



チュートリアル: インテル® MPI ライブラリー向け MPI Tuner

バージョン 5.1 Update 3 (Windows*)

著作権と商標について

本資料は、明示されているか否かにかかわらず、また禁反言によるとよらずにかかわらず、いかなる知的財産権のライセンスも許諾するものではありません。

インテルは、明示されているか否かにかかわらず、いかなる保証もいたしません。ここにいう保証には、商品適格性、特定目的への適合性、知的財産権の非侵害性への保証、およびインテル製品の性能、取引、使用から生じるいかなる保証を含みますが、これらに限定されるものではありません。

本資料には、開発の設計段階にある製品についての情報が含まれています。この情報は予告なく変更されることがあります。最新の予測、スケジュール、仕様、ロードマップについては、インテルの担当者までお問い合わせください。

本資料で説明されている製品およびサービスには、不具合が含まれている可能性があり、公表されている仕様とは異なる動作をする場合があります。

MPEG-1、MPEG-2、MPEG-4、H.261、H.263、H.264、MP3、DV、VC-1、MJPEG、AC3、AAC、G.711、G.722、G.722.1、G.722.2、AMRWB、Extended AMRWB (AMRWB+)、G.167、G.168、G.169、G.723.1、G.726、G.728、G.729、G.729.1、GSM AMR、GSM FR は、ISO、IEC、ITU、ETSI、3GPP およびその他の機関によって制定されている国際規格です。これらの規格の実装、または規格が有効になっているプラットフォームの利用には、Intel Corporation を含む、さまざまな機関からのライセンスが必要になる場合があります。

性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサ用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。

Intel、インテル、Intel ロゴは、アメリカ合衆国および / またはその他の国における Intel Corporation の商標です。

* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

Microsoft、Windows、Windows ロゴは、米国 Microsoft Corporation の、米国およびその他の国における登録商標または商標です。

Java は、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。

Bluetooth は商標であり、インテルは権利者から許諾を得て使用しています。

インテルは、Palm, Inc. の許諾を得て Palm OS ready マークを使用しています。

OpenCL および OpenCL ロゴは、Apple Inc. の商標であり、Khronos の使用許諾を受けて使用しています。

© 2016 Intel Corporation.

最適化に関する注意事項

インテル® コンパイラーでは、インテル® マイクロプロセッサに限定されない最適化に関して、他社製マイクロプロセッサ用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサ依存の最適化は、インテル® マイクロプロセッサでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサ用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。

注意事項の改訂 #20110804

概要



インテル® MPI ライブラリー向けの MPI チューナーを使用して、ランタイム・ライブラリーの最適化構成ファイルを自動的に取得する方法を説明します。また、このチュートリアルを通して基本的なトラブルシューティングのヒントを得ることができます。

チュートリアルの概要	<p>このチュートリアルでは、開発者のクラスターとアプリケーション向けにインテル® MPI ライブラリーのパフォーマンスを最適化するいくつかの方法を紹介します。以下を含みます。</p> <ul style="list-style-type: none">• クラスターのチューニング時間を最小化する• クラスターのチューニング中、デフォルトのパラメーター・グリッドの欠損を含める• アプリケーションのチューニング中の最適な設定の構成• MPI tuner を利用する際によくある問題の解決方法
所要時間	15-20 分
目的	<p>このチュートリアルでは、以下のトピックについて説明します。</p> <ul style="list-style-type: none">• クラスターやアプリケーションの構成に関連するインテル® MPI ライブラリーの最適な設定を見つけるため、MPI Tuner を使用します。• MPI Tuner を利用する際によくある問題の解決方法を示します。
その他の情報	<p>インテル® MPI ライブラリー向けの MPI Tuner に関する詳しい情報については、次のリソースをご覧ください。</p> <p>製品 Web サイト</p> <p>インテル® MPI ライブラリーのサポート</p> <p>インテル® クラスターツール製品</p> <p>インテル® ソフトウェア開発製品</p>

必要条件

インテル® MPI ライブラリー向けの MPI Tuner を使用する前に、ライブラリー、スクリプトおよびユーティリティーがインストールされていることを確認してください。インストール方法の詳細は、『インテル® MPI ライブラリー for Windows* インストール・ガイド』(英語)をご覧ください。

ナビゲーションのクイックスタート

MPI Tuner を使用するには:

1. `mpitune` ユーティリティを使用して、最適化された設定ファイルを作成します。
2. 通常の実行に `mpiexec` コマンドの `-tune` オプションを追加して設定ファイルを使用します。

注:

MPI Tuner を使用する前に、実行するタスクをチェックできます。実際に実行する前に、`mpitune` の適用範囲を確認するには、`--scheduler-only (-so)` オプションを使用します。

```
> mpitune ... -so
```

MPI Tuner を起動する

MPI Tuner を使用するには次のコマンドを起動します。

```
> mpitune
```

MPI Tuner コマンド

MPI Tuner ユーティリティは、次の 2 つのモードで動作します。

- クラスター固有。インテル® MPI ライブラリーの最適な設定を見つけるため、インテル® MPI Benchmarks やユーザーから提供されるベンチマーク・プログラムを使用して、クラスター環境を評価します。このモードがデフォルトで使用されます。
- アプリケーション固有。特定のアプリケーション向けにインテル® MPI ライブラリーの最適な設定を見つけるため、MPI アプリケーションの性能を評価します。

クラスター固有のチューニング

クラスター固有モードで MPI Tuner を使用するには次の操作を行います。

1. 次のコマンドを使用して、デフォルトの `<installdir>\<arch>\etc` ディレクトリーにチューニングされた設定ファイルを作成します。

```
> mpitune -hf <hostfile>
```

もしくは、`-odr` オプションを使用して、チューニングされた設定ファイルを作成するディレクトリーを指定します。

```
> mpitune -hf <hostfile> -odr <path_to_result_directory>
```

2. ディレクトリーを指定して、あるいはデフォルト・ディレクトリーから引数ファイルを読み込む場合は引数を指定せず、`-tune` オプションを使用します。次に例を示します。

```
> mpiexec -tune -ppn 8 -n 128 .\my_app
```

```
> mpiexec -tune <path_to_result_directory> -ppn 8 -n 128 .\my_app
```

アプリケーション固有のチューニング

アプリケーション固有モードで MPI Tuner を使用するには次の操作を行います。

1. 提供される環境とコマンド設定で指定するワークロードをチューニングするには、`--application (-a)` オプションを使用します。MPI Tuner は、新しい最適な設定を `myprog.conf` ファイルに生成します。

```
> mpitune --application \"mpiexec -n 32 .\myprog\" -of .\myprog.conf
```
2. アプリケーションの実行時に生成された最適な設定を読み込むには、`-tune` オプションを使用します。

```
> mpiexec -tune .\myprog.conf -n 32 .\myprog
```

タスク 1: クラスター固有モードでのチューニング時間を最小化する

クラスターのチューニング時間を減らすには、クラスターで最も頻繁かつ広範囲に使用されている MPI ワークロードについて考慮する必要があります。それらが以下に関してどのように実行されているかメモしておきます。

- 使用されるホスト数の範囲
- ホストごとのランク数
- ファブリックの使い方 (`I_MPI_FABRICS`)
- 典型的なメッセージサイズ
- 最も一般的な MPI 関数

ホストの範囲

例えば、大部分のワークロードがクラスター 4 から 16 の間のホスト数を使用する場合、`-hr <n:m>` オプションを使用してそれらの上限と下限を設定します。

```
> mpitune ...-hr 4:16
```

`mpitune` ユーティリティーは、4 から 16 の間の 2 の累乗となるすべてのホスト範囲を構築します。ここでは、4 つのホスト、8 つのホスト、そして 16 のホスト向けの設定を生成します。

ホストごとのランク数

ホストごとのランク数を設定するには、`-pr <n:m>` オプションを使用します。

```
> mpitune ...-pr 1:16
```

同様に、`mpitune` ユーティリティーは、1 から 16 の間の 2 の累乗となるランク範囲を構築します。例えば、ランク番号 1、2、4、8、16 のすべてのケース向けにチューニングされた設定を作成します。

```
> mpitune ...-pr 24:24
```

上限と下限が同じである場合、`mpitune` ユーティリティーは、`ppn=24` のみをチューニングします。

ファブリックの使い方 (I_MPI_FABRICS)

チューニングで使用するファブリックを指定するには、`-fl` オプションを使用します。

```
> mpitune ...-fl shm:dapl,dapl,shm:ofa,ofa
```

`mpitune` ユーティリティーは、列挙されたファブリックのみを使用します。

メッセージサイズ

チューニングするメッセージの対象サイズ範囲を指定するには、`-mr` オプションを使用します

```
> mpitune ...-mr 16:2097152
```

この場合、`mpitune` ユーティリティーは、指定された `16` から `2097152` バイト間の 2 の累乗のメッセージサイズの MPI 操作をチューニングします。

最も一般的な MPI 関数

MPI 関数の利用とパフォーマンスに関する統計情報がある場合、要求に応じてチューニング範囲を調整することができます。チューニングを開始する前に、『[インテル® MPI ライブラリー for Windows* ユーザーズガイド](#)』の MPI 関数の各種チューニング・オプションについてご覧ください。

最も広く使用される MPI ルーチンから始めて、より複雑な関数へと作業を進めることを推奨します。例えば、集合操作のチューニングの前に p2p のチューニングを行います。

最初に p2p に深く関連するオプションを指定します。

1. `option_set` 変数で、最も一般的な MPI 関数を集めます。

```
> set
option_set=I_MPI_RDMA_TRANSLATION_CACHE,I_MPI_DAPL_RNDV_BUFFER_ALI
GNMENT,I_MPI_SHM_FBOX_SIZE,I_MPI_SHM_CELL_SIZE,I_MPI_SSHM_BUFFER_S
IZE,I_MPI_EAGER_THRESHOLD,I_MPI_DAPL_BUFFER_SIZE,I_MPI_INTRANODE_E
AGER_THRESHOLD,I_MPI_DAPL_DIRECT_COPY_THRESHOLD
```

2. `option_set` でチューニング・セッションを実行します。ここでは、上記の環境変数に指定された値を基に、最適なインテル® MPI ライブラリー・クラスターの一連の設定を作成します。

```
> mpitune ...-os %option_set%
```

3. 次の集合操作をチューニングします。

```
> mpitune ...--collective-only
```

完了したら、2つの設定ファイルを1つにマージし、実行オプションに `-tune` または `-config` を指定してそのファイルを使用します。

注:

さらにチューニング時間を短縮するため、必要に応じて改善率を指定するか、許容可能なパフォーマンスを示すオプションを除外することができます。

タスク 2: クラスターのチューニング中、デフォルトのパラメーター・グリッドの欠損を含める

`mpitune` ユーティリティーは、2 の累乗であるほとんどの変数の値を列挙してチューニングするだけです。アプリケーションが非定型レイアウトやデータサイズを使用することが事前に判明している場合、カスタマイズしたセットで `mpitune` を実行しデフォルトを上書きすることができます。

`<installdir>\<arch>\etc` ディレクトリーに書き込み権限があることを確認してください。`mpitune` ユーティリティーは、自己構成のため `<installdir>\<arch>\etc` にある `*.xml` ファイルを使用します。クラスター固有モード向けに何がどのように実装されているかを示す 2 つのファイル `options.xml` と `Benchmarks\imb.xml` があります。

例えば、`I_MPI_EAGER_THRESHOLD` 環境変数のチューニングをカスタマイズしたい場合、変更方法については次のハイライトされたテキストを参照してください。

`options.xml`:

```
...
  <option name="I_MPI_EAGER_THRESHOLD" type="global" group="collective"
weight="1.0">
  <actions>
    <step order="1" storage="first">
      <additive>
        <env name="I_MPI_FALLBACK_DEVICE" type="global"
value="disable" />
      </additive>
      <range
name="range_vars">int_range(8192:524288:*:2)</range> <!-- explicit range
from 8k to 512k with power of 2 -->
        <format>@range_vars()</format>
        <result format="[msg_size]" limit="1" separator="" />
      </step>
    </actions>
    <requirements>
      <param name="hosts" value="2:2" /> <!-- use 2 hosts -->
      <param name="perhost" value="1:1" /> <!-- with 1 process on
host -->
      <param name="processes" value="2:2" /> <!-- and 2 processes
total -->
      <param name="devices" value="shm:dapl,shm:tmi" /> <!-- for
shm:dapl and shm tmi fabrics (I_MPI_FABRICS) -->
    </requirements>
    <result <!-- internal format description -->
      format="#first#"
      quotes="no"
      quotesInline="no"
    />
  </option>
...
Benchmarks/imb.xml:
  <test title="IMB Sendrecv" weight="1.0">
```

```

        <description>Sendrecv test from IMB benchmark for OUTPUT
mode</description>
        <executable>"IMB-MPI1" -npmin %proc% -iter 5 -msglen
@msglen_file() Sendrecv</executable>
        <function
title="msglen_file">range_file(768:1536+:256;"value[endl]")</function>
<!-- msg len file of IMB with range: 768, 1024, 1280 and 1536 bytes -->
        <launch_line>%mpiexec% %globals% %locals% %executable%</launch_line>
        <requirements> <!-- values for requirements section are calculated as
intersection with the same block from options.xml file. Results are in
the mpitune schedule -->
                <param name="hosts" value="1:-1" />
                <param name="perhost" value="1:-1" />
                <param name="processes" value="2:-1" />
                <param name="devices"
value="rdssm,rdma,shm,ssm,sock,shm:dapl,shm:tcp,dapl,tcp,shm,shm:ofa,shm:
tmi,ofa,tmi" />
        </requirements>
        <options_filter filter="exclusive"> <!-- this section enumerates
options to tune by this benchmark-->
                <option type="global" name="I_MPI_EAGER_THRESHOLD" />
                <option type="global" name="I_MPI_INTRANODE_EAGER_THRESHOLD" />
</options_filter>
<result <!-- format to parse benchmark output -->
        source="thtime"
        paramGroup="4"
        paramTitle="t[usec]"
        paramTarget="min"
        paramLeftMarginGroup="2"
        paramRightMarginGroup="3"
        paramChooseMode="heaviest"
        paramDiffDelta="0.001"
        msgGroup="0"
        msgTitle="Bytes"
        iterationCompare="min"
        startline=".*(\#bytes\s+\#repetitions).*"
        dataline="\s+(\d+)\s+(\d+)\s+([\d\.]+)\s+([\d\.]+)\s+([\d\.]+)"
        solidatalines="1"
/>
</test>
...

```

タスク 3: アプリケーション固有のチューニング

クラスター固有のチューニングが完了すると、アプリケーション向けにインテル® MPI ライブラリーを最適化することに集中できます。前述のクラスター固有のチューニング手法に、次のような変更を加えてアプリケーション固有のチューニングに適用できます。

1. `hosts` やホストごとのプロセス数などのアプリケーション固有の設定は、`mpitune` ではなく `mpiexec` のコマンドラインで指定されます。
2. 前述のマイクロベンチマークの代わりにアプリケーションを使用します。
3. `imb.xml` の代わりに、`app.xml` 設定ファイルを適用します。

トラブルシューティング

ここでは、MPI Tuner を実行する際に見られる一般的な問題の解決方法について説明します。

問題	原因と解決法
mpitune のスケジューラーが空である場合	<ol style="list-style-type: none">1. <code>mpitune</code> の引数が相互に矛盾しないことを確認してください。例えば、<code>--options-set</code> と <code>--options-exclude</code> がオーバーラップしないなど。2. ファブリックやデバイスのパスがチェックされない場合、同じ設定で MPI テスト・アプリケーションを実行してみてください。問題は、誤った <code>hostfile</code> や不適切なクラスター設定による可能性があります。
mpitune の実行時間が非常に長い場合	<ol style="list-style-type: none">1. <code>-so</code> オプションを使用して、実際に実行する前にスケジュールのみの確認を行ってください。 <p>タスク 1 とタスク 2 で説明した方法と (もしくは) それらを組み合わせることで不要なジョブをスキップします。</p>