新しいスーパーコンピュータ HPC2500 のサービス

— VPP800 からの移行 —

浅岡 香枝 *　　平野 彰雄 *

1　はじめに

京都大学学術情報メディアセンターでは、平成 16 年 3 月、スーパーコンピュータを、Fujitsu VPP800/63(以降、VPP) から **Fujitsu PRIMEPOWER HPC2500** (以降、HPC2500) にリプレースしました。

HPC2500 のシステムの概要やハードウェアの特徴に関しては、前号と前々号の広報 [1][2] で紹介していますので、本稿では、まず、HPC2500 がもつ高速化のための機能を紹介し、サービス形態について述べます。また、VPP からの移行という観点から、HPC2500 での並列化方法や基本的なサービスおよび、言語ソフトウェアの使い方を解説します。

2　高速化のための機能

HPC2500 では、プログラムの高速実行のために、ラージページメモリ、ハードウェアバリア、DTU(Data Transfer Unit) という機能があります。

ラージページメモリは、大規模演算のための機能で、メモリページのサイズをシステム標準の 8KB から 4MB に拡張したメモリ領域です。そして、ラージページメモリでは、メモリ領域を物理的に固定して、ページングを抑止しています。なお、ラージページメモリとして使用しない残りのメモリ領域は、ラージページメモリと対比して、**ノーマルページメモリ**と呼んでいます。ラージページを使用するには、結合編集時に-Klargepage=2 オプションを指定して実行ファイルを作成しておく必要があります。ラージページメモリに割付けられるのは、データ域、ヒープ域およびスタック域です。

ハードウェアバリアは、複数 CPU 上で実行されているスレッド間やプロセス間の同期をハードウェアを使って高速に処理する機能です。ハードウェアバリアを使用するには、結合編集時に-Khardbarrier オプションを指定して実行ファイルを作成しておく必要があります。

DTU は、プロセス間のデータ転送を行うための専用ハードウェアで、データ転送を高速に処理します。DTU を使用するには、-Klargepage=2 を結合編集時に指定して実行ファイルを作成しておく必要があります。

なお、**ハードウェアバリア、DTU は、NQS 実行時のみ利用できます。**

3　計算ノードの構成とメモリ資源

HPC2500 のノードの構成および用途を表 1 に示します。

HPC2500 には 11 台の計算ノードがありますが、運用では、このうちの 8 台を論理的に分割しています。分割は、1 物理ノードを 4 論理ノードに分けていますので、分割されたノード (以降、分割ノード) は、合計 4×8=32 台になります。分割ノード 1 台あたりの CPU 数は 32 個、メモリ容量は 128GB となります。また、残りのノード 3 台は非分割ですが、このうちの 1 台を TSS サービスを行うログインノードとしてサービスしています。

4　プログラミングモデルの対応と並列の種類

VPP でのプログラミングモデルは、HPC2500 では次のようになります。

○ ベクトル処理 (1PE)
　　→　自動並列化 (複数 CPU)
○ MPI (複数 PE)
　　→　MPI+自動並列化 (複数 CPU)
○ VPP Fortran (複数 PE)
　　→　XPFortran+自動並列化 (複数 CPU)

* あさおか かえ，ひらの あきお（京都大学 学術情報メディアセンター）

表 1: 計算ノードの構成および用途

ノードタイプ　(ノード数)	CPU 数	メモリ		用途
		ラージページ	ノーマルページ	
ログインノード　(1)	128	320GB	192GB	TSS および NQS サービス
非分割ノード　(2)	128	400GB	112GB	NQS サービス
分割ノード　(32)	32	100GB	28GB	NQS サービス

VPP のベクトル処理は、HPC2500 では**自動並列化**による並列処理が対応します。また、VPP の MPI(または VPP Fortran) は、各 PE でベクトル処理、そして、PE 間で並列処理を行うので、HPC2500 では自動並列化と MPI(または XPFortran) による 2 階層の並列処理が対応します。

この 2 階層の並列処理のうち、自動並列化の階層を**スレッド並列**といい、MPI(または、XPFortran) の階層を**プロセス並列**といいます。

スレッド並列は、1 つのプロセスの中にスレッドを複数個生成させ、各スレッドが並列に処理を行うという並列方法です。スレッド間ではメモリ空間を共有します。HPC2500 でスレッド並列を行う方法としては、自動並列化の他にも、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規格である **OpenMP** を用いる方法があります。自動並列化は、コンパイラが並列化可能であると判断できるループと配列演算のみの並列化ですので、きめ細かな並列化指示や、プログラム移植性を考えると、OpenMP の利用も必要になってくるかと思います。

一方、プロセス並列とは、複数プロセスを起動して、各プロセスがプロセス間通信を行いながら、並列に処理を行うという並列方法です。

スレッド並列＋プロセス並列を**ハイブリッド並列**と呼びます。図 1 に、ハイブリッド並列プログラムの実行イメージを示します。

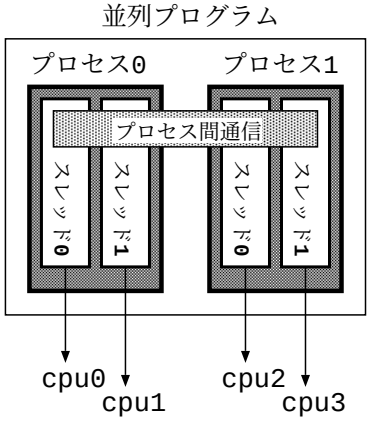

図 1: ハイブリッド並列のプログラムイメージ

図 1 は、2 プロセスで並列化、さらに、各プロセスを 2 スレッドで並列化したものです。

5　HPC2500 のサービス

5.1　ホスト名

HPC2500 のホスト名 (FQDN) は、次のとおりです。

```
hpc.kudpc.kyoto-u.ac.jp
```

なお、HPC2500 では、セキュリティー上の観点から、SSH(Secure SHell) による接続のみ許可し、telnet や ftp による接続は禁止しています。

SSH による接続方法については、今月号の広報 [3] で解説していますので、そちらを参照ください。

5.2　ファイル許可量

HPC2500 では、VPP と同様に、大きな演算結果の書き出しなどで使用していただくための大容量ファイルシステム/LARGE をサービスしています。

表 2 に、ホームディレクトリ、/LARGE の利用者あたりの許可量を示します。

表 2: 利用者あたりのファイル許可量

項目	容量	個数
ホームディレクトリ	512GB	30000 個
/LARGE	1024GB	2000 個

5.3　TSS のプロセスリミット

TSS でのプロセスリミットは、CPU 時間 (cputime) については、標準で 20 時間、最大 160 時間、仮想メモリサイズ (vmemoryuse) については、標準 2GB、最大 16GB としています。プロセスリミットは、limit コマンドで確認でき、unlimit コマンドで最大許可量まで拡張できます。

なお、上記の値はプロセスあたりの許可量なので、スレッド並列実行時には、上記の CPU 時間の許可量は、**全スレッドで消費する CPU 時間の合計に対しての許可量**となります。

表 3: キュー名と許可量

キュー名	最大 CPU 数	プロセス資源セット数 (プロセス数) (-lP)		プロセス資源セット当りの CPU 数 (-lp)		プロセス資源セット当りのラージページメモリサイズ (-lM)		CPU 時間 (-lT)	
		標準	最大	標準	最大	標準	最大	標準	最大
s8	8	1	1	8	8	1GB	20GB	6 時間	20 時間
s128	128	1	1	8	128	1GB	400GB	6 時間	20 時間
d32	32	1	32	8	32	1GB	100GB	6 時間	20 時間
d128	128	1	32	8	32	1GB	100GB	6 時間	20 時間
d512	512	1	128	8	32	1GB	100GB	6 時間	20 時間

5.4 NQS

ジョブの投入 (qsub)、キャンセル (qdel)、確認 (qstat) といった NQS 一連のコマンドおよび NQS 用スクリプトの記述方法は VPP と同じです。

表 3 に NQS のキュー名と許可量を示します。

5.4.1 キューの概要

キューは、ジョブの割付け先ノードおよび実行ジョブの種類により、大きく分けて 2 グループあり、これをキュー名の先頭の英字 (s,d) で表しています。s で始まるキューは、スレッド並列ジョブ用キュー で非分割ノードで実行されます。また、d で始まるキューはプロセス並列ジョブ用、ハイブリッド並列ジョブ用キューで、分割ノードに跨って実行されます。

5.4.2 許可量の指定

HPC2500 では、プロセスあたりの資源の許可量を管理している単位 (これを、プロセス資源セットと呼びます) があり、ジョブ全体として要求する資源の量は、プロセス資源セットの数 (すなわち、プロセス数)(-lP) とプロセス資源セット当りの各資源の量 (-lp,-lM) で決まります。

ジョブが要求する総 CPU 数は、プロセス資源セット数 (-lP) とプロセス資源セットあたりの CPU 数 (-lp) の積となります。この総 CPU 数の許可量が、表 3 の最大 CPU 数です。また、**ジョブが要求するラージページメモリサイズも、プロセス資源セット数 (-lP) とプロセス資源セットあたりのラージページメモリサイズ (-lM) の積**となります。-lM には、gb(ギガバイト)、mb(メガバイト) 単位で指定します。

-lP と-lp および-lM の指定は、例えば、次のようになります。

1) スレッド並列プログラムの場合

```
-lP 1 -lp 8 -lM 10gb
```

→ 1 × 8=8 個の CPU を要求

→ 1 × 10=10gb のラージページメモリを要求

2) プロセス並列+スレッド並列プログラムの場合

```
-lP 4 -lp 8 -lM 10gb
```

→ 4 × 8=32 個の CPU をの要求

→ 4 × 10=40gb のラージページメモリを要求

また、CPU 時間 (-lT) は、1CPU あたりの指定です。並列実行した CPU のうちの最大 CPU 時間が指定値を越えると実行が打ち切られます。

スタックサイズは、-ls オプションで、mb(メガバイト)、kb(キロバイト) 単位で指定します (例 -ls 10mb)。この指定値は、プロセスおよび各スレッド毎に確保される領域の大きさです。なお、実際に必要となったスタック量は、環境変数 **FLIB_USE_STACK_INFO** を指定してプログラムを実行することでわかります (ただし Fortran のみ)。

5.5 言語系ソフトウェア

翻訳／結合編集のコマンドやオプション、また、実行方法が VPP と異っているものありますので、ここでは言語毎に主な変更点について解説します。なお、コンパイラのオプションについては、ここにあげたもの以外にも、多くの変更がありますので、ホームページにある言語系の移行資料や各言語のマニュアルでご確認ください。

5.5.1 Fortran,C,C++

各言語毎のサポート仕様および、VPP と HPC2500 とのコマンド名の対応を表 4 に示します。また、HPC2500 では、アドレッシングモードの指定、ハードバリア、ラージページを使用するためのオプションなどは各言語に共通です。これらのオプションを表 5 に示します。

1) -K オプションの複数指定方法

-K ではじまるオプションは、例えば、次のようにカンマで区切ってまとめて指定することができます。

```
-KV9,hardbarrier,largepage=2,parallel
```

表 4: 翻訳／結合編集コマンド

言語	仕様	VPP	HPC2500
Fortran	ISO/IEC 1539-1:1997(JIS X3001-1:1998) (通称 Fortran95)	frt	frt
C	ISO/IEC 9899:1990,K&R	cc	fcc
	ISO/IEC 9899:1999 (通称 C99)	—	c99
C++	ISO/IEC 14882:1998	CC	FCC

表 5: 共通オプション

オプション	意味	指定
-KV9	64bit アドレッシングモードの指定	翻訳時と結合編集時
-Khardbarrier	ハードウェアバリアの指定	結合編集時
-Klargepage=2	ラージページの指定	結合編集時
-Kparallel	自動並列化の指定	翻訳時と結合編集時
-KOMP	OpenMP 仕様を解釈	翻訳時と結合編集時

2）アドレスモードの指定

2GB 以上のメモリ空間を扱う場合には、64bit アドレッシングモードの指定 (-KV9) が必要です。

3） 翻訳情報出力オプション (Fortran)

VPP で翻訳情報を出力する-P 系のオプションは、HPC2500 では-Q 系のオプションに変更されています。-Q 系オプションのうち、-Qt を指定しておけば、ソースリスト、有効となった翻訳オプション、最適化情報、並列化情報、消費するスタック情報など、とりあえず必要そうな情報が得られます。

また、翻訳情報は、ソースファイルの拡張子を.lst としたファイル名に出力されます。

4） 翻訳情報出力オプション (C,C++)

VPP で翻訳情報を出力する-Ksrc -Ksta オプションが HPC2500 でも使えます。ただし、ファイルに出力するための-Z オプションが HPC2500 にはありませんので、次のようにリダイレクションを使ってファイルに出力してください。

```
hpc% fcc -Ksrc,sta sample.c >& sample.lst
```

5） Fortran のデバッグオプション

未定義変数の引用、配列の添字チェック、仮引数と実引数の対応などを翻訳時や実行時にチェックするための VPP での-D 系のオプションが、HPC2500 では-H 系に変更されています。すべてのチェックを行うためには、-Haesux と指定してください。また、同様な機能として-Eg オプションがあり、これは、-Haesux に加えて、手続きの特性や共通ブロックの大きさの検査も行います。

5.5.2 自動並列化

-Kparallel の指定により、コンパイラが自動的にプログラムを並列化します。プログラム中の並列化された部分は、翻訳時メッセージや翻訳情報から確認できます。並列化に関する翻訳時メッセージの出力には、Fortran の場合は-Kpmsg もしくは-Et オプション、C/C++の場合は-Kpmsg オプションを指定します。

【翻訳／結合例】

```
hpc% frt -Kparallel,hardbarrier,V9,
             largepage=2 -Et -Qt sample.f
```

Fortran の場合は、-Qm オプションをつけることで、OpenMP プログラムへ変換でき、ファイル名には、拡張子の前に omp がつきます。

【OpenMP への変換】

```
hpc% frt -c -Kparallel -Qm sample.f
            → sample.omp.f が生成
```

実行は、環境変数 **PARALLEL** に並列数を指定します。

【実行例】

```
hpc% setenv PARALLEL 8
hpc% ./a.out
```

【NQS 用サンプルスクリプト】

```
# ----- サンプルスクリプト ----
# @$-q s8
# @$-eo
# @$-lP 1
# @$-lp 8
# @$-lM 2gb
#
PARALLEL=8;export PARALLEL
#
cd $QSUB_WORKDIR
./a.out
# -----------------------------
```

5.5.3 OpenMP

HPC2500 では、次の OpenMP の仕様をサポートしています。

- OpenMP Fortran API Version 2.0
- OpenMP C and C++ API Version 2.0

OpenMP プログラムの翻訳および結合編集には、**-KOMP** オプションを指定します。

【翻訳／結合例】

```
hpc% frt -KOMP,hardbarrier,V9,
                  largepage=2 -Qt sample.f
```

実行は、環境変数 **OMP_NUM_THREADS** に並列数を指定して実行します。

【実行例】

```
hpc% setenv OMP_NUM_THREADS 8
hpc% ./a.out
```

また、OpenMP でハードウェアバリアを使用するには、結合編集時の-Khardbarrier オプションの指定だけでなく、実行時に、環境変数 **FLIB_FASTOMP** に"TRUE"を設定する必要があります。

【NQS 用サンプルスクリプト】

```
# ----- サンプルスクリプト ----
# @$-q s8
# @$-eo
# @$-lP 1
# @$-lp 8
# @$-lM 2gb
#
OMP_NUM_THREADS=8;export OMP_NUM_THREADS
FLIB_FASTOMP="TRUE";export FLIB_FASTOMP
#
cd $QSUB_WORKDIR
./a.out
# ----------------------------
```

5.5.4 MPI

HPC2500 の MPI ライブラリは、VPP と同様に MPI2 規格に準拠しています。

HPC2500 でも MPI プログラムのための翻訳／結合編集コマンドが用意されています。VPP とのコマンドの対応を表 6 に示します。

表 6: MPI プログラムの翻訳／結合編集コマンド

言語	VPP	HPC2500
Fortran	mpifrt	mpifrt
C	mpicc	mpifcc
	—	mpic99 (*1)
C++	mpiCC	mpiFCC

(*1)C99 仕様での翻訳／結合編集

表 6 に示すコマンドには、対応する言語コンパイラ (frt,fcc 等) のオプションを指定することができます。

【MPI プログラムの翻訳／結合編集例】

```
hpc% mpifrt -KV9,hardbarrier,
   largepage=2,parallel sample.f
```

また、MPI プログラムの実行ですが、HPC2500 では、mpiexec コマンドを用います。

【MPI プログラムの実行例】

```
hpc% mpiexec -mode limited -n 4 ./a.out
```

【NQS 用サンプルスクリプト】

```
# ----- サンプルスクリプト ----
# @$-q d32
# @$-eo
# @$-lP 4
# @$-lp 8
# @$-lM 2gb
#
PARALLEL=8;export PARALLEL
OMP_NUM_THREADS=8;export OMP_NUM_THREADS
FLIB_FASTOMP="TRUE";export FLIB_FASTOMP
#
cd $QSUB_WORKDIR
mpiexec -mode limited -n 4 ./a.out
# ----------------------------
```

mpiexec コマンドの-mode オプションに"limited"を指定し[1]、-n オプションに並列数を指定します。

5.5.5 XPFortran(VPP Fortran 互換)

HPC2500 では、VPP Fortran 言語仕様 (!xocl) を包含した XPFortran 言語仕様をサポートしています。

XPFortran プログラムの翻訳／結合編集のためのコマンドは xpfrt です。また、実行には、xpfexec コマンドを使います。

【翻訳／結合例】

```
hpc% xpfrt -Kparallel,hardbarrier
  -Wx,-Rdst,-Ys sample.lst sample.f
```

xpfrt コマンドの-Wx,-Rdst,-Ys sample.lst は、ソースリスト等の翻訳情報を sample.lst というファイルへ出力するためのオプションです。

【実行例】

```
hpc% xpfexec -mode limited -vp 8 ./a.out
```

xpfexec コマンドの-vp オプションで並列数を指定します。

5.5.6 数値計算ライブラリ

HPC2500 でも、VPP で利用できた SSLII 等の数値計算ライブラリが利用できます。ライブラリ結合のオプションの対応を表 7 に示します。

[1] MPI2 の動的プロセス生成機能や並列入出力機能を使う場合には -mode オプションに"full"を指定します

また、HPC2500 には、OpenMP により、スレッド並列化された SSL II、C-SSL II、BLAS、LAPACK が用意されています (以降、スレッド並列版と呼びます)。スレッド並列版のライブラリ結合のオプションを表 8 に示します。

表 7: 数値計算ライブラリの結合オプション

ライブラリ	VPP	HPC2500
SSL II	-lssl2vp	-SSL2
C-SSL II	-Wg,-S -DMAIN_-main	-KSSL2
BLAS	-lblasvp	-SSL2
LAPACK	-llapackvp -lblasvp	-SSL2
SSL II (VPP Fortan)	-lssl2vpp	-KOMP,parallel -Wx,-SSL2XPF -SSL2BLAMP
ScaLAPACK	-scalapack	-SCALAPACK -SSL2

表 8: スレッド並列版ライブラリの結合オプション

ライブラリ	オプション
SSL II スレッド並列版	-KOMP -SSL2
C-SSL II スレッド並列版	-KOMP -KSSL2
BLAS スレッド並列版	-KOMP -SSL2BLAMP
LAPACK スレッド並列版	-KOMP -SSL2BLAMP

表 8 に示すライブラリのうち、BLAS、LAPACK は、従来のものと同じインタフェースで使えますので、結合オプションを変更するだけでスレッド並列版が使用できます。一方、SSL II と C-SSL II のスレッド並列版は、従来の SSLII とインタフェースが異なっていますので、ソース変更をしてからお使いください。

6 VPP の利用者ファイル

VPP の利用者のファイルは、ホームディレクトリ、/LARGE ともに HPC2500 のファイルシステムへセンター側で移行しています。ただし、VPP と HPC2500 では、OS やコマンドのパス名などが異なっているので、.cshrc 等の環境設定ファイルは、HPC2500 で正常に動作しない可能性があります。そこで、VPP のホームディレクトリ直下にあった .(ドット) ではじまるファイルについては、HPC2500 のホームディレクトリにそのままの形でコピーせずに、VPPRC というディレクトリをつくって、その下に移動しています。

7 実効 CPU 時間

HPC2500 は、複数の CPU がメモリを共有する SMP システムなので、他プロセスとのメモリ競合により、同一のプログラムを実行しても、CPU 時間が変動してしまいます。

そこで、HPC2500 には、CPU 時間をメモリ競合を考慮して補正する機能があります。補正された CPU 時間のことを**実効 CPU 時間**と呼び、補正前の CPU 時間のことを**実 CPU 時間**と呼んでいます。

NQS ジョブの CPU 時間制限および課金は、実 CPU 時間ではなく、実効 CPU 時間に対して処理しています。詳細はマニュアルをご覧ください。

8 おわりに

HPC2500 の機能とサービス形態、また、HPC2500 のサービスについてかけ足で解説しました。

今回のスーパーコンピュータリプレースは、ベクトル型からスカラ型へ、また、分散メモリ型から共有メモリ型へと、アーキテクチャが大きく変っていることもあり、今までのリプレースに比べると、慣れるまでは、難しいと思っています。このような中、より使いやすい、また、よりわかりやすいサービスのためには、利用者からの質問や要望がたいへん有効です。次のアドレスでご意見等を受付けていますので、ご活用ください。

consult@kudpc.kyoto-u.ac.jp

なお、HPC2500 のサービスに関する最新情報はセンターのホームページでも解説していきますので、そちらもあわせてご覧ください。マニュアルも置いてあります。

http://www.kudpc.kyoto-u.ac.jp/

参考文献

[1] 金澤正憲：超並列スーパーコンピュータへの移行 、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.2，No.5，2003

[2] 金澤正憲：新スーパーコンピュータのハードウェアについて、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.3，No.1，2003

[3] 赤坂浩一、浅岡香枝、平野彰雄：新しいスーパーコンピュータを利用するために、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.3，No.2，2004