Vol.15, No.1 2016

全国共同利用版

広報

新スーパーコンピュータサービス開始

ISSN 1347-3581

京都大学学術情報メディアセンター
Academic Center for Computing and Media Studies, Kyoto University

巻頭言

Vol. 15, No.1 号の発刊に当たって

京都大学学術情報メディアセンター
深沢　圭一郎

本号では，「新スーパーコンピュータサービス開始」というタイトルで、2016 年 10 月から導入が始まっている新スーパーコンピュータシステムについて特集いたします．サブシステム A，B，C からなる新システムですが，まず Xeon Phi（KNL）を搭載するサブシステム A が 2016 年 10 月から導入されており，2016 年 12 月末からサブシステム B と C が稼働を開始しています. 本号では各システムの利用方法を紹介していますが, 新システムはジョブスケジューラが旧システムと変更になっている点がこれまでとの大きな違いの一つになります. 3 つのシステムの中で, 最も性能が高く, ノード数も多く中心的なシステムが，サブシステム A になります．このシステムは世界でも早期に新しい Xeon Phi（KNL）を搭載したシステムであり，2016 年 11 月のスパコン世界ランキング（Top500）では 33 位，国内ランキングでは 4 位の性能を持つシステムになります. また, 新システム A は「Camphor 2」という名前がついておりますが，旧システム A「Camphor」とはかなり異なったシステムです．詳細は特集記事にありますが，Xeon Phi（KNL）は低めの周波数（1.4GHz）で多数のコア（68 コア）からなる CPU であり，SIMD 演算を効率的に行うことで，性能を高めています．そのため，非並列かつ SIMD が効きにくいアプリケーションは旧システム A より性能が出ない（1 コア当たりで）ことも予想されます．そのため，Xeon Phi（KNL）の性能を利用者の方に報告する意味でいくつかのベンチマーク評価を特集では紹介しています. スーパーコンピュータの評価によく利用されるベンチマークではありますが，他のシステムと比較することで，Xeon Phi （KNL）の特徴を少しでもご理解いただけると思います. サブシステム B についてのベンチマーク情報は，Vol. 16 No. 1 で紹介する予定です.

「イベント報告」では，センターが共催し，10 月末に京都大学国際科学イノベーション棟で開催された日本シミュレーション学会の年次大会である「The 35th JSST Annual Conference（JSST2016）」について報告されています．この会議は理工学・産業の多くの専門分野におけるシステム技法からソフトウェア、ハードウェア及び諸分野への応用に向けたシミュレーションの学理と技術に関する研究討論と情報交換を行う場として毎年開催されており，2011 年より国際会議として開催されています．会議では京都大学での開催ということもあり，本センターの中島浩教授に HPC に関する基調講演を行っていただきました．日本シミュレーション学会の会議であるため，スーパーコンピュータ関連ではなく，様々なアプリケーションに関する研究発表が多くある興味深い会議でした.

新スーパーコンピュータシステム導入に伴い，利用者の方にはご不便をおかけしている部分もありますが，安定した，より高効率な計算を行える計算機環境を作れるようにセンタ

ーとしても対応をしっかりとしていきたいと思います．新システムの中でも新しい Xeon Phi（KNL）は現在運用されている Xeon Phi（KNC）や通常の Xeon と比べて最適化手法が異なることが想定されています．センターでは，「プログラム高度化共同研究」などを通じて，皆様の研究，教育に新システムをご活用いただけるようにセンター教職員も尽力していきますので，今後ともご利用，ご支援のほど、よろしくお願いいたします．

新スーパーコンピュータ利用ガイド

尾形 幸亮　　池田 健二　　疋田 淳一

京都大学 企画・情報部

1　はじめに

学術情報メディアセンターでは、2016 年 10 月より新しいスーパーコンピュータ（以下、新スパコン）のサービスを開始しております。新スパコンは、Camphor 2（以下、システム A）、Laurel 2（以下、システム B）、Cinnamon 2（以下、システム C）の 3 種の演算システムおよび大規模ストレージで構成しています。システム A は最新の Intel 社のメニーコアプロセッサである Xeon Phi Knights Landing（KNL）を搭載したノードを 1800 台接続した大規模並列コンピュータです。2016 年 10 月より稼働開始しました。システム B は最新の Intel 社の Xeon Broadwell-EP プロセッサを 2 基搭載したノードを InfiniBand EDR により 850 台を接続した PC クラスタ型のシステムです。システム C は、16 台と少数ですが、3TB の大容量メモリと Intel 社の Xeon Haswell-EX プロセッサを 4 基搭載するノードを InfiniBand EDR 接続したシステムとなっています。システム B および C については、2017 年 1 月より稼働開始するために準備を進めています。これら 3 種の演算システムは高速通信網を経由して 24PB の大容量ストレージに接続しており、演算に必要な大規模データの入出力を高速に行える環境となっています。新システムの性能諸元値を記した構成図を図 1 に示しました。以下、本稿では新スパコンの利用方法について解説します。なお、2014 年から稼働している Camellia（システム E）も引き続きサービスしておりますが、本稿では省略します。

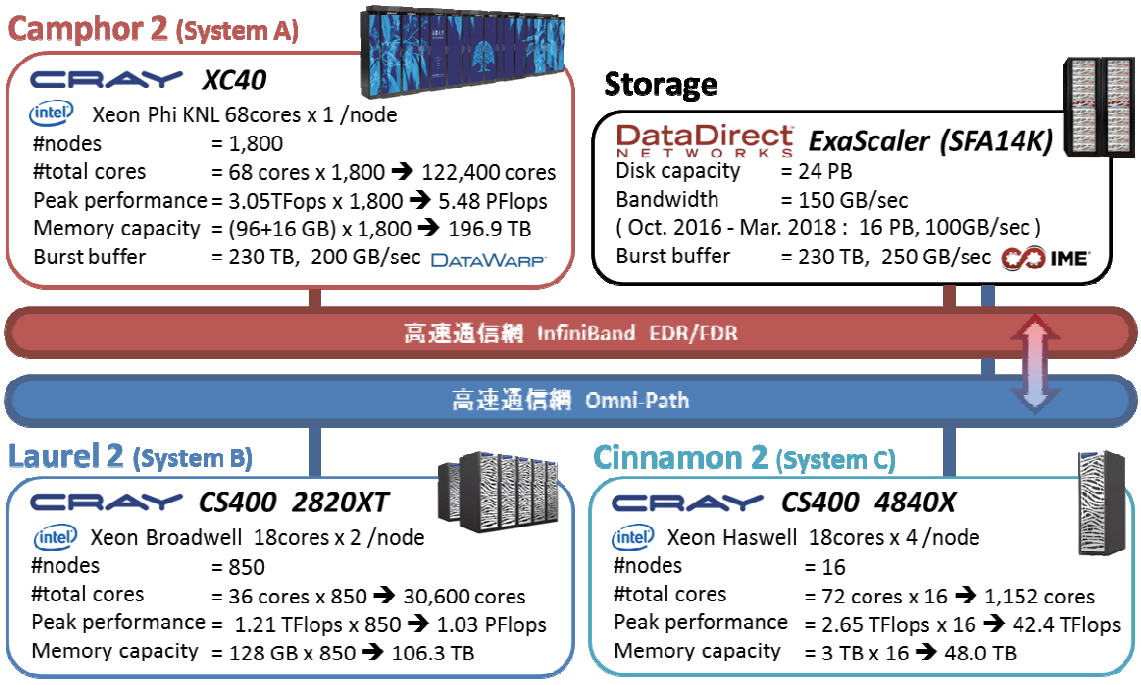


図 1 システム構成図

2　スパコンへのログイン

ログインノードのホスト名は、表 1 に示す通りです。システム A は 2 台、システム B および C はログインノードを共有しており、3 台で構成しています。複数台のログインノードは DNS ラウンドロビンにより負荷分散しています。

表 1　ホスト名

システム	ホスト名（FQDN）
A	camphor.kudpc.kyoto-u.ac.jp （システム A の利用者のみログイン可能）
B	laurel.kudpc.kyoto-u.ac.jp （全利用者がログイン可能）
C	laurel.kudpc.kyoto-u.ac.jp （ログイン後、コマンドでシステム C の環境に切り替え）

スパコンへのログイン手段は、公開鍵認証方式に限定しています。スパコンにログインするには、以下の流れで作業を行ってください。

1. SSH クライアントのインストール
2. 鍵ペアの作成（公開鍵、秘密鍵）
3. 利用者ポータルから公開鍵の登録
4. SSH クライアントで秘密鍵を使用してログイン

詳細な手順は次の URL で案内しています。
http://web.kudpc.kyoto-u.ac.jp/manual-new/ja/login

2.1 X 環境でのログイン

PC X サーバソフトウェアとして Exceed onDemand バージョン 8 を提供しています。Exceed onDemand は、データ圧縮技術により通信トラフィックを抑えているため、遠隔地でも快適に X 環境を利用することができます。インストールや利用方法は、次の URL で案内しています。
http://web.kudpc.kyoto-u.ac.jp/manual-new/ja/login/eod

3　ファイルシステム

3.1 ホームディレクトリ（$HOME）

ホームディレクトリのパスは、図 2 に示すように、/home の後に利用者番号の先頭のアルファベット一文字、次に利用者番号となっています。

利用可能な領域として、利用者あたり 100GB の容量を割り当てています。

記憶領域は、システム A，B，C で共有しています。ファイルシステムの実体は /home0 であり、/home にシンボリックリンクを張っています。

図 2　ホームディレクトリ

3.2 大容量ディスク領域

3.2.1 容量と構成

パーソナル、グループ、専用クラスタの各コースをご利用の方は、大容量ディスク領域を利用できます。割り当て容量は各コースの資源により、表 2 に示す値の通り決まっています(バックアップ領域を含む)。

表 2　コース毎の/LARGE 容量

コース	/LARGE 容量
パーソナル	2TB
グループ　タイプ A1、B1	1 ノードあたり 4TB
グループ　タイプ A2、B2	1 ノードあたり 2.4TB
グループ　タイプ A3、B3	1 ノードあたり 4TB
グループ　タイプ C1	1 ノードあたり 16TB
グループ　タイプ C2	1 ノードあたり 9.6TB
専用クラスタ	1 ノードあたり 4TB

大容量ディスクは、システム A、B、C では /LARGE0 と/LARGE1 の 2 つの領域で構成されており、初期設定では/LARGE1 はバックアップ領域となっています。また、図 3、図 4 に示すように、/LARGE0,1 の直下にユーザ名、グループ名のディレクトリを配置しています。

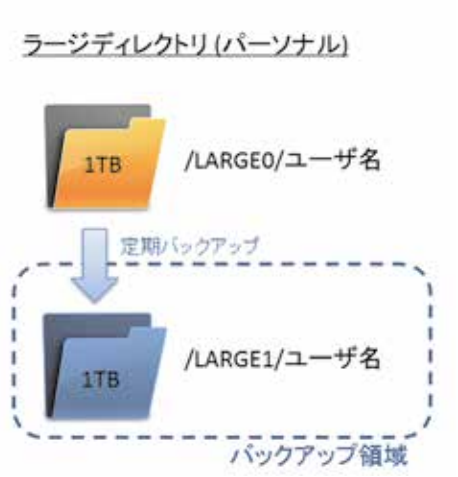


図 3　パーソナルの大容量ディスク

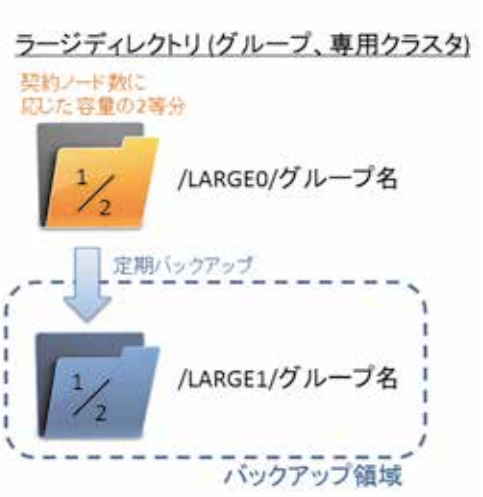


図 4　グループ・専用クラスタの大容量ディスク

3.2.2 バックアップ設定

バックアップ領域は、group_backup コマンドによりバックアップを解除することで通常の領域として利用できます。バックアップ設定の変更は、パーソナルコースの場合は利用者本人、グループ・専用クラスタコースの場合はグループ管理者（サービスの申請者）が行うことができます。

- 現在の設定を確認

```
$ group_backup -g gr10000 -l
 Num  Filesystem      Status  Filesystem        Status
  1)  /LARGE0/gr10000..Safe /LARGE1/gr10000 ... Backup
```

- バックアップを解除

```
$ group_backup -g gr10000 -u 1
/LARGE0/gr10000: Safe   => UnSafe
/LARGE1/gr10000: Backup => UnSafe
```

3.3 利用状況の確認（quota）

ディスクの利用状況はquota コマンドで確認することができます。確認できる項目を表 3 に示します。また、-g オプションを使用することで、大容量ディスクの利用状況を確認することができます。

- ホームディレクトリの利用状況

```
$ quota
Disk quotas for user b59999 (uid 59999):
fsys blocks quota limit grace files quota limit grace
 /home 100 100000000 105000000 - 50 600000 700000  -
```

表 3　利用状況の確認項目一覧

項目	内容
blocks	使用中のファイル容量(KB)
quota	ファイル容量／数の制限値(ソフトリミット)
limit	ファイル容量／数の絶対的制限値(ハードリミット)
grace	制限値越え(ソフトリミット超過)の許容期間
files	使用中のファイル数

4　利用環境のカスタマイズ

4.1 環境設定ファイル

標準のログインシェルは bash に設定しており、ログイン時に環境変数の設定などを行いたい場合は、.bash_profile ファイルや.bashrc ファイルに必要な設定を記述することで反映されます。tcsh を利用する場合には、.tcshrc に設定を記述することで反映されます。

なお、システム A、B、C では、ホームディレクトリが共有されているため、環境設定ファイルは同じものが読み込まれます。複数のシステムを利用している場合、例えば次のように hostname コマンドの結果に応じた条件文を記述することで、各システムに応じた処理を行うことができます。

- .bashrc の例

```
case `hostname` in
    camphor*)
        #システム A の場合の処理
    ;;
    laurel*)
        #システム B/C の場合の処理
    ;;
Esac
```

- .tcshrc の例

```
switch(`hostname`)
    case camphor*:
        #システム A の場合の処理
        breaksw
    case laurel*:
        #システム B/C の場合の処理
        breaksw
endsw
```

4.2 ログインシェルの変更

ログインシェルは chsh コマンドで変更することができます。対応しているシェルは sh、bash、csh、tcsh、zsh、ksh の 6 種類です。

- ログインシェルを tcsh に切り替える

```
$ chsh tcsh
```

4.3 システムからの通知メール転送

バッチ処理システムからのエラー通知など、システムからの通知メールはシステムのローカルユーザ宛てに送信されます。このメールは mutt コマンドでも確認できますが、いち早く閲覧できるようホームディレクトリに.forward ファイルを作成し、普段利用しているメールアドレスに転送することを推奨しています。

転送するには、次のコマンドのメールアドレス部分を自身のものに置き換えて実行してください。

- user@example.com にメールを転送

```
$ echo user@example.com > ~/.forward
```

4.4 メッセージの日本語化

SSH クライアントの文字コードを UTF-8 に設定した上で、次の通り環境変数 LANG を設定することで、コマンド出力、コンパイラリンク、オンラインマニュアル(man)などのメッセージを日本語表示に変更することができます。

- 環境変数 LANG を設定する(bash の場合)

```
$ export LANG=ja_JP.UTF-8
```

- 環境変数 LANG を設定する(tcsh の場合)

```
$ setenv LANG ja_JP.UTF-8
```

5 ソフトウェア環境設定ツール

コンパイラ、ライブラリ、アプリケーションを利用する際に必要になる環境設定を行うツールとして、module コマンドを提供しています。あらかじめ用意されているモジュールファイル(実行コマンドやライブラリのパスが定義されたファイル)と呼ばれる定義ファイルを module コマンドにより指定することで、異なるバージョンのソフトウェアを使用する際の、環境設定を簡単に切り替えて利用することができます。module コマンド一覧を表 4 に示します。詳細な使い方は module help と実行していただくことでヘルプが表示されます。

表 4 module コマンド

コマンド	説明
module list	ロード済みモジュールファイルの一覧表示
module avail	使用可能モジュールファイルの一覧表示
module show	モジュールファイルの設定内容表示
module load	モジュールファイルのロード
module unload	モジュールファイルのアンロード
module switch	モジュールファイルの切り替え

メインとなるコンパイラやライブラリはログイン時に自動で設定が行われます。システム A は Cray コンパイラ環境、システム B、C では、Intel コンパイラ環境が設定されます。

- ロードされている module ファイルの確認

```
$ module list
Currently Loaded Modulefiles:
 1) pbs/SystemA   2) cce/8.5.2   3) cray-mpich
```

- コンパイラの切り替え

```
$ module switch cce/8.4.6
```

ISV アプリケーションを利用する場合は、module avail コマンドにより利用可能なアプリケーションとバージョンを確認し、 module load コマンドにより環境設定を行った後で、起動コマンドを実行してください。

- matlab の起動手順

```
$ module  load  matlab
$ xrun matlab
※ xrun コマンドについては次章をご覧ください
```

6 プログラムの実行方法

6.1 ログインノードの利用

ログインノードは、多数の利用者が同時にログインし作業を行うため、エディタを使ったプログラミングやコンパイル、小規模な逐次処理のプログラム実行に限定して利用してください。過度な負荷がかかることを防止するために、表 5 に示すプロセスリミットを設定しています。

表 5 ログインノードのプロセスリミット

項目	初期値	最大値
CPU 時間(CPU Time)	4 時間	24 時間
メモリサイズ(Virtual memory)	4GB	4GB

bash を利用している場合は、ulimit コマンドを使って設定確認・変更を行うことができます。

- 設定確認(bash)

```
$  ulimit  -a
cpu time          (seconds, -t)  14400
virtual memory     (kbytes, -v)  4194304
```

- 最大値まで拡張(bash)

```
$  ulimit  -t  86400
$  ulimit  -v  4194304
```

tcsh を利用している場合は、limit コマンド、unlimit コマンドを使ってプロセスリミットの設定確認・変更を行うことができます。

- 設定確認(tcsh)

```
$  limit
cputime        4:00:00
vmemoryuse     4194304 kbytes
```

- 最大値まで拡張(tcsh)

```
$  unlimit
```

6.2 会話型ジョブの利用

会話型ジョブは通常のコマンド実行のように、対話的にジョブを実行する形式です。プログラム自体は計算ノード上で実行されますが、ログインしているターミナル上にプログラムの出力内容がすぐに返されるため、プログラムのデバッグや対話的に利用するアプリケーションを実行する際にご利用ください。

6.2.1 会話型ジョブ用の計算ノード

会話型ジョブ用の計算ノードは全利用者で共用するため、一部のユーザが占有しないように利用可能な資源に制限をかけています。制限値を表 6、表 7 に示します。

表 6 会話型ジョブ用計算ノードの制限値（標準）

	システム A	システム B	システム C
コア数	1	1	1
メモリ	1355MB	3413MB	42666MB
CPU 時間	制限なし		
経過時間	1 時間		

表 7 会話型ジョブ用計算ノードの制限値（最大）

	システム A	システム B	システム C
コア数	68	36	72
メモリ	90GB	120GB	3000GB
CPU 時間	制限なし		
経過時間	24 時間		

6.2.2 tssrun コマンド

会話型ジョブを実行するには tssrun コマンドを利用します。MPI プログラムを実行する場合は、システム A およびシステム B、C で利用方法が異なりますのでご注意ください。

- tssrun コマンドの利用方法

```
tssrun [-q queue] [-A p=x:t=x:c=x:m=x] ¥
 [-C cpulimit] [-W  elapsetime] ./a.out
※ オプションの詳細は表 9、表 10 を参照
```

- 【共通】逐次実行プログラム

```
$ tssrun ./a.out
```

- 【共通】OpenMP プログラム（4 スレッド）

```
$ tssrun -A p=1:t=4:c=4 ./a.out
```

- 【システム A】MPI プログラム（8 プロセス）

```
$ tssrun -A p=8 ./a.out
```

- 【システム B、C】MPI プログラム（16 プロセス）

```
$ tssrun -A p=16 mpiexec.hydra ./a.out
※ mpiexec.hydra コマンドを介してプログラムを起動
```

6.2.3 xrun コマンド

Exceed onDemand によりログインした状態で、xrun コマンドを利用することで、GUI のアプリケーションを計算ノードで実行させることができます。コマンドの利用方法は tssrun と同様です。

6.2.4 パーソナル、グループコース用のバッチキューの利用

tssrun および xrun コマンドは、会話型ジョブ用の計算ノードでプログラムを実行しますが、-q オプションにより、グループコースやパーソナルコースで利用可能なバッチキューを指定することで、前述の計算ノードの制限を受けずに、本格的なジョブ実行を対話的に行うことができます。

6.3 バッチジョブの利用

バッチ処理を実現するための、ジョブスケジューリングソフトウェアとして、システム A,B,C では PBS を導入しています。

PBS にバッチジョブを投入するには、実行したいプログラムを記述したシェルスクリプト（ジョブスクリプト）を作成し、専用のコマンドによってスクリプトの実行依頼を行います。

スパコンの計算資源は有限であるため、大規模な計算や本格的な実行は、バッチジョブで行っていただくことが基本となります。

6.3.1 バッチキュー名と投入権限

利用可能なバッチキューは申し込んだサービスコースにより決まります。キューの種類を表 8 に示します。

表 8 キューの一覧

キュー名	利用条件
eb	すべての利用者が利用可能
p{a|b|c}	パーソナルコース利用者
グループ名 + {a|b|c}	グループコース利用者
t{a|b|c}	会話型ジョブ用

6.3.2 ジョブスクリプト

ジョブスクリプトは、原則的には bash スクリプトとして記述してください。スクリプトはジョブ投入オプションを記述したジョブスケジューラオプション領域と実行するプログラムを記述したユーザプログラム領域から構成されます。以下にシステム A におけるサンプルを示します。

```
#!/bin/bash
#============ Options ============
#QSUB -q gr10001a              #キュー指定
#QSUB -ug gr10001              #実効グループ指定
#QSUB -W 10:00                 #経過時間上限指定
#QSUB -A p=4:t=1:c=1:m=1355M   #計算資源量の指定
#============ Shell Script ============
aprun -n $QSUB_PROCS -d $ QSUB_THREADS –N ¥
$QSUB_PPN ./a.out
```

ジョブスクリプトのサンプルを次の Web ページで公開しています。サンプルをベースにジョブスケジューラオプションの修正や必要なスクリプトを追記してください。

- システム A
  http://web.kudpc.kyoto-u.ac.jp/manual-new/ja/run/systema
- システム B、C
  http://web.kudpc.kyoto-u.ac.jp/manual-new/ja/run/systembc

6.3.3 ジョブスケジューラオプション

ジョブスクリプトのオプション領域にて、#QSUB 句の後にオプション指定することで、ジョブの属性を設定することができます。主要なオプションを表 9 に示します。

表 9　ジョブスケジューラオプション

オプション	説明
-q *QUEUENAME*	キュー指定
-ug *GROUPNAME*	実効グループ指定
-W *HOUR:MINUTE*	経過時間上限値指定
-A p=*X*:t=*X*:c=*X*:m=*X*	計算資源量の指定
-M *MAILADDR*	下記-m で送るメールのアドレス指定
-m be	ジョブ開始時と終了時にメール送信
-r n	障害発生時のジョブ再実行禁止

また、-A オプションで指定できる計算資源量を表 10 に示します。

表 10　ジョブスケジューラオプション(-A の詳細)

オプション	説明
p=procs	プロセス数　e.g.　p=8
t=threads	プロセスあたりのスレッド数　e.g. t=16
c=cores	プロセスあたりのコア数（基本的に t と同じ値を指定）　e.g. c=16
m=memory	プロセスあたりのメモリ容量(単位:M,G,T)　e.g.　m=1355M

6.3.4 ジョブスクリプトで利用可能な環境変数

ジョブスクリプト上で利用可能な PBS の環境変数のうち、代表的なものを表 11 に示します.

表 11　ジョブスクリプトで利用可能な環境変数

環境変数名	説明
QSUB_JOBID	当該ジョブのジョブ ID
QSUB_QUEUE	ジョブを投入したキュー
QSUB_WORKDIR	ジョブを投入したカレントディレクトリ
QSUB_PROCS	ジョブ実行時のプロセス数(-A オプションの p の値)
QSUB_THREADS	ジョブ実行時のスレッド数(-A オプションの t の値)
QSUB_CPUS	ジョブ実行時のコア数（-A オプションの c の値)
QSUB_MEMORY	ジョブ実行時のメモリ容量(-A オプションの m の値)
QSUB_PPN	ジョブ実行時のノードあたりのプロセス数（-A の値から自動で算出、システム A のみ)

6.3.5 プログラムの実行方法

シェルスクリプト領域に実行したいプログラムを記述しますが、システム A とシステム B、C でプログラム実行方法が異なりますのでご注意ください。

- システム A

逐次実行、OpenMP、MPI ともに、次のように aprun コマンドを介してプログラムを起動する必要があります。aprun を利用しないと計算ノード上で正しくプログラムが起動しません。

```
aprun -n $QSUB_PROCS -d $QSUB_CPUS ¥
 -N $QSUB_PPN ./a.out
```

表 12　aprun コマンドのオプション

オプション	説明
-n procs	プロセス数
-d cores	コア数
-N ppn	ノードあたりのプロセス数

- システム B、 C

MPI による並列実行の場合、次のように mpiexec.hydra コマンドを介してプログラムを起動します（逐次、OpenMP の場合は不要)。MPI で起動するプロセス数や計算ノードの情報は、mpiexec.hydra が PBS から直接データを取得するので、明示的な指定は不要です。

```
・MPI による並列実行の場合
    mpiexec.hydra ./a.out
・逐次実行、OpenMP による並列実行の場合
    ./a.out
```

6.3.6 ジョブ投入できるキューの確認（qstat -q)

ジョブを投入できるキューを確認するためには、システム A,B,C では qstat -q コマンドを使用します。以下にシステム A で qstat -q の実行例を示します（見易さのため一部省略しています)。

```
$ qstat -q
Queue    Memory CPUTime Walltime Node   Run   Que
-------- ------ -------- -------- ---- ----- -----
ta       --     --       --       --      0     0
pa       --     --       --       --      0     0
gr10001a --      --       --       --      0     0
```

左から 6，7 列目の RUN，QUE はそれぞれ実行中ジョブ，実行待ちジョブを表します.

6.3.7 ジョブ実行(qsub)

キューにジョブを投入するためには、qsub コマンドを使用します。qsub コマンドにジョブスクリプトファイルを指定しジョブを依頼すると、システムからジョブ ID が発行されます。以下にシステム A の実行例を示します。

```
$ qsub sample.sh
1146.sdb
```

6.3.8 ジョブの詳細情報を確認する(qs)

投入したジョブの詳細情報を確認するためには、qs コマンドを使用します。

```
$  qs
QUEUE  USER  JOBID  STATUS  PROC  THRD  CORE  MEM
ELAPSE(limit)
gr10001a b59999 123 RUN  4 8 8 1355M  00:06(01:00)
```

6.3.9 ジョブの途中経過を確認する(qcat)

実行中のジョブについて、ジョブの途中経過を確認するためには、qcat コマンドを使用します。ジョブの標準出力の表示は-o オプションを、標準エラー出力の表示は-e オプションを使用し、確認したいジョブの ID を指定します。

```
$  qcat  -o  5610
Tue May  1 00:00:01 JST 2016
Subroutine A step1 finished
Subroutine A step2 finished
Subroutine A step3 finished
```

```
$  qcat  -e  5610
Tue May  1 00:00:01 JST 2016
STDERR 1
```

6.3.10 グループのジョブを確認する(qgroup)

グループコースのキュー利用状況は、qgroup コマンドで確認できます。表示される項目の内容を表 13 に示します。

```
$  qgroup
QUEUE   SYS|RUN PEND OTHER|ALLOC(MIN/STD/MAX)
-------------------------------------------
gr10000a A |  1    0    0|   64( 136/ 272/ 544)
gr10001a A |  0    0    0|    0( 136/ 272/ 544)
QUEUE              USER|RUN(ALLOC)PEND(REQUEST)
OTHER(REQUEST)
-----------------------------------------------
gr10000a b59999|1(  64)     0(  0)      0(  0)
```

表 13　qgroup コマンドで表示される項目一覧

項目	説明
RUN、PEND、OTHER	ジョブの件数
ALLOC	割当コア数
REQUEST	要求コア数
MIN	キューに対する最低保障コア数 （すぐに利用可能なコア数）
STD	キューに対する標準コア数 （1 ジョブの利用コア数の上限）
MAX	キューに対する最大コア数 （全ジョブの利用コア数の合計の上限）

6.3.11 投入したジョブをキャンセルする(qdel)

投入したジョブをキャンセルするためには、システム A,B,C では qdel コマンドを使用します。キャンセルしたいジョブの ID を引数に指定することで、実行中、実行待ちに関わらずキャンセルすることができます。

```
$ qdel 15344
qdel: Job <15344> has finished
```

6.3.12 実行結果の確認

実行が終了すると、実行結果ファイルが qsub コマンドを実行したディレクトリに作成されます。デフォルトで作成される出力ファイルは、表 14 に示す形式で作成されます。

表 14　ジョブ実行結果ファイル

システム	内容	ファイル名
A	標準出力・ジョブ情報	A*mmddhh*.o*XXXXXX*
	標準エラー出力	A*mmddhh*.e*XXXXXX*
B	標準出力・ジョブ情報	B*mmddhh*.o*XXXXXX*
	標準エラー出力	B*mmddhh*.e*XXXXXX*
C	標準出力・ジョブ情報	C*mmddhh*.o*XXXXXX*
	標準エラー出力	C*mmddhh*.e*XXXXXX*

mmddhh : ジョブが開始された月・日・時

XXXXXX : ジョブ ID

7　プログラム開発環境

システムA、B、Cでは、Crayコンパイラ、Intelコンパイラ、PGIコンパイラ、GNUコンパイラが利用いただけます。また、ログインした際にデフォルトで利用いただけるコンパイラは、システムAではCrayコンパイラ、システムB、CではIntelコンパイラです。

7.1 システムAの場合

7.1.1 コンパイルコマンドとライブラリ

Cray コンパイラでサポートしているプログラミング言語とコンパイルコマンドを表 15 に、サポートしている数値計算ライブラリを表 16 に示します。

システムAでIntelコンパイラを利用する場合、module コマンドを実行し、コンパイラ環境を切り替えた上で、コンパイルコマンドを実行してください。module の設定状況に応じて動作が切り替わるため、どのコンパイラも同じコンパイルコマンド(ftn、cc、CC)で実行することができます。

表 15　プログラミング言語とコンパイルコマンド

言語	規格・バージョン	コマンド
Fortran	ISO/IEC 1539-1:2004 (Fortran 2003)	ftn
C	ISO/IEC 9899:1999(C99)	cc
C++	ISO/IEC 14882:2003	CC

表 16　MPI、数値計算ライブラリ

ライブラリ	サポート
MPI	Cray MPI
数値計算ライブラリ	LibSCi, MKL (BLAS,LAPACK,SCALAPACK)

7.1.2 Fortran

Fortran のコンパイル・リンクは ftn コマンドを利用します。

- Fortran でのコンパイル・リンク

```
$ ftn  sample.f90
```

Cray コンパイラはデフォルトで高い最適化を行うため、最初はオプションなしでコンパイル・リンクすることを推奨します。

- 数値計算ライブラリのリンク例

```
#LibSci を使用する場合
$ module load cray-libsci
$ ftn  sample.f90
```

LibSci ライブラリは、オプション指定なしに自動でリンク処理を行うため、コンパイル時のライブラリ指定は不要です。

7.1.3 C/C++

C のコンパイル・リンクは、cc コマンド、C++のコンパイル・リンクは、CC コマンドを利用します。

- C でのコンパイル・リンク

```
$ cc  sample.c
```

- C++でのコンパイル・リンク

```
$ CC  sample.cpp
```

- 数値計算ライブラリのリンク例

```
#Libsci を使用する場合
$ module load cray-libsci
$ cc  sample.c
```

7.1.4 コンパイルオプション

Cray コンパイラがサポートする主なオプションを表 17 に、メッセージ出力とデバッグのオプションを表 18 に、Fortran 言語固有のオプションを表 19 に示します。

表 17　主なオプション

オプション	説明
-o *FILENAME*	オブジェクトファイルの名前を指定します
-O{0|1|2|3}	最適化のレベルを指定します (デフォルトは 2)
-h omp	OpenMP 指示子を有効にしてコンパイルします（デフォルトで有効）
-h noomp	OpenMP 指示子を無効にしてコンパイルします
-h autothread	自動並列化を行います
-h byteswapio	ビッグエンディアン形式のファイルをサポートします
-h pic	2GByte を超えるメモリをサポートします (ライブラリを動的リンクとする必要があるため、下記の-dynamic を併用する必要があります)
-dynamic	ライブラリを動的リンクします

表 18　メッセージ出力とデバッグのオプション

オプション	説明
-ra	コンパイル時のレポートを作成します
-m2	すべての警告メッセージを表示します
-h msgs	実施した最適化についての情報を表示します
-h negmsgs	実施しなかった最適化についての情報を表示します
-Rb	配列の領域外参照を検出します

表 19　Fortran 言語固有オプション

オプション	説明
-f fixed	プログラムが固定形式で記述されていることを指示します
-f free	プログラムが自由形式で記述されていることを指示します
-em	モジュールを有効にします

7.1.5 自動並列化

ftn、cc、CC のコンパイルコマンド実行時に、-h autothread オプションを指定することで、コンパイラが自動的にプログラムを並列化します。

- 自動並列化のコンパイル・リンク

```
$ ftn -h autothread sample.f90
```

7.1.6 OpenMP

OpenMP は、共有メモリ型並列計算機における並列プログラミングの標準インターフェイスです。Fortran では!$OMP、C/C++では#pragma omp で始まる指示子をユーザがソースプログラム中に挿入し、-h omp オプションを指定することで、プログラムを並列化します。

- OpenMP のコンパイル・リンク

```
$ ftn -h omp sample.f90
```

7.1.7 MPI

MPI(Message Passing Interface)は、並列処理アプリケーション用メッセージパッシングライブラリです。システム A では MPI プログラムも ftn、cc、CC の各コマンドでコンパイルを行います。

- MPI のコンパイル・リンク

```
$ ftn sample_mpi.f90
```

7.1.8 Intel コンパイラへの切り替え

Cray コンパイラから Intel コンパイラに切り替えるには、次のコマンドを実行してください。

```
$ module switch PrgEnv-cray PrgEnv-intel
```

7.2 システム B、C の場合

7.2.1 コンパイルコマンドとライブラリ

Intel コンパイラでサポートしているプログラミング言語とコンパイルコマンドを表 20 に、サポートしているライブラリを表 21 に示します。

表 20　プログラミング言語とコンパイルコマンド

言語	規格・バージョン	コマンド
Fortran	ISO/IEC 1539-1:2004 (Fortran 2003)	ifort
C	ISO/IEC 9899:1999(C99)	icc
C++	ISO/IEC 14882: 2003	icpc

表 21　MPI、数値計算ライブラリ

ライブラリ	サポート
MPI	Intel MPI
数値計算ライブラリ	MKL、IMSL、NAG

7.2.2 Fortran

Fortran のコンパイル・リンクは ifort コマンドを利用します。

- Fortran でのコンパイル・リンク

```
$ ifort sample.f90
```

- 数値計算ライブラリのリンク例

```
#MKL を使用する場合
$ module load intel
$ ifort sample.f90 -mkl

#IMSL を使用する場合
$ module load imsl
$ ifort $F90FLAGS sample.f90 $LINK_FNL

#NAG を使用する場合
$ module load nag
$ ifort $NAGFLAGS sample.f90 $NAGLINK
```

7.2.3 C/C++

C のコンパイル・リンクは icc コマンド、C++のコンパイル・リンクは icpc コマンドを利用します。

- C でのコンパイル・リンク

```
$ icc  sample.c
```

- C++でのコンパイル・リンク

```
$ icpc  sample.cpp
```

- 数値計算ライブラリのリンク例

```
#MKL を使用する場合
$ module  load  intel
$ icc  sample.c  -mkl

#NAG を使用する場合
$ module load nag
$ icc  $NAGFLAGS  sample.c  $NAGLINK
```

7.2.4 コンパイルオプション

システム B、C のメインコンパイラである Intel コンパイラがサポートする主なオプションを表 22 に、メッセージ出力とデバッグのオプションを表 23 に、Fortran 言語固有のオプションを表 24 に示します。

表 22　主なオプション

オプション	説明
-o *FILENAME*	オブジェクトファイルの名前を指定します
-fast	プログラムの速度が最大になるように最適化します
-O{0|1|2|3}	最適化のレベルを指定します (デフォルトは 2)
-ipo	複数ファイル間で、手続き間の処理を最適化します
-qopenmp	OpenMP 指示子を有効にしてコンパイルします
-parallel	自動並列化を行います
-mcmodel=medium -shared-intel	2Gbyte を超えるメモリをサポートします
-convert big_endian	ビッグエンディアン形式のファイルをサポートします

表 23　メッセージ出力とデバッグのオプション

オプション	説明
-qopt-report	実施した最適化についての情報を表示します
-qopt-report -qopt-report-phase=par	実施した自動並列化についての情報を表示します
-qopt-report -qopt-report-phase=vec	実施したベクトル化についての情報を表示します

表 24　Fortran 言語固有オプション

オプション	説明
-nofree	プログラムが固定形式で記述されていることを指示します
-free	プログラムが自由形式で記述されていることを指示します
-warn all	すべての警告メッセージを表示します
-CB	配列の領域外参照を検出します

7.2.5 自動並列化機能

ifort、icc、icpc のコンパイルコマンド実行時に、-parallel オプションを指定することで、コンパイラが自動的にプログラムを並列化します。

- 自動並列化のコンパイル・リンク

```
$ ifort  -parallel  sample.f90
```

7.2.6 OpenMP

コンパイル時に-qopenmp オプションを指定することで、プログラムを並列化します。

- OpenMP のコンパイル・リンク

```
$ ifort  -qopenmp  sample.f90
```

7.2.7 MPI

システム B、C では MPI プログラムのコンパイルに、mpiifort (Fortran)、mpiicc (C)、mpiicpc (C++) を使用します。

- MPI のコンパイル・リンク

```
$ mpiifort  sample_mpi.f90
```

8 ISV アプリケーション

スパコンを用いた大規模な科学技術計算、および計算化学、構造解析、統計解析、可視化などのために様々なアプリケーションソフトウェアのサービスを提供しています(表 25～表 30)。利用できるアプリケーションはシステム毎に異なります。利用者の所属により利用が制限される場合がありますので、詳細は次の URL でご確認ください。

http://web.kudpc.kyoto-u.ac.jp/manual-new/apps

表 25 可視化・図形処理アプリケーション

名称	システム		
	A	B	C
AVS/Express	—	○	○
ENVI/IDL	—	○	○
Tecplot	—	○	○

表 26 数式処理アプリケーション

名称	システム		
	A	B	C
Maple	—	○	○
Mathematica	—	○	○

表 27 技術処理アプリケーション

名称	システム		
	A	B	C
MATLAB	—	○	○

表 28 計算化学アプリケーション

名称	システム		
	A	B	C
Gaussian09	○	○	○
GaussView	—	○	○
MOPAC	—	○	○

表 29 構造解析・機構解析アプリケーション

名称	システム		
	A	B	C
MSC Nastran	—	○	○
Patran	—	○	○
Adams	—	○	○
Marc	—	○	○
Marc Mentat	—	○	○
LS-DYNA	—	○	○
ANSYS	—	○	○

表 30 統計解析アプリケーション

名称	システム		
	A	B	C
SAS	—	○	—

9 おわりに

本稿では、システム A、B、C の各スパコンシステムの利用に必要となる情報を解説しました。より詳細な情報、最新の情報は以下 Web ページをご確認ください。

- スーパーコンピュータの使い方
  http://web.kudpc.kyoto-u.ac.jp/manual-new/ja

新スーパーコンピュータ Camphor 2 のベンチマーク評価報告

平石 拓 *

*学術情報メディアセンター

1　はじめに

京都大学学術情報メディアセンターは，2016 年 10 月に新しいスーパーコンピュータ Camphor 2 のサービスを開始した．また，2016 年 12 月からはさらに Laurel 2 と Cinnamon 2 のサービスを開始する予定である．

本稿では，すでにサービスを開始している Camphor 2 および従来からサービスを行っているスーパーコンピュータにおける，いくつかのマイクロベンチマークおよび実アプリケーションに基づくベンチマークプログラムによる性能評価の結果を示し，両者の性能を比較する．

2　システム構成

2016 年度，2014 年度，2012 年度に稼動を開始した（開始予定の）スーパーコンピュータの性能諸元をそれぞれ表 1，表 2，表 3 に示す．

Camphor 2 は 2012 年度に導入された Camphor と同じく高い電力性能比を指向したシステムであり，Camellia（2014 年度稼動開始）に搭載されている Knights Corner（KNC）の後継である最新の Xeon Phi メニーコアプロセッサ Knights Landing（KNL）を採用している．世界のスパコンの Linpack 性能を競う Top500 では 33 位（日本では 4 位），電力比性能を競う Green500 では 9 位（同 3 位）にランキングされた（いずれも 2016 年 11 月時点）[7].

Camellia の KNC は，通常の Xeon プロセッサノードに追加するコプロセッサとして搭載されていたが，Camphor 2 の KNL はホスト CPU であり，Linux 等の OS も KNL 上で直接動作する．そのため，Camellia のようにホストーコプロセッサ間のデータ転送を意識するプログラミングは不要となる．KNL は SIMD 演算性能の強化，アウトオブオーダ実行ができるようになっているなどコアあたりの性能も KNC より大幅に強化されている．

また，Camphor 2 のノードは通常の DDR4 メモリ（96GB）に加えて MCDRAM と呼ばれる高速メモリ（16GB）を有している．この MCDRAM は，DDR4 メモリのキャッシュとして使えるモード（キャッシュモード）のほか，MCDRAM と DDR4 のアドレスを分離してどちらのメモリにどのデータを置くかをユーザレベルで使い分けることができるモード（フラットモード），この両者のモードを併用するモード（ハイブリッドモード）のいずれかで利用することができるようになっている．ただし，本稿執筆時点ではユーザはキャッシュモードのみを利用できる運用となっており，後日残りのモードの利用も解放される予定である．

さらに，これも本稿執筆時点ではユーザに解放されていない機能だが，DataWarp というバーストバッファを備えていることも Camphor 2 の大きな特徴である．これは，計算中に発生する大量のファイル I/O を高速･低容量なストレージを経由して行うことにより，計算時の I/O による遅延を最小限に抑えるための仕組みである．なお同様の機能は，Infinite Memory Engine（IME）により Laurel 2 および Cinnamon 2 でも利用できるようになる予定である．

3　測定条件

3.1　計算機環境

今回の性能評価では新旧世代の Xeon Phi および Xeon プロセッサの性能を比較するため，Camphor 2（新サブシステム A），Magnolia（サブシステム D）

表 1: スーパーコンピュータの性能諸元（2016 年度導入システム）

名称	サブシステム名 愛称 機種名	サブシステム A Camhpor 2 Cray XC40	サブシステム B Laurel 2 CS400 2820XT	サブシステム C Cinnamon 2 CS400 4840X
稼動期間		2016 年 10 月～	2016 年 12 月～（予定）	
ノード数		1800	850	16
理論演算性能		5.4 PFlops	1.03 PFlops	42.4 TFlops
Linpack 性能		3.07 PFlops	—	—
Top500	初出	33rd (Nov. 2016)	—	—
CPU	製品名，周波数	Intel Xeon Phi Knights Landing 1.4 GHz	Intel Xeon Broadwell 2.1 GHz	Intel Xeon Haswell 2.3 GHz
	ソケット数 / node	1	2	4
	コア数 / node (/ socket)	68 (68)	36 (18)	72 (18)
メモリ	規格	MCDRAM + DDR4-2133	DDR4-2400	DDR3-1600
	総バンド幅	921 GB/s (MCDRAM) + 102 GB/s (DDR4)	154 GB/s	205 GB/s
	容量	16GB (MCDRAM) + 96 GB (DDR4)	128 GB	3 TB
ネットワーク	リンク	Cray Aries (15.75 GB/s)	Intel Omni-Path (12 GB/s)	Intel Omni-Path × 2 (12 GB/s × 2)
	トポロジ	Dragonfly	Fat tree	Fat tree
	バイセクション バンド幅	13.5 TB/s	half-bisection	full-bisection
ストレージ		DDN ExaScaler (16 PB, 100 GB/s)（2018 年度に 8 PB, 50 GB/s を増強予定）		
バーストバッファ		Cray DataWarp (230 TB, 206 GB/s)	DDN IME (230 TB, 250 GB/s)	

表 2: スーパーコンピュータの性能諸元（2014 年度導入システム）

名称	サブシステム名 愛称 機種名	サブシステム D Magnolia Cray XC30	サブシステム E Camellia Cray XC30
稼動期間		2014 年 7 月～2016 年 12 月（予定）	2014 年 4 月～
ノード数		416	482
理論演算性能		428.6 TFlops	583.6 TFlops
Linpack 性能		307.2 TFlops	380.5 TFlops
Top500	初出 最終・最新	174th (Nov. 2014) 456th (Jun. 2016)	101st (Jun. 2014) 455th (Nov. 2016)
CPU	製品名，周波数	Intel Xeon Haswell 2.6 GHz	Intel Xeon Ivy Bridge 2.6 GHz + Intel Xeon Phi Knights Corner 1.053 GHz
	ソケット数 / node	2	1 (Xeon) + 1 (Xeon Phi)
	コア数 / node (/ socket)	28 (14)	10 (Xeon) + 60 (Xeon Phi)
メモリ	規格	DDR4-2133	DDR3-1600 (Xeon) + GDDR5 (Xeon Phi)
	総バンド幅	136 GB/s	51.1 GB/s (DDR4) + 352 GB/s (GDDR5)
	容量	64GB	32 GB (DDR4) + 8 GB (GDDR5)
ネットワーク	リンク	Cray Aries (15.75 GB/s)	Cray Aries (15.75 GB/s)
	トポロジ	Dragonfly	Dragonfly
	バイセクション バンド幅		4.6 TB/s
ストレージ		DDN SFA12K (3 PB, 24 GB/s)	

および Camellia（サブシステム E）の 3 種類のサブシステムでそれぞれ測定を行った．

Camellia では KNC をネイティブ実行モード（ホストプロセッサを使わず，KNC 上でのみプログラムを実行するモード）でのみ用いた．そのため，メモリが 8 GB しか使用できず，一部のベンチマークでは他のサブシステムに比べて小さい問題サイズで測定を行っている．

3.2 ベンチマークプログラム

本稿で示す評価に用いたベンチマークプログラムは以下の通りである．

(a) HPLinpack: Innovative Computing Laboratory, University of Tennessee (ICL-UT) が配布する HPCC ベンチマーク [1] に含まれる，密行列の連立一次方程式を解くベンチマークプログラム

表 3: スーパーコンピュータの性能諸元（2012 年度導入システム）

名称	サブシステム名 愛称 機種名	サブシステム A Camhpor Cray XE6	サブシステム B Laurel Appro GreenBlade 8000	サブシステム C Cinnamon Appro 2548X
稼動期間		2012 年 4 月〜2016 年 8 月	2012 年 4 月〜2016 年 12 月（予定）	
ノード数		940	601	16
理論演算性能		300.8 TFlops	242.6 TFlops (含 GPU) 193.0 TFlops (除 GPU)	10.6T Flops
Linpack 性能		251.7 TFlops	135.4 TFlops (除 GPU)	—
Top500	初出 最終	73rd (Jun. 2012) 405th (Nov. 2015)	126th (Jun. 2012) 494th (Jun. 2014)	— —
CPU	製品名，周波数	AMD Opteron Abu Dhabi 2.6 GHz	Intel Xeon Sandy Bridge 2.6 GHz	Intel Xeon Sandy Bridge 2.6 GHz
	ソケット数 / node	2	2	4
	コア数 / node (/ socket)	32 (16)	16 (8)	32 (8)
GPU		—	NVIDIA Tesla M2090 （64 ノードに搭載）	—
メモリ	規格 総バンド幅 容量	DDR3-1600 102 GB/s 64GB	DDR3-1600 102 GB/s 64 GB	DDR3-1066 136 GB/s 1.5 TB
ネットワーク	リンク	Cray Gemini (9.3 GB/s or 4.6 GB/s per link)	Infiniband FDR × 2 (6.8 GB/s × 2)	Infiniband FDR × 2 (6.8 GB/s × 2)
	トポロジ	3D-torus	Fat tree	Fat tree
	バイセクション バンド幅	1.7 TB/s	3.1 TB/s	217 GB/s
ストレージ		DDN SFA10000 (5 PB, 54 GB/s)		

であり，Top500 の評価にも用いられる．本評価では，複数ノード（各サブシステムの一部）を使用したときの演算性能を測定した．測定に用いた HPCC のバージョンは 1.4.2 である．

(b) 高速フーリエ変換プログラム（FFTE）：上記 HPCC ベンチマークに含まれる．複数ノードによる性能測定を行った．

(c) FDTD 法による電磁場解析プログラム（FDTD）: 電場と磁場の時間発展をステンシル計算（各格子点の次の時刻の状態を，近傍の格子点の値を使って求める）によって導く．参照局所性を高めるため，時空間タイリングという手法が用いられている [6]．MPI と OpenMP によるハイブリッド並列化が行われており，単一ノードおよび複数ノードによる性能測定を行った．

(d) 並列ポアソンソルバ（Poisson）: 反復法によるポアソン方程式のソルバであり，マルチグリッド法による収束性向上やブロック化等による性能改善が施されている [3, 4]．本測定では，OpenMP 版による単一ノードでの評価と，Flat MPI 版による複数ノードでの評価を行った．

(e) Intel MPI Benchmarks（IMB）: Intel 社が配布 [2] する，一対一通信，集団通信等の MPI の基本性能を測定するベンチマークプログラムである．今回測定に用いたバージョンは 4.1.10 である．8 ノード ×1 プロセス実行の評価を行った．

(f) メモリアクセス性能評価ベンチマーク（STREAM）: John D. McCalpin 氏が作成・配布 [5] する，メモリ性能を実測により測定するベンチマークである．（性能が最大になるような）任意の数のスレッドを生成し，それぞれのスレッドが大きなサイズの配列に対するロード・ストアを行うことでメモリの性能評価を行う．単一ノードによる性能評価を行った．

3.3 コンパイラ・ライブラリ

ほとんどプログラムのコンパイルには，Intel Compiler バージョン 16（一部のベンチマークについては，Magnolia および Camellia ではバージョン 14）を用いた．Camphor 2 の HPLinpack では Cray Compiler バージョン 8.5.2 を用いた．

MPI ライブラリには，Cray MPI を用いた．バージョンは Camphor 2 では 7.4.2，Magnolia および Camellia では 7.3.2 である．

HPLinpack では，数値ライブラリとして Camphor 2 では Cray Libsci 16.07.1 を，Magnolia および Camel-

lia では Intel MKL バージョン 11.3.2 および 11.2.4 をそれぞれ用いた．

4　測定結果

各ベンチマークの性能評価結果を表 4–表 9 に示す．なお，これらの表中の「システム」は，「新 A」が Camphor 2，「D」が Magnolia，「E」が Camellia を意味する．

(a) HPLinpack　測定結果を表 4 に示す．プロセス分割数などのチューニングや問題サイズが十分ではなく，Top500 用の測定値ほどの対理論性能は得られていない．HPL ではノード数の比例に近い性能を得られることが期待できるため，ノードあたりでは Camphor 2 は Magnolia や Camellia より高い性能が得られると考えられる．

(b) FFTE　測定結果を表 5 に示す．Camphor 2 の性能は Magnolia よりやや低くなっているが，Camellia よりは大幅に改善されていることが確認できる．

(c) FDTD　測定結果を表 6 に示す．このプログラムはベクトル化率が比較的高いため，単一ノード，複数ノードともに Camphor 2 は Magnolia より高い性能が得られている．

(d) Poisson　測定結果を表 7 に示す．このプログラムもベクトル化率は高く，単一ノード実行での Camphor 2 の性能は Magnolia を上回っているが，複数ノード実行の両者の性能は逆転している．この一因として，多数の MPI プロセスを起動したことで，通信コストの増大の悪影響が大きくったことが考えられる．なお，KNL と KNC 上での OpenMP 版の実行では，ハイパースレッディングを利用して，1 物理コアに 2 スレッド割り当てることでコアあたり 1 スレッドの割り当てより 20%程度高い性能が得られた．

(e) IMB　測定結果を表 8 に示す．表 1，表 2 が示す通り，Camphor 2，Magnolia，Camellia のスペック上の通信性能に大きな差はないが，実測性能は Magnolia ＞ Camphor 2 ＞ Camellia の順となっている．これは，通信性能が CPU コアの性能に律速されているためだと考えられる．

表 4: HPLinpack の性能測定結果

システム	新 A	D	E
使用ノード数	8	32	4
プロセス数 × スレッド数	544×1	896×1	240×1
N（問題サイズ）	200000	400000	25000
測定値（HPL TFLOPS）	6.94	19.7	0.465

表 5: FFTE の性能測定結果

システム	新 A	D	E
使用ノード数	8	19	3
プロセス数	512	512	128
N（問題サイズ）	200000	400000	25000
測定値（MPIFFT GFLOPS）	66.8	153	6.70

表 6: FDTD の性能測定結果

システム	新 A	D	E
スレッド数	64	28	60
測定値（GFLOPS）	87.5	65.8	27.3

(a) 単一ノード性能

システム	新 A	D	E
使用ノード数	8	8	4
プロセス数 × スレッド数	32 × 17	16 × 14	24 × 10
測定値（GFLOPS）	418	379	42.0

(b) 複数ノード性能

表 7: Poisson の性能測定結果

システム	新 A	D	E
スレッド数	128	28	120
測定値（Mpoints/s）	479	304	81.7

(a) 単一ノード性能

システム	新 A	D	E
使用ノード数	8	8	8
プロセス数	512	224	480
測定値（Gpoints/s）	2.33	3.91	0.228

(b) 複数ノード性能

表 8: IMB の性能測定結果

システム	新 A	D	E
ノード数 × プロセス数	8 × 1		
PingPong 4MB (MB/s)	7392	9165	3604
PingPong 8B (MB/s)	2.33	3.94	0.5
Sendrecv 4MB (MB/s)	10671	11494	3673
Sendrecv 8B (MB/s)	2.46	5.62	0.63
Allreduce 4MB (μs)	6279	2504	14050
Reduce 4MB (μs)	12547	4355	47976
Reduce-Scatter 4MB (μs)	3836	1310	7138
Allgather 4MB (μs)	6553	5148	17034
Alltoall 4MB (μs)	9480	6636	17587
Bcast 4MB (μs)	1802	1445	3288

表 9: STREAM の性能測定結果

システム		新 A	D
スレッド数		64	28
性能値（copy, GB/s）	$S = 0.3\,\mathrm{GB}$	268	108
	$S = 3\,\mathrm{GB}$	267	—
	$S = 14.9\,\mathrm{GB}$	64.3	—

（S：配列サイズ）

通信性能をフルに使い切るためには，1 ノード内で複数プロセスが同時に通信するようにするなどの方策が必要となると考えられる．

(f) STREAM　測定結果を表 9 に示す．Camphor 2 では，メモリアクセスが MCDRAM のみで完結するようにした場合と，DDR4 メモリもアクセスされる場合との性能を比較するため，いくつかの配列サイズを試した．予想通り，小さな配列サイズでは MCDRAM による高いアクセス性能が得られるが，サイズを大きくすると性能が落ち込んでしまうことが確認できる．

5　おわりに

本稿では，メディアセンターの新しいスパコンと旧スパコンにおけるいくつかのベンチマークの測定結果を示し，システム間の実性能を比較した．Camphor 2（KNL）のコアあたりの性能は Magnoila（Haswell）より低いが，ベクトル化率を高めたプログラムを多数のプロセス・スレッドを使って並列実行すれば，Magnolia より高いノードあたり性能を得ることができる．ベクトル化率や並列度が十分でなかったり，MPI プロセス数を増やしすぎた場合には性能が落ち込んでしまう．ただし，KNL におけるそのような性能の落ち込みは KNC ほど極端ではなく，より使いやすいメニーコアプロセッサになっていると評価できる．また，ネットワークの通信性能を使い切れるようにすることや，MCDRAM の有効活用も Campphor 2 の性能を生かし切るための重要なポイントである．

なお，Laurel 2 と Cinnamon 2 や，バーストバッファの性能評価結果についても，次号以降の記事で報告する予定である．

本稿がシステムの利用検討およびアプリケーション開発の一助になれば幸いである．

参考文献

[1] Innovative Computing Laboratory, University of Tennessee: HPC Challenge. http://icl.cs.utk.edu/hpcc/.

[2] Intel Corporation: Getting Started with Intel MPI Benchmarks. https://software.intel.com/en-us/articles/intel-mpi-benchmarks.

[3] Kawai, M., Iwashita, T. and Nakashima, H.: *SIMD Implementation of a Multiplicative Schwarz Smoother for a Multigrid Poisson Solver on an Intel Xeon Phi Coprocessor*, Springer International Publishing, pp. 57–65 (2015).

[4] Kawai, M., Iwashita, T., Nakashima, H. and Marques, O.: *Parallel Smoother Based on Block Red-Black Ordering for Multigrid Poisson Solver*, Springer Berlin Heidelberg, pp. 292–299 (2013).

[5] McCalpin, J. D.: STREAM: Sustainable Memory Bandwidth in High Performance Computers. http://www.cs.virginia.edu/stream/.

[6] Minami, T., Hibino, M., Hiraishi, T., Iwashita, T. and Nakashima, H.: *Automatic Parameter Tuning of Three-Dimensional Tiled FDTD Kernel*, Springer International Publishing, pp. 284–297 (2015).

[7] TOP500.org: Top500 Supercomputer Sites. https://www.top500.org.

The 35th JSST Annual Conference（JSST2016） 開催報告

江原 康生

京都大学学術情報メディアセンター

1 JSST とは

本会議は、日本シミュレーション学会が主催する年次大会で、理工学・産業の多くの専門分野におけるシステム技法からソフトウェア、ハードウェア及び諸分野への応用に向けたシミュレーションの学理と技術に関する研究討論と情報交換を行う場として毎年開催しており、本年で第35回となります。

2011年より、国際化の推進を目的として、国際会議 JSST2011 という位置付けによる開催に移行し、全セッション、企画の英語化が行われています。翌年以降も国際会議として各地で持ち回りによる開催が行われており、今年（JSST2016）は、2016年10月27日～29日に京都大学国際科学イノベーション棟での開催となりました。

2 JSST2016 での講演

今年度は初めての試みとして、毎年企画されているオーガナイズドセッションに加えて、4件のテーマに関するシンポジウム（数値解析と非線形問題における視覚分析、HPC（ハイパフォーマンス）技術とアプリケーション、気候変動適応技術の社会実装、京都学生研究シンポジウム）を併催し、各シンポジウムで基調講演、招待講演等が企画されました。

その中で HPC 技術とアプリケーションのシンポジウムには、本センターの深沢先生にシンポジウムチェアを務めていただき、中島先生には基調講演をしていただきました（図1）。

JSST2016 企画の基調講演では、「A big challenge for safe automated driving - Potential problems of Human Factors」というタイトルで、産総研の北崎智之先生にご講演いただきました。加えて、各シンポジウム企画による基調講演として、「Molecular simulation for soft and hard matters」というタイトルで核融合研の中村浩章先生に、さらに前述の通り、「Regularity: A new important player in the game of high-performance simulations in manycore era」というタイトルで本学学術情報メディアセンターの中島浩先生にご講演をしていただきました。

図1 基調講演の様子

他にも、下記の 7 つの幅広い研究テーマによるオーガナイズドセッションが企画され、併せて80件の論文発表が行われました。会場では、7 つの協賛企業による機器展示をしていただきました。

3日間の合計で約140名の参加があり、参加者間で活発な議論が交わされました。次回の JSST2017 は東京電機大学で開催される予定です。

◆ オーガナイズドセッションテーマ

- Multi dimensional communication network
- Computational Electromagnetism and Its Applications
- Kansei and higher brain functions
- Numerical Computations
- Motion Dynamics in Human, Animal and Robot
- Complex Networks and Complex Systems
- Simulation Technology in Origami

◆ JSST2016 Web ページ
http://jsst2016.jsst.jp/

システム A 運転状況 （2015 年 10 月 ～ 2016 年 3 月 ）

1）保守作業に伴うサービス休止およびシステムダウン障害発生状況

保守作業に伴うサービス休止

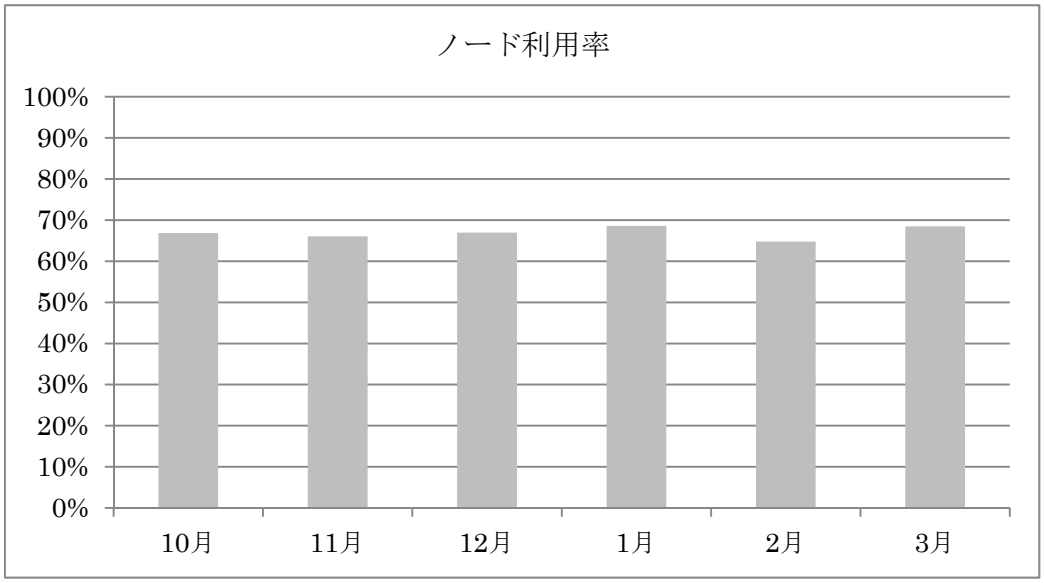
保守開始日時	サービス再開日時	保守時間[h]
2015/10/12 6:00	2015/10/14 9:30	51.50
2015/12/10 9:00	2015/12/11 9:30	24.50
2016/03/30 9:00	2016/04/01 0:00	39.00

システムダウン障害発生状況

障害発生日時	サービス再開日時	ダウン時間[h]
2015/12/07 10:30	2015/12/07 13:45	3.25
2015/12/30 18:06	2016/01/04 11:23	113.28
2016/01/06 12:30	2016/01/06 14:30	2.00
2016/01/07 17:25	2016/01/07 19:10	1.75

2）サービス状況

	サービス時間[h]	バッチ					
		処理件数	経過時間[h]	占有時間[h]	CPU時間[h]	平均稼動ノード数	ノード利用率
10 月	692.50	33,719	66,673	11,120,900	8,982,300	752.7	67 %
11 月	720.00	18,377	80,341	11,268,700	9,779,630	751.9	66 %
12 月	686.35	19,118	80,567	13,126,800	11,245,200	852.1	67 %
1 月	656.87	16,752	112,173	15,647,300	13,234,900	930.3	69 %
2 月	696.00	16,212	72,118	13,146,300	10,683,800	939.8	65 %
3 月	705.00	17,990	56,438	15,165,400	13,254,000	939.9	67 %
計	4156.72	122,168	468,310	79,475,400	67,179,830	861.1	67 %



- 占有時間 = 合計(経過時間×占有コア数)
- 平均稼動ノード数 = 電源 ON 状態のノード数の月平均 (10 分間隔のサンプリングデータより算出)
- ノード利用率 = 稼動ノードに対するジョブが実行されているノードの割合

システム B 運転状況 （2015 年 10 月 ～ 2016 年 3 月 ）

1）保守作業に伴うサービス休止およびシステムダウン障害発生状況

保守作業に伴うサービス休止

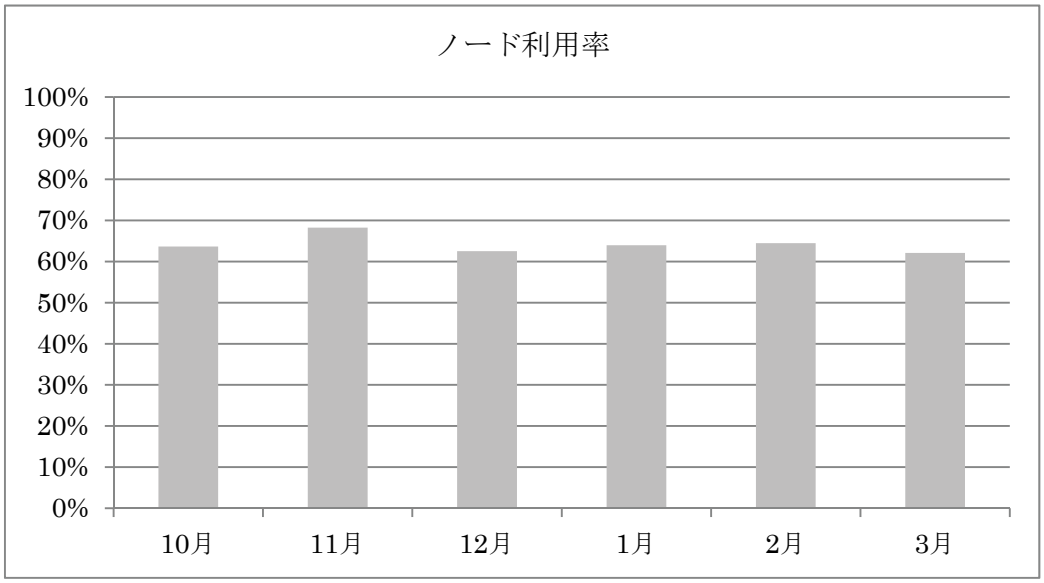
保守開始日時		サービス再開日時		保守時間[h]
2015/10/12	6:00	2015/10/14	9:30	51.50
2015/12/10	9:00	2015/12/11	9:30	24.50
2016/03/30	9:00	2016/04/01	0:00	39.00

システムダウン障害発生状況

障害発生日時	サービス再開日時	ダウン時間[h]

2）サービス状況

	サービス時間[h]	バッチ					
		処理件数	経過時間[h]	占有時間[h]	CPU時間[h]	平均稼動ノード数	ノード利用率
10 月	692.50	217,866	339,230	3,220,350	247,318	533.7	64 %
11 月	720.00	72,508	348,253	4,109,530	297,642	538.9	68 %
12 月	719.50	88,644	327,470	3,825,420	287,188	539.0	63 %
1 月	744.00	90,927	280,875	4,017,930	292,689	545.0	64 %
2 月	696.00	51,842	351,829	3,732,590	277,192	545.0	64 %
3 月	705.00	65,486	259,252	3,998,810	296,083	510.7	61 %
計	4277.00	587,273	1,906,909	22,904,630	1,698,112	535.4	64 %



- 占有時間 = 合計(経過時間×占有コア数)
- 平均稼動ノード数 = 電源 ON 状態のノード数の月平均 (10 分間隔のサンプリングデータより算出)
- ノード利用率 = 稼動ノードに対するジョブが実行されているノードの割合

システム C 運転状況 （2015 年 10 月 ～ 2016 年 3 月 ）

1）保守作業に伴うサービス休止およびシステムダウン障害発生状況

保守作業に伴うサービス休止

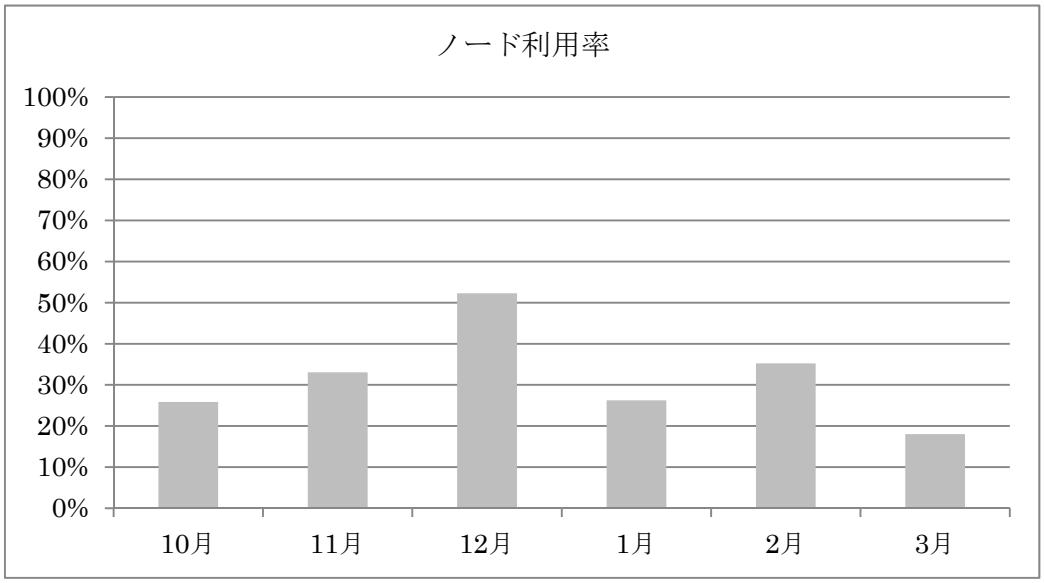
保守開始日時		サービス再開日時		保守時間[h]
2015/10/12	6:00	2015/10/14	9:30	51.50
2015/12/10	9:00	2015/12/11	9:30	24.50
2016/03/30	9:00	2016/04/01	0:00	39.00

システムダウン障害発生状況

障害発生日時	サービス再開日時	ダウン時間[h]

2）サービス状況

	サービス時間[h]	バッチ					
		処理件数	経過時間[h]	占有時間[h]	CPU時間[h]	平均稼動ノード数	ノード利用率
10 月	692.50	7,216	4,388	47,849	14,136	11.9	26 %
11 月	720.00	1,769	5,386	66,251	24,227	12.0	33 %
12 月	719.50	2,218	8,081	147,321	55,307	14.8	52 %
1 月	744.00	938	3,359	91,660	31,506	16.0	26 %
2 月	696.00	1,053	10,792	98,063	34,605	16.0	35 %
3 月	705.00	609	6,109	52,846	18,455	16.0	18 %
計	4277.00	13,803	38,116	503,990	178,235	14.4	32 %



- 占有時間 = 合計(経過時間×占有コア数)
- 平均稼動ノード数 = 電源 ON 状態のノード数の月平均 (10 分間隔のサンプリングデータより算出)
- ノード利用率 = 稼動ノードに対するジョブが実行されているノードの割合

システム D 運転状況 （2015 年 10 月 ～ 2016 年 3 月 ）

1）保守作業に伴うサービス休止およびシステムダウン障害発生状況

保守作業に伴うサービス休止

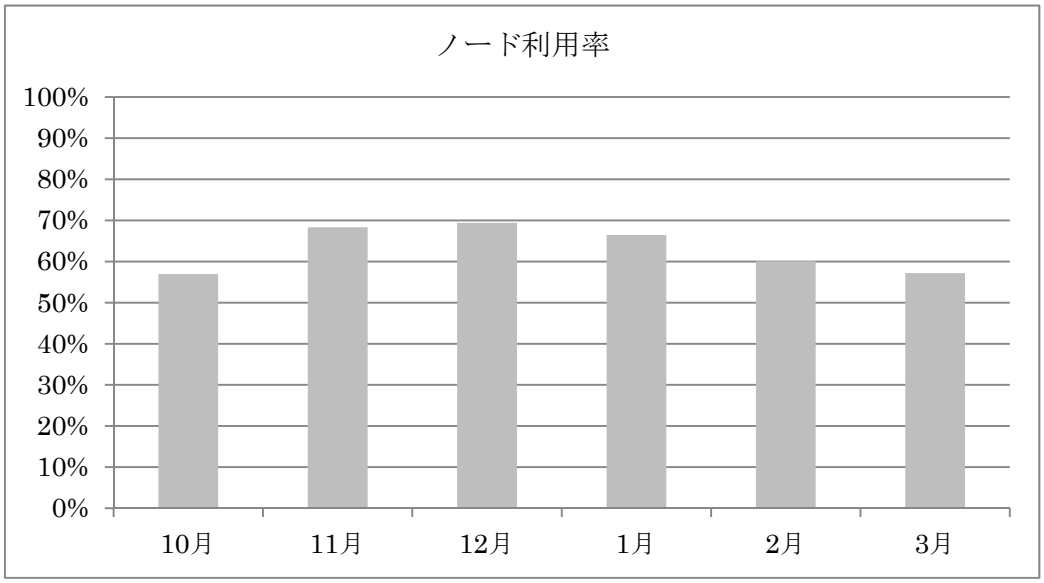
保守開始日時		サービス再開日時		保守時間[h]
2015/10/12	6:00	2015/10/14	9:30	51.50
2015/12/10	9:00	2015/12/11	9:30	24.50
2016/03/30	9:00	2016/04/01	0:00	39.00

システムダウン障害発生状況

障害発生日時	サービス再開日時	ダウン時間[h]

2）サービス状況

	サービス時間[h]	バッチ					
		処理件数	経過時間[h]	占有時間[h]	CPU時間[h]	平均稼動ノード数	ノード利用率
10 月	692.50	22,951	51,527	4,368,630	201,544	416.0	57 %
11 月	720.00	8,776	54,945	5,507,240	290,159	416.0	68 %
12 月	719.50	8,814	62,969	6,055,770	264,069	416.0	69 %
1 月	744.00	8,986	73,108	5,746,340	186,356	416.0	66 %
2 月	696.00	10,127	53,657	5,012,120	157,341	416.0	60 %
3 月	705.00	8,328	63,455	5,063,170	122,198	416.0	55 %
計	4277.00	67,982	359,661	31,753,270	1,221,667	416.0	63 %



- 占有時間 = 合計(経過時間×占有コア数)
- 平均稼動ノード数 = 電源 ON 状態のノード数の月平均 (10 分間隔のサンプリングデータより算出)
- ノード利用率 = 稼動ノードに対するジョブが実行されているノードの割合

システム E 運転状況 （2015 年 10 月 ～ 2016 年 3 月 ）

1）保守作業に伴うサービス休止およびシステムダウン障害発生状況

保守作業に伴うサービス休止

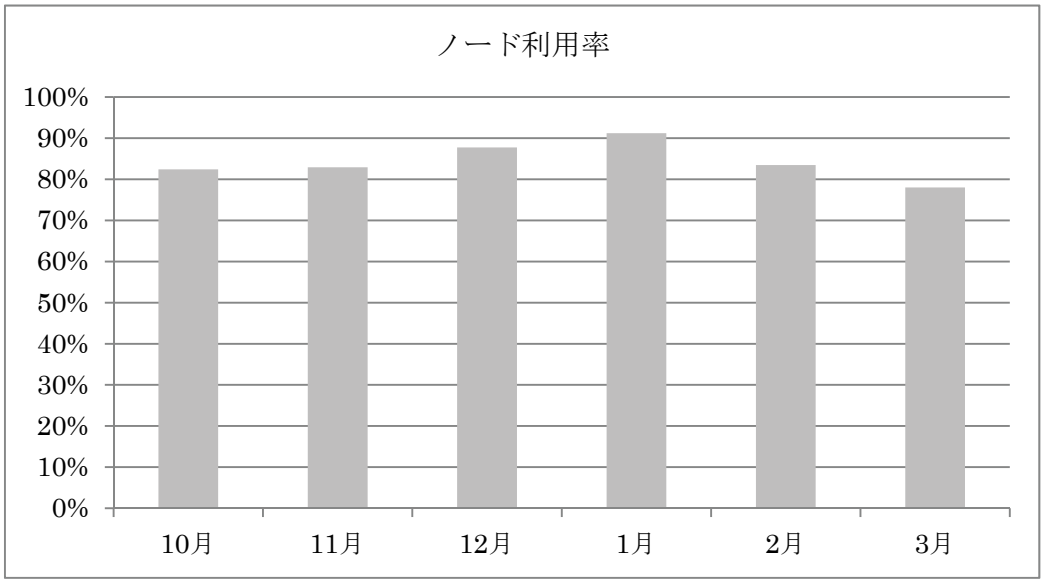
保守開始日時		サービス再開日時		保守時間[h]
2015/10/12	6:00	2015/10/14	9:30	51.50
2015/12/10	9:00	2015/12/11	9:30	24.50
2016/03/30	9:00	2016/04/01	0:00	39.00

システムダウン障害発生状況

障害発生日時	サービス再開日時	ダウン時間[h]

2）サービス状況

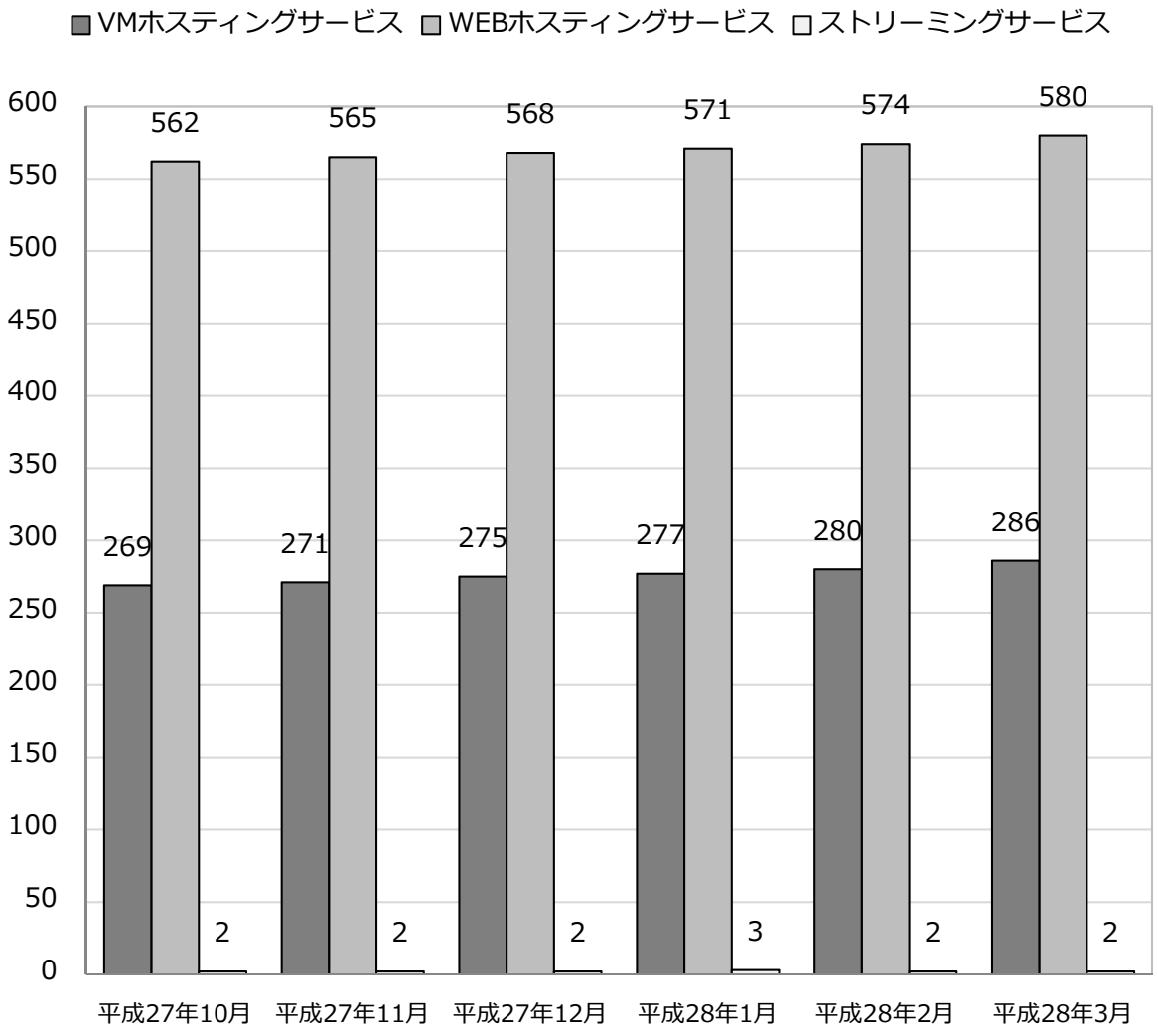
	サービス時間[h]	バッチ					
		処理件数	経過時間[h]	占有時間[h]	CPU時間[h]	平均稼動ノード数	ノード利用率
10 月	692.50	3,513	5,018	1,917,920	208,216	360.0	82 %
11 月	720.00	4,039	6,905	2,175,290	634,217	360.0	83 %
12 月	719.50	3,780	20,257	2,355,670	83,843	360.4	88 %
1 月	744.00	12,644	11,875	2,407,910	1,119,580	360.0	91 %
2 月	696.00	3,541	17,337	2,061,510	123,795	360.0	83 %
3 月	705.00	3,956	16,775	2,070,430	1,154,250	363.5	75 %
計	4277.00	31,473	78,166	12,988,730	3,323,901	360.6	84 %



- 占有時間 = 合計(経過時間×占有コア数)
- 平均稼動ノード数 = 電源 ON 状態のノード数の月平均 (10 分間隔のサンプリングデータより算出)
- ノード利用率 = 稼動ノードに対するジョブが実行されているノードの割合

汎用コンピュータシステムのサービス状況

1. ホスティング・ホームページサービス利用状況



(平成27年10月から平成28年3月)

大型計算機システム利用承認件数について

平成28年3月末現在、大型計算機システムの利用件数は、2,918件となっています。

別表1 スーパーコンピュータシステム

コース	タイプ	セット	利用負担額	提供サービス					
				システム	バッチ	システム資源	経過時間（時間）	ストレージ（TB）	無料利用者数
エントリ	−	基本	12,600 円/年	B	共有	最大1ノード相当((36コア、128GBメモリ)×1)	1	0.2	−
パーソナル	タイプA	基本	100,000 円/年	A	共有	最大4ノード相当((68コア、16+96GBメモリ)×4)	168	2.0	−
	タイプB	基本	100,000 円/年	B	共有	最大4ノード相当((36コア、128GBメモリ)×4)		2.0	
	タイプC	基本	100,000 円/年	C	共有	最大1ノード相当((72コア、3072GBメモリ)×1)		2.0	
	タイプE	基本	100,000 円/年	E	共有	最大2ノード相当((10コア、32GBメモリ+ 1MIC)×2)		2.0	
グループ	タイプA1	最小	230,000 円/年	A	優先	4ノード((68コア、16+96GBメモリ)×4)	336	16.0	8
		追加単位	115,000 円/年			2ノード((68コア、16+96GBメモリ)×2)		8.0	4
	タイプA2	最小	276,000 円/年		準優先	8ノード((68コア、16+96GBメモリ)×8)		19.2	16
		追加単位	69,000 円/年			2ノード((68コア、16+96GBメモリ)×2)		4.8	4
	タイプA3	最小	690,000 円/年		占有	8ノード((68コア、16+96GBメモリ)×8)		32.0	16
		追加単位	345,000 円/年			4ノード((68コア、16+96GBメモリ)×4)		16.0	8
	タイプB1	最小	240,000 円/年	B	優先	4ノード((36コア、128GBメモリ)×4)		16.0	8
		追加単位	120,000 円/年			2ノード((36コア、128GBメモリ)×2)		8.0	4
	タイプB2	最小	288,000 円/年		準優先	8ノード((36コア、128GBメモリ)×8)		19.2	16
		追加単位	72,000 円/年			2ノード((36コア、128GBメモリ)×2)		4.8	4
	タイプB3	最小	720,000 円/年		占有	8ノード((36コア、128GBメモリ)×8)		32.0	16
		追加単位	360,000 円/年			4ノード((36コア、128GBメモリ)×4)		16.0	8
	タイプC1	最小	150,000 円/年	C	優先	1ノード((72コア、3072GBメモリ)×1)		16.0	8
		追加単位	150,000 円/年			1ノード((72コア、3072GBメモリ)×1)		16.0	8
	タイプC2	最小	180,000 円/年		準優先	2ノード((72コア、3072GBメモリ)×2)		19.2	16
		追加単位	90,000 円/年			1ノード((72コア、3072GBメモリ)×1)		9.6	8
	タイプE1	最小	280,000 円/年	E	優先	4ノード((10コア、32GBメモリ+ 1MIC)×4)		16.0	8
		追加単位	140,000 円/年			2ノード((10コア、32GBメモリ+ 1MIC)×2)		8.0	4
	タイプE2	最小	336,000 円/年		準優先	8ノード((10コア、32GBメモリ+ 1MIC)×8)		19.2	16
		追加単位	84,000 円/年			2ノード((10コア、32GBメモリ+ 1MIC)×2)		4.8	4
	タイプE3	最小	840,000 円/年		占有	8ノード((10コア、32GBメモリ+ 1MIC)×8)		32.0	16
		追加単位	420,000 円/年			4ノード((10コア、32GBメモリ+ 1MIC)×4)		16.0	8
大規模ジョブ	タイプA	最小	23,000 円/週(7日)	A	占有	8ノード((68コア、16+96GBメモリ)×8)	168	−	−
		追加単位	11,500 円/週(7日)			4ノード((68コア、16+96GBメモリ)×4)			
	タイプB	最小	24,000 円/週(7日)	B	占有	8ノード((36コア、128GBメモリ)×8)			
		追加単位	12,000 円/週(7日)			4ノード((36コア、128GBメモリ)×4)			
	タイプC	最小	15,000 円/週(7日)	C	占有	2ノード((72コア、3072GBメモリ)×2)			
		追加単位	7,500 円/週(7日)			1ノード((72コア、3072GBメモリ)×1)			
	タイプE	最小	28,000 円/週(7日)	E	占有	8ノード((10コア、32GBメモリ+ 1MIC)×8)			
		追加単位	14,000 円/週(7日)			4ノード((10コア、32GBメモリ+ 1MIC)×4)			
専用クラスタ	−	最小	720,000 円/年	B	−	8ノード((36コア、128GBメモリ)×8)	−	32.0	16
		追加単位	360,000 円/年			4ノード((36コア、128GBメモリ)×4)		16.0	8
ストレージ容量追加			10,000 円/年	ストレージ容量10TBの追加につき					
ライセンスサービス			20,000 円/年	可視化ソフト（AVS,ENVI/IDL)およびプリポストウェアの1ライセンスにつき					

備考

1. 利用負担額は、年度単位で算定している。また、総額表示である。パーソナルコース、グループコース又は専用クラスタコースを、年度途中から利用を開始する場合及び年度途中で利用を終了する場合の利用負担額は、上記表中の利用負担額を12で除した後、利用月数を乗じて算出するものとし、100円未満に端数が出た場合は、10円単位を四捨五入するものとする。
   なお、月途中から利用を開始する場合及び月途中で利用を終了する場合は、それぞれ1月の利用とする。

2. 大型計算機システムの全ての利用者は、上記表のサービスの他、次のサービスを受けることができる。
   1） 大判プリンタサービス
   2） その他、大型計算機システムが提供するサービス、機器の利用

3. 上記表の大規模ジョブコース、ストレージ容量追加、ライセンスサービスの申請には、スーパーコンピュータシステムの利用者であることが必要である。

4. 「共有」: 当該カテゴリのユーザ間で一定の計算資源を共有するベストエフォートのスケジューリングを行う。
   「準優先」: 定常稼働状況において記載値（以上）の計算資源が確保されるように優先スケジューリングを行う。
   また、稼働状況によらず記載値の1/4の計算資源が確保されることを保証する。
   「優先」: 定常稼働状況において記載値（以上）の計算資源が確保されるように優先スケジューリングを行う。
   また、稼働状況によらず記載値の1/2の計算資源が確保されることを保証する。
   「占有」: 稼働状況によらず記載値の計算資源が確保されることを保証する。

5. ストレージ容量はバックアップ領域（最大で総容量の1/2）を含む。

6. グループコース及び専用クラスタコースの利用者番号は利用者あたり年額5,000円を負担することで追加できる。

7. 機関・部局定額制度
   他機関又は学内における部局（『国立大学法人京都大学の組織に関する規程』第3章第2節から第11節で定める組織をいう。）の組織が、その組織単位でグループコースサービスを利用申請する場合の利用負担額は、別表1に規定するの1.5倍の額とする。なお、利用負担額が年額150万円未満の場合は100人、年額150万円を超える場合は、150万円毎に100人までの利用者を認める。ストレージは、1.5倍の容量とする。

8. スパコン連携サービス
   学術情報メディアセンターのスーパーコンピュータシステムと密な連携により、学内における部局の組織が計算サーバ等を設置する場合、下記の負担額を支払うものとする。

冷却方式	利用負担額	利用負担額算定単位
水冷	11,300 円/月	水冷冷却方式の計算サーバ等の定格電力 1kWにつき
空冷	13,200 円/月	空冷冷却方式の計算サーバ等の定格電力 1kWにつき

別表２（汎用コンピュータシステム）

区分	利用負担額	単位
VM ホスティングサービス	72,000 円／年	1 仮想マシンにつき
ホームページサービス	6,000 円／年	1 ドメイン名につき
ストリーミングサービス	6,000 円／年	1 申請につき

備考

1. 利用負担額は、総額表示である。
2. 上記表の汎用コンピュータシステムのサービスを利用するためには、大型計算機システムの利用者であることが必要である。
3. VM ホスティングサービスにおいて、下記の負担額を支払うことにより CPU、メモリ、ディスクを増量することができる。

区分	利用負担額	単位
CPU 増量	18,000 円／年	2 コアにつき(最大 8 コアまで)
メモリ増量	18,000 円／年	8GB につき(最大 64GB まで)
ディスク増量	18,000 円／年	200GB につき(最大 1,000GB まで)

4. VM ホスティングサービスにおいて VMware を用いる場合は、下記の負担額を支払うことにより VMware の利用及び CPU、メモリ、ディスクを増量することができる。ただし、システム資源が限られているためサービスの提供を限定することがある。

区分	利用負担額	単位
VMware 利用	72,000 円／年	1 仮想マシンにつき
CPU 増量	36,000 円／年	2 コアにつき(最大 8 コアまで)
メモリ増量	36,000 円／年	8GB につき(最大 64GB まで)
ディスク増量	18,000 円／年	200GB につき(最大 1,000GB まで)

5. ホームページサービス及びストリーミングサービスにおいて、下記の負担額を支払うことにより公開スペースの上限を拡大することができる。

区分	利用負担額
公開スペース 20GB プラン	3,000 円／年
公開スペース 50GB プラン	9,000 円／年

6. 利用負担額は、当該年度(4 月から翌年 3 月まで)の利用に対して年額として算定するが、年度途中から利用を開始する場合には月数に応じて減額する。

※民間機関利用

別表3　スーパーコンピュータシステム

システム	システム資源	経過時間（時間）	ストレージ（TB）	無料利用者数	利用負担額
A	8ノード(68コア、16+96GBメモリ)×8)	336	19.2	16	1,104,000 円/年
	12ノード(68コア、16+96GBメモリ)×12)	336	28.8	24	1,656,000 円/年
	16ノード(68コア、16+96GBメモリ)×16)	336	38.4	32	2,208,000 円/年
B	8ノード(36コア、128GBメモリ)×8)	336	19.2	16	1,152,000 円/年
	12ノード(36コア、128GBメモリ)×12)	336	28.8	24	1,728,000 円/年
	16ノード(36コア、128GBメモリ)×16)	336	38.4	32	2,304,000 円/年
C	2ノード(72コア、3072GBメモリ)×2)	336	19.2	16	720,000 円/年
	3ノード(72コア、3072GBメモリ)×3)	336	28.8	24	1,080,000 円/年
	4ノード(72コア、3072GBメモリ)×4)	336	38.4	32	1,440,000 円/年
E	8ノード(10コア、32GBメモリ+ 1MIC)×8)	336	19.2	16	1,344,000 円/年
	12ノード(10コア、32GBメモリ+ 1MIC)×12)	336	28.8	24	2,016,000 円/年
	16ノード(10コア、32GBメモリ+ 1MIC)×16)	336	38.4	32	2,688,000 円/年

備考

1. 利用負担額は、年度単位で算定している。また、総額表示である。パーソナルコース、グループコース又は専用クラスタコースを、年度途中から利用を開始する場合及び年度途中で利用を終了する場合の利用負担額は、上記表中の利用負担額を12で除した後、利用月数を乗じて算出するものとし、100円未満に端数が出た場合は、10円単位を四捨五入するものとする。
   なお、月途中から利用を開始する場合及び月途中で利用を終了する場合は、それぞれ1月の利用とする。

2. ストレージ容量はバックアップ領域（最大で総容量の1/2）を含む。

全国共同利用版広報・Vol.14(2015)総目次

[巻頭言]

[スパコン応用研究]

[研究会開催報告]

[スーパーコンピュータ共同研究制度（若手研究者奨励枠）研究報告]

[プログラム高度化支援事業研究報告]

[スーパーコンピュータ共同研究制度（大規模計算支援枠）研究報告]

[サービスの記録・報告]

[資料]

[編集後記]

― サービス利用のための資料一覧 ―

1. スーパーコンピュータシステム・ホスト一覧

- システム A：camphor.kudpc.kyoto-u.ac.jp
- システム B・C：laurel. kudpc.kyoto-u.ac.jp
  - システム B（SAS 利用時）：sas.kudpc.kyoto-u.ac.jp
- システム E：camellia.kudpc.kyoto-u.ac.jp

※ ホストへの接続は SSH(Secure SHell) 鍵認証のみ、パスワード認証は不可

2. 問い合わせ先 & リンク集

- 情報環境機構のホームページ
  http://www.iimc.kyoto-u.ac.jp/

- 学術情報メディアセンターのホームページ
  http://www.media.kyoto-u.ac.jp/

- スーパーコンピュータシステムに関する問い合わせ先
  - 利用申請などに関する問い合わせ先
    【情報環境支援センター】
    E-mail：zenkoku-kyo@media.kyoto-u.ac.jp / Tel：075-753-7424
    URL: http://www.iimc.kyoto-u.ac.jp/ja/services/comp/

  - システムの利用など技術的な問い合わせ先
    【スーパーコンピューティング掛】
    E-mail：consult@kudpc.kyoto-u.ac.jp / Tel：075-753-7426
    URL: http://www.iimc.kyoto-u.ac.jp/ja/services/comp/contact.html

- ホームページ・ホスティングサービス関する問い合わせ先
  【クラウドコンピューティング掛】
  E-mail：whs-qa@media.kyoto-u.ac.jp / Tel：075-753-7494
  URL: http://www.iimc.kyoto-u.ac.jp/ja/services/whs/

編 集 後 記

既報および本広報誌の紹介の通り，2016 年 10 月より新スーパーコンピュータシステム Camphor 2 のサービスを，また当初予定よりも遅れてしまいましたが 12 月末より Laurel 2 および Cinnamon 2 のサービスも開始いたしました．本広報誌もこの遅れの影響を受け，当初予定していた Laurel 2 についての性能評価報告は間に合いませんでしたが，次年度以降にバーストバッファを含めた評価報告の掲載を検討しておりますのでいましばらくお待ちいただければ幸いです．

（副部会長）

京都大学学術情報メディアセンター全国共同利用版広報　Vol. 15, No. 1

2017年 2月 6日 発行

編集者　京都大学学術情報メディアセンター
　　全国共同利用版広報編集部会
発行者　〒606-8501　京都市左京区吉田本町
　　京都大学学術情報メディアセンター
　　Academic Center for Computing and Media Studies
　　Kyoto University
　　Tel. 075-753-7400
　　http://www.media.kyoto-u.ac.jp/
印刷所　〒616-8102 京都市右京区太秦森ヶ東町 21-10
　　株式会社エヌジーピー

目次

Cover : Designed by TANI Takushi / T&T Design Lab. (Feb,2O17)