,680 (368,640)	24 時間	－	28	★	★
(interactive-o) ※4						
interactive-o_n1	1 (48)	30 分	30 分	28	○	○
interactive-o_n12	2 ～ 12 (576)	10 分	10 分	〃	○	○
prepost	1 (56)	6 時間	3 時間	340	○	○
prepost1_n1 ～ prepost4_n1	1 (56)	1～6 時間	1～3 時間	340	○	○
prepost1_n4	1 ～ 4 (224)	1～6 時間	1～3 時間	340	○	○
prepost1_n8	1 ～ 8 (448)	1～6 時間	1～3 時間	340	○	○

★ 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-o，debug-o，short-o を小文字で指定する
regular-o 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 ノード当り 1.00。ただし priority-o は優先利用ノード群のためトークン消費係数は 1.50
※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）

pjsub --interact -g グループ名 -L “rscgrp=interactive-o,node=ノード数”

Wisteria/BDEC-01 スーパーコンピュータシステム（Wisteria-A） ジョブクラス制限値（2021 年 5 月 14 日）

キュー名※1	ノード数・GPU 数※2（最大 GPU 数）	制限時間（経過時間） 正式運用期間	試験運用期間	メモリー容量（GiB）※3	一般申込	公募制度による申込	GPU 専有申込	ノード固定
debug-a	1 ノード (8)	30 分	30 分	448	○	○	○	○
short-a	1 ～ 2 ノード (16)	2 時間	1 時間	448	○	○	○	○
(regular-a)								
small-a	1 ～ 2 ノード (16)	48 時間	12 時間	448	○	○	○	○
medium-a	3 ～ 4 ノード (32)	〃	〃	〃	○	○	○	○
large-a	5 ～ 8 ノード (64)	24 時間	6 時間	〃	○	○	○	○
share-debug	1, 2, 4 GPU	30 分	30 分	56	○	○	○	○
share-short	1, 2, 4 GPU	2 時間	1 時間	56	○	○	○	○
(share)								
share-1	1 GPU	48 時間	12 時間	56	○	○	○	○
share-2	2 GPU	〃	〃	〃	○	○	○	○
share-4	4 GPU	24 時間	6 時間	〃	○	○	○	○
challenge-a	1 ～ 39 ノード (312)	24 時間	－	448	★	★	★	★
任意	1 ノード (8)	任意 ※4	－	448	×	×	○	○
interactive-a ※5	1 ノード (8)	10 分	10 分	56	○	○	○	○
share-interactive	1 GPU	〃	〃	〃	○	○	○	○

★ 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-a，debug-a，short-a を小文字で指定する
regular-a 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 1GPU 当り 3.00
※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-a,node=ノード数”

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 一般	グループコース 申込ノード固定
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	〃	○	○	○

★　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，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

センターから

サービス休止等のお知らせ

2021 年 11 月下旬からの計算機サービス予定は以下のとおりです。

Wisteria-BDEC01 スーパーコンピュータシステム

○ Wisteria/BDEC-01 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
11 月 26 日（金）	9:00 ～ 22:00 までサービス休止	月末処理
12 月 17 日（金）	9:00 ～ 22:00 までサービス休止	月末処理
1 月 28 日（金）	9:00 ～ 22:00 までサービス休止	月末処理

・ Wisteria/BDEC-01 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。

○ Wisteria/BDEC-01 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
11 月 25 日（木）8:30 ～ 17:30 まで 12 月 16 日（木）8:30 ～ 17:30 まで 1 月 27 日（木）8:30 ～ 17:30 まで

・上記期間中，Wisteria/BDEC-01 の debug-o/a, short-o/a, regular-o/a, priority-o, interactive-o/a, prepost, share, share-debug, share-short, share-interactive, ノード固定及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。

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
11 月 26 日（金）	9:00 ～ 20:00 までサービス休止	月末処理
12 月 17 日（金）	9:00 ～ 20:00 までサービス休止	月末処理
1 月 28 日（金）	9:00 ～ 20:00 までサービス休止	月末処理

・ 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
11 月 25 日（木）8:30 ～ 17:30 まで 12 月 16 日（木）8:30 ～ 17:30 まで 1 月 27 日（木）8:30 ～ 17:30 まで

・上記期間中，Oakbridge-CX の debug, short, regular, interactive, prepost, ノード固定 及び 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
11 月 25 日（木）	9:00 ～ 22:00 までサービス休止	月末処理
12 月 15 日（水）	9:00 ～ 22:00 までサービス休止	月末処理
1 月 26 日（水）	9:00 ～ 22:00 までサービス休止	月末処理

・ 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水曜日）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
11 月 24 日（水）8:30 ～ 17:30 まで 12 月 14 日（火）8:30 ～ 17:30 まで 1 月 25 日（火）8:30 ～ 17:30 まで

・上記期間中，Oakforest-PACS の regular-flat，regular-cache，debug-flat，debug-cache，interactive-flat，interactive-cache 及び 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【 11 月 30 日 9:00 ～ サービス終了 】

○ 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
11 月 30 日（火）	9:00 ～ サービス終了	サービス終了

・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
11 月 29 日（月）9:00 ～ 11 月 30 日（火）9:00 まで

・上記期間中， Reedbush-H の h-debug,h-short，h-regular，h-interactive 及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 及び 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。

＊ Wisteria/BDEC-01，Oakbridge-CX 及び Oakforest-PACS において，新型コロナウイルス感染症拡大防止に配慮し，一部条件を変更して実施しております。詳細は Web ページ(https://www.cc.u-tokyo.ac.jp/guide/hpc/)をご覧ください。なお Reedbush-H においては引き続き中止しております。

システム変更等のお知らせ

(2021.9.1 － 2021.10.31 変更)

1. ハードウェア

1.1 Wisteria/BDEC-01 スーパーコンピュータシステム … なし

1.2 Oakbridge-CX スーパーコンピュータシステム … なし

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

1.4 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 8 (Wisteria/BDEC-01)

➢ Odyssey (Wisteria-O)

OpenFOAM	2106	(2021.09.22)

➢ ログインノード/プリポストノード

FFmpeg	4.2.4-2	(2021.09.22)

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

2.3 Red Hat Enterprise Linux 7, CentOS 7 (Oakforest-PACS)

ABINIT-MP	Open Ver.1 Rev.22	(2021.10.27)

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。

2.4 Red Hat Enterprise Linux 7 (Reedbush-H/L) … なし

3. その他

3.1 Reedbush-H/L サービス終了について

Reedbush-H/L スーパーコンピュータシステムは 2021 年 11 月 30 日 9:00 をもってシステムを停止し、すべてのサービスを終了致します。詳細については Web ページ、メール、スーパーコンピューティングニュースをご参照ください。

Reedbush-H/L サービス終了にあたっては以下の点にご注意ください。

- サービス終了後のスーパーコンピュータのご利用につきましては Wisteria/BDEC-01、Oakbridge-CX をご検討ください。
- 一般利用にて Reedbush-H/L をご利用の方は「トークン移行」を行うことが可能です。Wisteria/BDEC-01、Oakbridge-CX への移行をご検討の利用者様につきましては「トークン移行」も併せてご参考ください。「トークン移行」についての詳細は Web ページ(https://www.cc.u-tokyo.ac.jp/guide/application/transfer_token.php)をご参照ください。
- Reedbush において、11 月 30 日 9:00 の時点にて、残トークンがあったとしてもこれ以降残トークンを消費することはできません。

- 利用者データ等はサービス終了後すべて削除されます。センターでは利用者データ等のバックアップは取得しておりません。サービス終了後にデータの回復等は対応いたしかねますので、11 月 30 日 9:00 のサービス終了時までに、**必ず利用者の皆様にて、必要なデータ等についてバックアップの取得**をお願いいたします。

3.2 Oakforest-PACS サービス終了について

Oakforest-PACS スーパーコンピュータシステムは 2022 年 3 月末をもってすべてのサービスを終了致します。詳細については Web ページ、メール、スーパーコンピューティングニュースをご参照ください。

Oakforest-PACS サービス終了にあたっては以下の点にご注意ください。

- サービス終了後のスーパーコンピュータのご利用につきましては Wisteria/BDEC-01、Oakbridge-CX をご検討ください。
- 一般利用にて Oakforest-PACS をご利用の方は「トークン移行」を行うことが可能です。Wisteria/BDEC-01、Oakbridge-CX への移行をご検討の利用者様につきましては「トークン移行」も併せてご参考ください。「トークン移行」についての詳細は Web ページ(https://www.cc.u-tokyo.ac.jp/guide/application/transfer_token.php)をご参照ください。
- Oakforest-PACS の利用者様におきましては、2022 年に稼働する東京大学情報基盤センター「大規模共通ストレージシステム(Ipomoea-01)」へのファイル移行サービスについてのご案内を、最先端共同 HPC 基盤施設(JCAHPC)よりメールで送らせていただいております。詳細につきましてはメール「Oakforest-PACS「移行サービス」のご案内」をご確認ください。

2022 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

2022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、**2022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、2022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

Wisteria/BDEC-01、Oakbridge-CX のグループコース

2022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2022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、**2022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：2022年2月11日（金）（早めに手続きを行ってください）

※2022年度より、パーソナルコースは廃止され、グループコースのみとなります。

[1] 企業、若手・女性、大規模 HPC チャレンジ、萌芽共同研究公募課題「All for HPC」、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

Wisteria/BDEC-01、Oakbridge-CXのグループコース

2022年1月下旬に代表者の方に2021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。2022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、2021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。

Oakforest-PACSにおきましては、2022年度3月を持ちまして**サービスを終了となります。**
Wisteria/BDEC-01、またはOakbridge-CXへトークン移行にてお申込みください。

■利用申込期限：2022年2月11日（金）（早めに手続きを行ってください）

※2022年度より、パーソナルコースは廃止され、グループコースのみとなります。

３．問い合わせ先

〒277-0882
千葉県柏市柏の葉6-2-3（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

東京大学柏キャンパス一般公開 2021

スーパーコンピューティング部門

2021 年度の東京大学柏キャンパス一般公開は、新型コロナウイルスの感染拡大に配慮しオンラインにて 10 月 22 日(金)～29 日(金)に開催されました。情報基盤センターでは「Wisteira/BDEC-01 がもたらす新しい科学の夜明け」をテーマに、スーパーコンピュータシステムの概要説明、並列プログラミングやシミュレーションなどのミニ講座、スーパーコンピュータを利用した研究紹介など、スーパーコンピュータに関する様々な話題を紹介しました。10 月 22 日(金)から 24 日(日)にはウェブ会議システム(Zoom)を利用して講義を開催し質問にお答えしました。

■柏キャンパス一般公開 2021 情報基盤センター
https://www.cc.u-tokyo.ac.jp/kashiwa_open/
■東京大学柏キャンパス一般公開 2021(オンライン開催)
https://park-ssl.itc.u-tokyo.ac.jp/utkk/opc2021/
■東京大学柏キャンパス一般公開 2021 参加部局イベント案内
https://park-ssl.itc.u-tokyo.ac.jp/utkk/opc2021/oc50.php#b06

【開催期間中ののべ数】

柏キャンパス一般公開 2021 情報基盤センターウェブアクセス数 3,136 件
動画 視聴回数 710 回
(ウェブ会議システム)講義 参加者 509 名

2021年度 インターン・後期
東京大学情報基盤センター「若手・女性利用者推薦」採択課題

スーパーコンピューティングチーム

東京大学情報基盤センター（以下、センター）では、若手研究者（2021年4月1日時点において40歳以下、学生を含む）及び女性研究者（年齢は問わない）による、スーパーコンピュータ、大規模ネットワーク機器などの大型計算資源を使用した研究を対象とした公募型プロジェクトを実施しています。センターの教員による審査の上、年間で数十件の優れた研究提案課題を採択する予定です。採択された課題では申請した計算資源を無料で使用することができます。

前期・後期に募集を行う一般枠と、学部学生・大学院生を対象とし、主に夏期における利用を想定したインターン制度があります。一般枠（前期・後期）では、1人で行う研究課題を募集します。一般枠の課題は、1年または半年単位（後期は半年のみ）の実施となります。インターン制度では、1人で行う研究課題または2 人以上のメンバーで構成された研究グループで行う研究課題を募集します。

一般枠で新規に採択された課題のうち、特に優れた課題で「学際大規模情報基盤共同利用・共同研究拠点（JHPCN）」の萌芽型共同研究課題の条件を満たすものについては、本センターより同拠点萌芽型共同研究課題として推薦する予定です。同拠点共同研究課題審査委員会で審査の上、JHPCNの萌芽型共同研究課題としても採択された場合、毎年7月に開催されるJHPCNのシンポジウムにて発表の機会が与えられる場合があります。本制度に採択された課題は終了後、得られた成果をもとに、「学際大規模情報基盤共同利用・共同研究拠点（JHPCN）」の公募型共同研究（一般課題、国際課題、企業課題）等へと進展することが大いに期待されます。なお、インターン制度で採択された課題はJHPCN萌芽型共同研究課題としての推薦は行いません。

このたび、以下の基準による厳正な審査のうえ、2021年度インターンは1件の課題を、後期は8件の課題を採択いたしました。

- 本制度が提供する計算機システムを利用することで、学術的にインパクトがある成果を創出できると期待される提案を積極的に採択します。
- スーパーコンピュータの利用環境の改善に寄与すると期待されるソフトウェア開発に関する提案も歓迎します。
- 現状の環境にとどまらず、メニーコア、10万コア超える超並列環境など、将来の先端的なスーパーコンピュータ環境を目指した提案は特に歓迎します。

本制度の詳細は、以下のHPをご覧ください。
https://www.cc.u-tokyo.ac.jp/guide/young/

■ 2021年度 インターン 採択課題

課題名	階層型直交格子と再帰的なフィッティングによる回転翼の非定常空力予測
代表者名（所属）	菅谷 圭祐（東京大学大学院 工学系研究科）
利用システム名	Oakbridge-CX
実施期間	半年

近年、環境負荷を低減することが可能な電動・ハイブリッド航空機が注目されている。また都市部での新しい乗り物として、垂直離着陸な小型の航空機が注目されている。これらの革新的な航空機の空力設計では、数値流体力学（CFD）を用い、機体周りの複雑な流れを予測することが重要である。しかし、航空機や回転翼は形状が複雑であり、CFD を用いた流れの予測に必要な計算格子を生成することが解析のボトルネックになる。そこで、複雑な形状に対し自動で格子を生成可能な直交格子法が注目されている。本研究の目的は、回転翼の非定常乱流解析に向けた、自動格子生成が可能な新しい手法を開発することである。計算には、三次元複雑形状周りに自動で物体適合格子を生成可能で、支配方程式である保存則が厳密に満足される、直交格子法のひとつである RecursiveFitting 法を用いる。また物体の回転を再現するため、移動格子法を組み合わせる。加えて、壁面近傍の乱流境界層を精度良く予測するため、壁関数を組み合わせて解析を行う。

■ 2021年度 後期 採択課題

課題名	Feedforward 型の NeuralNetwork を用いた多体電子系波動関数の基底状態の解法
代表者名（所属）	乾 幸地（東京大学大学院 工学系研究科）
利用システム名	Reedbush-H Reedbush-L Wisteria-A
実施期間	半年
近年、深層学習に代表されるニューラルネットを用いて量子多体系の波動関数を近似良く求めようという研究が盛んに行われている。フェルミオン系におけるこれまでの研究の多くは、反交換関係を表現するために行列式やパフィアンなどによる一体の波動関数と、相互作用効果を記述するニューラルネットを組み合わせて用いている。これまでの研究では、一体の波動関数部分に対する研究はそこまで盛んに行われてこなかった。そこで我々は畳み込み型ニューラルネットと行列式を組み合わせることで、精度を向上させることを目的に、系統的な計算を実行する。	

課題名	ダイマー粒子を用いたガラスの Johari-Goldstein beta 緩和の理解
代表者名（所属）	白石 薫平（東京大学 大学院総合文化研究科）
利用システム名	Oakbridge-CX
実施期間	半年
液体を急冷すると、融点で結晶化を避けて過冷却液体状態になる。この状態では、僅かな温度低下によって系の緩和時間が急激に増大し、仕舞には実験の時間スケールを越え、分子の構造が乱雑なまま凍結したガラスが生じる。この過冷却液体が示す遅い動力学は、統計力学の通常の理解が適用できない現象である。様々な過程から成る緩和動力学を理解することは、ガラス転移の理解に直結する重要な問題である。本研究では、Johari-Goldstein（JG）beta 緩和過程に着目する。これは、他の主要過程である alpha 緩和よりも短時間に現れる過程であり、分子性ガラスの実験で普遍的に観測されているが、数値的な研究は限定的である。それは、多くのシミュレーションが依拠してきた等方的な球によるモデルガラス系には、JGbeta 緩和過程が出現しないからである。この研究では、近年 JG beta 緩和を示すことが報告されたダイマー粒子系の分子動力学シミュレーションを行う。そして、レプリカ交換法や isoconfigurational アンサンブルといった計算物理学の手法を駆使し、JGbeta 緩和の微視的な運動と発生メカニズムに迫る。	

課題名	グラフ構造を入力として応力分布を出力とする GAN ベースの有限要素法の Surrogate モデルの研究
代表者名（所属）	中井 優（東京大学 工学部）
利用システム名	Wisteria-A
実施期間	半年

有限要素法の深層学習による Surrogate モデルとして、メッシュを入力データとして、メッシュの各ノードにおける応力分布を出力とするような、GAN をベースとした、有限要素法による応力分布の生成過程をニューラルネットワークによってモデル化した生成モデルを提案する。提案手法は、深層学習の持つモデルの表現力と計算量のトレードオフの選択の自由度が大きいという特徴を利用して、有限要素法による解析よりも高速な応力分布の推論が可能にする。さらに、グラフのような大きさ (ノード数) の変化する入力に適用することが困難であるという問題に対して、GNN と呼ばれるネットワーク構造を採用してグラフ構造の扱いを可能にすることで、また入出力のパターンを学習する識別モデルではなくデータの生成過程そのものを学習する生成モデルを利用することで、従来の識別モデルを利用した深層学習による Surrogate モデルを上回る性能を持ち、それらのモデルでは達成できなかった応力分布そのものの生成を可能にするような、新たな Surrogate モデルを構築する。

課題名	公共データを活用した転写因子結合ダイナミクスの解析
代表者名（所属）	植野 和子（国立国際医療研究センター）
利用システム名	Oakbridge-CX
実施期間	半年

本研究課題では申請者らが過去の研究で発見した、RNA ポリメラーゼ II と共役してゲノム DNA 上を「動く」転写因子と同じ性質を持つ分子が他にどれくらい存在するのかを明らかにするため、公共データベースに登録された大量の次世代シークエンス解析データを用いて、各々の転写因子の結合動態を網羅的に解析する。具体的には、米国のポストゲノム解析プロジェクトである ENCODE(ENCyclopedia Of DNA Elements) において収集されたマウスおよびヒトの転写因子の ChIP-seq データを収集し、遺伝子配列上における分布を解析することで、その挙動を明らかにしていく。またその際に、同サンプルから得られた遺伝子発現データを併用して解析することで、転写制御機能と分子運動を関連づけた解析を行う。以上の解析から、遺伝子配列上を動く新規の転写因子を同定すると共に、転写因子の構造と分子運動、機能を関連づける普遍性のある法則を見出すための基盤的データを創出する。

課題名	波形インバージョンによる地球マントル最下部の地震波速度構造推定
代表者名（所属）	大鶴 啓介（東京大学大学院 理学系研究科）
利用システム名	Oakforest-PACS
実施期間	半年

地球マントルの最下部数百 km の領域は、固体岩石で構成されるマントルと高温の液体鉄合金である外核が接する核-マントル境界（CMB）の直上に位置し、熱及び化学組成の不均質を生み出す主要な領域であると考えられる。これまでに行われた地震波による構造推定からは、アフリカ下と太平洋下の 2 箇所に巨大な低速度異常域（LLSVP）が存在することが知られている。しかし、これが複数の熱的な上昇流の束なのか、化学組成の異なる物質の塊なのかわかっておらず、その判別にはより高い解像度の構造推定が必要である。詳細な構造推定を行う際に必要となる正確な理論波形についても、短周期までの計算には多くの計算資源が必要となる。そこで本課題では、地震波形に含まれる情報を余すことなく活用することでより詳細な構造推定を可能にする波形インバージョン手法と、スペクトル要素法により理論波形を計算する SPECFEM3D_GLOBE というソフトを使用して、LLSVP 内部の詳細な地震波速度構造を推定することを目指す。

課題名	地震波形インバージョンによるマントル最下部の S・P 波速度構造同時推定—地球深部の熱・化学進化の理解に向けて—
代表者名（所属）	佐藤 嶺（東京大学大学院 理学系研究科）
利用システム名	Oakforest-PACS
実施期間	半年

地球の核-マントル境界 (CMB) は、固体岩石のマントルと液体鉄合金の外核が接する地球内部の最も主要な熱・化学組成境界であるため、CMB 直上数 100km(D″ 領域) は地球の熱・化学進化の理解に重要な領域である。D″ 領域では地温勾配とマントル組成のソリダスが近いために部分溶融に伴う化学分化を起こすマグマが定常的に発生する可能性が高く、また典型的な沈み込み領域下においては海洋プレートの沈み込みに伴う化学組成不均質が形成されると考えられている。従って観測情報から化学組成異常のサイズと度合いを制約することが重要である。しかし、地震波速度不均質を温度・化学組成効果に定量的に分離するには S・P 波速度構造を同程度の解像度で同時に推定する必要があったが、データセットの質や種類が異なる等課題があった。また正確な理論地震波形の計算も必要であるが、短周期までの計算は計算資源の問題から困難であった。本申請研究では地震波形に含まれる情報を余すことなく活用できる波形インバージョン手法と、スペクトル要素法による理論地震波形計算ソフトウェア SPECFEM3D_GLOBE を用いて D″ 領域の S・P 波速度構造を同時推定する。

課題名	分子動力学計算を用いた DNA 分解酵素の失活メカニズムの解明
代表者名（所属）	大滝 大樹（長崎大学 生命医科学域（医学系））
利用システム名	Oakbridge-CX
実施期間	半年

酵素は生物が生きていく上で必要な消化・分解・吸収・代謝などのあらゆる過程に関係する。酵素における変異の多くは酵素作用を失活させることが分かっているが、その分子的なメカニズムについては不明な点が多い。本研究では、免疫系疾患に関係する DNA 分解酵素に着目し、分子動力学シミュレーションにより多くの変異型を調べる。相互作用などの解析から、「変異の影響がどのような経路をたどり、最終的にどこに影響を及ぼすか」を分子論的に明らかにすることを目的とする。

課題名	ディープラーニングによる高精度マルウェア分析
代表者名（所属）	三橋 力麻（東京大学大学院 情報理工学系研究科）
利用システム名	Wisteria-A
実施期間	半年

サイバーセキュリティの現場では、セキュリティアナリストとマルウェアの戦いが続いている。マルウェアの構造や動作は従来よりも複雑さを増しているため、セキュリティアナリストは、新しいマルウェアを入手すると、その一つ一つを丁寧に解析せざるを得ない状況にあり、その作業負担は膨大になっている。その一方で、マルウェアの攻撃はますます高度になっており、日々発見される新しい脆弱性や、検知動作の検知を回避する仕組みを取り込みながら、次々に新しいマルウェアが量産されている。ただし、マルウェアは一度開発されると、検知を回避するためプログラムの構造の一部を変更した「亜種」が数多く生産される特徴がある。そこで、未知のデータセットの中から、マルウェア亜種を自動的に既知の仲間に分類できれば、セキュリティアナリストによるマルウェア解析の負担を大きく減らすことができる。本課題では、ディープラーニングを活用した高精度なマルウェア分析を実現し、セキュリティアナリストによるマルウェア解析作業の補助や新たなマルウェア亜種の速やかな検出などマルウェアによる被害を早期に防止するためのシステムを実現する。

以上

スーパーコンピュータシステム「大規模 HPC チャレンジ」採択課題のお知らせ

1. はじめに

Wisteria/BDEC-01、Oakbridge-CX、Oakforest-PACS スーパーコンピュータシステムでは、「大規模 HPC チャレンジ」を実施しています。「大規模 HPC チャレンジ」は、スーパーコンピュータシステムがもつ最大規模のノード数を、最大 24 時間・1 研究グループで計算資源の占有利用ができる公募型プロジェクトです[1]。(※)

課題審査委員会による厳正な審査の結果、以下の課題を採択しましたのでお知らせいたします。

※ 新型コロナウイルス感染症拡大防止に配慮し、通常から一部条件、実施時間等を変更し実施しています。

- 実施時間は 8 時間（実施日当日 9:00～17:00）
- Oakforest-PACS スーパーコンピュータシステムにおいては、flat モード（4,200 ノード）、cache モード（3,200 ノード）の各設定の上限までの利用とします（各月 flat モード 1 件、cache モード 1 件の最大 2 件まで受入可能、ただし 1 グループで flat モード、cache モード両方利用することも可能）
- Wisteria/BDEC-01 スーパーコンピュータシステムにおいては、Wisteria-O（Odyssey）の 6,144 ノード（294,912 コア）又は Wisteria-A（Aquarius）の 36 ノード（288 基）、或いは両方の利用とします。

2. 採択課題

システム：Oakbridge-CX

募集期間：2021 年度 第 2 回再募集　2021 年 7 月 1 日～2021 年 8 月 2 日

1 件の応募があり、以下の課題を採択しました。

採択課題一覧

課題名	三次元強震動シミュレーションとリアルタイムデータ同化の融合
代表者名（所属）	中島　研吾（東京大学情報基盤センター）
本研究では，JDXnet（Japan Data eXchange network）によって得られる地震観測データを利用したリアルタイムデータ同化と高精度な三次元強震動シミュレーションの融合による計算を，Oakbridge-CX システム（OBCX）を使用して実施する。具体的には，新潟県中越沖地震（2007 年 7 月 16 日）における各地地震計における観測データを，外部から OBCX の外部接続ノードに取り込んでフィルタリング処理を施し，古村等によって開発された三次元強震シミュレーション・リアルタイムデータ同化融合コード Seism3D/OpenSWPC-DAF を実行する。本研究では，OBCX のような外部接続ノードを有するシステム上で，Seism3D/OpenSWPC-DAF を使用したシミュレーション・データ同化融合を実施することが，地震発生時の正確な波動伝播予測，適切な避難情報提供に有効であることを示し，リアルタイム災害予測システムの研究開発に資する。	

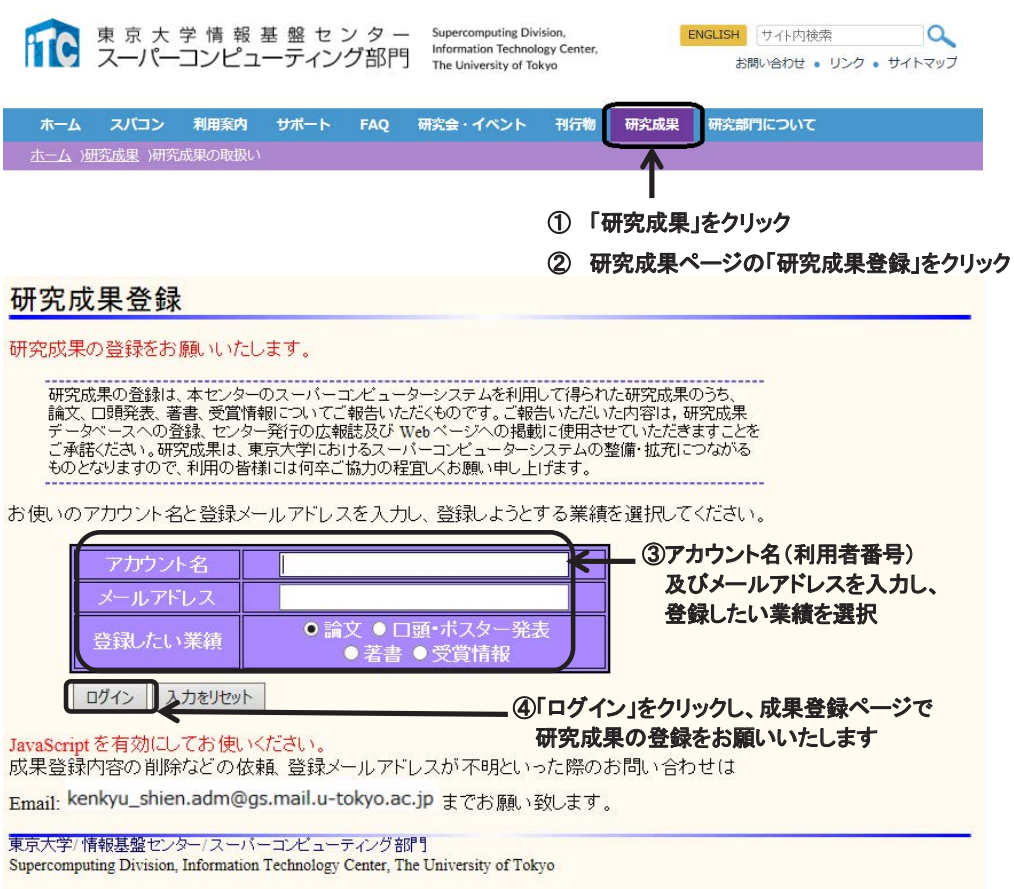
[1]「大規模 HPC チャレンジ」
https://www.cc.u-tokyo.ac.jp/guide/hpc/

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。

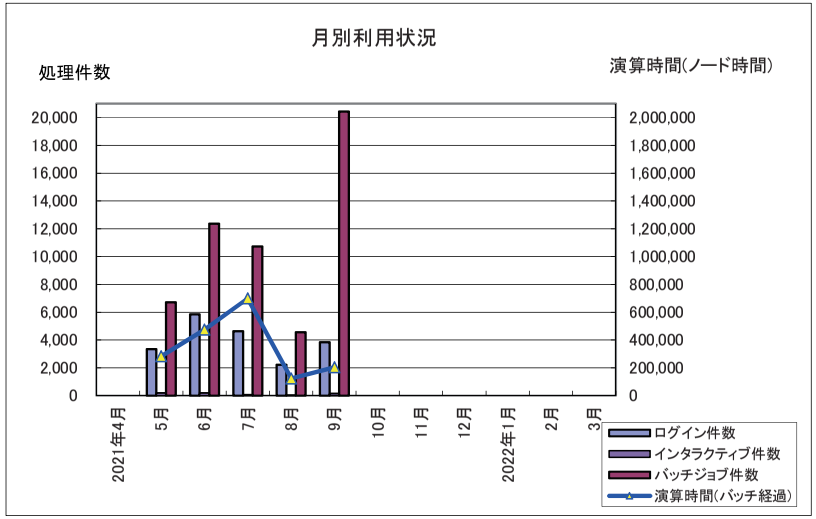


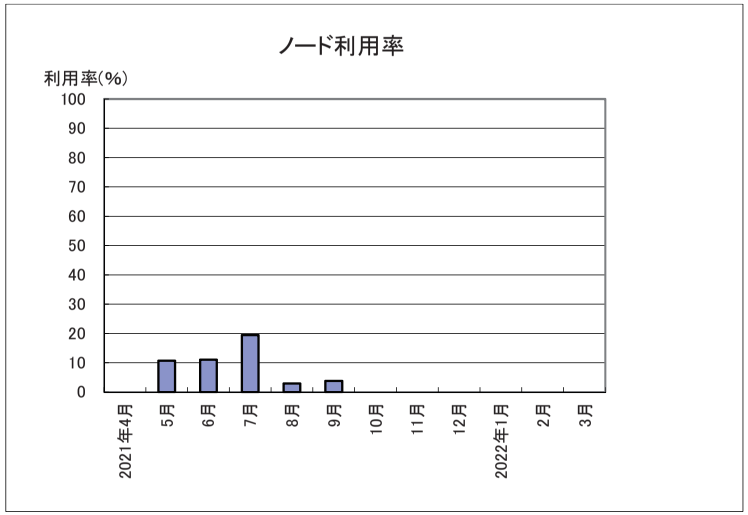
8・9月のジョブ統計

1. Wisteria/BDEC-01スーパーコンピュータシステム（Odyssey）ジョブ処理状況 （RedHat Enterprise Linux 8）

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		
2021年5月	860	206	3,348	54	188	6,718	13,623	148	66,121	697.32	40	35	282,715	822.0	10.7
6月	974	270	5,845	97	186	12,367	23,491	361	189,018	1,398.60	122	36	473,446	842.5	11.0
7月	1,103	229	4,638	66	64	10,730	24,723	438	211,696	1,670.10	74	10	698,405	1,492.2	19.4
8月	939	147	2,212	82	28	4,551	9,866	383	128,337	2,218.48	255	6	123,322	219.4	2.9
9月	1,058	196	3,835	76	150	20,431	19,769	434	246,063	1,944.79	281	40	204,279	289.0	3.8
合計			19,878	375	616	54,797	91,472			7,929	772	127	1,782,167		

・試運転は、2021年5月14日より開始。正式サービスは、2021年8月2日より開始
・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
・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



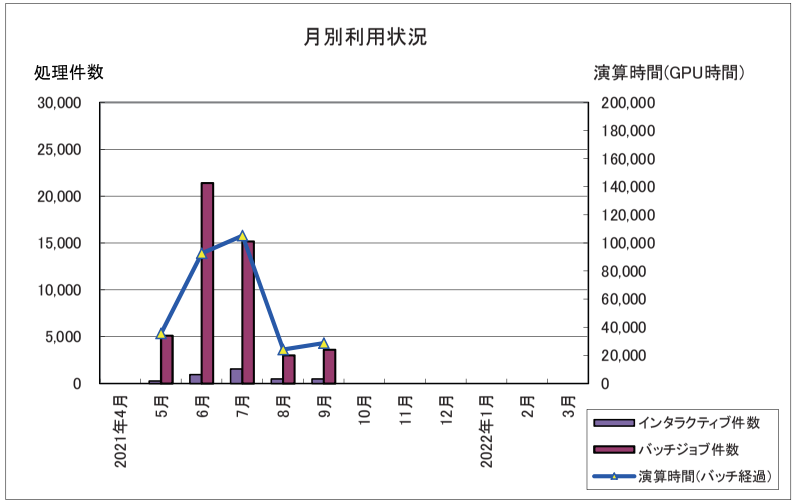


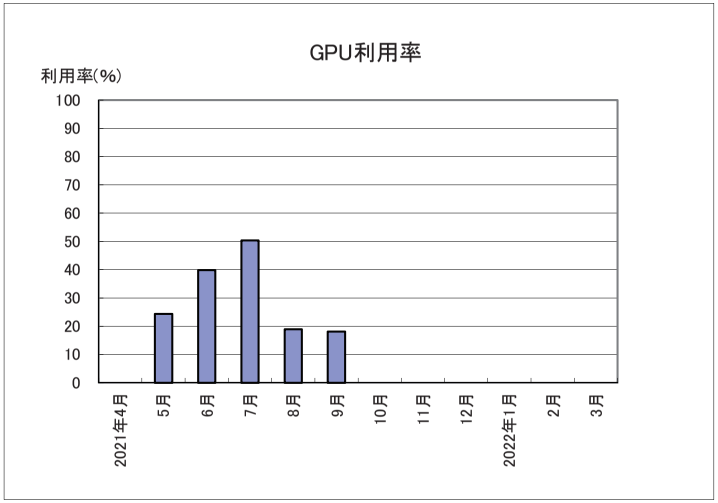
2. Wisteria/BDEC-01スーパーコンピュータシステム（Aquarius）ジョブ処理状況 （RedHat Enterprise Linux 8）

年月	処理件数		演算時間 [GPU時間](経過時間)		平均GPU利用数(GPU)	GPU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		
2021年5月	252	5,095	158	35,645	87.5	24.3
6月	953	21,394	1,245	92,643	143.3	39.8
7月	1,540	15,155	1,531	105,239	180.9	50.3
8月	466	3,004	470	24,188	67.9	18.9
9月	480	3,599	358	28,702	65.1	18.1
合計	3,691	48,247	3,762	286,417		

・試運転は、2021年5月14日より開始。正式サービスは、2021年8月2日より開始
・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Wisteira/BDEC-01(Odyssey) と共通。

・GPU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1GPUが100%動作したと仮定した場合の利用GPU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
・GPU利用率： サービスGPUに対する利用比率。 計算式＝GPU利用数÷サービスGPU数×100

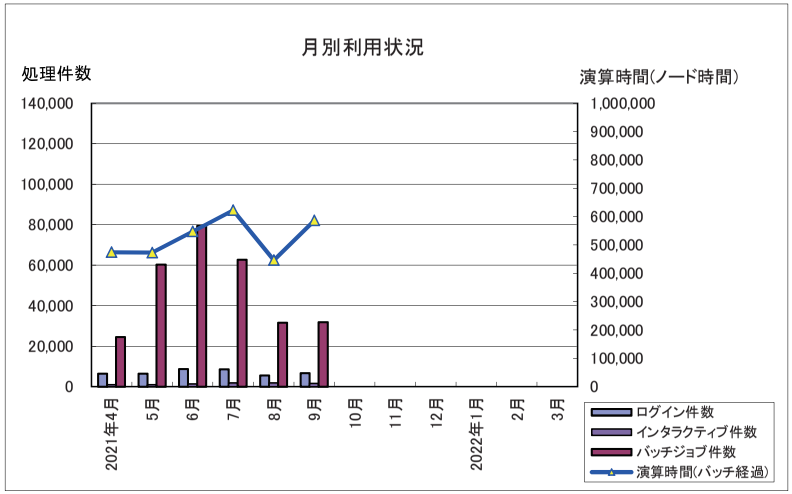


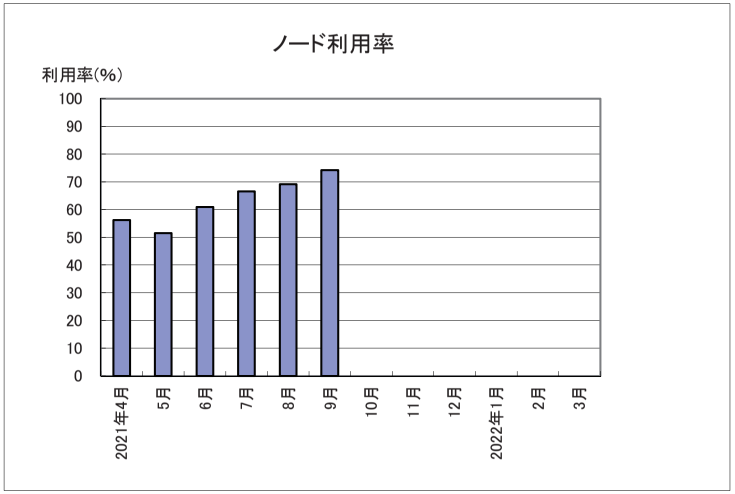


3. Oakbridge-CX スーパーコンピュータシステムジョブ処理状況（Red Hat Enterprise Linux 7、CentOS 7）

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	/work		プリポスト	インタラクティブジョブ	バッチジョブ		
2021年4月	1,070	323	6,405	2	927	24,498	27,351	913	1,339,482	1,242.42	0	385	474,088	769.1	56.2
5月	1,196	299	6,395	0	982	60,233	38,699	874	1,292,853	8,996.00	0	992	472,558	704.1	51.5
6月	1,087	358	8,649	43	1,356	79,376	48,740	993	1,357,480	2,853.51	1	1,093	547,424	832.8	60.9
7月	1,132	366	8,545	42	1,860	62,733	57,857	1,017	1,430,169	16,807.20	2	674	622,202	909.5	66.5
8月	1,141	278	5,494	59	1,814	31,559	22,299	1,060	1,499,355	4,136.73	5	592	446,153	945.0	69.1
9月	1,067	284	6,700	17	1,585	31,851	32,068	1,095	1,592,894	2,314.47	0	592	586,073	1,015.7	74.2
2020年9月	1,535	299	11,804	48	1,152	80,846	39,017	1,130	1,539,134	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	1,160	1,700,805	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	1,172	1,878,875	15,583.41	17	980	716,567	1,086.7	79.9
12月	1,214	241	11,452	56	1,475	75,087	43,511	1,191	1,872,689	9,074.40	253	860	768,743	1,127.9	82.9
2021年1月	1,105	215	10,238	49	1,601	64,441	53,307	1,021	1,891,719	3,322.22	253	982	754,704	1,110.5	81.7
2月	1,110	161	7,326	117	2,215	40,336	28,469	1,049	1,963,710	2,864.16	398	893	591,306	941.2	69.2
3月	1,080	170	7,803	67	1,373	63,001	35,810	1,115	1,502,680	2,660.82	387	671	591,775	854.0	62.8
合計			106,148	567	19,736	668,414	522,355			76,240	1,835	9,543	7,303,785		

・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
・2020年9月分は合計に含まない
・2021年4月分より、集計方法を変更いたしました。下線部分が前回から変更となっている箇所となっております。
・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



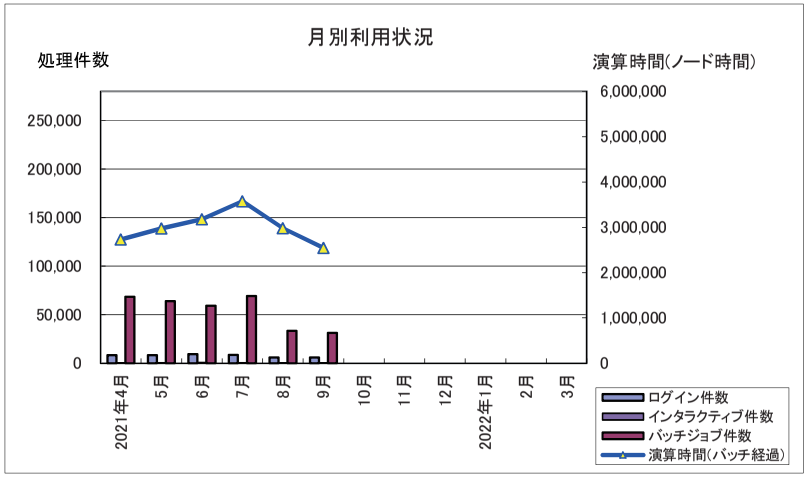


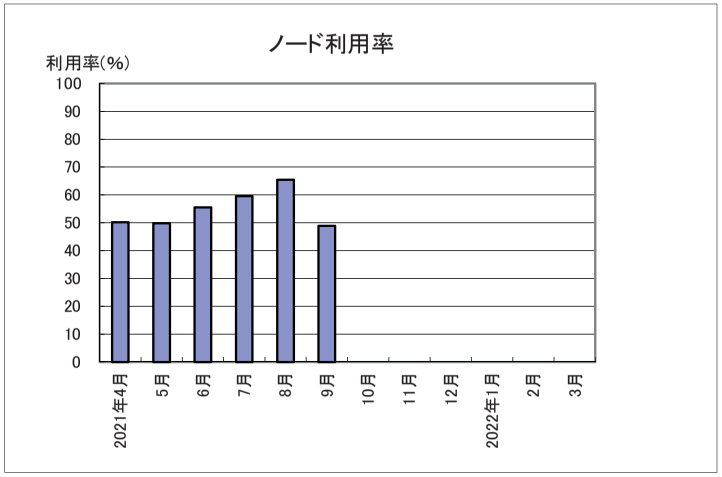
4. Oakforest-PACSスーパーコンピュータシステムジョブ処理状況 （RedHat Enterprise Linux 7、CentOS 7）

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		
2021年4月	1,690	450	8,322	128	181	68,373	47,950	2,618	6,233,823	3,528.98	310	65	2,731,333	4,117.0	50.2
5月	1,732	347	8,252	192	288	63,829	63,997	2,216	4,729,782	2,036.24	411	78	2,977,248	4,089.8	49.8
6月	1,397	378	9,348	274	511	59,097	66,215	2,236	4,830,259	2,088.30	470	92	3,179,605	4,557.0	55.5
7月	1,329	380	8,428	193	415	69,240	74,087	2,199	4,947,221	2,050.97	420	203	3,574,862	4,895.8	59.6
8月	1,336	308	5,994	189	237	33,416	28,905	2,177	5,049,162	5,565.78	523	160	2,981,494	5,366.4	65.4
9月	1,373	319	5,891	184	171	31,112	38,671	2,246	5,093,072	1,940.11	430	108	2,547,260	4,011.9	48.9
9月	1,732	420	9,811	293	795	52,222	56,925	2,441	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	2,552	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	2,535	6,618,475	3,297.48	516	125	4,468,068	6,400.5	78.0
12月	1,789	471	13,407	419	375	99,699	66,020	2,648	6,634,466	6,273.23	630	149	4,837,811	6,633.6	80.8
2021年1月	1,804	475	12,636	495	556	210,638	84,101	2,687	6,512,605	3,174.92	787	233	5,112,137	6,993.7	85.2
2月	1,786	431	11,384	314	282	79,481	78,257	2,787	6,581,603	5,791.25	546	197	4,733,131	7,166.3	87.3
3月	1,727	472	12,986	384	302	271,840	102,642	2,760	6,629,043	6,329.17	706	174	5,484,594	7,523.7	91.7
合計			121,820	3,214	4,038	1,105,076	808,431			48,537	6,147	2,294	47,065,467		

・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
・2020年9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＝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

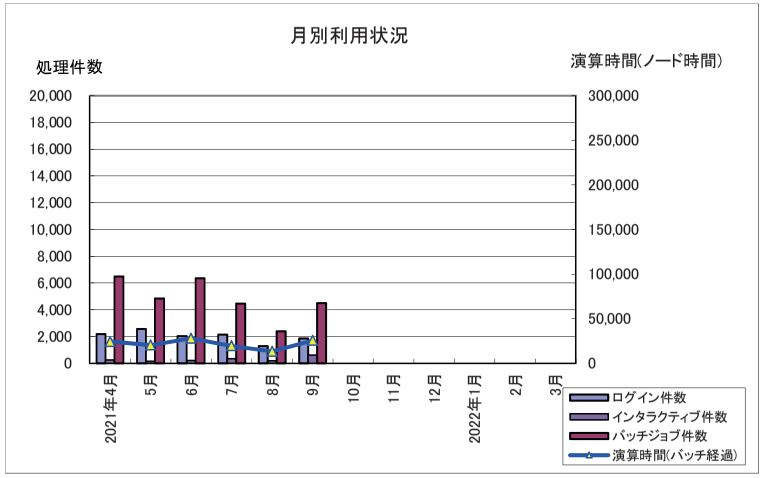


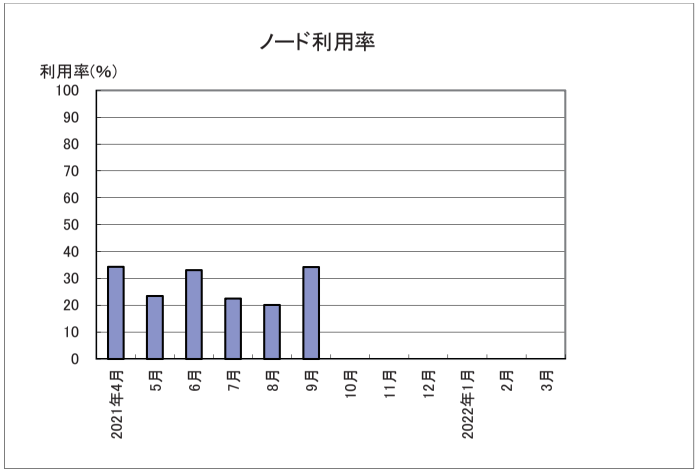


5. Reedbushスーパーコンピュータシステム（Reedbush-H）ジョブ処理状況 （RedHat Enterprise Linux 7）

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		
2021年4月	986	127	2,183	19	253	6,489	3,140	146	192,825	12.31	2	284	24,386	41.1	34.3
5月	644	116	2,571	0	149	4,839	3,857	134	203,491	13.35	0	179	20,465	28.0	23.4
6月	592	102	2,022	0	200	6,355	3,567	135	200,432	2.60	0	200	27,992	39.6	33.0
7月	587	111	2,149	2	348	4,452	3,564	142	210,761	4.95	3	285	19,598	27.0	22.5
8月	585	91	1,285	1	195	2,384	2,378	137	218,311	5.10	2	200	13,374	24.2	20.1
9月	610	97	1,862	0	608	4,502	3,302	136	222,755	2.59	0	635	25,653	41.0	34.2
2021年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
12月	1,094	236	5,252	0	1,545	16,439	8,449	180	349,615	6.64	0	1,979	47,463	67.1	55.9
2021年1月	1,112	195	4,585	0	1,010	12,940	7,523	181	356,486	7.35	0	1,173	44,632	62.2	51.8
2月	1,057	188	3,758	0	809	12,800	5,831	188	354,254	16.00	0	954	56,532	86.5	72.1
3月	1,056	176	4,140	0	776	7,345	5,273	189	316,348	13.20	0	920	73,742	102.4	85.4
合計			40,027	22	7,479	99,009	60,722			110	7	8,457	429,663		

・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
・2020年9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



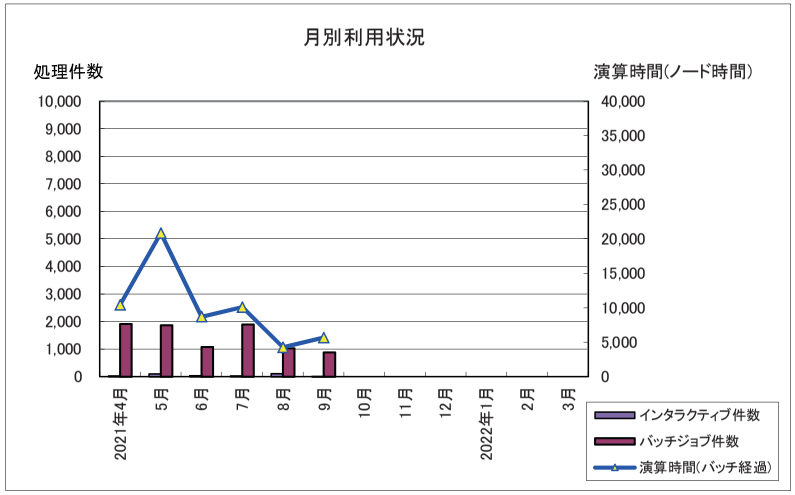


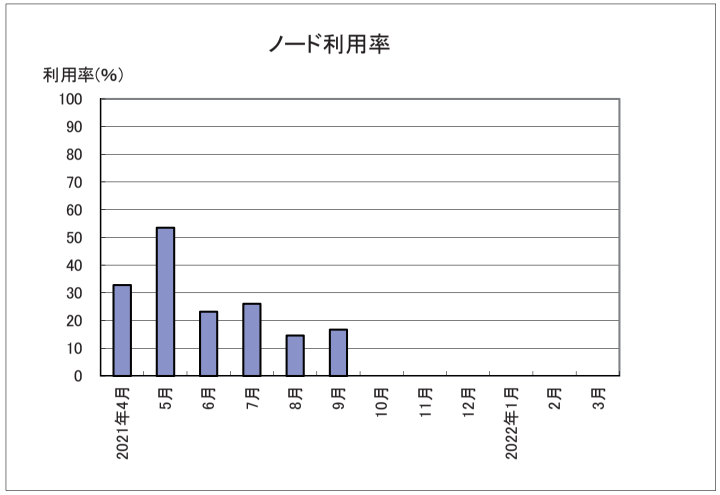
6. Reedbushスーパーコンピュータシステム（Reedbush-L）ジョブ処理状況 （RedHat Enterprise Linux 7）

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		
2021年4月	19	1,913	3	10,413	17.4	32.8
5月	95	1,867	45	20,833	28.3	53.5
6月	34	1,075	70	8,689	12.3	23.2
7月	25	1,887	44	10,081	13.8	26.0
8月	105	1,031	87	4,249	7.7	14.6
9月	1	874	0	5,657	8.8	16.7
2020年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
12月	105	2,734	106	21,208	28.9	53.6
2021年1月	100	3,171	46	26,404	35.9	66.7
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	768	22,058	714	211,409		

・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。
・2020年9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





Numerical simulation of deep-water oil blowouts

Daniel Cardoso Cordeiro

大阪大学基礎工学研究科

1. Introduction

In 2010, the largest offshore blowout of all time happened in the Gulf of Mexico, USA. The accident became known as the Deep-water Horizon oil spill in which around 4.9 million oil barrels were spilled into the ocean, and chemical dispersants were applied in a large scale into the subsea as a way to treat the oil blowout plume [1].

Given the possible negative outcomes that subsea injection of chemical dispersants may cause to the environment, it is necessary to assess clearly, whether this technique is specifically adequate for a certain accident scenario before its application. However, since real-scale experiments are unpractical due to their obvious environmental risk, numerical models and laboratory-scale experiments are the only available tools to investigate the blowout phenomenon. Therefore, the development of an accurate numerical blowout model would represent a valuable asset in the decision-making process for the remediation of such accidents with chemical dispersants.

A number of challenges stand in the way of achieving a more exact blowout model. Among them are the sensitivity of model predictions to complex chemical and biological phenomena, the need for a coupling of ocean circulation models with the near-field blowout plume dynamics and the accuracy of the oil/gas droplet size distribution (DSD) [2]. Particularly, the droplet size distribution (DSD) represents a serious threat to the preciseness of the spilled oil path inside the sea as a small variation in the droplet diameter results in a large discrepancy in the distance that the oil could travel both laterally and vertically inside the ocean. This could induce the response team to choose unsuitable remediation techniques with further damage to the environment.

In this research, we propose the use of a LES (Large Eddy Simulation) for turbulence with a hybrid Euler-Euler and volume of fluid (VOF) models to accurately predict the DSD by calculating the actual breakup and coalescence caused by the atomization of the turbulent oil jet into a quiescent water media.

2. Numerical methods

To simulate the oil blowout phenomena, a hybrid Eulerian multifluid solution framework with a volume of fluid (VOF) sharp interface capturing algorithm was used. It improves the accuracy of the traditional Euler-Euler method without the computational burden of coupling an entire interface-capturing model [3]. The present model was validated against a series of laboratory-scale oil blowout experiments. The deterministic approach based on the large-eddy simulation does not rely on the input of any empirical

data nor calibration parameters, increasing the reliability to the real-scale prediction of the DSD in deep-water blowouts.

The governing equations solved are the Continuity (Eq. 1) and Navier-Stokes (Eq. 2) equations:

$$\frac{\partial \alpha_k}{\partial t} + \boldsymbol{u_k}.\nabla \alpha_k = 0 \tag{1}$$

$$\frac{\partial(\rho_k \alpha_k \boldsymbol{u_k})}{\partial t} + (\rho_k \alpha_k \boldsymbol{u_k}.\nabla)\boldsymbol{u_k} = -\alpha_k \nabla p + \nabla.(\mu\alpha\nabla \boldsymbol{u_k}) + \rho_k \alpha_k \boldsymbol{g} + \boldsymbol{F_{D,k}} + \boldsymbol{F_{S,k}} + \boldsymbol{F_{vm,k}} \tag{2}$$

where u is the velocity, is the volume fraction, t is time, ρ is the density, p is the pressure, g is the gravity acceleration, $F_{S,k}$ is the surface tension force, $F_{D,k}$ is the drag force, $F_{vm,k}$ is the virtual mass force and the subscript k indicates the fluid phase. The Euler-Euler method was used to model the multiphase flow coupled with a Schiller-Naumann drag model. The open-source OpenFOAM v4.0 was used for the calculations.

Table 1: Configuration of the cases: dispersants to oil ratio, DOR; surface tension, σ [mN/m]; inlet diameter, D [mm], inlet velocity, U_{in} [m/s]; the jet Reynolds number, Re; the jet Weber number, We; the oil-water viscosity ratio, ν_o/ν_w; the oil-water density ratio, ρ_o/ρ_w ; simulated time interval, ΔT.

	Case	DOR (%)	σ	D	U_{in}	Re	We	ν_o/ν_w	ρ_o/ρ_w	$\Delta T U_{in}/D$
——	1	0%	28.3	14	0.65	1076	179	8.46	0.856	428
– – – –	2	1%	0.6	14	0.65	1076	8438	8.46	0.856	428
·····	3	0%	28.3	9	0.65	691	115	8.46	0.856	667
—▲—	4	0%	28.3	14	0.16	264	11	8.46	0.856	114
– –△– –	5	1%	0.6	14	0.16	264	511	8.46	0.856	114
—●—	6	0%	28.3	14	0.65	2151	179	4.23	0.856	428
– –○– –	7	0%	28.3	14	0.65	4302	179	2.11	0.856	428
—■—	8	0%	28.3	14	0.65	1076	125	8.46	0.600	428
– –□– –	9	0%	28.3	14	0.65	1076	84	8.46	0.400	428

The domain was modeled based on the experimental data [4], as a 1.0 m height cuboid tank with a 0.3 m per 0.3 m width. Oil flows through a round inlet at the center of the tank initially filled with water. The DSD was computed a posteroiri by isolating the droplet in the isosurface alpha = 0.1.

3. Results and Discussion

The effects of different parameters over the jet atomization in the DSD were showed using non-dimensional numbers and figures of the jet behavior.

The velocity inlet altered the regime in different cases: 1-2 were atomization and 4-5 were second wind induced breakup. Both the regimes showed very different characteristics regarding the droplet size distribution and the velocity of the particles as per Fig. 1.

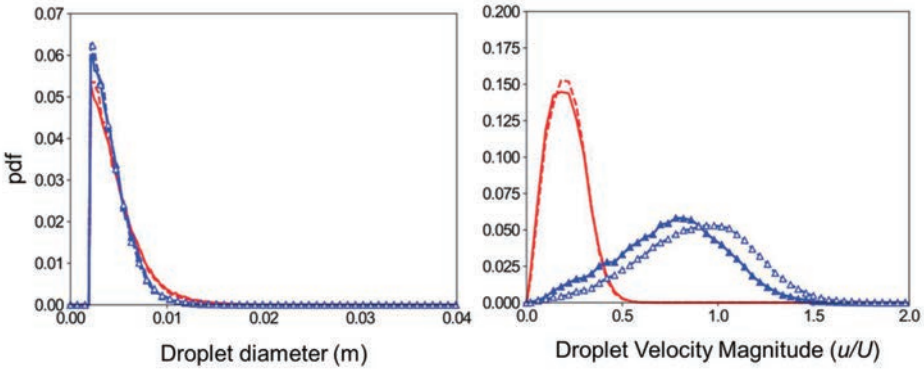


Figure 1: Droplet diameter distribution (left) and droplet velocity distribution (right) for cases 1, 2, 4 and 5.

From the droplets local information, it was possible to conclude that for the case of surface tension reduction, the droplet diameter was slightly reduced while the droplet velocity distribution behaved differently for the cases in different breakup regimes. As Fig. 1 also shows, in the atomization regime (Cases 1 and 2), the droplet velocity was reduced and increased in the second wind induced regime (Cases 4 and 5).

The inlet diameter reduction did not change the flow regime; however, it intensified Kelvin-Helmholtz instabilities near the inlet and narrowed the droplet size distribution (Fig. 2).

Figure 2: Effect of the diameter inlet reduction in the snapshots of the isosurface alpha = 0.1 for cases 1 (left) and 3 (right). The details show the Kelvin-Helmholtz instabilities.

The cases with variation in the density (1, 8, 9) and the viscosity ratio (1, 6, 7), showed in Figs. 5 and 6 respectively, illustrates the impact of the near-field instabilities in both the droplet size distribution and in the droplet velocity distribution. The viscosity reduction decreased the Kelvin-Helmholtz effect while keeping similar DSD, which is caused by the increase in the droplet Re. The density reduction acted to enhance the instabilities near the inlet, inducing breakup much

earlier than in the higher density cases.

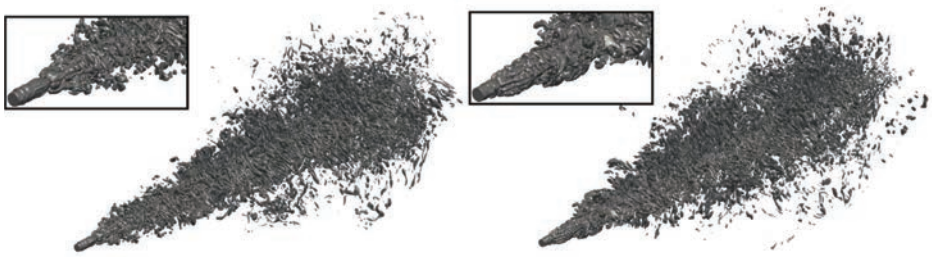


Figure 3: Effect of the viscosity ratio reduction in the snapshots of the isosurface alpha = 0.1 for cases 1 (left) and 6 (right). The details show the Kelvin-Helmholtz instabilities near the inlet.

Figure 4: Effect of the density ratio reduction in the snapshots of the isosurface alpha = 0.1 for cases 1 (left) and 8 (right). The details show the Kelvin-Helmholtz instabilities near the inlet.

4. Conclusion

Using the hybrid volume-of-fluid/Euler-Euler model with a large eddy simulation turbulent model was effective in modeling the laboratory-scale oil blowout in a water tank. The effects of different physical parameters on the atomization were complex and it shows the difficulty in understanding how the distribution of droplets are affected by this phenomenon.

Further research is necessary to create and overarching theory that could explain the jet breakup in a more clear and concise way.

References

[1] Kujawinski et al., Environ. Sci. Technol., vol. 45, pp. 1298-1306, 2011.

[2] Socolofsky et al., Oceanography, vol. 29, pp. 64-75, 2016.

[3] Wardle, Weller, International Journal of Chemical Engineering, vol. 13, 2013.

[4] Miyata et al., JIME, vol. 51, pp.367-374, 2016.

動的膜の分子動力学計算

小山 志穂里
（株）豊田中央研究所

1．はじめに

マイクロ・ナノスケールの輸送システムは，バイオやエネルギー貯蔵などの幅広い分野への応用が期待されている技術で，小規模な流れに特有の現象を利用した様々なデバイスが提案されている。一般的なマイクロ・ナノ流体デバイスでは，固液界面に形成された電気二重層に電場や化学ポテンシャルなどの外場を印加することで動電現象が引き起こされ，界面において液体の流れを伴う。マイクロ・ナノスケールのシステムでは，体積に対する表面積の割合が大きくこのような現象が促進されるため，小規模な輸送システムの原動力として応用することが研究されている。

代表的な動電現象としては，外部電場により誘起される電気浸透流[1]が挙げられる。電気浸透流は電場によって直接誘起されるため，機械的な可動部がなく，ポンプを小型化するための重要な技術として認識されている。これまでの研究では，ナノサイズのシステムで流量を予測するための理論解析[2-5]や分子動力学（MD）シミュレーション[6-9]，ガラス多孔質材料[10,11]やポリカーボネート膜を加工した実験[12]などが行われている。

電気浸透流の実用化に向けた課題の一つは，外場として直流電圧を長時間印加することにより生じる溶液の電気分解である。これを防ぐために，進行波電位の利用[13]や電気分解の特性時間スケールよりも短い周期の交流電圧を用いる[14,15]など，多くの試みがなされてきた。これらの方法では，順方向と逆方向の電気浸透流を整流して，正味の一方通行の流れを誘発することが重要な要素となる。

流れの整流と制御は，様々なタイプの小規模システムで幅広く研究されており，特に，ナノスケールでの整流は関心が高いトピックである。このような整流装置の例としては，円錐形の細孔[14-16]やラチェット形の流路[17]などの幾何学的に非対称な構造を用いたもの，固液界面の表面電荷分布の非対称性[18-21]，指向性印加電圧[22-24]，ナノ多孔質媒体とイオン交換膜の組み合わせによる非対称性[25]などが考えられている。また，古典的なバルブ構造もナノスケールシステムへの応用が検討されている[26]。このアプローチは，前方と後方の流れの間に大きな差を機械的に作り出すことができるので有望であるが，複雑なナノ加工法や，繊細な構造の機械的な誤作動が懸念点となっている。これを克服するシステムとして，例えば，アクティブフィルターの目詰まりに基づくマイクロスケールの整流[27,28]のような，自発的かつ機械的に逆流を防止できるより単純な構造のシステムであれば，ナノスケールの電気浸透流を効率的に整流することができ，応用の可能性が広がる。

本研究では，より簡易な構造で交流電圧下での整流が可能な電気浸透流ポンプとして，細孔密度の異なる2種類の膜と帯電したコロイド粒子からなる電気浸透流ダイオードを提案した。この電気浸透流ダイオードは，順方向に電場を印加する際は，コロイド粒子は細孔密度の高い膜に近づくため，発生する電気浸透流にはほとんど影響を与えない。一方で，逆方向に電場を印加したときにはコロイド粒子が細孔を閉塞するため，流れを準機械的に抑制する。この流れの非対称性から，周期がコロイド運動の緩和時間よりも十分に長ければ，交流電場で誘起される電気浸透流から正味の一方向の流れを得ることが期待される。本研究では，まず，提案システムのMDシミュレーションを行い，電流と流量の両方について電場に対する強い非線形性，すなわち，電気浸透流ダイオードとしての性能を確認した。さらに，この現象を確認するために，電気浸透流ダイオードの数理モデルを構築し，シミュレーション結果との比較を行った。モデリングでは，コロイド粒子の存在確率に応じて変化する流路の有効径と呼ばれる新しいパラメータを導入し，有限長の円筒形細孔を通過する電気浸透流の理論モデルを拡張した。シミュ

レーション結果との比較より，提案したモデル方程式がシステムの振る舞いを再現することを示した。

2．問題設定と分子動力学（MD）シミュレーション

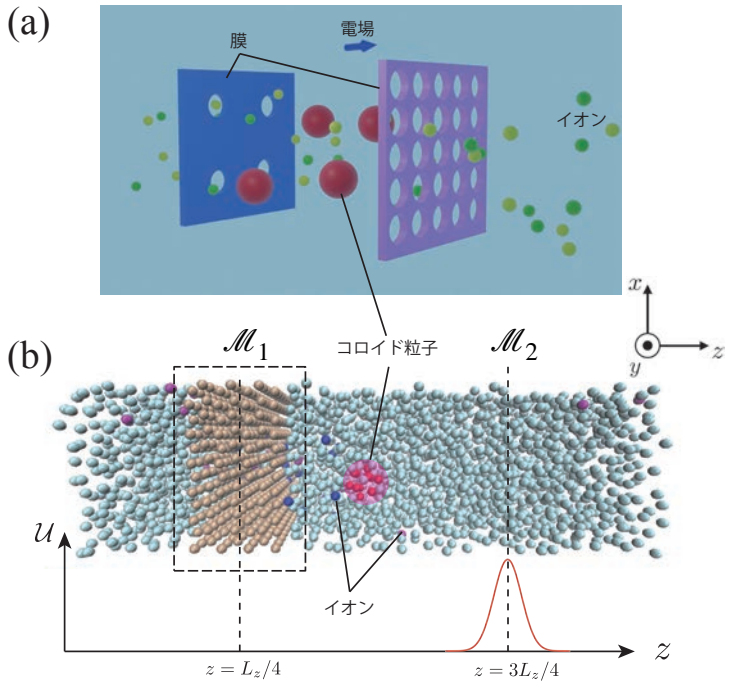


Fig. 1: 電気浸透流ダイオードの概要。

(a)は模式図，(b)はシミュレーションで検討したシステムである。シミュレーション系では、膜$\mathscr{M}_1$は物理的な細孔を持ち、膜$\mathscr{M}_2$はコロイド粒子のみに作用するエネルギー障壁としてモデル化されている（文献[29]より一部改変して転載; CC BY 4.0）。

Fig.1(a)のように，z軸に垂直な方向に平行に置かれた2種類の膜$\mathscr{M}_1$と$\mathscr{M}_2$に帯電したコロイド粒子が挟まれた**電気浸透流ダイオード**を考える。コロイド粒子の大きさは膜の孔径Dよりもわずかに大きく，1粒子あたり電荷$q > 0$に帯電している。膜が有する細孔の数は膜$\mathscr{M}_1$がコロイド粒子の数と同程度，膜$\mathscr{M}_2$が$\mathscr{M}_1$に比べてはるかに多いとし，細孔壁は負に帯電している。ここにz方向の電場E_zを印加すると，コロイド粒子は電場の向きに応じて$\mathscr{M}_1$あるいは$\mathscr{M}_2$側に電気泳動する。$E_z < 0$ではコロイド粒子は$\mathscr{M}_1$の近くに移動するため，誘起された電流や電気浸透流は細孔近傍の粒子によって遮断されると予想される。

本研究では，膜，コロイド粒子，溶媒を理想的な粒子でモデル化し，分子動力学（MD）シミュレーションを行った。粒子間の相互作用には，粒子間相互作用のポテンシャルエネルギーが$r_* < r_c$で$\mathscr{E} = 4\epsilon\left[(\sigma/r_*)^{12} - (\sigma/r_*)^6\right]$，それ以外で0となるLennard-Jones（LJ）ポテンシャルを用いた。ここで，r_*は相互作用する粒子の中心間の距離，ϵとσはそれぞれ粒子のポテンシャル深さと直径に対応するパラメータ，r_cはカットオフパラメータである。本研究では，すべてのLJ粒子についてLJパラメータをϵ_0とσ_0とし，カットオフ距離は$r_c = 2.5\sigma_0$とした。本研究では，物理量を以下の特徴量を用いてスケーリングした；長さσ_0，質量m_0，時間$\tau_0 = \sigma_0\sqrt{m_0/\epsilon_0}$，エネル

ギーϵ_0，温度$T_0 = \epsilon_0/k_B$，電荷$q_0 = \sqrt{4\pi\varepsilon_0\sigma_0\epsilon_0}$，電場$E_0 = \epsilon_0/\sqrt{4\pi\varepsilon_0\sigma_0^3\epsilon_0}$。ここで k_B はボルツマン定数，ε_0は真空の誘電率である。

実際の計算でのMDシミュレーションのスナップショットをFig.1(b)に示す。膜$\mathscr{M}_1$はz方向に六角密着構造を7層積層し，膜面中心から$1.2\sigma_0$以下の距離にある粒子を除去して細孔としたものを作成し，その中間層が$z = L_z/4$となるように配置した（L_zはz方向のシステムサイズ）。細孔壁には12個の一価の陰イオン（電荷$-q_0$）を一様に分布させた。細孔密度の高い膜$\mathscr{M}_2$は，コロイド粒子のみに作用する仮想的なエネルギー障壁でモデル化し，具体的にはガウスポテンシャル障壁$\mathscr{U} = U_0 \exp\left[-a_0(z-z_0)^2\right]$，$z < \sigma_0$を仮定した。ここで，ポテンシャル高さ$U_0$はコロイド粒子が$\mathscr{M}_1$を通過しないために十分な大きさである$30\epsilon_0$に設定し，$\mathscr{M}_2$の中心は$z_0 = 3L_z/4$，ポテンシャルの幅，すなわち膜厚を決めるパラメータa_0は$10\sigma_0^{-2}$とした。コロイド粒子は，対角線の長さが$2\sigma_0$の正二十面体の頂点に12個の陽イオンが位置するように構成され，コロイド粒子の正味の電荷はq，すなわち，各イオンの電荷は$q/12$とした。電解質溶液は，20個の一価の陰イオンと$32-q$個の一価の陽イオンを含む2410個の粒子から構成されており，陽イオンの数は，系全体の電荷（細孔壁の表面電荷を含む）がゼロになるように決定した。MDシミュレーションには，オープンソースパッケージであるLAMMPSを使用した。各粒子のニュートン方程式の時間積分には速度Verlet法，長距離のクーロン相互作用の取り扱いには，particle-particle-particle-mesh（PPPM）法を採用した。また，すべての方向に周期境界条件を適用し，NVTアンサンブルを用いて系の温度をT_0に維持した。シミュレーションの時間ステップは$dt = 0.002\tau_0$とした。系のサイズはx, y 方向が$L_x = L_y = 10\sigma_0$，z方向は$L_z \sim 40\sigma_0$で，L_zの正確な値は，z方向の圧力が$1.0 \pm 0.1\epsilon_0/\sigma_0^3$となるように決定した。

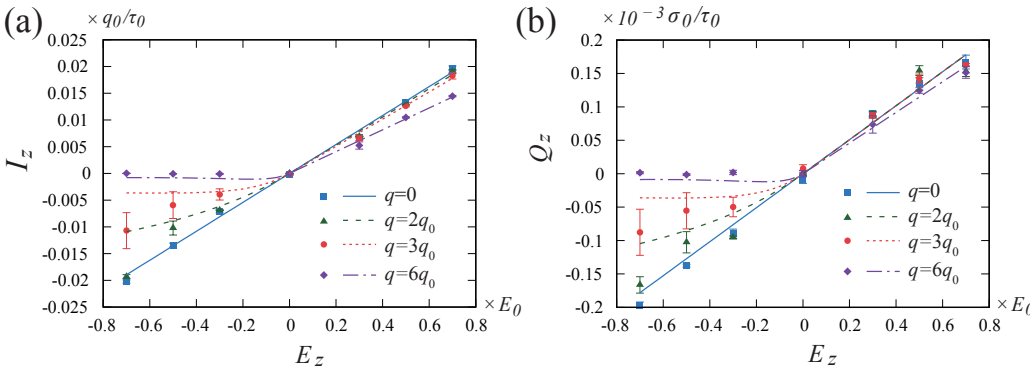


Fig. 2: MDシミュレーションから得られた電気浸透流ダイオードの非線形応答。
(a)が電流I_z，(b)が電気浸透流Q_zを示し，コロイド粒子の電荷qに対する電場E_zの関数としてプロットしている（文献[29]より転載; CC BY 4.0）。

コロイド粒子の電荷qを様々な値に設定した場合のシミュレーション結果をFig.2に示す。ここでは，電流I_zと電気浸透流Q_zを，$-0.7E_0 \le E_z \le 0.7E_0$の範囲で電場$E_z$の関数としてプロットした。シミュレーションは，まず10^5ステップの間，圧力をϵ_0/σ_0^3，温度をT_0に保ったまま，電場なしでシステムの平衡化を行い，その後，電場を印加して，10^7ステップ以上の本計算を行っ

た。電流と流量は，それぞれ$I_z = (S/V)\sum_{\text{ions}} q_i v_{zi}$，$Q_z = (1/N)\sum_{\text{solution}} v_{zi}$のように計算し，ここで，$q_i$と$v_{zi}$は粒子の電荷と$z$方向の速度，$S$と$V$はそれぞれ$x$-$y$平面上の面積と体積，$N$は対象領域内の粒子の数である。これらの値は，本計算の最後の8×10^6ステップの平均値として算出した。計算は，各ケースについて異なる3種類の初期設定を用いて実行した。図中のシンボルとエラーバーはこれらの平均値と分散であり，線は後述するモデル方程式による予測値である（詳細は4.2節を参照）。

$q > 0$の場合，電流I_zと流量Q_zはともに電場E_zに対して明らかな非線形性を示した。$E_z < 0$の場合の流れ，つまり逆流は抑制され，$q = 6q_0$では流れは完全にブロックされる。$E_z < 0$の場合，コロイド粒子は$\mathscr{M}_1$の細孔に接近し，細孔を流れる電流や流れを遮断するバルブの役割をする。この効果はコロイド粒子の電荷が大きいほど高くなった。コロイド粒子の挙動を示すため，$q = 6q_0$，$E_z = \pm 0.7E_0$の場合についてz軸方向のコロイド粒子の確率分布関数をFig.3に示す。コロイド粒子は，電場が負（$E_z = -0.7E_0$）の条件ではほとんどが膜$\mathscr{M}_1$の近くに存在するのに対し，電場が正（$E_z = 0.7E_0$）の条件では$\mathscr{M}_1$から十分に離れていることが確認された。これとMDの結果から，電流と流れの非線形な挙動は，コロイド粒子によって流路の実質的な径が小さくなることに起因することが示唆された。

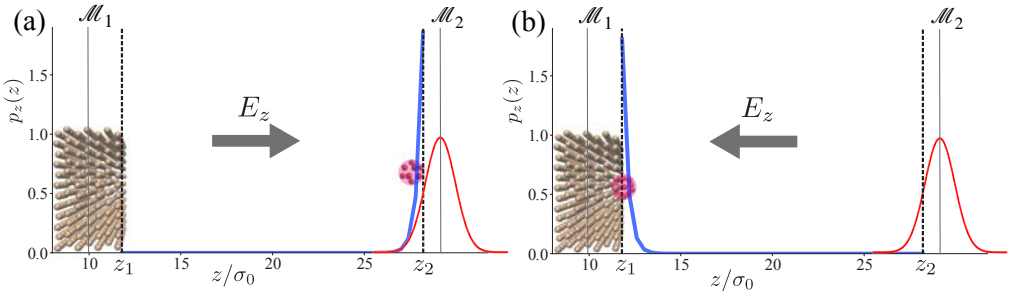


Fig. 3: 電場印加時のコロイド粒子の挙動。

(a)順方向（$E_z > 0$），(b)逆方向（$E_z < 0$）の電場に対する応答を示す。横軸はz座標，縦軸はz方向のコロイド粒子の存在確率$p_z(z)$を示す（文献[29]より一部改変して転載; CC BY 4.0）。

そこで，本研究ではコロイド粒子の存在確率に依存する流路の実質的なサイズ，すなわち有効径を導入することで，非線形応答を捉える理論モデルを構築した。この有効径を用いて，Sherwoodらによって提案された有限長の円筒形細孔に誘起される電流と電気浸透流を与える理論モデル[4]を拡張する。次節ではモデル方程式を構築し，モデルとMDシミュレーションの結果を比較する。

3．電気浸透流ダイオードの数理モデルの構築

本節では，Fig.2のMDシミュレーションの結果を再現する電気浸透流ダイオードの数理モデルを構築する。

3．1．ナノポアを通過する電気浸透流

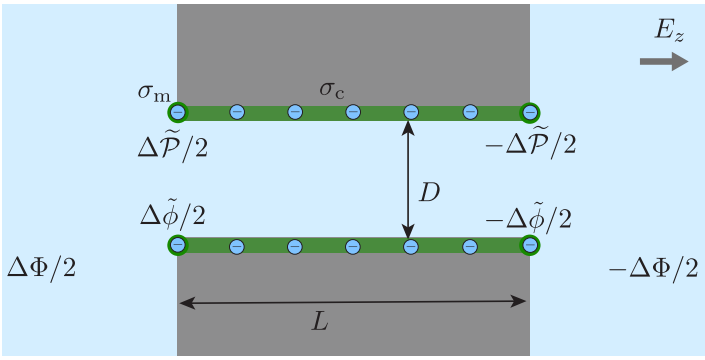


Fig. 4: 有限長さの円筒形細孔の模式図。

長さL，流路径Dで，電荷密度は細孔の淵がσ_{m}，円筒壁面はσ_{c}である。細孔の両端で圧力差$\Delta\widetilde{\mathcal{P}}$，電位差$\Delta\tilde{\phi}$を持つ（文献[29]より転載; CC BY 4.0）。

まず，Sherwoodらが提案した有限長の円筒形細孔内の電流および電気浸透流のモデル式を概説し，コロイド粒子のない流れを表現するために採用する。ここでは，文献[3,4]を基に，Fig.4に示すような有限の長さL，直径Dの円筒形細孔を流れる電流と電気浸透流のモデル方程式の簡単な導出を行う。線形応答領域では，電流I_zと電気浸透流Q_zは次のように書ける：

$$I_z = -G\Delta\Phi, \tag{1}$$

$$Q_z = -H\Delta\Phi\,. \tag{2}$$

ここで，Gはコンダクタンス，Hは電気浸透流係数，$\Delta\Phi$は膜の両側の電位差である。有限長さの円筒形細孔に対するGとHの明示的な式を与える前に，非常に薄い膜（$L \ll D$）と厚い膜（$L \gg D$）という極端なケースを考える。

$L \ll D$の場合，膜は孔のある厚さ0のシートとみなされ，入口効果が支配的となる。このとき，コンダクタンスは直径に比例し，$G_{\mathrm{m}} = \kappa D$（$\kappa$は溶液のバルク電気伝導度）となる。また，細孔径$D$に比べてデバイ長$\lambda_{\mathrm{d}}$が十分に小さい領域（$\lambda_{\mathrm{d}} \ll D$，今回のMDシミュレーションではこれに相当）に限定すると，電気浸透流係数は$H_{\mathrm{m}} \sim D\sigma_{\mathrm{m}}\lambda_{\mathrm{d}}/\mu$となる。ここで，$\mu$は溶液粘度，$\sigma_{\mathrm{m}}$は細孔の縁に沿った電荷密度である。一方，膜厚が大きい$L \gg D$の場合は円筒壁面の効果が支配的になり，コンダクタンスは$G_{\mathrm{c}} = \pi D^2\kappa/(4L)$，$\lambda_{\mathrm{d}} \ll D$の場合での電気浸透流係数は$H_{\mathrm{c}} \sim \pi D^2\sigma_{\mathrm{c}}\lambda_{\mathrm{d}}/(4\mu L)$（$\sigma_{\mathrm{c}}$は円筒壁面の表面電荷密度）となる。なお，以下では，$L \ll D$の場合の量を下付きのmで表し，$L \gg D$の場合の量を下付きのcで表す。

次に，有限長Lの円筒形細孔のGとHを上述の極限のケースの係数を用いて示す。コンダクタンスGは，単純に入口と円筒壁面の効果を直列に組み合わせることで得られる：

$$G = \left(G_{\mathrm{m}}^{-1} + G_{\mathrm{c}}^{-1}\right)^{-1} = \kappa\left(\frac{4L}{\pi D^2} + \frac{1}{D}\right)^{-1}. \tag{3}$$

電気浸透流係数Hの表現には，有限長の円筒の両端間に発生する圧力差$\Delta\widetilde{\mathcal{P}}$の影響を考慮する

必要がある（例えば，文献[30]を参照）。以下の議論では，便宜上，円筒の両端間の電位差を$\Delta\tilde{\phi}$と表す。$\Delta\widetilde{\mathscr{P}}$と$\Delta\tilde{\phi}$の値は電流と流量に関する2つの連続の式を用いて求められる。具体的には，膜を流れる電流の連続性は次のように書ける：

$$I_z = G_{\mathrm{m}}(\Delta\Phi - \Delta\tilde{\phi}) = G_{\mathrm{c}}\Delta\tilde{\phi}\,. \tag{4}$$

ここで，2つ目の等式の左辺は膜の外側の電流，右辺は膜の内側の電流である。流量Q_zの連続性は，電位差と圧力差の両方で発生する流量を考慮して次のように書ける[30]：

$$Q_z = H_{\mathrm{m}}(\Delta\Phi - \Delta\tilde{\phi}) - \mathscr{L}_m\Delta\widetilde{\mathscr{P}} = H_{\mathrm{c}}\Delta\phi + \mathscr{L}_{\mathrm{c}}\Delta\widetilde{\mathscr{P}}\,. \tag{5}$$

ここで，$\mathscr{L}_{\mathrm{m}} = -D^3/(24\mu)$は無限に薄いシート上の細孔の透過率に関するSampsonの公式[31]，$\mathscr{L}_{\mathrm{c}} = -\pi D^4/(128L\mu)$は円筒形流路の透過率に関するPoiseuilleの法則である。2つ目の等式の左辺は電位差と圧力差によって誘起された膜外の流量の和であり，右辺は円筒内の流量である。ここで，$\Delta\tilde{\phi}$と$\Delta\widetilde{\mathscr{P}}$の明示的な表現は，式(4)，(5)を解くことで与えられ，続いて式(5)の第一等式を用いて，電気浸透流係数は次のように計算できる：

$$H = \left(\frac{Q_z}{\Delta\Phi}\right) = \frac{\mathscr{L}_{\mathrm{m}}H_{\mathrm{c}}G_{\mathrm{c}}^{-1} + \mathscr{L}_{\mathrm{c}}H_{\mathrm{m}}G_{\mathrm{m}}^{-1}}{(\mathscr{L}_{\mathrm{m}} + \mathscr{L}_{\mathrm{c}})(G_{\mathrm{m}}^{-1} + G_{\mathrm{c}}^{-1})} = \frac{\left(\lambda_{\mathrm{d}}/\mu\right)\left(D\sigma_{\mathrm{m}} + 16L\sigma_{\mathrm{c}}/3\pi\right)}{(1 + 4L/\pi D)\left(1 + 16L/3\pi D\right)}\,. \tag{6}$$

３．２．流路の有効径

電気浸透流ダイオードのモデルでは，電流と流量の変化を，式(3)および(6)においてDに代わる有効径D_{eff}を導入することで捉え，細孔の入り口付近にコロイド粒子が存在する場合の影響を考える。本研究では，D_{eff}のモデル方程式を以下の形で提案する：

$$D_{\mathrm{eff}} = D\exp\left[-\alpha\left(P_z(q, E_z) - P_z^0\right)\left(P_{xy}(q) - P_{xy}^0\right)\right]. \tag{7}$$

ここで，$\alpha > 0$はモデルパラメータ（定数），$P_z(q, E_z)$はコロイド粒子が膜$\mathscr{M}_1$に近づく確率（に比例する）量，$P_{xy}(q)$はコロイド粒子が細孔の周りにある確率に関係する量で，P_z^0とP_{xy}^0は$q = 0$のときの値である。P_{xy}はx-y平面上のコロイド粒子の運動を表すので，E_zには依存しないと仮定する。この単純なモデル方程式を導出するにあたって，D_{eff}の微小変動に伴う確率p^*(式(7)の指数関数の変数に対応する)の変化が，$\mathrm{d}p^* \sim -(D_{\mathrm{eff}}\mathrm{d}D_{\mathrm{eff}})/(\pi D_{\mathrm{eff}}^2)$のように見積もることができる，すなわち，$p^*$の変化が流路断面積の変化率に比例すると仮定する。この関係式を$p^* \to 0$で$D_{\mathrm{eff}} = D$という条件で積分することで，式(7)は得られ，これは電気浸透流ダイオードの本質的な特徴を記述する。コロイド粒子が$\mathscr{M}_1$から十分に離れている場合（$P_z \sim 0$）は有効径は細孔径と等しく（$D_{\mathrm{eff}} \sim D$），コロイド粒子が$\mathscr{M}_1$付近にあり（$P_z \gg 0$），かつ細孔近傍に存在する場合（$P_{xy} \gg 0$），流路は完全に閉塞する（$D_{\mathrm{eff}} \sim 0$）。次節では，Fokker-Planck方程式の解を用いて，P_zとP_{xy}の解析的な表現について説明する。

３．３．コロイド粒子の運動に対するFokker-Planck方程式

ここでは，Fokker-Planck方程式を用いてコロイド粒子の運動を考察し，有効径のモデル方程式(7)に含まれるパラメータP_zとP_{xy}の具体形を導出する。電場が印加された溶液中のコロイド粒子には，電気泳動による力と溶媒粒子からのランダム力が作用し，これにより粒子の移流・拡散が生じる。以下では，このような条件下での確率密度を支配するFokker-Planck方程式を，$\mathscr{M}_1$に対して垂直な方向（z軸方向）の運動と平行な方向（x-y平面内）の運動に分けて定式化する

まず，z軸方向の確率密度関数p_zに関するFokker-Planck方程式は，

$$\frac{\partial p_z}{\partial t} = \frac{\partial}{\partial z}\left[(A\,p_z) + \frac{\partial}{\partial z}(\mathscr{D}p_z)\right], \tag{8}$$

と書け，ここで，$A(q, E_z)$は移流速度，$\mathscr{D}$は粒子の拡散係数で，qとE_zには依存しないと仮定する。定常状態では，解析解は次のように得られる：

$$p_z(z) = p_0 \exp\left(-\frac{A\,z}{\mathscr{D}}\right),\quad p_0 = A\,\mathscr{D}^{-1}\left[\exp\left(-\frac{A}{\mathscr{D}}z_1\right) - \exp\left(-\frac{A}{\mathscr{D}}z_2\right)\right]^{-1}. \tag{9}$$

ここで，z_1とz_2は，それぞれ$\mathscr{M}_1$と$\mathscr{M}_2$の表面における境界の位置である（Fig.3の破線参照）。

次に，コロイド粒子のx-y方向の運動に注目する。円筒座標系において動径方向の存在確率分布p_rを考えると，Fokker-Planck方程式は次のように書ける：

$$\frac{\partial p_r}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}\left(A_r r p_r\right) + \mathscr{D}\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial}{\partial r}\right)\right]p_r\,. \tag{10}$$

さらにここでは，コロイド粒子と細孔壁の電気的相互作用による径方向の移流を考慮して，係数A_rはr^2に反比例する，すなわち$A_r(q, r) = C_A(q)/r^2$と仮定する。このとき，次のような解析解が得られる：

$$p_r(r) = p_{r0}\exp\left(\frac{C_A}{\mathscr{D}r}\right),\quad p_{r0} = \left[\int_0^{r_{\max}} r\,\exp\left(\frac{C_A}{\mathscr{D}r}\right)\mathrm{d}r\right]^{-1}. \tag{11}$$

以上から，有効径の式(7)のパラメータは，$P_z = p_z(z_1)$および $P_{xy} = \int_0^{\delta r} r' p_r(r')\mathrm{d}r'$（$\delta r$は細孔近傍領域の半径）と定義した。これらのパラメータの具体的な値については，2章の問題設定に対応して独立して行った移流と拡散のMDシミュレーションの結果から推定した。

４．モデルとMDシミュレーションの比較

４．１．モデルパラメータの決定

ここでは，MDシミュレーションに用いた系のモデルパラメータの具体的な値を決定する。

まず，溶液中のコロイド粒子の拡散係数$\mathscr{D}$を推定するために，バルク電解質溶液中のコロイド粒子のブラウン運動について，独立したMDシミュレーションを実行した。コロイド粒子の平均二乗変位から拡散係数は$\mathscr{D} = 1.7\times10^{-2}\sigma_0^2/\tau_0$と求められた。コロイド粒子の電荷は拡散運動にほとんど影響しないため，以下の議論では全てのケースについてこの値を使用する。

次にz方向の移流の影響を考える。この移流の効果は式(9)のパラメータ$A(q, E_z)$を介してモデルに組み込まれている。（$\mathscr{M}_2$と同様の）2つの仮想膜の間に電荷qを持つコロイド粒子が挟まれた系に電場E_zを印加するMDシミュレーションから，z軸に沿ったコロイド粒子の確率密度関数を求め，上記で得られた$\mathscr{D}$の値を用いて式(9)で与えられる解析解をフィッティングすることで，各(q, E_z)における係数Aの値を決定した。その結果，$A = \beta q E_z$という関係が得られ，$\beta = -1.3 \times 10^{-2} \sigma_0^2 / \epsilon_0 \tau_0$と求められた。

最後に，式(11)のC_Aの値を決定するために，x-y方向の移流の影響を調べた。ここでは，膜$\mathscr{M}_1$の近傍でコロイド粒子の運動が2次元に限定される状況を考える。この制約は，$\mathscr{M}_2$と同様の仮想的な膜を$\mathscr{M}_1$からコロイド粒子の粒子径だけ離れた位置に配置することで実現した。x-y平面での運動に注目するため，物理的な細孔は考慮せず，$\mathscr{M}_1$内の電荷は2章の問題設定と同様に分布させた。z方向のときと同様に$\mathscr{D}$の値を用いて，式(11)で与えられる解析解をこれらの制約付きMDシミュレーションで得られた確率密度関数にフィッティングすることで，C_Aの値を決定した。その結果，C_Aの値は$C_A = \gamma q$の形で得られ，$\gamma = 6.2 \times 10^{-4} \sigma_0^3 / \tau_0 q_0$と求められた。

４．２．モデル方程式の検証

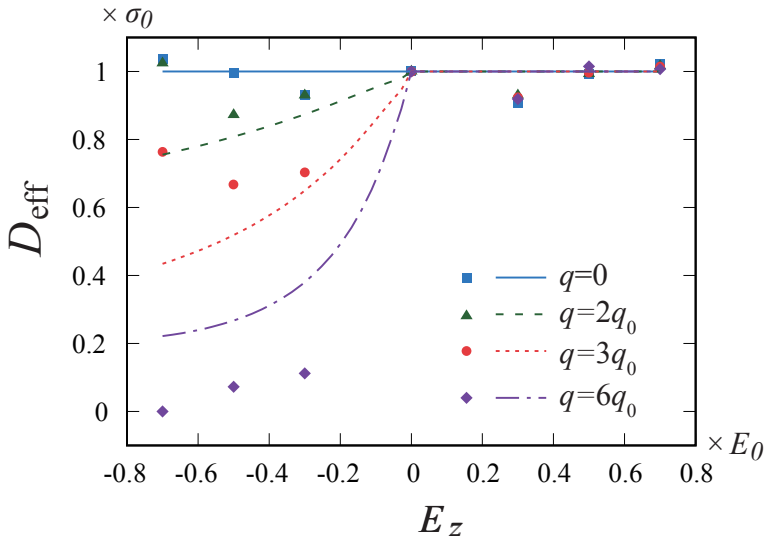


Fig. 5: 有効径D_{eff}の比較。

様々なqに対して，電場E_zの関数としてプロットしている。シンボルはシミュレーションから推定された値を，線はモデル方程式(9)で与えられた値を示す（文献[29]より転載; CC BY 4.0）。

ここまでで，独立したMDシミュレーションから得られたモデルパラメータ$\mathscr{D}$, A, C_Aの値を用いて，コロイド粒子が存在する場合の流路の有効径D_{eff}を推定することができた。式(1), (2), (3), (6)で与えられたモデルにより，式(7)で定義された有効径D_{eff}を用いて推定した電流と流量をFig.2に線で示す。ここでは，自由パラメータαは$\alpha = 6.7$とした。電流，流量ともに，MDシミュレーションの結果とよく一致しており，モデル推定値は電気浸透流ダイオードの正しい非線形挙動を示した。この結果から，自由パラメータを適切に設定すれば，電気浸透流ダイオードの性能は，コロイド粒子の電荷qと電場E_zの関数として，モデル方程式から近似的に推定できることがわかった。

有効径D_{eff}の値を直接比較したものをFig.5に示す。ここでは，MDシミュレーションのD_{eff}の値は，直径Dを未知のパラメータとして，式(1), (2), (3), (6)を用いて，Fig.2のMDの結果から逆算した。MDシミュレーションとモデル方程式の間には良好な一致が得られ，モデル方程式が電気浸透流ダイオードの特徴をよく捉えられていることが確認できた。モデルの他の物理パラメータに関して，表面電荷密度σ_{c}やσ_{m}などのパラメータは，幾何学的な設定から決定でき，電解質溶液の特性に関連する電気伝導度κと粘度μも独立して測定することができる。しかし，上記の比較では，これらのパラメータの値を単独で測定することを回避するために，細孔近傍にコロイド粒子が存在せず純粋な電流と流量が近似的に得られる領域$E_z \geq 0$における電流と流量の線形近似から求めた。

5．おわりに

本研究では，細孔密度の異なる2種類の膜の間に帯電したコロイド粒子を配置した電気浸透流ダイオードを提案し，MDシミュレーションとモデル方程式の構築を行った。シミュレーションでは，2枚の膜の間の細孔密度の違いにより，コロイド粒子がナノバルブの役割を果たすことによる電流と電気浸透流の整流が観察された。モデル構築では，流路の有効径の概念を導入することで，提案した電気浸透流ダイオードの重要な特徴であるコロイドナノバルブ（コロイド粒子が流路を塞いで逆方向の電流と流れを防ぐ現象）を捉える数理モデルを確立した。有効径は，細孔周辺の存在確率密度関数に依存しており，式(7)で定義される。この有効径の考え方は，分子ふるい，ナノフィルトレーション，多孔質媒体中の輸送，交通システムの渋滞解析など，他の用途にも応用できる可能性がある。モデル方程式のパラメータはコロイド粒子の移流・拡散に関するFokker-Planck方程式の解析解から得られ，これらの値を用いて，MDシミュレーションの全結果をモデル方程式により再現することに成功した。これにより，提案したモデル方程式がコロイドナノバルブの本質的なメカニズムを記述できていることが確認された。比較の際には，自由パラメータαを一度だけ校正する必要があるが，αの値が特定できれば，同様の系に対して様々な条件下でのダイオードの性能を推定することができる。

実用面では，本ダイオードは，整流の特性緩和時間よりも十分に長い周期であれば，交流電圧下での流れの整流に応用できると期待される。さらに，このナノバルブは，物理的なパラメータを適切に選択すれば非対称な流れを可能にし，逆方向の流れをほぼ完全に遮断することができるため，既存の交流電気浸透流システムと比較して，より効率的な整流が実現できる可能性がある。本研究では，コロイド粒子，膜，電解質溶液は，抽象的な材料を想定していたが，これらは実用的な材料に置き換えて検討することができる。これらの実験的検討は今後の研究トピックである。

6．謝辞

本研究は東京大学情報基盤センター若手・女性利用者推薦制度の助成を受けて実施された。

参 考 文 献

[1] V. Pretorius, B. J. Hopkins, and J. D. Schieke, Electro-osmosis: A new concept for high-speed liquid chromatography, J. Chromatogr. A **99**, 23 (1974).

[2] C.-Y. Wang, Y.-H. Liu, and C. C. Chang, Analytical solution of electro-osmotic flow in a semicircular microchannel, Phys. Fluids **20**, 063105 (2008).

[3] M. Mao, J. D. Sherwood, and S. Ghosal, Electro-osmotic flow through a nanopore, J. Fluid Mech. **749**, 167 (2014).

[4] J. D. Sherwood, M. Mao, and S. Ghosal, Electroosmosis in a finite cylindrical pore: Simple models of end effects, Langmuir **30**, 9261 (2014).

[5] D. V. Melnikov, Z. K. Hulings, and M. E. Gracheva, Electro-osmotic flow through nanopores in thin and ultrathin membranes, Phys. Rev. E **95**, 063105 (2017).
[6] R. Qiao and N. R. Aluru, Charge Inversion and Flow Reversal in a Nanochannel Electro-Osmotic Flow, Phys. Rev. Lett. **92**, 198301 (2004).
[7] Y. Chen, Z. Ni, G. Wang, D. Xu, and D. Li, Electroosmotic flow in nanotubes with high surface charge densities, Nano Lett. **8**, 42 (2008).
[8] B. Rotenberg and I. Pagonabarraga, Electrokinetics: Insights from simulation on the microscopic scale, Mol. Phys. **111**, 827 (2013).
[9] H. Yoshida, H. Mizuno, T. Kinjo, H. Washizu, and J.-L. Barrat, Molecular dynamics simulation of electrokinetic flow of an aqueous electrolyte solution in nanochannels, J. Chem. Phys. **140**, 214701 (2014).
[10] S. Yao and J. G. Santiago, Porous glass electroosmotic pumps: Theory, J. Colloid Interface Sci. **268**, 133 (2003).
[11] S. Yao, D. E. Hertzog, S. Zeng, J. C. Mikkelsen, Jr., and J. G. Santiago, Porous glass electroosmotic pumps: Design and experiments, J. Colloid Interface Sci. **268**, 143 (2003).
[12] K. Bengtsson and N. D. Robinson, A large-area, all-plastic, flexible electroosmotic pump, Microfluid. Nanofluid. **21**, 178 (2017).
[13] A. Ramos, H. Morgan, N. G. Green, A. González, and A. Castellanos, Pumping of liquids with traveling-wave electroosmosis, J. Appl. Phys. **97**, 084906 (2005).
[14] X.Wu, P. Ramiah Rajasekaran, and C. R. Martin, An alternating current electroosmotic pump based on conical nanopore membranes, ACS Nano **10**, 4637 (2016).
[15] W. Li, W. Wang, Q. Hou, Y. Yan, C. Dai, and J. Zhang, Alternating electric field-induced ion current rectification and electroosmotic pump in ultranarrow charged carbon nanocones, Phys. Chem. Chem. Phys. **20**, 27910 (2018).
[16] L. Jubin, A. Poggioli, A. Siria, and L. Bocquet, Dramatic pressure-sensitive ion conduction in conical nanopores, Proc. Natl. Acad. Sci. USA 115, **4063** (2018).
[17] H. Sugioka and S. Segawa, Effective symmetry breaking of flow in AC electro-osmotic pump using a ratchet structure, J. Phys. Soc. Jpn. **88**, 084602 (2019).
[18] R. Karnik, C. Duan, K. Castelino, H. Daiguji, and A. Majumdar, Rectification of ionic current in a nanofluidic diode, Nano Lett. **7**, 547 (2007).
[19] I. Vlassiouk and Z. S. Siwy, Nanofluidic diode, Nano Lett. **7**, 552 (2007).
[20] C. B. Picallo, S. Gravelle, L. Joly, E. Charlaix, and L. Bocquet, Nanofluidic Osmotic Diodes: Theory and Molecular Dynamics Simulations, Phys. Rev. Lett. **111**, 244501 (2013).
[21] S. Prakash, H. A. Zambrano, M. Fuest, C. Boone, E. Rosenthal-Kim, N. Vasquez, and A. T. Conlisk, Electrokinetic transport in silica nanochannels with asymmetric surface charge, Microfluid. Nanofluid. **19**, 1455 (2015).
[22] W. Guan, R. Fan, and M. A. Reed, Field-effect reconfigurable nanofluidic ionic diodes, Nat. Commun. **2**, 506 (2011).
[23] S. Wu, F. Wildhaber, O. Vazquez-Mena, A. Bertsch, J. Brugger, and P. Renaud, Facile fabrication of nanofluidic diode membranes using anodic aluminium oxide, Nanoscale **4**, 5718 (2012).
[24] S. Wu, F. Wildhaber, A. Bertsch, J. Brugger, and P. Renaud, Field effect modulated nanofluidic diode membrane based on Al2O3/W heterogeneous nanopore arrays, Appl. Phys. Lett. **102**, 213108 (2013).
[25] A. Yaroshchuk, E. E. Licón, E. K. Zholkovskiy, M. P. Bondarenko, and T. Heldal, Asymmetric electroosmotic pumping across porous media sandwiched with perforated ion-exchange membranes, Faraday Discuss. **199**, 175 (2017).
[26] V. Tesař, Valve-less rectification pumps, in *Encyclopedia of Microfluidics and Nanofluidics*, edited by D. Li (Springer, Boston, MA, 2008).
[27] S. M. McFaul, B. K. Lin, and H. Ma, Cell separation based on size and deformability using microfluidic funnel ratchets, Lab Chip **12**, 2369 (2012).
[28] Y. Yoon, S. Kim, J. Lee, J. Choi, R.-K. Kim, S.-J. Lee, O. Sul, and S.-B. Lee, Clogging-free microfluidics for continuous size-based separation of microparticles, Sci. Rep. **6**, 26531 (2016).
[29] S. Koyama, D. Inoue, A. Okada, and H.Yoshida, Electro-osmotic diode based on colloidal nano-valves between double membranes, Phys. Rev. Res. **3**, 033289 (2021).
[30] H. Yoshida, T. Kinjo, and H. Washizu, Coupled lattice Boltzmann method for simulating electrokinetic flows: A localized scheme for the Nernst-Plank model, Commun. Nonlinear Sci. Numer. Simulat. **19**, 3570 (2014).
[31] R. A. Sampson, XII. On Stokes's current function, Philos. Trans. R. Soc. A **182**, 449 (1891).

A new method for monocular depth estimation

Zhaofeng Niu
Nara Institute of Science and Technology

1. Introduction

Depth information is very important for various applications [1], such as 3D reconstruction, autonomous driving and augmented reality. Nowadays, there are several ways to obtain the depth information: 1) using depth camera, which measures the depth by structure light [2], etc.; 2) using stereo images or video streams, which provides some spatial and temporal information to estimate the depth map [3, 4]; 3) using monocular images [5], for which the input is a single RGB image without temporal information. Among these three ways, using the depth camera is the simplest way but it is expensive and usually has lots of limitation, such as short sensing range and low resolution, while using single RGB images is the most convenient way with lowest cost, but it is very challenging due to the limited information from the input RGB image.

My research is depth estimation on single RGB images, which is the third way mentioned above. There are lots of existing methods aiming to achieve the high accurate depth estimation with deep learning method, which is proved to have better performance than the traditional methods [6]. For example, Liu *et al*. [7] propose a convolutional neural network (CNN) model with a conditional random field (CRF) loss, which is used to minimize the log-likelihood between neighboring superpixels generated by the model, while Cao *et al*. [8] design a fully connected CRF to do the post-processing for refining the output. In addition, since depth is a kind of geometric information, combining other geometric information such as surface normal [9] and semantic information [10], may help to improve the accuracy of depth estimation. Zhang *et al*. [11] propose a multi-task network, which can predict the surface normal, semantic segmentation and depth map simultaneously. However, the performance can be improved by better utilizing the multi-scale information of the input images, which is proved to be one of the keys for generating high-accuracy depth estimation.

Inspired by BTS method [12], a new monocular depth estimation method is proposed in this work, *i.e.*, HMA-Depth method [13] (which means a Hierarchical Multi-scale Attention method for Depth estimation). In BTS method, a multi-scale method is proposed for depth estimation. For each scale of the image features, a local planar guidance (LPG) module is introduced to guide the features back to the original resolution of the input image. However, it uses a convolutional layer to combine the results from each scale, which does not fully utilize the advantages of the scaled information. Instead of using LPG module, HMA-Depth method adopts a hierarchical multi-scale attention method to do the depth estimation. Similar to BTS method, HMA-Depth also tries to obtain the various advantages among different resolution scales. Specifically, higher resolution can show more details of the image while lower resolution has relatively clear object contour information. HMA-Depth extracts the features of multiple scales and adopts an attention mechanism to generate the important area in the estimated depth maps. Different from BTS method, HMA-Depth generates the depth map for each scale, as well as the attention map, which can show the area that is important for estimation. The experiment results prove that HMA-Depth outperforms BTS method and other state-of-art methods.

2. Method

To obtain more precise local information from the image while keeping a good understanding of the global context, HMA-Depth method is proposed, which adopts a multi-scale attention frame to do depth estimation. The network architecture is shown in Fig.1. Firstly, a backbone network extracts the features into different scales, *i.e.*, *H/8*, *H/4*, *H/2* and *H* (*H/s* indicates the scales of *H/s* × *W/s* for short, where $s \in \{1, 2, 4, 8\}$, and *H* and *W* represent the height and weight of the input image, respectively). Specifically, an atrous spatial pyramid pooling (ASPP) module, skip connections and bilinear interpolation are used in the up-sampling process. The ASPP module, using convolutional kernels with different dilation rates ($r \in \{3, 6, 12, 18, 24\}$), is used to improve the feature quality.

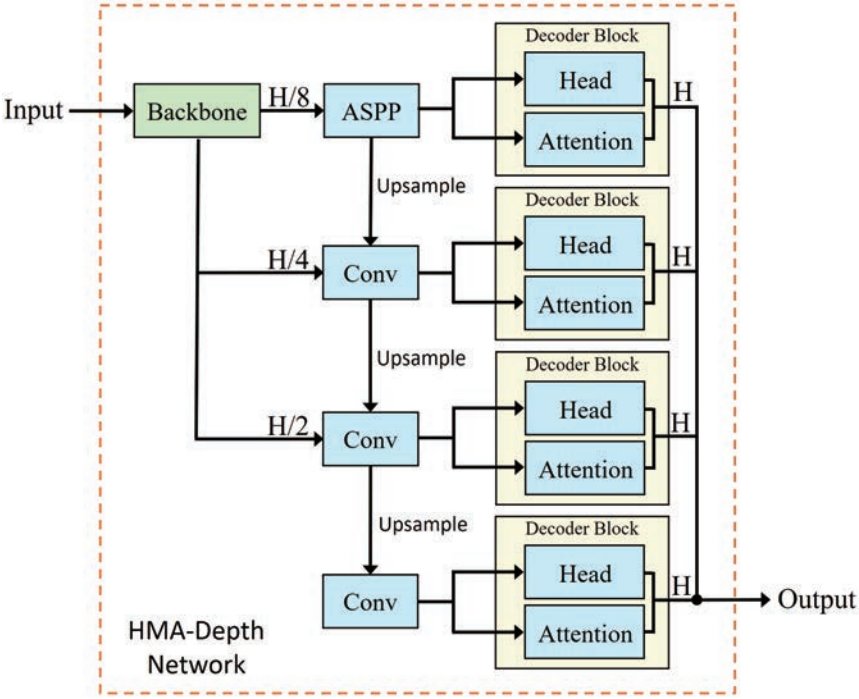


Fig.1 Network architecture

For each scale of resolution, a decoder block, which involves an attention module and a depth module, is used to generate the estimation results for the corresponding scale. The depth module estimates the depth map for each scale while the attention module can extract the preferred regions for depth map. To make each scale focus on independent parts of the input image, the calculation of attention masks can be explained as follows:

$$M_{H/8} = A_{H/8} \quad (1)$$

$$M_{H/4} = A_{H/4}(1 - A_{H/8}) \quad (2)$$

$$M_{H/2} = A_{H/2}(1 - A_{H/8})(1 - A_{H/4}) \quad (3)$$

$$M_{H} = (1 - A_{H/8})(1 - A_{H/4})(1 - A_{H/2}) \quad (4)$$

$A_{H/8}$, $A_{H/4}$ and $A_{H/2}$ indicate the attention maps for the scales of *H/8*, *H/4* and *H/2*, respectively, then $M_{H/8}$, $M_{H/4}$, $M_{H/2}$, and M_H indicate the corresponding weighted masks for each scale. These calculations can make each attention module focus on independent parts of the input, and the sum of masks equals to 1.

The scaled depth maps and attention maps are illustrated in Fig.2. It can be seen that the global information such as the object contour is estimated better in the lower resolution while higher resolution module predicts fine details. And the sum of four masks should be a whole white mask.

The final output can be represented as follows:

$$D_{final} = \sum_{s \in S} M_s \cdot D_s^m \quad (5)$$

where D^m is the scaled depth map generated from the depth module.

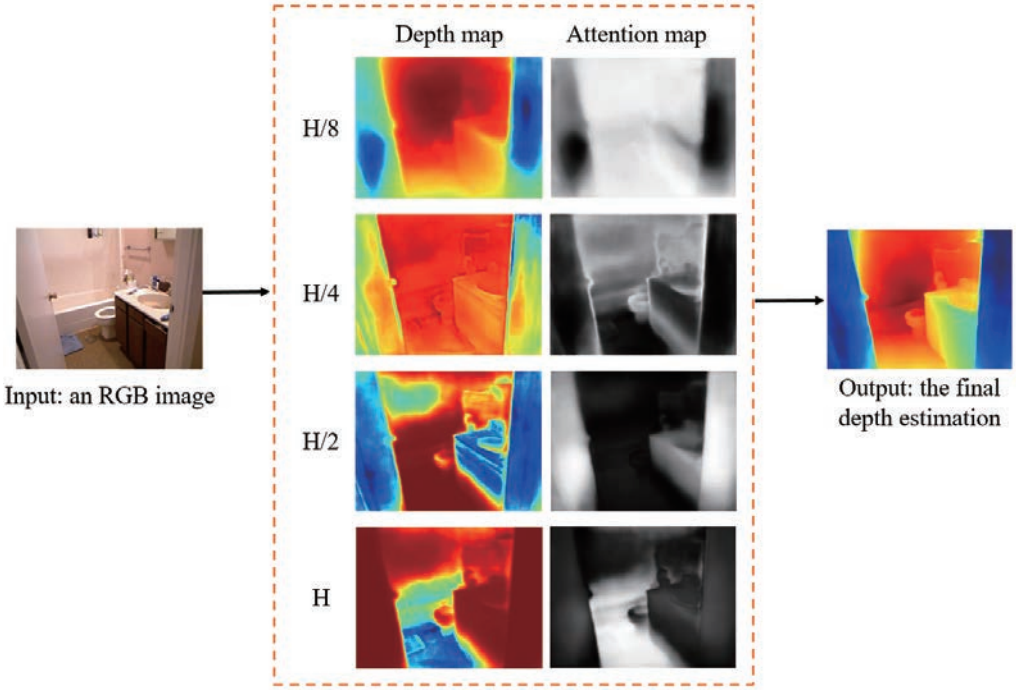


Fig.2 Depth and attention maps generated at different scales

As for the loss function, the scale-invariant error proposed by Eigen *et al.* [6] is adopted to calculate the error between the predicted y and the ground truth y^*, and the formula is shown as follows:

$$\text{Loss} = \frac{1}{n}\sum_i g_i^2 - \frac{\lambda}{n^2}\left(\sum_i g_i\right)^2 \quad (6)$$

in which $g_i = \log y_i - \log y_i^*$ and $\lambda \in [0, 1]$; n represents the number of pixels that have valid depth values. Similar to BTS method [12], $\lambda = 0.85$ is set in the experiments.

3. Experimental results

In the experiment, PyTorch [14] is used to implement the whole network. The number of the epoch is set as 50 and the batch size is 16. Multiple networks, such as ResNet 50, ResNeXt 50, are used as the backbone network, extracting the dense feature. A part of the experiments is conducted with Reedbush-H server. To verify the effectiveness of the proposed method, one commonly-used dataset, *i.e.*, KITTI dataset [15], is adopted to conduct the experiments.

KITTI dataset is obtained by an autonomous driving platform, which is equipped with a laser scanner, a GPS localization system and a stereo camera rig. To compare with other methods, the commonly used Eigen split [6] is adopted in the experiments, involving 23488 images from 32 scenes for training and 697

images from 29 scenes for testing. Tab.I shows the comparison result on KITTI dataset.

Tab.I. Quantitative results on KITTI dataset

Methods	**δ1↑**	**δ2↑**	**δ3↑**	**AbsRel↓**	**RMSE↓**	**RMSElog↓**
Make3D [16]	0.601	0.820	0.926	0.280	8.734	0.361
Eigen *et al*. [6]	0.702	0.898	0.967	0.203	6.307	0.282
Liu *et al*. [17]	0.680	0.898	0.967	0.201	6.471	0.273
Kuznietso *et al*. [18]	0.862	0.960	0.986	0.113	4.621	0.189
Yin *et al*. [19]	0.938	0.990	**0.998**	0.072	3.258	0.117
DORN[5]	0.932	0.984	0.994	0.072	**2.727**	0.120
BTS-ResNet 50 [12]	0.950	0.991	**0.998**	0.062	2.878	0.101
BTS-DenseNet 161 [12]	0.952	0.992	**0.998**	0.062	2.871	0.094
HMA-Depth-ResNet 50	0.953	0.992	**0.998**	0.062	2.870	0.096
HMA-Depth-ResNeXt 50	0.951	0.992	**0.998**	0.062	2.867	0.094
HMA-Depth-DenseNet 121	0.952	0.991	**0.998**	0.063	2.874	0.096
HMA-Depth-DenseNet 161	**0.955**	**0.993**	**0.998**	**0.060**	2.850	**0.092**

* ↑ indicates that the performance is better when the value is greater; ↓ indicates the performance is better when the value is lower. The bold value represents the best value for each metric.

According to the table above, HMA-Depth method outperforms other methods for all the metric, except the root mean square error (RMSE) metric, which is a little larger than DORN method. In addition, Fig.3 shows some qualitative results to prove the visualization performance. It compares BTS method and HMD-Depth method by depth estimation results of two RGB images from KITTI dataset. It can be seen that HMD-Depth method can generate more accurate boundary of the objects and more smooth surface.

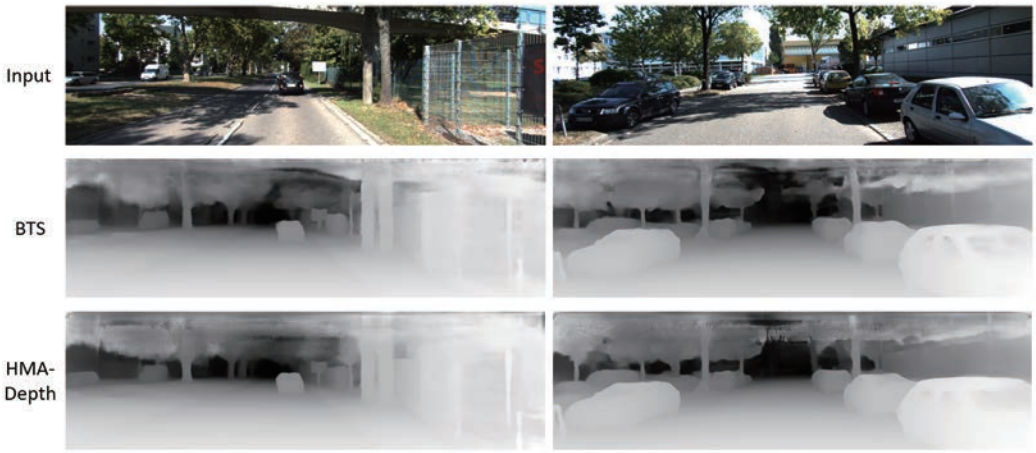


Fig.3 Visualization results of the KITTI dataset

4. Conclusion

In this work, a new method, name HMA-Depth method, is proposed for depth estimation of single RGB images. It utilizes the advantages of different scales of the image and adopts an attention mechanism to extract the important areas of each scale. Specifically, HMA-Depth makes each scale focus on independent area of the images, aiming to amplify the advantages of each scale. The quantitative results and qualitative results both prove that HMA-Depth method outperforms other methods on KITTI dataset. To obtain more solid comparison, however, another common-used dataset, NYU V2 dataset [20] as an example, need to be used. Besides, some ablation studies are also necessary to prove the effectiveness of 4-scale frame. The ablation study may include 3-scale, 5-scale frame or 4-scale frame without attention mechanism.

Therefore, the experiments on another datasets and ablation experiments will be conducted for the next step. To improve the performance further, combining depth estimation task with semantic segmentation results, which can help smooth the object surface, may also be considered in the future work.

Reference

[1] Silvan Weder, Johannes Schonberger, Marc Pollefeys, and Martin R Oswald, “RoutedFusion: Learning real-time depth map fusion,” in IEEE CVPR, 2020, pp.4887–4897.

[2] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al., “KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera,” in Annual ACM Symposium on User Interface Software and Technology, 2011, pp. 559–568.

[3] Jamie Watson, Oisin Mac Aodha, Daniyar Turmukham-betov, Gabriel J Brostow, and Michael Firman, “Learning stereo from single images,” in ECCV, 2020, pp. 722–740.

[4] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf, “Consistent video depth estimation,” arXiv preprint arXiv:2004.15021, 2020.

[5] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao, “Deep ordinal regression network for monocular depth estimation,” in IEEE CVPR, 2018, pp. 2002–2011.

[6] David Eigen, Christian Puhrsch, and Rob Fergus, “Depth map prediction from a single image using a multi-scale deep network,” in Advances in Neural Information Processing Systems, 2014, pp. 2366–2374.

[7] Fayao Liu, Chunhua Shen, and Guosheng Lin, “Deep convolutional neural fields for depth estimation from a single image,” in IEEE CVPR, 2015, pp. 5162–5170.

[8] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen, “Estimating depth from monocular images as classification using deep fully convolutional residual networks,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 11, pp. 3174–3182,2017.

[9] Bo Li, Chunhua Shen, Yuchao Dai, Anton Van Den Hen-gel, and Mingyi He, “Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs,” in IEEE CVPR, 2015, pp. 1119–1127.

[10] Peng Wang, Xiaohui Shen, Zhe Lin, Scott Cohen, Brian Price, and Alan L Yuille, “Towards unified depth and semantic prediction from a single image,” in IEEE CVPR, 2015, pp. 2800–2809.

[11] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, NicuSebe, and Jian Yang, “Pattern-affinitive propagation across depth, surface normal and semantic segmentation,” in IEEE CVPR, 2019, pp. 4106–

4115.
[12] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh, “From big to small: Multi-scale local planar guidance for monocular depth estimation,” arXiv preprint arXiv:1907.10326, 2019.
[13] Zhaofeng Niu, Yuichiro Fujimoto, Masayuki Kanbara, and Hirokazu Kato. "HMA-Depth: A New Monocular Depth Estimation Model Using Hierarchical Multi-Scale Attention," IEEE 17th International Conference on Machine Vision and Applications (MVA), pp. 1-5, 2021.
[14] Adam Paszke, Sam Gross, Francisco Massa, AdamLerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al., “Pytorch: An imperative style, high-performance deep learning library,” in Advances in Neural Information Processing Systems, 2019, pp. 8026–8037.
[15] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun, “Vision meets robotics: The KITTI dataset,” International Journal of Robotics Research (IJRR), 2013.
[16] Ashutosh Saxena, Min Sun, and Andrew Y Ng, “Make3D: Learning 3D scene structure from a single still image,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 5, pp. 824–840, 2008.
[17] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid, “Learning depth from single monocular images using deep convolutional neural fields,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no.10, pp. 2024–2039, 2015.
[18] Yevhen Kuznietsov, Jorg Stuckler, and Bastian Leibe, “Semi-supervised deep learning for monocular depth map prediction,” in IEEE CVPR, 2017, pp. 6647–6655.
[19] Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan, “Enforcing geometric constraints of virtual normal for depth prediction,” in IEEE ICCV, 2019, pp. 5684–5693.
[20] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus, “Indoor segmentation and support inference from RGBD images,” in ECCV, 2012, pp. 746–760.

Python 環境(2021 年版)

塙 敏 博

東京大学情報基盤センター

１．はじめに

東大情報基盤センターで運用しているスパコンを利用して，Deep Learning を始めとした機械学習ユーザが増えていることから，2019 年 7 月に「スパコンで機械学習 (Python 環境構築編)」として記事を掲載しました[1]。多くの方に参考にしていただいたようで，お役に立てて幸いです。

ところが，当該記事でベースに用いた Anaconda[2]について，2020 年 4 月に利用規約が変更され，無償利用の条件が厳しくなっております。東大情報基盤センターのシステムでも注意喚起と回避策の概略についてご紹介します。

また，5 月には，東大情報基盤センターに導入された Wisteria/BDEC-01 が導入され，Odyssey ノードでも Aquarius ノードでも Python は利用可能です。ここでは Aquarius ノードでの利用を想定して，miniconda，JupyterHub/JupyterLab，singularity による利用方法を説明します。

２．Anaconda の利用規約

現在，Anaconda の利用規約は 2021 年 5 月 26 日版となっています[3]。利用規約では，無償利用では商用(commercial activity)は許可されないとあり，以下は commercial activity ではない，としています。

- 個人がビジネス以外の個人的な目的で使用する場合
- 教育機関の学生または従業員が教育活動に関連して使用する場合
- 非営利団体の従業員やボランティアが、慈善事業に関連して使用する場合
- 非営利研究機関が、非商業的な活動に関連して使用する場合で、収益につながらない場合
- 総従業員数が 200 人未満の企業にて使用する場合

ここから考えると，当センターのスパコンシステム(JCAHPC の OFP も含む)では，大半のユーザは commercial activity とみなされることになりそうです。

そこで 2021 年度より，Anaconda は取りやめ，Anaconda 互換の Miniconda および conda-forge の利用に切り替えました。

３．Miniconda と conda-forge の利用

Miniconda は，Anaconda でも使われている管理コマンド conda のみを含む最小限のパッケージです[4]。conda-forge とは，*conda で利用可能なパッケージのリポジトリで，Anaconda 社ではなくコミュニティーで管理されているものです[5]。Anaconda 社としても，Miniconda と conda-forge の組み合わせであればライセンスを侵害しない，と回答しています[6]。

４．Python 環境の選択，環境構築

4.1 インストール済みモジュールの利用

当センターのスパコンシステムでは，ユーザ環境の切り替えのために Environment Modules [7]を用いています。Reedbush，Oakforest-PACS，Oakbridge-CX，Wisteria/BDEC-01 全てのシステムで，python を用いるためのモジュールを用意しています。また，Reedbush，Oakbridge-CX，Wisteria/BDEC-01 では，miniconda のモジュールを用意しています。運良く，使いたいバージョンのソフトウェアがモジュールとして存在していれば，すぐに使用することができます。

- モジュールの一覧表示，その後使いたいモジュールをロード（Reedbush の例）

```
$ module avail
…
-------- /lustre/app/modulefiles/cuda10 -------
…  miniconda3/py39_4.9.2  … pytorch/1.8.1 …
…
$ module load pytorch/1.8.1
All modules were reset, cuda10/10.0.130 miniconda3/py39_4.9.2 pbsutils intel/19.4.243
openmpi/3.1.4/intel gnu/gcc_7.5.0 are loaded
```

※ プロンプト，例えば [y12345@reedbush-h1 ~]$ を $ と表す。また，コマンドラインで入力する文字列は下線で表す。以下同様。

- モジュールの一覧表示（Oakforest-PACS，Oakbridge-CX，Wisteria/BDEC-01 で可能）

```
$ show module
ApplicationName   ModuleName                    Node          BaseCompiler/MPI
-------------------------------------------------------------------------------
…
Miniconda         miniconda/py38_4.9.2          aquarius
…
PyTorch+Horovod   pytorch-horovod/1.8.1-0.21.3  aquarius      cuda/11.1
```

4.2 Miniconda の利用

機械学習やデータ解析でよく用いられる Python ですが，前提にしている Python のモジュール・ライブラリのバージョンがツールごとに異なり，依存関係がたちまち成立しなくなるため，一つの環境にまとめることは，ほぼ不可能です。そこで，使用するツールごとに環境を分ける必要が生じてきます。

Python の仮想環境を作って切り替えるツールには，知る限りでは，pyenv，virtualenv，venv，{Ana,mini}conda,最近では pipenv と言ったものが存在しているようです。この中で，conda 系は，BLAS などのライブラリ群が Intel MKL をベースに構成される(pip 系では OpenBLAS がデフォルト)という点や，チャネルを指定して提供元を切り替えられる(intel や pytorch など，conda-forge 以外にもそれぞれチャネルが選べる)という利点があります。また，共用のスパコンで使う上では，ビルド済みバイナリで提供されること(pip ではソースからビルドする物

も多い)，create した環境同士は基本的に干渉せず独立に扱えるなどの点からも，conda 系が扱いやすいと考えています。

2019 年 7 月号に Anaconda による環境構築を執筆しましたが，Miniconda でも同じように実現できます。但し，Miniconda もデフォルトチャネルは anaconda のレポジトリになっていますので，conda-forge に限る必要があります。

```
$ conda config --add channels conda-forge
$ conda config --remove channels defaults
```

この操作をすると，結果的に以下の“.condarc”の channels:の項のようになるはずですのでご確認ください。

さて，今回は，なるべくストレージ容量を節約し，高速に環境を構築することを念頭に，インストール済みの miniconda 環境をローカルに clone（可能な場合にはシンボリックリンク）して使うことを考えてみます。注意すべき点は，計算ノードで実行できるディレクトリにインストールすること，です。例えば，Wisteria/BDEC-01 では **/data** または**/work** 以下，Oakforest-PACS，Oakbridge-CX では **/work** 以下，Reedbush では **/lustre** 以下のディレクトリしか計算ノードは参照できないので，そこにインストールする必要があります。

Wisteria/BDEC-01 を例に，ユーザグループ gg12，ユーザ名 a12345 として，/work/gg12/a12345/miniconda にインストールすることにします。

準備

システムにインストール済みの Miniconda をモジュールでロードします。

```
$ module load aquarius miniconda/py39 4.9.2
$ conda info
…
base environment : /work/opt/local/x86_64/cores/miniconda/py39_4.9.2  (read only)
…
```

もし，ホームディレクトリに“.conda”，“.condarc”がある場合には，元のファイルをリネームして，/work 領域とシンボリックリンクを張って共通化しておきます。

```
$ mv  ~/.conda  ~/.conda.bak   (あれば)
$ mv  ~/.condarc  ~/.condarc.bak   (あれば)
$ touch  /work/gg12/a12345/.condarc
$ mkdir  /work/gg12/a12345/.conda
$ ln  -s  /work/gg12/a12345/.conda*  ~/
```

“.condarc”は以下のように編集します。もし不具合が起こるようでしたら，ファイル中の *softlinks を削除した後に環境の再構築をお試しください。

```
channels:
  - conda-forge
allow_softlinks: true # 容量を節約したい場合
always_softlink: true # 容量を節約したい場合
# root_prefix: /work/gg12/a12345/miniconda # miniconda をインストールするpath
```

クローン

上記の base environment のパスから，自分のディレクトリにクローンします。

```
$ conda create -p /work/gg12/a12345/miniconda --clone base
$ conda install -p /work/gg12/a12345/miniconda conda
```

conda および全モジュールをアップデートします。

```
$ conda update -p /work/gg12/a12345/miniconda conda
$ conda update -p /work/gg12/a12345/miniconda --all
```

その上で“.condarc”中の `root_prefix`の前のコメント `#` を削除し，有効にします。

使い方

以下のようにしてconda環境を有効にします。

```
$ source /work/gg12/a12345/miniconda/etc/profile.d/conda.sh
```

これによって conda コマンドも使えるようになります。さらに activate すれば準備完了です。

```
$ conda info --envs
# conda environments:
#
base              *  /work/gg12/a12345/miniconda
$ conda activate
(base) $
```

ここでは，実際に執筆時点で最新版のpytorchをインストールしてみます。

```
(base) $ conda create -n pytorch pytorch=1.10.0=cuda112py39h4e14dd4_0 cudatoolkit=11.2
(base) $ conda activate pytorch
(pytorch) $
```

以下に`conda`コマンドの主な使い方をまとめておきます。

コマンド	意味
conda create -n 環境 [-c チャネル] パッケージ	指定された名前の環境の作成
conda create -p パス [-c チャネル] -clone 元環境	元環境をクローンしてパスの名前の環境を作成
conda info [--envs]	環境やインストールパスなどの表示
conda install [-c チャネル] パッケージ[=バージョン等]	現在の環境にパッケージをインストール
conda list	パッケージ一覧表示
conda search パッケージ [-c チャネル] [--info]	パッケージの検索，バリエーションや依存関係を表示
conda remove パッケージ	パッケージの削除
conda remove -n 環境 --all	環境の削除
conda update [-c チャネル] パッケージ or --all	パッケージの更新

５．JupyterHub の利用

Wisteria/BDEC-01システムでは，Webブラウザを用いてPythonを利用可能なJupyterHub環境を提供しています[8]。以下のURLにWebブラウザで接続することで利用できます。サインインの際は，ポータルと同じパスワードを利用してください。なお，Mac環境の方は，Safariの場合にはターミナルで文字が入力できない場合があります。Firefox，Chromeなどをご利用ください。

https://wisteria08.cc.u-tokyo.ac.jp:8000/jupyterhub/

Wisteria/BDEC-01 システムでは，JupyterHub 中の Jupyter Notebook から，pysub コマンドを利用して，Notebook 上に記載されている Python コードの行番号を指定してジョブ投入ができます。

あらかじめ，以下のテンプレートを ~/.notebook/template_job.sh に用意しておきます。

```
#!/bin/sh
#PJM -L rscgrp=share-a
#PJM -L gpu=1
```

また，カレントディレクトリを/work に移動しておきます。

```
[1] !pwd
/home/a12345/.notebook
[2] %cd /work/gg12/a12345
/work/01/gg12/a12345
```

続いて以下のように Python のスクリプトを入力したとします。一旦入力を完了させて行番号が振られたことを確認します。

```
[3] import os
    import numpy as np
    A=np.random.rand(5,5)
    B=np.random.rand(5,5)
    print (A,B)
    C=np.dot(A,B)
    print (C)
```

続いてこの行番号 3 を使って pysub でジョブをサブミットします。その前に一旦左上の save ボタンを押してこの notebook を保存しておきます。この notebook のファイル名が“Untitled.ipynb”だとすると，以下のように実行することができます。

```
[4] !pysub  -f  Untitled.ipynb  3
```

JupyterHub には，ターミナル機能もありますので，“New Launcher”からターミナルを起動して結果を確認することも容易です。ただし，この結果出力を再び Python で使用して解析を行う，といった連携はまだできるようになっていません（実現方法は現在検討中です）。

自分好みの JupyterLab を構成する

JupyterLab では，様々な拡張機能が利用可能です。デフォルトでは，拡張機能が利用できるように見えますが，共有領域に設定を書き込もうとしてしまうため，うまくいきません。ここではそれを回避する方法を説明します。

まず，**4.2** 節で述べた方法で，JupyterLab の環境を構築します。拡張機能に必要なモジュールもあらかじめインストールしておきます。

```
(base) $ conda create -n myjupyter jupyterlab
(base) $ conda activate myjupyter
(myjupyter) $
```

次に，~/.jupyter/jupyter_lab_config.py を作成し，上で構成した myjupyter 環境における JupyterLab のパスをセットします。ファイルの内容は以下の 1 行です。

```
c.LabApp.app_dir = '/work/gg12/a12345/miniconda/envs/myjupyter/share/jupyter/lab'
```

最後に，JupyterLab 上で，“File”→“Hub Control Panel”→“Stop My Server”→“Start My Server”の順に再起動すると，自分が構築した JupyterLab 環境を使用して起動します。拡張機能のいくつかは，JupyterLab の環境とバージョンが合わずに動作しないものも見受

けられますが，是非ご利用いただき，便利な使い方がありましたらご紹介ください。

注意点としては，ブラウザから切断しただけでは，環境が起動中のまま残ることです。明らかに不要な場合には，“File”→“Log Out”を選んでください。逆に言えば，明示的にログアウトするまでは処理がそのまま継続できます。また，複数のユーザの Jupyter 環境が同じサーバ上で動作しますので，あまり重い処理は実行しないようにお願いいたします。

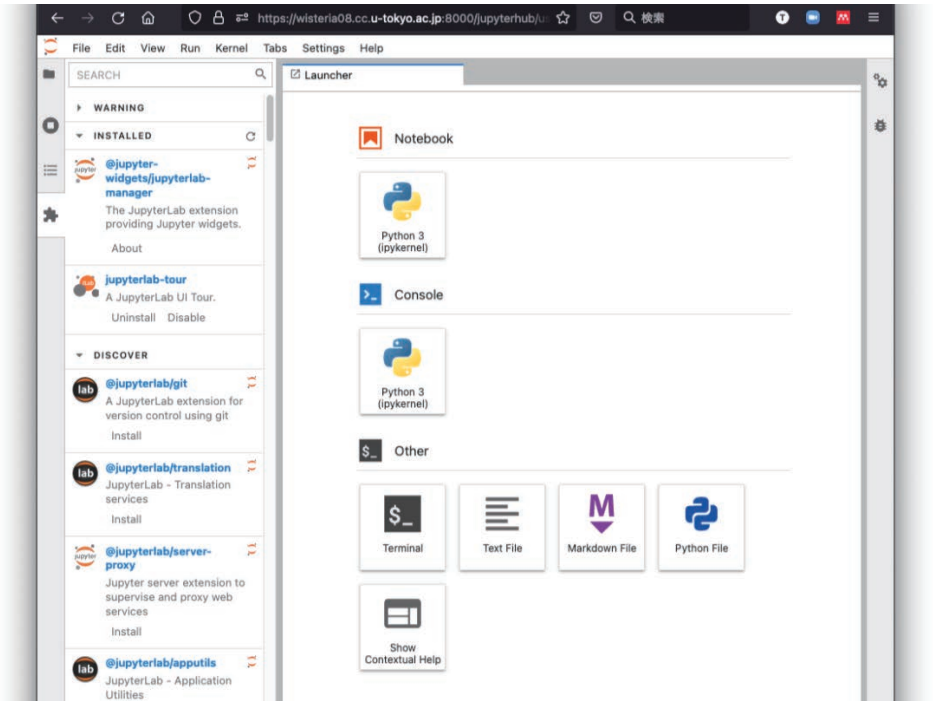


図 1 JupyterLab の画面

6．Singularity の利用

例えば TensorFlow の場合，CUDA ライブラリ等の依存関係の問題で，conda，pip 等で環境構築するのは非常に困難です。このような場合，コンテナ仮想化によって，開発元などが作成したイメージ一式を使い，確実に動く環境を容易に手に入れることができます。

コンテナ仮想化環境として，お手元の PC では，Docker コンテナを利用することが多いと思いますが，スパコンでは Docker ではなく，Singularity というソフトウェアを利用します[9]。

これ以降は，Docker Hub に登録されている Tensorflow コンテナ，具体的には https://hub.docker.com/r/tensorflow/tensorflow の“latest-gpu”タグで登録されているものを，tf-image.file というファイル名でダウンロードする場合を例に説明します。

```
$ module load singularity
$ singularity build tf-image.file docker://tensorflow/tensorflow:latest-gpu
```

（注）ファイルサイズは 3 GB 弱ほどになりますので実行の際はご注意ください。

ダウンロード元としては以下のような既存のコンテナリポジトリの指定が可能です。

- docker:// Docker Hub
- shub:// Singularity レジストリ

実行する際には，以下のようにします(Wisteria/BDEC-01 の場合です。Singularity のバージョンによってオプションが異なる場合があります)。

```
$ singularity exec --nv --bind $PWD tf-image.file python スクリプト名
```

Singularity の主なコマンドの使い方を以下にまとめておきます。

コマンド	意味
singularity build [--sandbox] container.img ソース	コンテナを singularity 用に build して container.img に保存する。--sandbox の場合は，container.img 以下のディレクトリに展開される。
singularity shell [--nv] [--writable] [--fakeroot] container.img	コンテナ環境内で閉じて実行する。--nv は NVIDIA GPU を利用する場合に用いる。--writable はコンテナイメージを書き込み可能にする場合，--fakeroot と合わせて使用する。
singularity exec [--nv] [--writable] [--fakeroot] [--bind dir1[:dir2]] container.img コマンドライン	コンテナ環境内にあるコマンドを用いてコマンドラインを実行する。--bind はコンテナ外の dir1 をコンテナ内の dir2 にマップする。dir2 を省略すると dir1 がコンテナ内でも使われる。その他のオプションは shell の場合と同じ。

Wisteria/BDEC-01 におけるジョブスクリプトは以下のようになります。カレントディレクトリの keras-tf-mnist.py を使う例です。特に jobenv の指定は忘れないようにしてください。

```
#!/bin/bash
#PJM -L rg=share-a
#PJM -L gpu=1
#PJM -g gg12
#PJM -L jobenv=singularity
#PJM -L elapse=00:15:00
#PJM -j
module load aquarius cuda singularity
singularity exec --bind $PWD /work/gg12/a12345/tf-image.file python keras-tf-mnist.py ¥
>& keras-tf-mnist.log.$PJM_JOBID
```

５．おわりに

本稿では，Wisteria/BDEC-01 の環境を中心に，Python 環境の構築方法について，最新の状況を説明しました。

新しい UI として準備した JupyterHub は，スパコンに対するハードルを下げ，利便性を高める環境として我々も期待しています。

また，Singularity は，環境構築のコストを下げる切り札として，スパコンでも広く使われるようになり，NVIDIA によるスパコン向けベンチマークの配布もコンテナイメージを使って行われるようになっています[10]。当センターでは，早くから全システムに Singularity を整備してきました。以前は MPI 環境への対応に難がありましたが，Wisteria/BDEC-01 では特に不都合なく使えています。

Python 環境は(Jupyter も)，アップデートのサイクルが非常に早いので，また大きな変化がありましたらご報告したいと思います。本執筆記事の感想，コメントをお待ちしております。

参 考 文 献

[1] 塙 敏博，スパコンで機械学習(Python 環境構築編)，スーパーコンピューティングニュース，Vol. 21，No. 4，2019 年 7 月
https://www.cc.u-tokyo.ac.jp/public/VOL21/No4/11.201907python.pdf

[2] Anaconda Inc.，Anaconda Installers & Packages，https://repo.anaconda.com

[3] Anaconda Inc.，Terms of Service，https://www.anaconda.com/terms-of-service

[4] Miniconda，https://docs.conda.io/en/latest/miniconda.html

[5] conda-forge，https://conda-forge.org/

[6] Anaconda CEO の回答，
https://www.reddit.com/r/Python/comments/iqsk3y/anaconda_is_not_free_for_commercial_use_anymore/g4xuabr/

[7] Environment Modules，http://modules.sourceforge.net/

[8] Jupyter，JupyterHub，https://jupyter.org/hub

[9] Sylabs.io，Singularity，https://sylabs.io/singularity/

[10] NVIDIA Corp.，NVIDIA HPC Benchmarks，
https://ngc.nvidia.com/catalog/containers/nvidia:hpc-benchmarks

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，8 年目となった。2021 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。

まず今年度もコロナ禍により多くの授業日程が非常事態宣言中になり，対面では行えなかった。前半を座学とし，後半を演習としているが，前半の授業は，オンライン双方向授業を行い，質問対応はチャットなどで対応した。後半の演習の授業では，緊急事態宣言が解除されている期間もあり，一部希望者は対面でも対応し，簡単なプログラムの実行や作成をさせた。また今年度は OpenMP や MPI のプログラムの作成，デバッグは Google Colaboratory 上にて C 言語で行わせるようにした。この環境では実行結果がブラウザ上で確認でき，こちらが作成したコードも参照可能となるため，スムーズに学習させることができた。そのためスパコンの利用は，デバッグが終わったあとに，ジョブを投入して計測をするようにさせた。

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を渡し，それの OpenMP，MPI を用いた並列化とした。

成績については，期末のプログラミングを伴うレポート課題と，対面で設定した期末試験により評価した。今年度の履修人数は 77 人であり，実際に単位を取得したのは 65 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，7 月末まで授業用アカウントを利用させて頂いた。

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学会発表を行う学生もいる。学部 3 年生のうちから本制度の支援を受け，スパコンの利用に関して認証手続きからジョブ投入の方法，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，この後で学生が卒業研究を進める上でも有用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
4 月 13 日	15:35-17:20 対面	1-1　CPU の仕組み，OS の機能 プロセス，キャッシュ，仮想メモリ空間
4 月 20 日	15:35-17:20 対面	1-2 並列性の分類 {命令，スレッド，プロセス}レベル並列性 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
4 月 27 日	15:35-17:20 オンライン	キャッシュの一貫性，計算と通信のコスト， αβモデル 2-2 共有・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
5 月 11 日	15:35-17:20 オンライン	3-1 分散メモリ型の並列処理 MPI 基礎
5 月 18 日	15:35-17:20 オンライン	集団通信関数 プログラム例：数値積分，内積コード
5 月 25 日	15:35-17:20 オンライン	3-2 Google Colaboratory を利用した MPI のプログラム演習
6 月 1 日	15:35-17:20 オンライン	3-3 マルチスレッドと排他制御 mutex_lock，デッドロックとその検知
6 月 8 日	15:35-17:20 オンライン	3-4 OpenMP の書き方と例題 False sharing 実行例等
6 月 15 日	15:35-17:20 オンライン	4-1 並列プログラムの実践 1 計算環境，認証，実行手順， 数値積分による円周率の計算の OMP 化
6 月 22 日	15:35-17:20 対面，オンライン同時	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
6 月 29 日	15:35-17:20 対面，オンライン同時	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）
7 月 6 日	15:35-17:20 対面，オンライン同時	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）
7 月 13 日	15:35-17:20 対面	授業内 期末テスト

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

東京大学情報基盤センター

本稿は 2021 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 Wisteria/BDEC-01 スーパーコンピュータのシミュレーションノード群（Odyssey）を使用する。Wisteria/BDEC-01 は本年度導入されたシステムであり、本実験では今年度から使用することとなった。

本年度は 2 組，5 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：

- Wisteria/BDEC-01 へのログイン
- 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

第 157 回お試しアカウント付き並列プログラミング講習「GPU プログラミング入門」

星野哲也

東京大学情報基盤センター

本稿は，2021 年 6 月 9 日にオンライン会議システム Zoom を用いて開催された『第 157 回お試しアカウント付き並列プログラミング講習会 : GPU プログラミング入門[1]』の開催報告である。

本講習会では、GPU を用いるために必須である、GPU の特性やプログラミング手法に関する講義および実習を行う。対象 GPU としては主に NVIDIA 社の Tesla GPU を、GPU 向けの並列化プログラミング環境としては主に OpenACC を用いる。実習では拡散方程式のカーネルなど、HPC において利用される基本的な計算問題を題材として、GPU 向けのプログラムを作成する方法や、GPU プログラミングを行う上で役に立つツールの使用方法、CPU-GPU 間のデータ転送の最適化を適用するまでの手順を学ぶ。本講習会は 2021 年 5 月 14 日より運用を開始した Wisteria/BDEC-01 の Aquarius ノードを用いて実習を行なった。Aquarius ノードには 8 基の A100 GPU が搭載されているが、講習会受講者が利用できるのは 4 基までである。受講者には 1 ヶ月有効なアカウントが与えられ、1 ノード 30 分までのジョブが実行可能である。講義内容の詳細については，ウェブページ [1] 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。また講習会に用いたサンプルコードは https://github.com/hoshino-UTokyo/lecture_openacc にて公開されている。

合計 12 名の事前登録者があり，11 名（学生 : 4 名，研究機関 : 3 名，企業 : 4 名）が受講した。講習会終了後にアンケートを実施した（10 名より回収）。表 1 は質問項目と回答（5 段階評価）の人数分布である。

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	9		
(b) 講習会講義内容（プレゼン）	簡単⇔難	1		7	2	
(c) 配布資料内容	簡単⇔難		1	8	1	
(d) サンプルプログラム内容	簡単⇔難		1	7	2	
(e) 満足度	不満⇔満足		1	2	3	4

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

第161回お試しアカウント付き並列プログラミング講習会「Wisteria実践」実施報告

塙　敏博
東京大学情報基盤センター

2021年9月9日（木）に、第161回お試しアカウント付き並列プログラミング講習会「Wisteria実践」が開催されました。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、学部学生：8名、大学院学生：4名、大学・研究機関教職員：9名、企業の方：1名、計22名の方にご参加いただきました。

1ヶ月有効となるお試しアカウントが与えられ、5月14日に運用を開始したばかりのWisteria/BDEC-01スーパーコンピュータシステムを用いて、Wisteria/BDEC-01の利用方法、特に、OpenMP, OpenACCおよびMPI(Message Passing Interface)を用いた実践的なプログラミングと実行方法について、講義と演習が以下の日程で行われました。

当日のプログラムを、以下に掲載します。初回だった前回は，全体的に時間が不足していたので，今回は午前中から実施しました。

- 9月9日（木）
  - 10:30 – 12:00　Wisteria/BDEC-01システム紹介
  - 13:30 – 15:15　Odysseyノード：A64FXにおけるOpenMP最適化、MPI+OpenMPハイブリッド並列、性能分析(講義＋演習)
  - 15:30 – 17:15　Aquariusノード：A100 GPUの利用、OpenACC, MPI並列化、性能分析(講義＋演習)
  - 17:15 – 17:30　Q&A

17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示します。

プログラミング経験については、5年未満が8名、5年以上が9名、35年という方もいらっしゃいました。並列プログラミングについては、知識を前提にしていましたが、経験なしの方も9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、Fortranが12名と多いですが、Python, CとC++が同数の8名（複数回答可）となり、

Python のユーザが引き続き増えてきています。

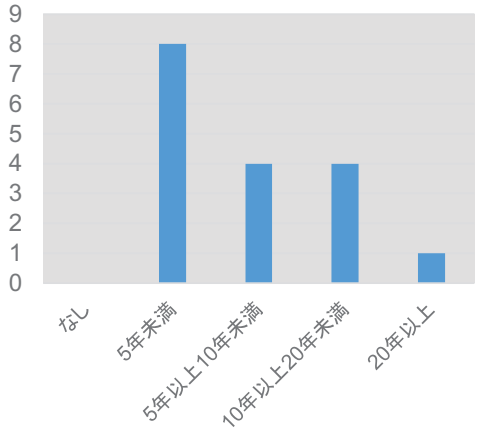


図 1 プログラミング経験

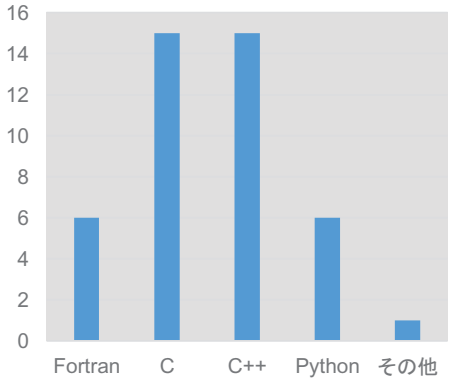


図 2 普段使用するプログラミング言語

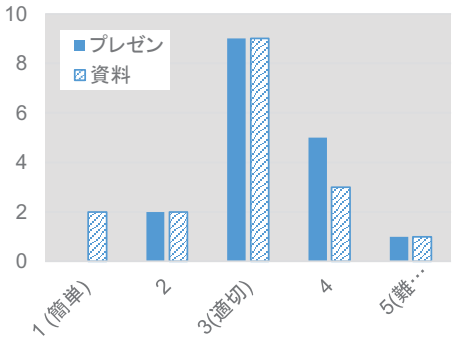


図 3 講習会の内容

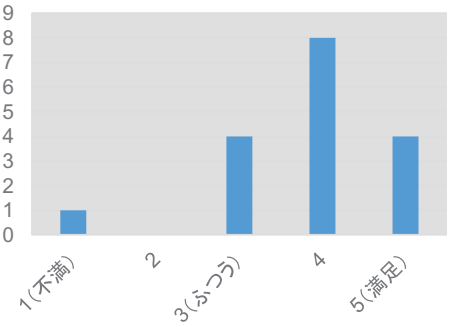


図 4 講習会参加の満足度

講習会の満足度は図 4 に示すようにおおむね満足度が高かったようです。平均値は 3.8 でした。

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主な回答は

- いろいろなデバイスで参加できる
- スライドを使用し実践も織りなされていたこと
- 具体的な実践演習の応用方が見られるので，利用にあたっての注意点等が確認しやすい点
- マルチディスプレイで講習を受けられる
- 地方からの参加であり時間と費用で大変助かっています

一方悪かったことについては、

- 見学会がオンラインだったこと。実際に Wisteria を見たいです。

- 内容が複雑だったこと
- 基本的にエラーを独力で解決する必要があること（今回はエラーが出なかったので問題なかったです）
- 特になし（2名）

との意見をいただきました。遠隔地からの参加、使いやすい環境で講義を受けられることで、概ね好評だったようです。

また、以下のような感想をいただきました。

- 午前中の概要説明はもう少し短くしても良いのではと思います
- 非常によくわかりました。
- Wisteria のマニュアルに書ききれない高速化のコツを聞けたのが良かったです。

同様の講習会があれば、「また受けたい」という回答が 13 名、「どちらともいえない」が 3 名で、感想からもその他の講習会にも期待されていることが伺えます。

Oakforest-PACS においては「KNL 実践」講習会と「OFP 実践」講習会を合わせて 8 回開催し、好評を得ました。Wisteria/BDEC-01 についても同様に、今後も内容を見直しながら、「Wisteria 実践」を定期的に開催していきたいと考えております。

また、本講習会資料、録画データも公開しております。復習に役立てたり、参加が適わなかった方に自習に使っていただければ幸いです。

しばらくは新型コロナウイルス感染症対策でオンラインのみの開催が続きます。オンライン講習会にはオンサイト講習会にない利点があることも分かってきたので、今後オンサイト開催が可能になってもオンラインを考慮しながら内容を検討していく予定です。

以上

第 162 回お試しアカウント付き並列プログラミング講習会「OpenFOAM 初級」実施報告

今野 雅

東京大学情報基盤センター客員研究員

2021 年 9 月 28 日（火），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）、オープン CAE 学会との共催で，第 162 回お試しアカウント付き並列プログラミング講習会「OpenFOAM 初級」がオンラインで開催された。本講習会は，センターに設置されたスーパーコンピューター(以降，スパコン)の利用促進とスパコンを用いた数値流体解析の普及を目的として実施されたものである。なお，本講習会はセンターのお試しアカウント付き並列プログラミング講習会として行われた 20 回目の OpenFOAM の講習会である。受講者は，大学・研究機関教職 1 名，大学院学生 5 名，学部学生 1 名，企業の方 4 名であり，事前申込者 13 名，受講者合計は 11 名であった。センターが運営するスパコン Wisteria/BDEC-01 を用い，Wisteria/BDEC-01 の概要，利用方法，OpenFOAM の演習が 1 日終日の日程で行われた。当日のプログラムを表 1 に掲載する。なお，講習会終了後約 1 ヶ月有効なお試しアカウント(Wisteria-O，最大ノード数 12，最大実行時間 15 分)が受講者に与えられた。

表 1 講習会プログラム

【2021 年 9 月 28 日（火）】
10：00 - 12：00　自動車空力解析演習 I
　　イントロダクション（約 5 分間）
　　Wisteria/BDEC-01 概要（約 55 分間）
　　講習会の準備（約 15 分間）
　　module による OpenFOAM の環境設定（約 15 分間）
13：00 - 18：00　自動車空力解析演習 II
　　解析モデル（約 40 分間）
　　ベース格子生成（25 分間）
　　ベース格子の可視化（約 30 分間）
　　格子生成（60 分間）
　　最終格子の可視化（約 10 分間）
　　流体解析（約 50 分間）
　　解析結果の可視化（約 10 分間）
　　実測値との比較（約 10 分間）
　　解析演習の説明（約 10 分間）
　　その他の話題・質疑応答

講習会終了後のアンケート集計結果(回答数 10)を表 2 に示すが，参加した満足度の平均は 5 点満点中，4.80 と高かった。また，参加者から表 3～5 に示すご意見を頂いた。今後の講習会の参考にしたい。

表 2 アンケート集計結果

評点	講習会の時間		講習会の講義内容(プレゼン)		配布資料の内容		サンプルプログラム内容		参加した満足度	
1	短い	0	簡単	0	簡単	0	簡単	1	不満	0
2		0		0		1		0		0
3	適切	6	適切	7	適切	8	適切	7	普通	0
4		4		2		1		2		2
5	長い	0	難	1	難	0	難	0	満足	8
	平均	3.40	平均	3.40	平均	3.00	平均	3.00	平均	4.80

表 3 Zoom によるオンライン講習会で良かったこと(原文ママ)

- 使いやすい、音声や画像も良い
- 大画面で実習することができること.
- 時間、場所が選べること。
- 遠方からも参加できる

表 4 Zoom によるオンライン講習会で悪かったこと(原文ママ)

- 特にありません
- 人数が増えるとエラー時対応が難しい。
- ありません

表 5 本講習会に対するご意見(原文ママ)

- 今野先生、どうもありがとうございました。
- OpenFOAM の説明が少し難しかった.
- 少人数であったので、良かったと思います。先生の指導も上手でした。Windows でのフォワーディングができていません。やり方が分かりましたら、ご通知をお願い致します。
- 今回から，vncserver を用いてディレクトリの編集を行いましたが，動きにラグや小さくて見づらかったので，以前のように x 転送を用いて行っていただきたいです.

第163回お試しアカウント付き並列プログラミング講習会「スーパーコンピューター超入門」

芝　隼人

東京大学情報基盤センター

2021年9月30月に、第163回お試しアカウント付き並列プログラミング講習会「スーパーコンピューター超入門」（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））を開催したので、本稿にてその実施報告を行う。

東京大学情報基盤センターにおいて「超入門」と題した講習会は、2020年9月18日（第138回）に初めて実施し（初回は河合・芝の２名により担当）、2021年4月26日（第152回）に２回目を行い、いずれも好評を博した。いずれの回も定員に到達、締切より前に申込を打ち切らせていただくことなったことから、今年度２回目となる同趣旨の講習会を行った。特に、前回は東大を含めた大学学部・大学院における学期中の授業と日程が重なったことで申し込めなかった方も多いとみられる。多くの学部では新学期の開始前となる日程を選ぶことで、学部生・大学院生の参加が促進されることを意図した。この意図は適切に反映され、当日出席者８名のうち学部生１名、大学院生５名であった。

今回も、内容としては前回までと同様Linuxの簡単な操作・ログインなどの事柄から始めてスーパーコンピューターを用いるための基本を学ぶ講義及び実習を行った。講習会の時間割は次の通りとした。

13:00 - 14:00　ガイダンス・初めてのログイン（講義・実習）
14:15 - 16:00　サンプルプログラムによる並列計算超入門（講義・実習）
16:15 - 17:00　より進んだ利用に向けて（講義）

また、第３部については、並列計算に必要なトークンの見積もり方についての講義のほか、統合開発ツールをログインノード上に接続する使用方法の紹介などを行った。

申込受付時にいずれの言語を使用するかのアンケートを行ったが、事前申込のあった10名の内訳として

・FORTRAN　4名
・C/C++　3名
・Python　3名

となっていた。地球惑星科学や機械系、物理学など、科学技術計算分野への従来的なスパコン利用分野からの参加が多かったと考えられる。

次の表１には、受講者からのアンケートの結果の集計を示す。当日参加者の全員からご回答をいただくことができた。

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			8		
(b) 講習会講義内容	簡単⇔難		3	4	1	
(c) 配布資料内容	簡単⇔難		1	7		
(d) サンプルプログラム内容	簡単⇔難		3	4	1	
(e) 参加した満足度	不満⇔満足			1	2	5

アンケートの自由記述欄には次のような意見が寄せられた。

- 話すスピードが早く、追いつくのが少し大変でした。
- 内容については、知りたいと思っていたことが詰まっており、参加してよかったです。ありがとうございました。
- 実際スパコンを使った実習を体験できてよかった。
- 実習部分は少なめに感じましたが、スパコンの使い方の初歩から手を動かして学ぶことができたのは良かったです。最後の 45 分の講義で、この後実際にスパコンを使うとしてどのような技術的、事務的手順を踏めばいいのか全体像が見えてとても有意義に感じました。

また、Zoom によるオンライン開催については次のような感想が寄せられた。

- 現地に行く必要がないため、自宅で受講できたことがよかった。
- 今回は、質問などは出なかったが、質問がある際に対面のほうが気軽にできる。
- 軽い質問であれば質問しやすい。分からないところを直接聞くことができた。
- スライドや講師の画面が手元に表示されているため見やすい。
- 自分が持っているデスクトップ環境から参加可能なのが良い。
- 内容がより高度なものになった場合、支障が出るかもしれないと思いました。

当日の講習会は、前回に引き続きビデオ収録を行ってほぼそのままの形で動画サイト YouTube に掲載、公開した。ビデオを見返してみると私自身の講義技術にまだまだ不足を感じるところであるが、このような講習会の実施を貴重な機会として、引き続き教育等技術の向上に努めてまいりたい。

講義資料および実習資料を当センターの第 163 回講習会ホームページ（ https://www.cc.u-tokyo.ac.jp/events/lectures/163/ ）において公開している。また同じ場所から、上述の講義・実習の YouTube 動画へのリンクを貼っている。

第 164 回お試しアカウント付き並列プログラミング講習「GPU プログラミング入門」

星野哲也

東京大学情報基盤センター

本稿は，2021 年 10 月 5 日にオンライン会議システム Zoom を用いて開催された『第 164 回お試しアカウント付き並列プログラミング講習会：GPU プログラミング入門[1]』の開催報告である。

本講習会では、GPU を用いるために必須である、GPU の特性やプログラミング手法に関する講義および実習を行う。対象 GPU としては主に NVIDIA 社の Tesla GPU を、GPU 向けの並列化プログラミング環境としては主に OpenACC を用いる。実習では拡散方程式のカーネルなど、HPC において利用される基本的な計算問題を題材として、GPU 向けのプログラムを作成する方法や、GPU プログラミングを行う上で役に立つツールの使用方法、CPU-GPU 間のデータ転送の最適化を適用するまでの手順を学ぶ。本講習会は Wisteria/BDEC-01 の Aquarius ノードを用いて実習を行なった。Aquarius ノードには 8 基の A100 GPU が搭載されているが、講習会受講者が利用できるのは 4 基までである。受講者には 1 ヶ月有効なアカウントが与えられ、1 ノード 30 分までのジョブが実行可能である。講義内容の詳細については，ウェブページ[1]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。また講習会に用いたサンプルコードは https://github.com/hoshino-UTokyo/lecture_openacc にて公開されている。

合計 10 名の事前登録者があり，7 名（学生：3 名，研究機関：1 名，企業：3 名）が受講した。講習会終了後にアンケートを実施した（4 名より回収）。表 1 は質問項目と回答（5 段階評価）の人数分布である。

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			4		
(b) 講習会講義内容（プレゼン）	簡単⇔難		1	3		
(c) 配布資料内容	簡単⇔難		1	3		
(d) サンプルプログラム内容	簡単⇔難		1	3		
(e) 満足度	不満⇔満足					4

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

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，A 4 サイズで、ページの余白は上下 20mm、左右 26mm、ヘッダー15mm、フッター 10mm に設定してください。詳しくは原稿様式をご参照ください。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 採用された原稿は、本センターの Web ページに掲載いたします。
https://www.cc.u-tokyo.ac.jp/public/news.php

【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://wisteria-www.cc.u-tokyo.ac.jp/ (Wisteria/BDEC-01) 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	スーパーコンピュータシステム利用申込書提出先 uketsuke@cc.u-tokyo.ac.jp 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
Wisteria/BDEC-01 スーパーコンピュータシステム (Wisteria-O/A)	wisteria.cc.u-tokyo.ac.jp 以下のホストの何れかに接続します※ wisteria0{1-6}.cc.u-tokyo.ac.jp
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
〒277-0882　千葉県柏市柏の葉6-2-3
（電話）　04-7133-4663（ダイヤルイン）

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.6(2021.11)

目　　次

センターから

サービス休止等のお知らせ …… 1
システム変更等のお知らせ …… 3
2022年度の利用申込（新規・継続）について …… 5
東京大学柏キャンパス一般公開 2021（オンライン開催）10/22-29 …… 7
2021年度インターン・後期
　東京大学情報基盤センター「若手・女性利用者推薦」採択課題 …… 8
「大規模HPCチャレンジ」採択課題のお知らせ …… 14
研究成果登録のお願い …… 15
８月・９月のジョブ統計 …… 16

ユーザーから

Numerical simulation of deep-water oil blowouts …… 22
動的膜の分子動力学計算 …… 26
A new method for monocular depth estimation …… 36

資料

Python環境（2021年版）…… 42

教育活動報告

工学院大学３年次講義「並列・分散システム」 …… 51
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 …… 53
第157回お試しアカウント付き並列プログラミング講習会
　「GPUプログラミング入門」 …… 54
第161回お試しアカウント付き並列プログラミング講習会
　「Wisteria実践」 …… 55
第162回お試しアカウント付き並列プログラミング講習会
　「OpenFOAM初級」 …… 58
第163回お試しアカウント付き並列プログラミング講習会
　「スーパーコンピューター超入門」 …… 60
第164回お試しアカウント付き並列プログラミング講習会
　「GPUプログラミング入門」 …… 62

原稿募集 …… 63