Windows* 環境での MPI プログラムの作成と実行

2016 年 4 月

内容

- 必要要件と各ツール
- インストール
- コンパイルと実行

必要なツールと環境

- プロセッサーと Windows* OS
- コンパイラーとリンカー
- MPI ライブラリー
- クラスター診断/最適化ツール

プロセッサーと Windows* OS

- インテル® 64 アーキテクチャー・ベースのシステム
  - 1 コアあたり 1GB のメモリーと 1GB 以上の空きディスク容量
- Windows Server 2012 R2 + HPC Pack 2012 R2
  - ドメイン認証やジョブ・コントロールの機能を使用する際に必要
- Windows 7/8/8.1/10
  - パスワード認証でクライアント・システムに接続

参考資料: HPC Pack 2012 R2で最新のクラスタ！InfiniBandも使えます - その1: ヘッドノード構築編

コンパイラーとリンカー

- Visual Studio* 2010/2012/2013/2015

- 最適化コンパイラーとライブラリー
  - インテル® C/C++ および Fortran コンパイラー V14.0 以降
  - その他の最適化コンパイラー

OpenMPI* ライブラリー

Windows* 環境でメッセージ・パッシング・インターフェイスを提供するライブラリー

- インテル® MPI ライブラリー
- Microsoft* MPI (HPC Pack 2012)
- その他オープンソースの MPI ライブラリー

クラスター診断/最適化ツール

クラスターを正しく構成する、そして MPI アプリケーションを効率よくクラスター上で動作させるにはツールの活用が便利

- インテル® MPI ライブラリー/mpitune ユーティリティー/インテル® Trace Collector & Analyzer
- Microsoft HPC Pack
- その他サードパーティー製ツール (Allinea* Forge など)

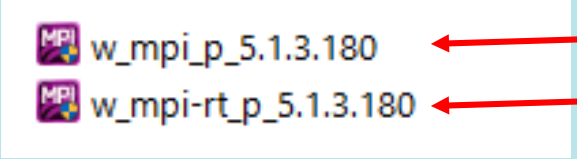
インテル® Parallel Studio XE Cluster Edition

フェーズ	製品名	機能	利点
ビルド	**インテル® Advisor XE**	スレッド設計支援 (Studio 製品のみ)	• 並列アプリケーションの設計を簡単かつ明確にし、迅速化
	インテル® Parallel Studio の各エディション	• C/C++ および Fortran コンパイラー • インテル® TBB • インテル® Cilk™ Plus • インテル® IPP • インテル® MKL	• マルチコアと将来のメニーコアのパフォーマンスおよびスケーラビリティーを引き出すアプリケーションを開発するためのソリューション
	インテル® MPI ライブラリー	ハイパフォーマンスな MPI ライブラリー	• ハイパフォーマンス、スケーラビリティー、インターコネクトの独立性、実行時のファブリック選択、アプリケーション・チューニングを実現
検証 & チューニング	**インテル® VTune™ Amplifier XE**	アプリケーションのパフォーマンスとスケーラビリティーを最適化するパフォーマンス・プロファイラー	• 従来の推測作業を排除し、短時間で容易にパフォーマンスとスケーラビリティーのボトルネックを特定
	インテル® Inspector XE	コードの品質を高める動的なメモリー/スレッド化解析とスタティック解析	• 生産性とコードの品質を高め、コストを削減し、早期にメモリー/スレッド/セキュリティーの問題を発見
	インテル® Trace Analyzer & Collector †	アプリケーションの正当性と動作を理解するための MPI パフォーマンス・プロファイラー	• MPI プログラムのパフォーマンスを解析し、並列アプリケーションの動作と通信パターンを可視化して、hotspot を特定

効率良く、素早く、そして信頼性の高いアプリケーションを開発

インテル® MPI ライブラリーのインストール

- パッケージ (SDK と RTO) を入手します



SDK (開発システムにインストール)

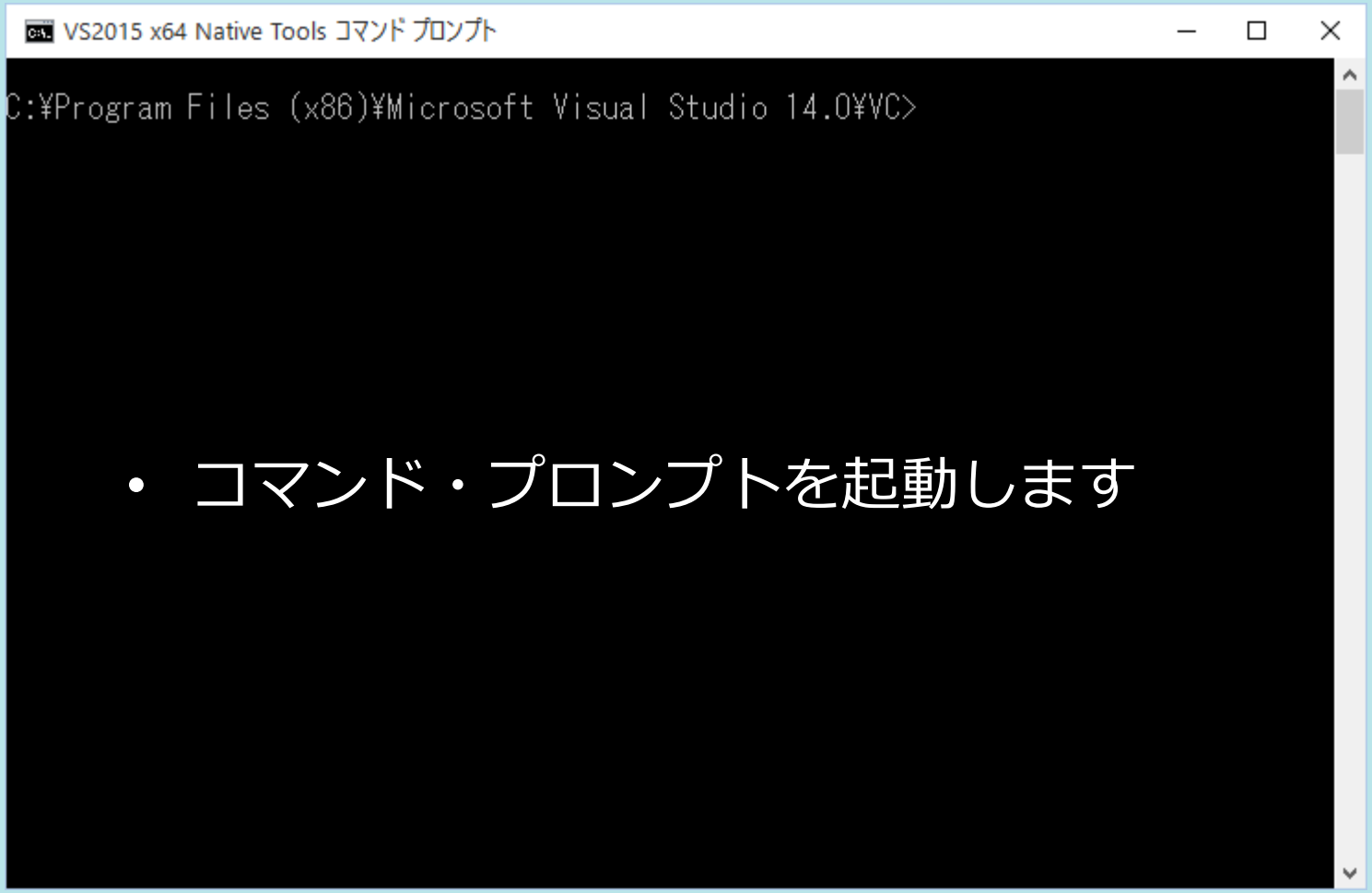
RTO (各実行ノードにインストール)

RTO は以下で要求ください:
https://registrationcenter.intel.com/RegCenter/ComForm.aspx?productid=1395

- パッケージをダブルクリックして、インストールを開始します
  - 以前のバージョンがインストールされている場合、アンインストールする必要はありません
  - インストールには管理者権限が必要です
- デフォルトのインストール先:
  - C:¥Program Files (x86)¥IntelSWTools¥mpi¥[5.1.3.180]

[R2.1][U4.1]

コンパイルとリンク (コマンドライン)



- コマンド・プロンプトを起動します

コンパイルとリンク (コマンドライン)

Intel(R) MPI Library 5.1 Update 3 for Windows* Target Build Environment for Intel(R) 64 applications

```
C:¥Program Files (x86)¥Microsoft Visual Studio 14.0¥VC>cd C:¥Program Files (x86)
¥IntelSWTools¥mpi¥5.1.3.180¥intel64¥bin

C:¥Program Files (x86)¥IntelSWTools¥mpi¥5.1.3.180¥intel64¥bin>mpivars

Intel(R) MPI Library 5.1 Update 3 for Windows* Target Build Environment for Inte
l(R) 64 applications
Copyright (C) 2007-2015 Intel Corporation. All rights reserved.


C:¥Program Files (x86)¥IntelSWTools¥mpi¥5.1.3.180¥intel64¥bin>
```

- mpivars.bat を実行します

```
SET I_MPI_ROOT= C:¥Program Files (x86)¥IntelSWTools¥mpi¥5.1.3.180
SET PATH=%I_MPI_ROOT%¥intel64¥bin;%PATH%
SET LIB=%I_MPI_ROOT%¥intel64¥lib;%LIB%
SET INCLUDE=%I_MPI_ROOT%¥intel64¥include;%INCLUDE%
```

- インテル® コンパイラーのコマンド・プロンプトでは、これは自動的に反映されます
- これで、MPI スクリプトが利用できるようになります (mpicc、mpicl、mpiicc、mpiifort など)

コンパイルとリンク (コマンドライン)

コンパイラーのコマンド	利用可能なコンパイラー	サポートされる言語	サポートされる ABI
一般的なコンパイラー			
mpicc.bat	cl.exe	C	64 ビット
mpicxx.bat	cl.exe	C++	64 ビット
mpifc.bat	ifort.exe	Fortran 77/Fortran 95	64 ビット
Microsoft* Visual C++* コンパイラー			
mpicl.bat	cl.exe	C/C++	64 ビット
インテル® Fortran および C++ コンパイラー 14.0 から 16.0 およびそれ以降			
mpiicc.bat	icl.exe	C	64 ビット
mpiicpc.bat	icl.exe	C++	64 ビット
mpiifort.bat	ifort.exe	Fortran 77/Fortran 95	64 ビット

コンパイルとリンク (コマンドライン)

環境変数または、-cc オプションでコンパイラーを選択できます

例: -cc=cl.exe

Intel Compiler 16.0 Update 1 Intel(R) 64 Visual Studio 2015

```
------------------------------------------------------------------------
The following environment variables are used:
   I_MPI_ROOT       Intel(R) MPI Library installation directory path
   I_MPI_{CC,CXX} or MPICH_{CC,CXX}
                    the path/name of the underlying compiler to be used
   I_MPI_{CC,CXX}_PROFILE or MPI_{CC,CXX}_PROFILE
                    name of profile file (without extension)
   I_MPI_COMPILER_CONFIG_DIR
                    folder which contains configuration files *.conf
   VT_ROOT          Intel(R) Trace Collector installation directory path
------------------------------------------------------------------------
```

[R2.1.1]

コンパイルとリンクの例

```
C:¥mpiicc pi.c
C:¥mpiicc pi.c /Qopenmp /QxHOST
C:¥mpiicc pi.c /Qopenmp /QxHOST /Zi -trace

C:¥mpicc pi.c
C:¥mpicc pi.c –openmp /arch:AVX
C:¥mpicc pi.c –openmp /arch:AVX /Zi -trace

C:¥mpicc pi.c /Qopenmp /QxHOST /Zi      // エラー !!
```

コンパイルとリンク (Visual Studio*)

1. Visual Studio* で、WinXX コンソール・プロジェクトを作成します
2. x64 ソリューション・プラットフォームを選択します
3. インクルード・パスに <installdir>¥intel64¥include を追加します
4. ライブラリー・パスに <installdir>¥intel64¥lib¥<configuration> を追加します。<configuration> に次を設定します：
   - Debug: シングルスレッド版のデバッグ向けライブラリー
   - Release: シングルスレッド版の最適化されたライブラリー
   - Debug_mt: マルチスレッド版のデバッグ向けライブラリー
   - Release_mt: マルチスレッド版の最適化されたライブラリー
5. ターゲットリンク・コマンドに適切なインテル® MPI ライブラリーを追加します：
   - C アプリケーションには impi.lib を追加します。
   - C++ アプリケーションには、impi.lib と impicxx.lib (Release) または、impid.lib と impicxxd.lib (Debug) を追加します。

MPI プログラムを実行する前に

MPI プログラムを実行する前に以下を確認します;

1. 各ノードにインテル® MPI ライブラリーのランタイム・パッケージ (RTO) がインストールされている

2. 各ノードでプロセス管理 (Hydra) サービスが起動されている

impi_hydra	3032	Intel(R) MPI Library Hydra Process Manager	実行中
impi_smpd	3276	Intel(R) MPI Library Process Manager	実行中

3. ノードの認証方法が設定されている
   - パスワードベースの認証でログイン
   - アクティブ・ディレクトリー認証でログイン

4. MPI プログラムやデータを格納する共有ドライブが設定されている

[R2.4] [U7.1]

Hydra プロセス管理

Hydra プロセス管理は、インテル® MPI ライブラリーの SDK や RTO をインストールすると、自動的に Windows* システムにインストールされサービスが起動されます

制御方法:

- Windows* タスクマネージャーの「サービス」で以下を [開始] [停止] [再起動] できます

impi_hydra	3032	Intel(R) MPI Library Hydra Process Manager	実行中
impi_smpd	3276	Intel(R) MPI Library Process Manager	実行中

- コマンドプロンプトから「hydra_service.exe」を起動、オプションには以下を指定できます

オプション		オプション	
-install | -register	Hydra サービスをインストール	-start	Hydra サービスを開始
-uninstall | -remove | -unregister	Hydra サービスをアンインストール	-stop	Hydra サービスを停止
-status <ホスト名>	<ホスト名> のノードのサービスの状態を取得	-restart <ホスト名>	<ホスト名の> サービスを再起動
-register_spn	ノードを Windows ドメインに登録	-remove_spn	ノードを Windows ドメインから削除

認証方法の設定

[R3] [U10.3]

インテル® MPI ライブラリーは、Windows* クラスター全体でユーザー認証を行う 2 つの方法をサポートしています:

- パスワードベースの認証
  mpiexec –register または wmpiregister コマンドでレジストリーへログイン情報を暗号化して登録

- ドメインベースの認証
  これはクラスター管理者によって定義されるドメインポリシーに基づいて管理されます。マシン上にユーザー情報 (ユーザー名とパスワード) を保存する必要はありません

I_MPI_AUTH_METHOD 環境変数に認証方法を設定:

password	パスワードベースの認証を使用します。これは、デフォルト値です
Delegate	委任機能を持つドメインベースの認証を使用します
impersonate	制限付きドメインベースの認証を使用します

ジョブ開始コマンド

[R2.2] [U5]

インテル® MPI ライブラリーとリンクしたアプリケーションを実行するには、mpiexec コマンドを使用します：

1. mpiexec.hydra <g-オプション> <l-オプション> <実行ファイル>
2. mpiexec.hydra <g-オプション> <l-オプション> <実行ファイル> :¥
   <l-オプション> <実行ファイル>
3. mpiexec.hydra -configfile <ファイル>

オプション：

<g-オプション>	すべての MPI プロセスに適用するグローバル・オプション
<l-オプション>	単一の引数セットに適用するローカルオプション
<実行ファイル>	a.exe または path¥a.exe などのファイル名
<ファイル>	コマンドライン・オプションを含むファイル

サンプルコード

```
MPI_Init(&argc, &argv) ;
MPI_Comm_rank(MPI_COMM_WORLD, &my_id) ;
MPI_Comm_size(MPI_COMM_WORLD, &numprocs) ;
MPI_Get_processor_name(name, &namelen);
my_steps = num_steps/numprocs ;

printf("Rank %d Threads %d Running on %s¥n", my_id, 1, name);
for (i=my_id*my_steps; i<(my_id+1)*my_steps ; i++){
   double x = (i+0.5)*step;
   sum += 4.0/(1.0+x*x);
}
sum *= step ;
MPI_Reduce(&sum, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD) ;
MPI_Finalize();
end = clock();

if(my_id == 0)
printf("Pi = %f Time = %f ¥n", pi, (double)(end - start)/CLOCKS_PER_SEC);
```

ジョブ開始コマンドの例

[R2.2] [U5.1]

1. pi_mpi.exe
2. mpiexec.exe -n <プロセス数> pi_mpi.exe
3. mpiexec.exe -hosts 2 host1 4 host2 8 pi_mpi.exe
4. mpiexec.exe -f hostfile –n <プロセス数> pi_mpi.exe
5. mpiexec.exe -machine machinefile pi_mpi.exe
6. mpiexec.exe -genv I_MPI_FABRICS shm -n <プロセス数> pi_mpi.exe
7. mpiexec.exe -configfile config_file

ジョブ開始コマンドの例 (続き)

[R2.2] [U5.1]

8. mpiexec –f hfile –map w:¥¥xxx.xxx.xxx.xxx¥share w:¥mpi¥pi_mpi.exe

9. mpiexec –machine mfile1 –map w:¥¥xxx.xxx.xxx.xxx¥share w:¥mpi¥pi_mpi.exe

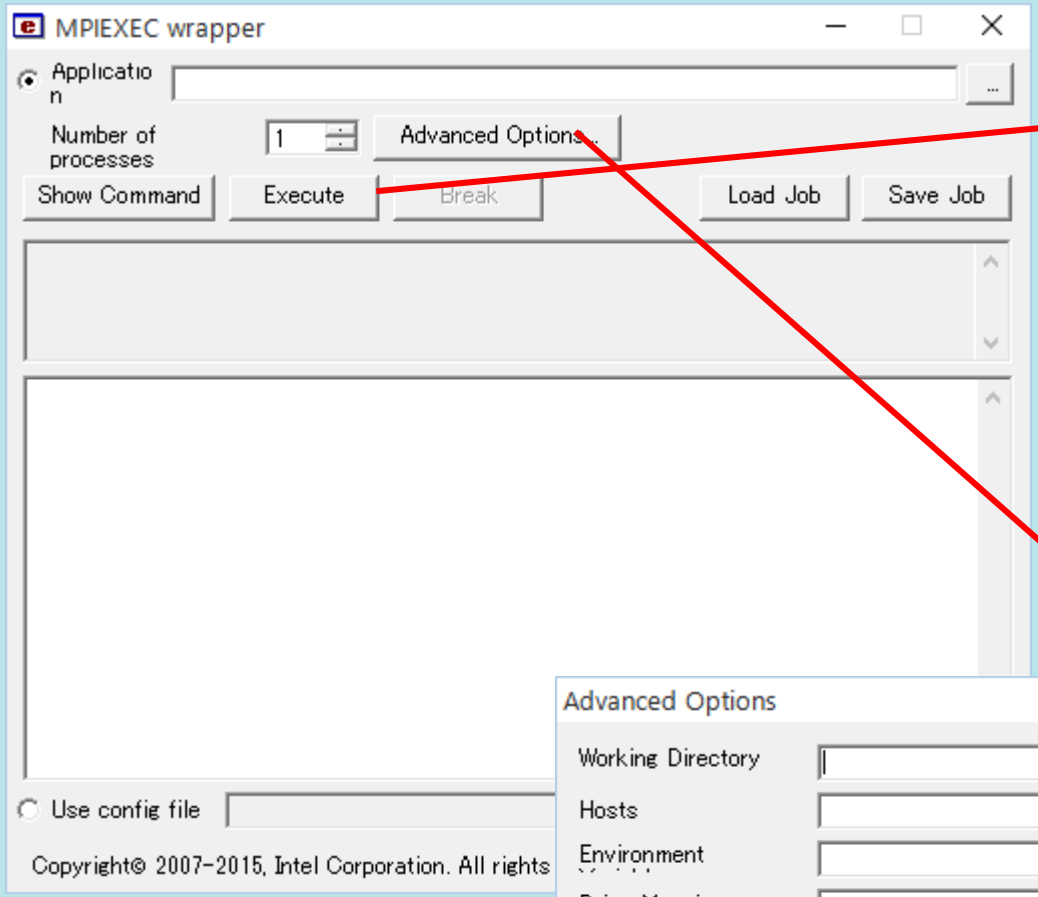
10. mpiexec –machine mfile2 –map w:¥¥xxx.xxx.xxx.xxx¥share ¥
    -genv OMP_NUM_THREADS=4 w:¥mpi¥pi_mpiomp.exe

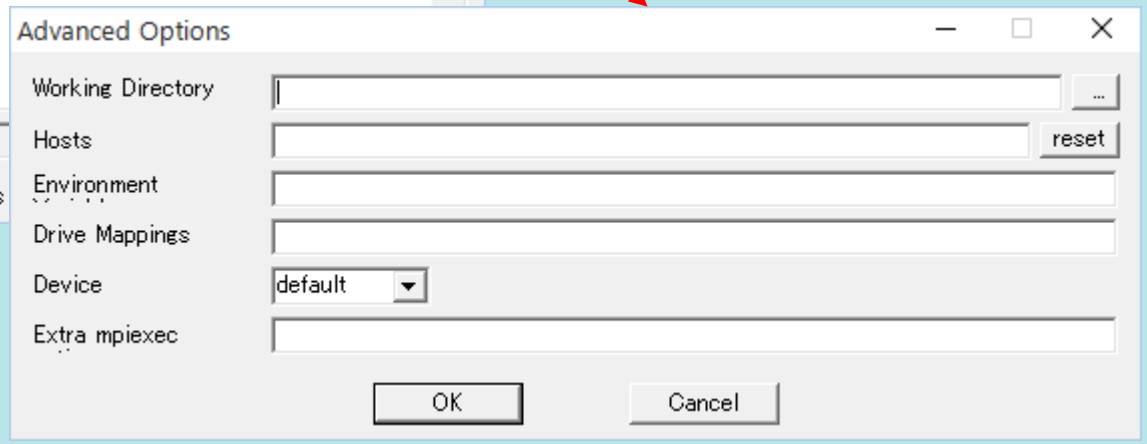
ホストファイル:
xxx.xxx.xxx.xx1
xxx.xxx.xxx.xx2
…
…

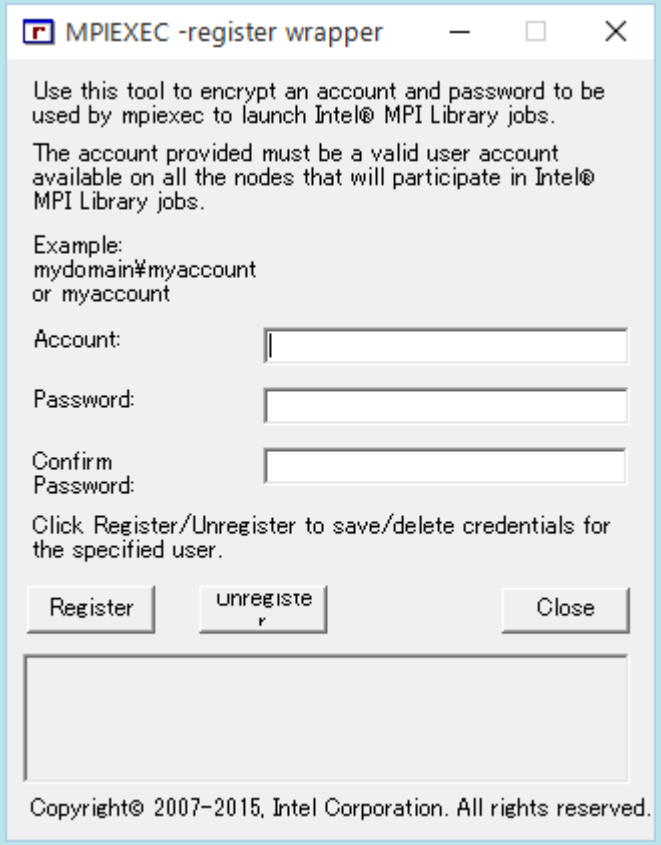
マシンファイル1:
xxx.xxx.xxx.xx1:4
xxx.xxx.xxx.xx2:2
…
…

マシンファイル2:
xxx.xxx.xxx.xx1:1
xxx.xxx.xxx.xx2:1
…
…

WMPIEXEC コマンドでジョブを開始







環境変数

インテル® MPI ライブラリーには次の環境変数が用意されています:

- コンパイルとリンクを制御する環境変数 [U2.1.4]
- ジョブの開始を制御する環境変数 [U2.2.3]
- ジョブ管理システムを制御する環境変数 [U2.4.6]
- ユーザー認証を制御する環境変数 [U3.3]
- チューニングを制御する環境変数 [U4]

Windows* と Linux* が混在したクラスター構成

インテル® MPI ライブラリー (Windows* 版と Linux* 版)では、同じ Hydra プロセス管理を利用するため、OS が混在したクラスターでもジョブを実行することができます

-hostos <windows | linux>
特定のホストにインストールされているオペレーティング・システムを指定します。MPI プロセスは、このオプションの指示に従って各ホスト上で起動されます。デフォルトは windows です

このオプションは、-host オプションと組み合わせて使用されます。例えば、次のコマンドラインは、host1 で a.exe を実行し、host2 で a.out を実行します:

```
> mpiexec -n 1 -host host1 -hostos windows a.exe :¥
          -n 1 -host host2 -hostos linux ./a.out
```

ジョブ・スケジューラーのサポート

インテル® MPI ライブラリーは、HPC 市場で一般的に利用されているジョブ・スケジューラーの大部分をサポートしています。Windows* では、次のジョブ・スケジューラーがサポートされます:

- **Microsoft* HPC Pack***

  > job submit /numprocessors:4 /stdout:test.out mpiexec -delegate test.exe

- **Altair* PBS Pro***

  > qsub -C "REM PBS" job

```
REM PBS -l nodes=4:ppn=2
REM PBS -l walltime=1:00:00
cd %PBS_O_WORKDIR%
mpiexec test.exe
```

インテル® Xeon Phi™ コプロセッサーの利用

クラスター環境でインテル® Xeon Phi™ コプロセッサーを利用するには、幾つかの選択肢があります:

1. Windows* / Linux* ホストに搭載された、コプロセッサーへホストからオフロードを行う MPI* アプリケーションを利用

2. Linux* ホストに搭載されたホストから MPI* ライブラリーを介してコプロセッサーのネイティブ MPI* アプリケーションを利用

3. Windows* ホストに搭載されたホストから MPI* ライブラリーを介してコプロセッサーのネイティブ MPI* アプリケーションを利用 (コプロセッサー上のネイティブ MPI* を開発するため、Linux* 版の MPI* ライブラリーが必要)

一般的なクラスターに関する考察

MPI を利用する上で考慮すべき一般的なクラスターに関す情報を説明します:

- 使用するノードの定義
- ヘテロジニアス・システムとジョブ
- ユーザー認証
- デバッグとテスト

参考資料

インテル® MPI ライブラリー関連日本語マニュアル:

- ゲッティング・スタート・ガイド (Windows | Linux)
- インテル® MPI ライブラリー・ユーザーズ・ガイド (Windows | Linux)
- インテル® MPI ライブラリー・リファレンス・マニュアル (Windows | Linux)

- MPI tuner チュートリアル (Windows | Linux)
- インテル® Trace Analyzer & Collector チュートリアル:
  MPI アプリケーションを解析する (OS 共通)

参照

iSUS のインテル® MPI ライブラリー製品サイト：

http://www.isus.jp/intel-mpi-library/

iSUS のインテル® Parallel Studio XE 製品サイト：

http://www.isus.jp/intel-parallel-studio-xe/

日本語マニュアルのダウンロード：

http://www.xlsoft.com/jp/products/intel/cluster/mpi/index.html

参考資料

- ゲッティング・スタート・ガイド 日本語版 (Windows | Linux)
- インテル® MPI ライブラリー・ユーザーズ・ガイド 日本語版 (Windows | Linux)
- インテル® MPI ライブラリー・リファレンス・マニュアル 日本語版 (Windows | Linux)

- MPI tuner チュートリアル 日本語版 (Windows | Linux)
- インテル® Trace Analyzer and Collector チュートリアル:
  MPI アプリケーションを解析する 日本語版 (OS 共通)

MPI と OpenMP* のハイブリッド化

```
printf("Rank %d Threads %d Running on %s¥n", my_id, omp_get_max_threads(), name);
#pragma omp parallel for reduction(+:sum)
for (i=my_id*my_steps; i<(my_id+1)*my_steps ; i++)
{
  double x = (i+0.5)*step;
  sum += 4.0/(1.0+x*x);
}
```

各ノードに分散された for ループを OpenMP のワークシェア機能を使用して
スレッドで並列実行

iSUS IA Software User Society