Windows* 環境での インテル® MPI ライブラリー向けの パフォーマンス解析とチューニング およびデバッグ

2016 年 4 月

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

内容

- デバッグとテスト**
- チューニング
  - MPI tuner (インテル® MPI ライブラリー**)

- パフォーマンス解析
  - インテル® MPI ライブラリー**
    - 統計収集
  - インテル® Trace Analyzer & Collector***
    - アプリケーションのトレース
    - Performance Snapshot
  - インテル® VTune™ Amplifier XE**

**:インテル® Parallel Studio XE Cluster Edition に含まれる、またはスタンドアロン版も入手可能
***: インテル® Parallel Studio XE Cluster Edition が必要

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

MPI アプリケーションのデバッグとテスト

- インテル® MPI ライブラリーは、Microsoft* Visual Studio* デバッガーをサポートします

  Microsoft* MPI Windows* デバッガー拡張 (MS-MPI v6 WinDbg 拡張以降) で利用可能なメッセージ・キュー・デバッガー機能をサポートしています。この機能は、Windows* プラットフォーム上の任意の MPI ツールやデバッガーが、標準化されたメッセージ・キュー・インターフェイスを使用してインテル® MPI ライブラリーと相互運用することを可能にします

- インテル® MPI ライブラリーはデバッグ機能をサポートします
  - ログ (デバッグ情報) の取得
  - アプリケーションのトレース
  - 正当性チェック

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

インテル® MPI ライブラリーのログ取得機能

アプリケーションのデバッグよりも、ログを利用することが有効なことがあります。環境変数 **I_MPI_DEBUG** は、実行中の MPI アプリケーションの情報を取得する非常に便利な機能を提供します:

```
> mpiexec.exe -genv I_MPI_DEBUG 5 -n 8 ./my_application
```

注意

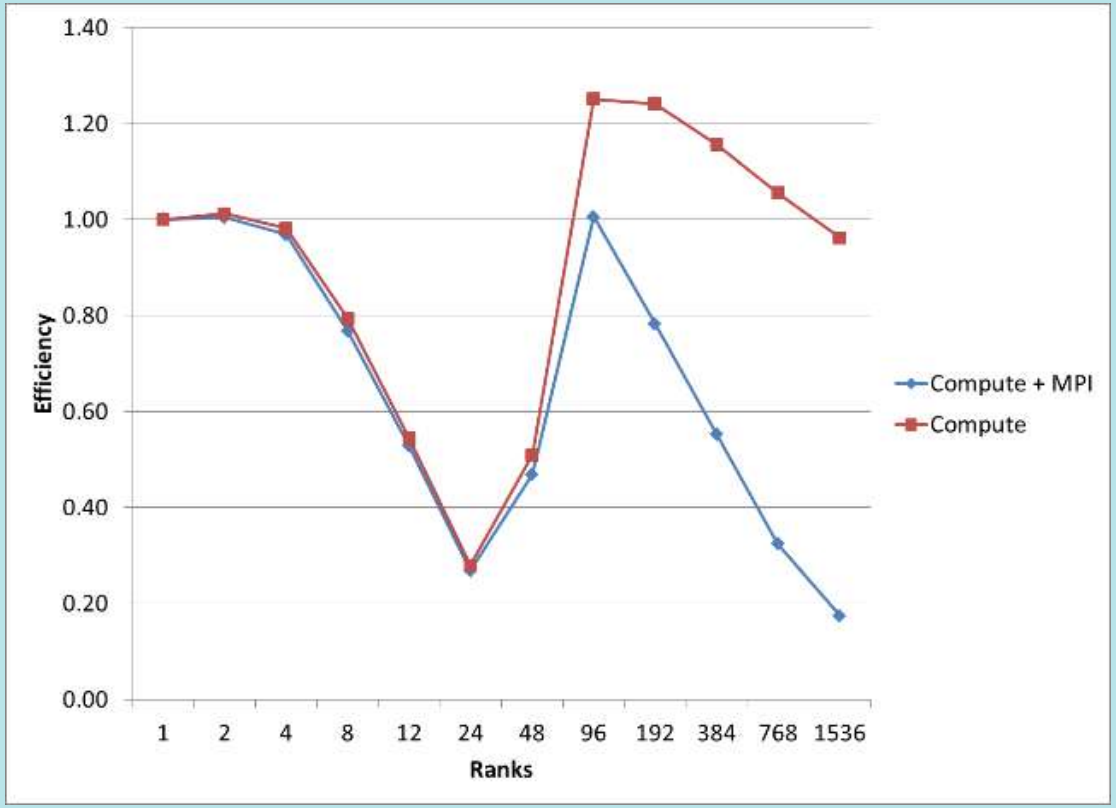
I_MPI_DEBUG に大きな値を設定すると、多くの情報を取得できますが、アプリケーションの実行パフォーマンスを大幅に低下させます。始めに一般的なエラーを検出するには、I_MPI_DEBUG=5 で開始することを推奨します。I_MPI_DEBUG に関する詳しい情報は、『インテル® MPI ライブラリー for Windows* リファレンス・マニュアル』をご覧ください

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

ステップ 1 - クラスターレベルの解析
インテル® MPI ライブラリー

アプリケーションの計算時間の並列効率を個別に計算し描画

- 並列効率: E[p] = S[p]/p
- 48 コア (2 ノード) までの MPI 時間はごくわずか
- 96 ランク (4 ノード) を超えると計算パフォーマンスが超線形 (スーパーリニア) にスケーリング
- ほぼ同じ辺り (96 ランク付近) で MPI 時間が低効率の最大の原因となる



© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

ステップ 2 - ランタイム解析

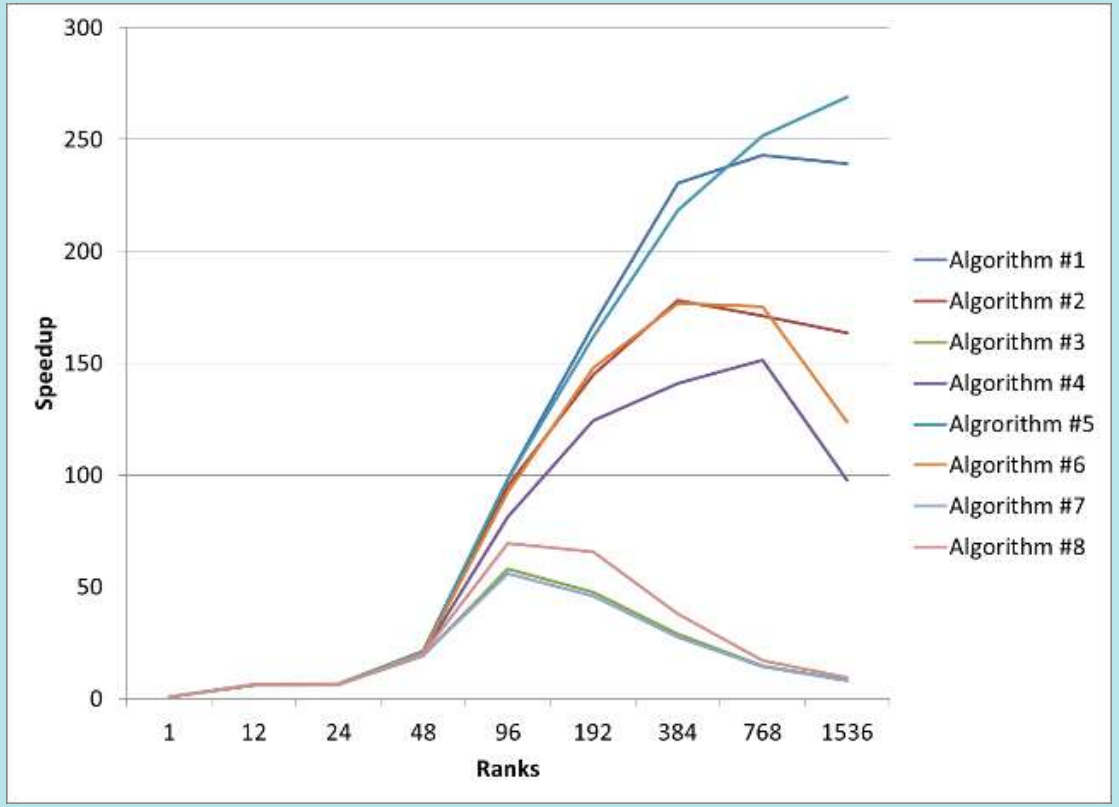
- ソースコードを変更せずに MPI パフォーマンスを向上することが可能
- インテル® MPI ライブラリーの環境変数を使用するか、計算ノードへのランクのプロセス割り当てを変更する
- ノードへのプロセス割り当てを変更するには、マシンや設定ファイルで行うか、またはコミュニケーター内でランクを並べ替える
- 1 つの方法はグローバル操作に絞ってチューニングを開始
- 環境変数 I_MPI_DEBUG = 5 に設定
  - 使用されている変数、ネットワーク・ファブリック、プロセスの割り当てに関する情報を出力
  - I_MPI_DEBUG を 6 に設定すると、集合アルゴリズムのデフォルトのアルゴリズムも出力される

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

ステップ 2 – ランタイム解析

- 『インテル® MPI ライブラリー・リファレンス・マニュアル』に **MPI_Allreduce** の **8** つの異なるアルゴリズムが記載されている

- 環境変数 **I_MPI_ADJUST_ALLREDUCE** を 1 ～ 8 の整数値に設定することで簡単にアルゴリズムの変更が可能



© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

インテル® MPI ライブラリーのチューニング
MPI tuner

インテル® MPI ライブラリーには、特定のクラスターやアプリケーション向けの最適な設定が含まれる、設定ファイルを作成する mpitune ユーティリティー (MPI tuner) が含まれています

クラスター固有モード:

- 利用するクラスター構成で一度だけ実行します。クラスター環境に最適化された設定ファイルが生成されます

アプリケーション固有モード:

- 利用するアプリケーション毎に実行します。アプリケーションに最適化された設定ファイルが生成されます

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

なぜ自動チューニング?

- インテル® MPI ライブラリーは、幅広い設定オプションで利用できます - さまざまな環境変数が用意されています
  - 集合操作のための異なるアルゴリズム
  - 異なるファブリックのサポート
  - ライブラリーの振る舞いによる異なるしきい値 (rendezvous 、キャッシュバイパスなど...)
  - 異なるバッファーサイズ
  - …
- アプリケーションではなく、ライブラリー自身をチューニング
- しかし、ライブラリーには、すでに適切なアウトオブボックス設定があります ...
- MPITune は、自動的に空間を探索できるツールです

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

mpitune によるクラスター固有のチューニング

特定のクラスターやアプリケーション環境上で、ライブラリー・チューニングの最適値を探します

- インストール後に一度、そしてクラスターの設定を変更した後に実行します
- 最良の構成は、通信デバイス、ノード数、MPI ランク、そしてプロセスの分散モデルの組み合わせごとに記録されます
- 設定はインテル® MPI ライブラリー・フォルダーに保存され、すべてのユーザーが利用できます

```
# 設定値を収集:                    # 収集された値を再利用:
$ mpitune [options]                $ mpirun –tune ./application
```


© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

mpitune によるアプリケーション固有のチューニング

特定のクラスターやアプリケーション環境上で、ライブラリー・チューニングの最適値を探します

- アプリケーションごとに、そしてアプリケーションもしくはクラスターの構成を変更した後に実行します
- 最良の構成は、通信デバイス、ノード数、MPI ランク、そしてプロセスの分散モデルの組み合わせごとに記録されます
- 設定はユーザーのホームに保存されます

```
# 設定値を収集:
$ mpitune [options]  --application ¥”mpirun application¥”

# 収集された値を再利用:
$ mpiexec –tune ./app.conf ./application
```


© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

mpitune パラメーター

パラメーター	効果
--application | -a ¥"<app_ cmd_line>¥“	アプリケーション固有モード
--fabric-list | -fl　[f1[,f2...[,fN]]]	使用するファブリックの選択
--host-range | -hr {min:max|min:|:max}	テストに使用するホストの範囲を設定
--perhost-range| -pr {min:max|min:|:max}	ホストごとのプロセッサー数の範囲を設定
--application-regexp | -ar　¥"reg-expr¥"	パフォーマンス・インデックスとしてアプリケーションの出力値を使用
--application-value | -avd {min|max}	値の最適化の方向性を指定
--session-file | -sf [file-path]	チューニング・プロセスを継続 ...
--collectives-only | -co	集合操作のみチューニング
--valuable-improvement | -vi {percent}	最小の成功基準を定義
--iterations | -i <count>	各チューニング過程での実行回数を定義
--options-set | -os <opt1,...,optN>	要求したオプションのみをチューニング ...

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

mpitune ランタイムの削減 - スケジュールを理解

- mpitune の実行には長い時間がかかることがあります
- アプリケーションだけをチューニングする固有の設定があります
- プロセスを実際に起動することなくスケジュールをチェックできます

$ mpitune --scheduler-only | -so …

```
30'Jul'14 08:31:40     | Checking of ability to tune required options has been completed.
30'Jul'14 08:31:40     | Set monopoly access for options to be tuned...
30'Jul'14 08:31:40     | Complete.
30'Jul'14 08:31:40     | Current schedule:
-------------------------------------------------------------------------------
 # |                    Option                  |           Test          | S |
-------------------------------------------------------------------------------
  1|I_MPI_RDMA_SCALABLE_PROGRESS                |...the best range options| - |
  2|I_MPI_RDMA_TRANSLATION_CACHE                |...the best range options| - |
  3|I_MPI_WAIT_MODE                             |...the best range options| - |
  4|I_MPI_EAGER_THRESHOLD                       |Tuning thresholds        | - |
  5|I_MPI_INTRANODE_EAGER_THRESHOLD             |Tuning thresholds        | - |
  6|I_MPI_RDMA_EAGER_THRESHOLD                  |Tuning thresholds        | - |
  7|I_MPI_DYNAMIC_CONNECTION                    |...the best range options| - |
  8|I_MPI_ADJUST_ALLREDUCE                      |... I_MPI_ADJUST_* family| - |
-------------------------------------------------------------------------------
```


© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

MPI tuner の使い方例

クラスター固有モード:

> mpitune

> mpitune -hf <hostfile>

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

> 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モード:

> mpitune --application ¥"mpiexec -n 32 .¥myprog¥" -of .¥myprog.conf

> mpiexec -tune .¥myprog.conf -n 32 .¥myprog

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

インテル® MPI ライブラリー統計収集

インテル® MPI ライブラリーは、アプリケーションの実行を妨げることなく、パフォーマンス・データを収集する組込み統計収集機能を持っています。収集された情報はテキストファイルに書き込まれます

環境変数 **I_MPI_STATS** で統計収集を制御します

モード	説明
ネイティブ統計形式	全ての MPI 関数の情報と通信およびデータ転送
IPM 統計形式	統合パフォーマンス・モニタリング (IPM) サマリー形式 (ウォールクロック、システム、ユーザー、MPI 時間と主要な MPI 関数の情報)

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

インテル® MPI ライブラリー統計収集

インテル® MPI ライブラリーは、アプリケーションの実行を妨げることなく、パフォーマンス・データを収集する組込み統計収集機能を持っています。収集された情報はテキストファイルに書き込まれます

環境変数 **I_MPI_STATS** で統計収集を制御します

I_MPI_STATS=[native:][n-] m

n, m	出力情報の統計レベル。
1	各プロセスが送信したデータ量を出力
2	セル数と転送されたデータ量を出力
3	統計出力は、実際の引数に応じて組み合わせられます
4	バケットリストで定義された統計出力
10	すべての通信コンテキストの集団操作の統計出力
20	すべての MPI 関数の時間情報を追加して出力

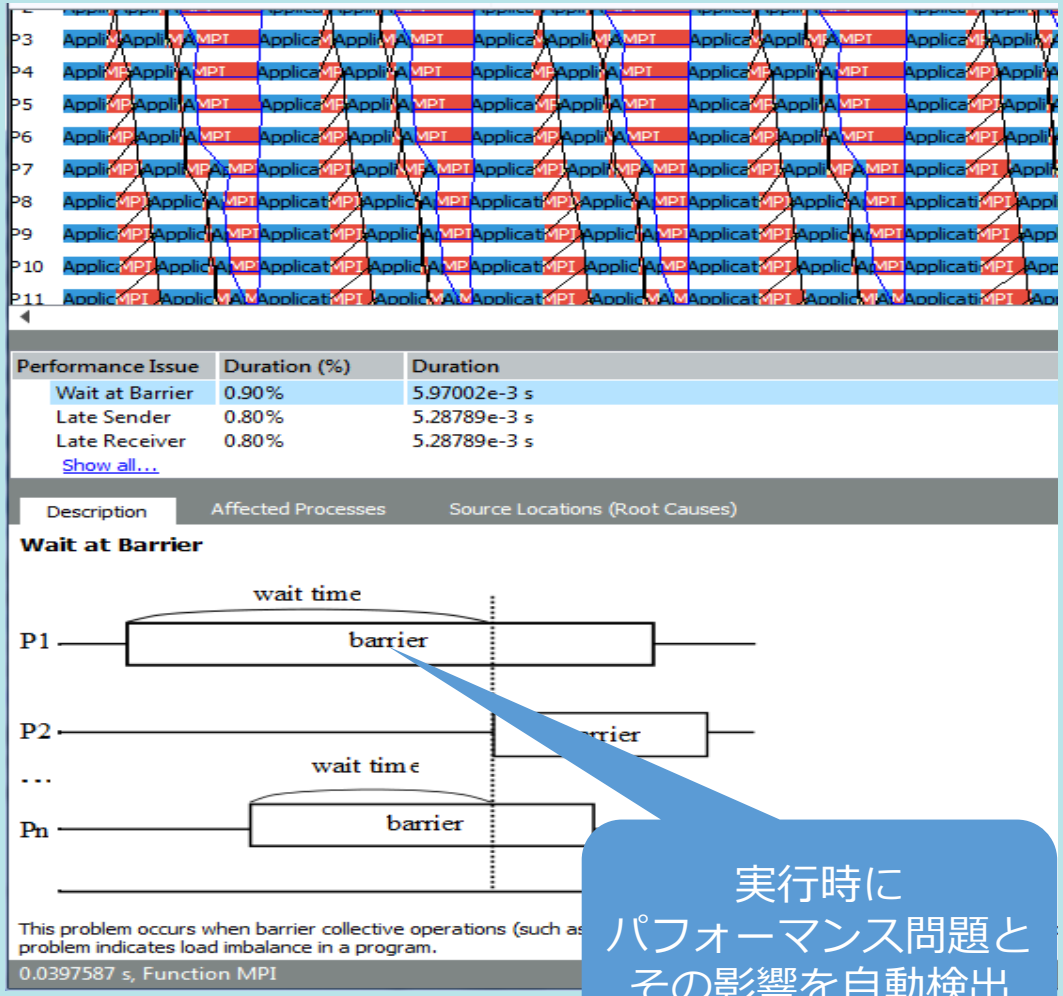
n,m は正の整数値です。出力する情報の範囲を定義します。レベル n から レベル m の統計 (n と m を含む) が出力されます。n が指定されない場合、デフォルト値は 1 です

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

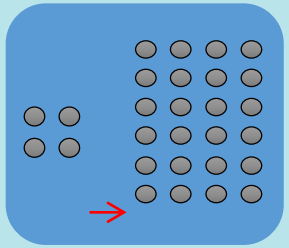
インテル® Trace Analyzer & Collector

- 開発者を支援
  - 並列アプリケーションの動作を視覚化して確認
  - プロファイル統計とロードバランスを評価
  - 通信 hotspot を特定
- 機能
  - イベントベースのアプローチ
  - 低オーバーヘッド
  - 優れたスケーラビリティー
  - 強力な集合およびフィルター関数
  - イデアライザー (理想化)
  - パフォーマンス・スナップショット

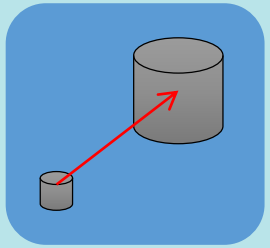


© 2016 iSUS. 無断での引用、転載を禁じます。* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

MPI とハイブリッド・クラスターのスケーラブルなプロファイル
MPI Performance Snapshot

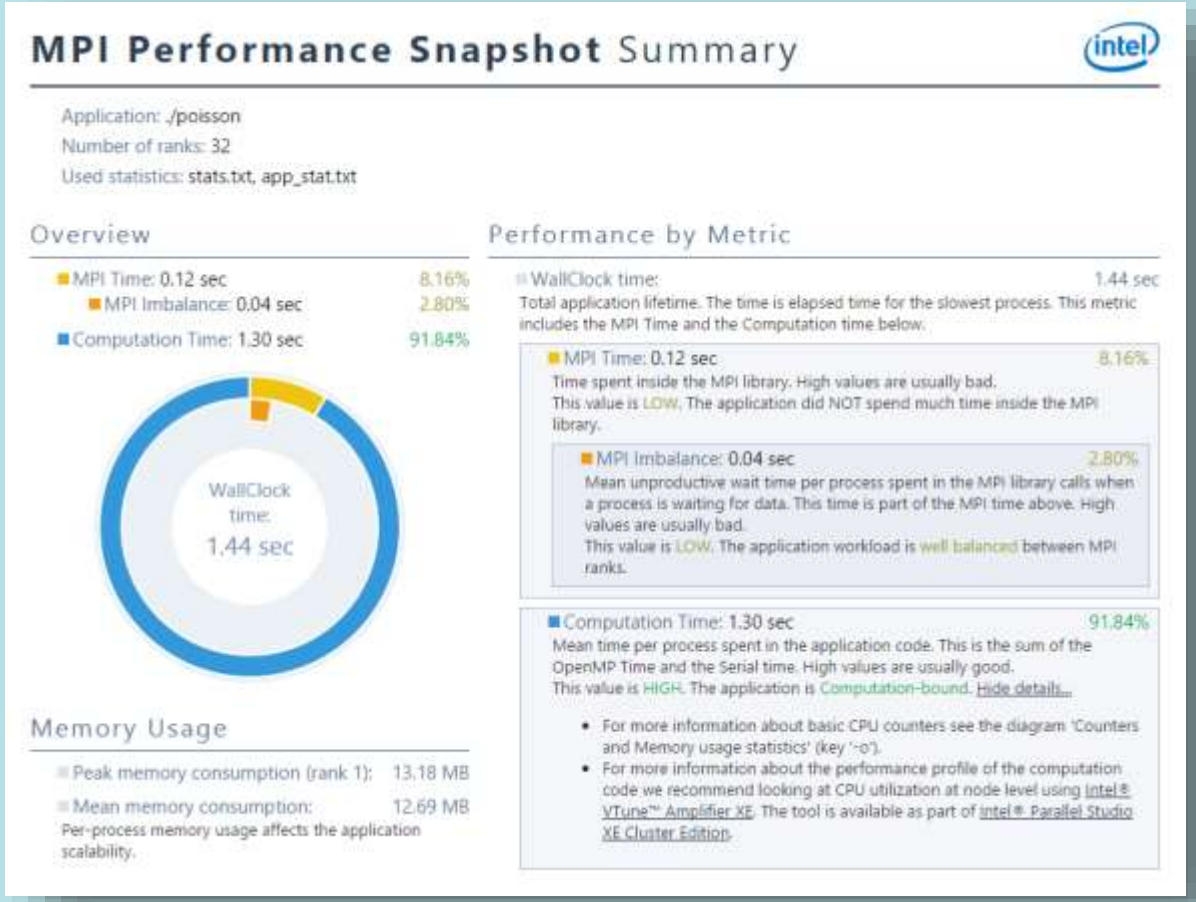


軽量 – 低オーバーヘッド、最大 32K ランクをプロファイル



スケーラビリティー - スケーリングによるパフォーマンスの変化を迅速に検出

主要メトリック – PAPI カウンター、MPI/OpenMP* のインバランスを表示



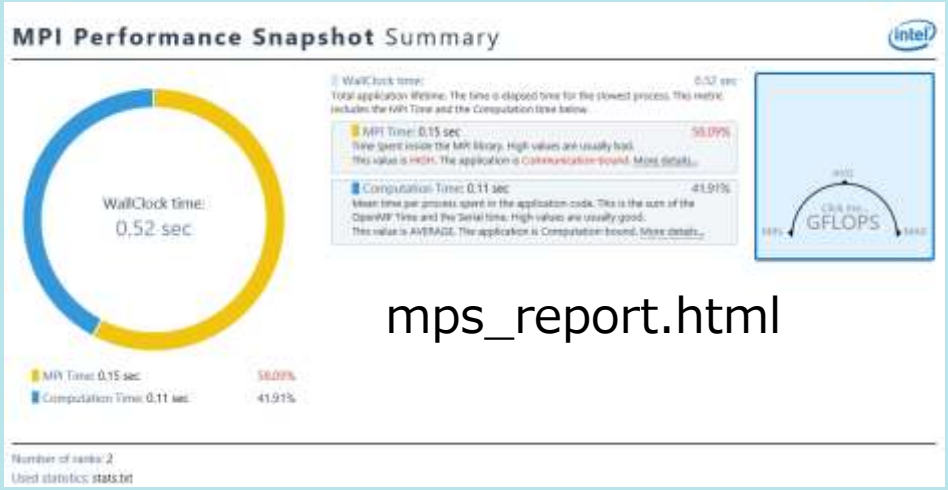
実行時のパフォーマンス・スナップショットの取得は、Linux* 環境でのみサポートされます

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

Performance snapshot の取得

Windows* 環境でパフォーマンス・スナップショット・レポートを取得する手順：

1. I_MPI_STATS=20 を設定
2. mpiexec コマンドで MPI プログラムを実行
3. mps –g stats.txt を実行



mps_report.html

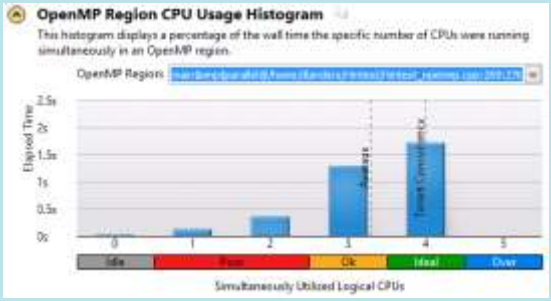
実際のクラスターでの取得例:

> mpiexec –machine mfile2 –genv I_MPI_STATS=20 pi_mpiomp

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

インテル® VTune™ Amplifier XE

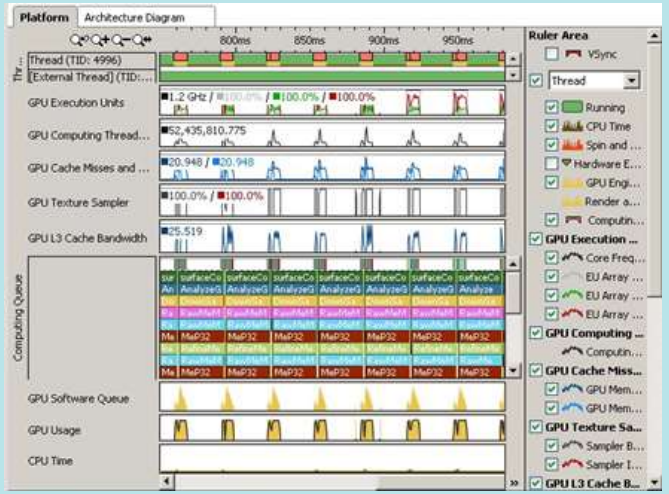
- OpenMP* のスケーラビリティーを迅速にチューニング
- 使いやすい MPI ハイブリッド解析
- 簡単な OpenCL*/GPU 解析
- 優れた帯域幅解析
- 素早く、簡単にインストールして使用可能
- VM サポート
- 最新のプロセッサーと OS



Top OpenMP Processes by MPI Communication Spin Time

This section lists process with the lowest MPI communication spin time.

Process	PID	MPI Communication Spinning	(%)	OpenMP Potential Gain	(%)	Serial Time	(%)
bt-mz.B.4	125904	0.020s	0.2%	3.392s	31.2%	0.251s	2.3%
bt-mz.B.4	125902	0.040s	0.4%	3.431s	31.6%	0.291s	2.7%
bt-mz.B.4	125905	0.321s	3.0%	3.025s	27.9%	0.659s	6.1%
bt-mz.B.4	125903	0.441s	4.1%	3.147s	29.0%	0.608s	5.6%



© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

OpenMP* の効率とスケーラビリティーをチューニング

インテル® VTune™ Amplifier XE で必要なデータを素早く取得

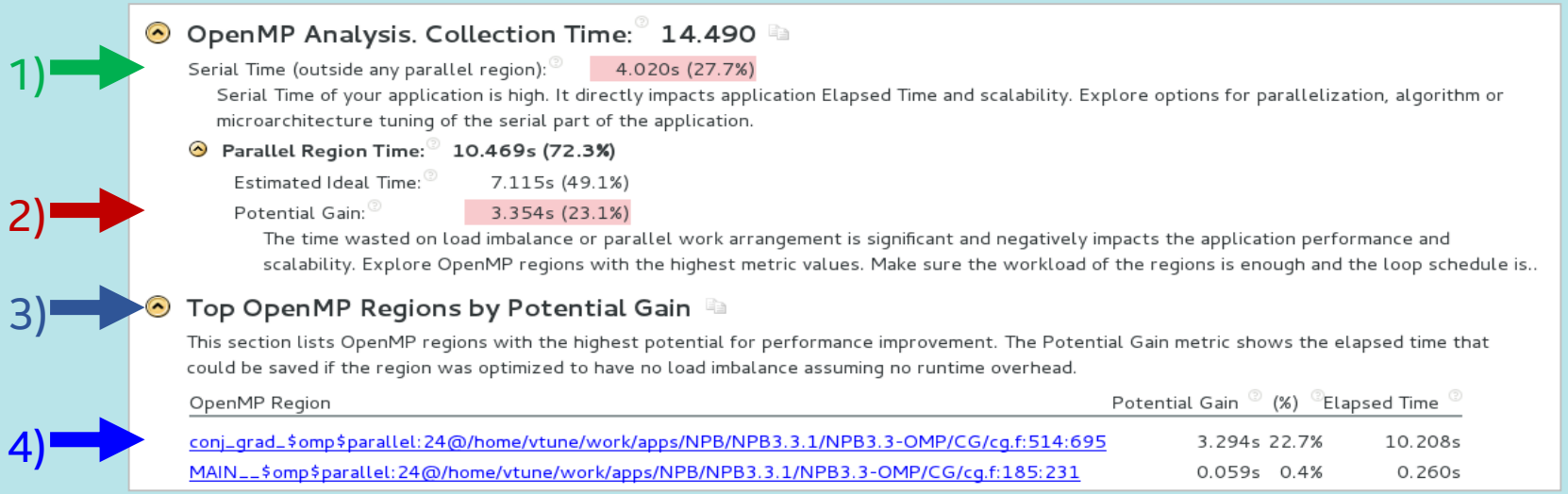
必要なデータ:

1) アプリケーションのシリアル時間はスケーリングに影響するほど長いか?

2) OpenMP* をチューニングすることでどの程度パフォーマンスが向上するか?

3) チューニングにより最も大きな利点が得られるのはどの OpenMP* 領域/ループ/バリアか?

4) 各領域の効率が悪い原因は? (リンクをクリックすると詳細が表示される)



インテル® VTune™ Amplifier XE のサマリーレポート:

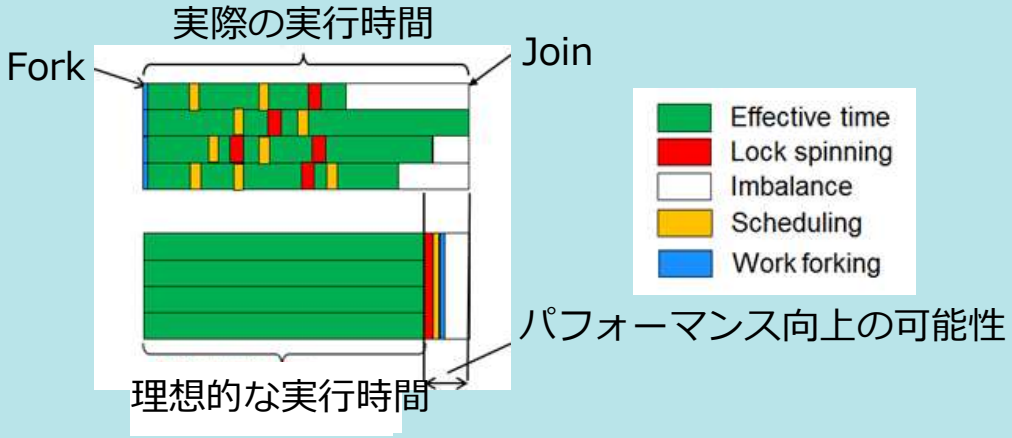
© 2016 iSUS. 無断での引用、転載を禁じます。* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

OpenMP* の効率とスケーラビリティーをチューニング

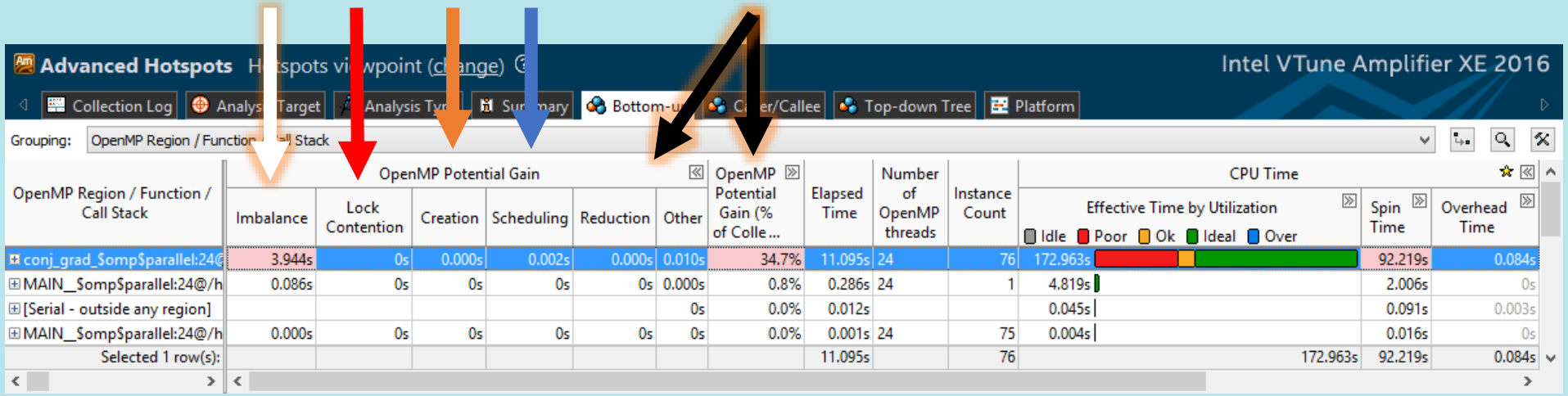
効率が悪い部分のウォールクロック時間を確認し原因を特定

重要な点に注目

- 効率が悪いのはどの領域か?
- チューニングにより十分なパフォーマンスの向上が得られるか?
- 効率が悪い原因は？インバランス？スケジュール？ロックスピン？
- インテル® Xeon Phi™ コプロセッサー搭載システムをサポート



インバランス　ロック　Fork　スケジュール　　パフォーマンス向上の可能性



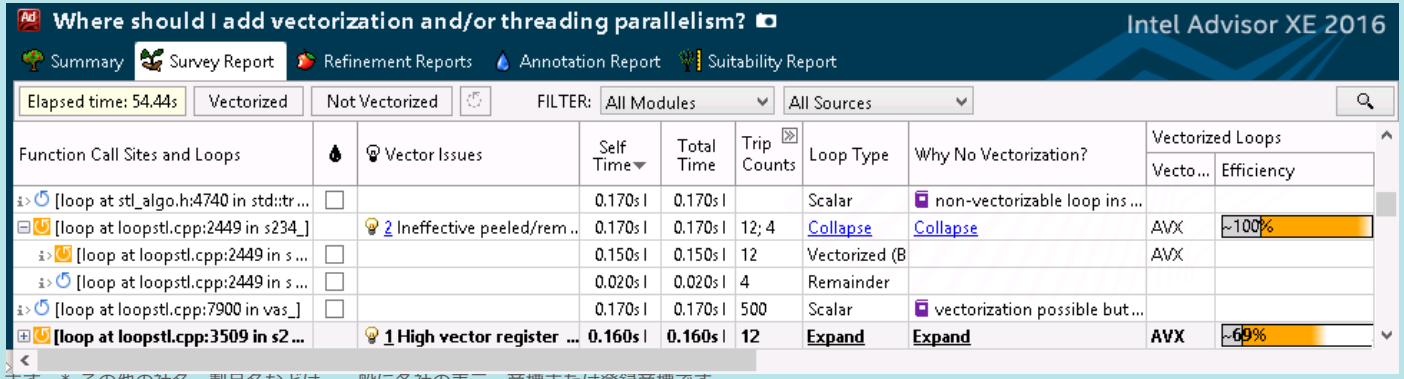
© 2016 iSUS. 無断での引用、転載を禁じます。* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

新機能: インテル® Advisor XE
ベクトル化による最適化

- 問題:
  - インテル® AVX2 向けに再コンパイルしてもパフォーマンスがそれほど向上しない
  - どこをベクトル化すべきか?
  - 新しいアーキテクチャー向けに組込み関数のコードを変更したい
  - コンパイラー・レポートの見方が良く分からない

New!

- データに基づくベクトル化:
  - どのベクトル化により最も大きな利点が得られるか?
  - ベクトル化を妨げているものは? その原因は?
  - ループがベクトル化に適しているか?
  - データを再構成することでパフォーマンスが向上するか?
  - #pragma simd だけで大丈夫か?



© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

設計して実装

インテル® Advisor XE - スレッド・プロトタイプの生成

- **並列化の設計**
  - 通常の開発作業に影響なし
  - すべてのテストケースを継続して使用可能
  - 実装する前にチューニングしてデバッグ

並列化の実装

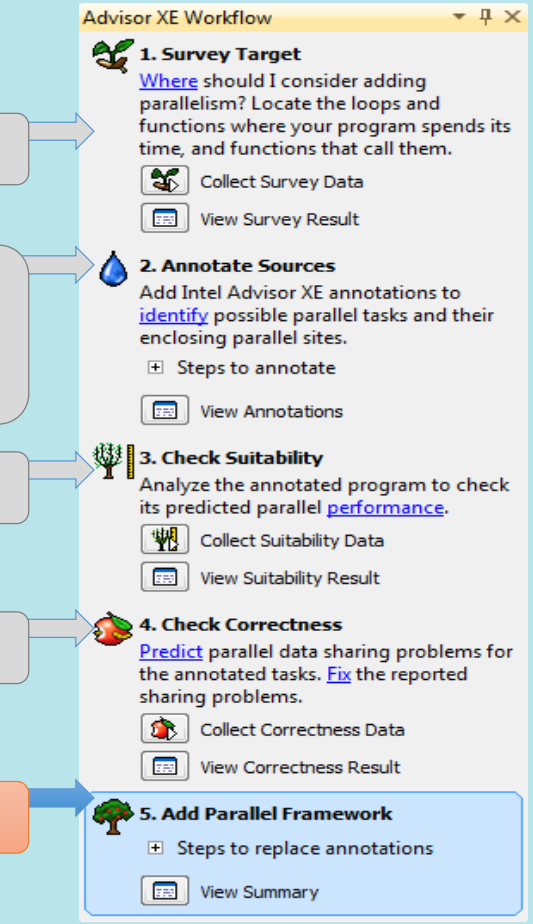
1) 解析

2) 設計
(コンパイラーはこれらのアノテーションを無視)

3) チューニング

4) 検証

5) 実装!



より少ない労力とリスクで、より大きな効果

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

インテル® Advisor XE

MPI アプリケーションのヒントを得る

mpiexec コマンドからインテル® Advisor XE を起動して、各ノード上でインテル® Advisor XE で解析を行う:

```
> mpiexec -n 10 advixe-cl -collect survey --project-dir my_proj your_app
```

インテル® Advisor XE を使用した、MPI アプリケーションの解析はコマンドラインのみがサポートされます

- クラスターノード上にインテル® Advisor XE が無い場合、ホストシステムへ結果をコピーして解析します
- クラスターノードでは、コマンドラインでレポートを取得できます
- 異なるランクの結果を統合しないで、個別に解析します

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

まとめ

インテル® MPI ライブラリー、インテル® Trace Analuzer and Collector およびインテル® VTune™ Amplifier XE を活用して、MPI アプリケーションを;

- 効率よくデバッグ
- 自動的にクラスター環境をチューニング
- 最適化のヒントを取得

- 通信と計算を解析
- パフォーマンスのボトルネックを特定

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。

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)

© 2016 iSUS. 無断での引用、転載を禁じます。* 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。