intel Software

# インテル® ソフトウェア開発製品によるソースコードの近代化

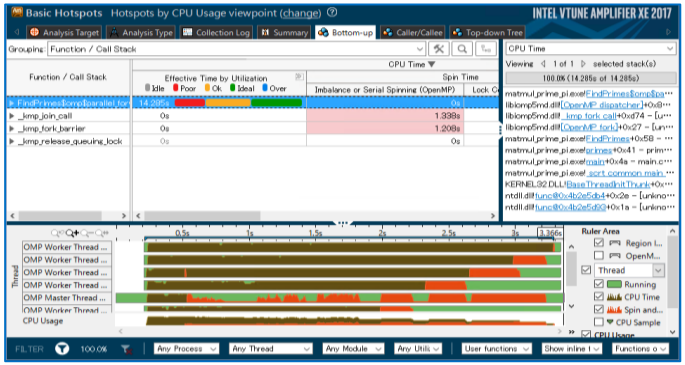
エクセルソフト株式会社
黒澤 一平

# ソースコードの近代化

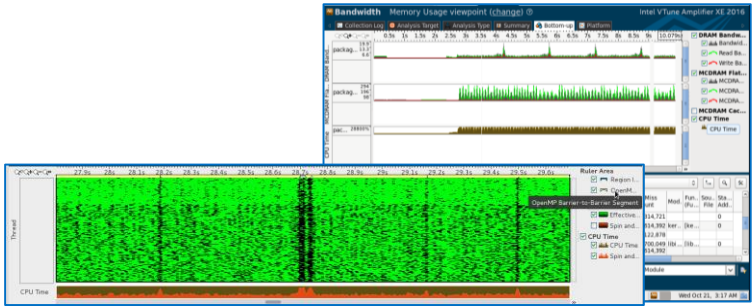
- インテル® Xeon Phi™ プロセッサーや、将来のインテル® Xeon® プロセッサー上での実行に向けた準備と適用

# インテル® ソフトウェア製品

| 名称 | 概要 |
|---|---|
| インテル® Composer XE for Fortran and C++ | Fortran, C/C++ 言語に対応した最適化コンパイラー |
| インテル® VTune™ Amplifier XE | CPU 内部の、処理の効率性やさまざまな情報を取得 |
| インテル® Advisor | ベクトル化、マルチスレッド化の解析を行い、最適化アドバイスを提供 |
| インテル® Trace Analyzer & Collector | MPI アプリケーションの動作状況やボトルネックを解析 |
| インテル® Performance Snapshot | アプリケーション全体の性能をシンプルに解析 |




インテル® VTune™ Amplifier XE




インテル® Xeon Phi™ プロセッサー
(開発コード名: Knights Landing) 対応

# 最新ツールを使用する利点

- より高度な最適化を行えるようになります。
- 初心者でも上級者に近い最適化を行えるようになります。
- 時間を大幅に短縮することができます。
- ツールからアドバイスを得ることができます。
- 数年先まで有効なコードを作ることができるようになります。

# ソフトウェア開発者が考慮するべき並列性

## ベクトル化

コアの命令セットを利用

- 1 コアごとの性能向上
- 複数のデータ要素を同時に処理 (SIMD)

## スレッド並列化

複数コアを利用

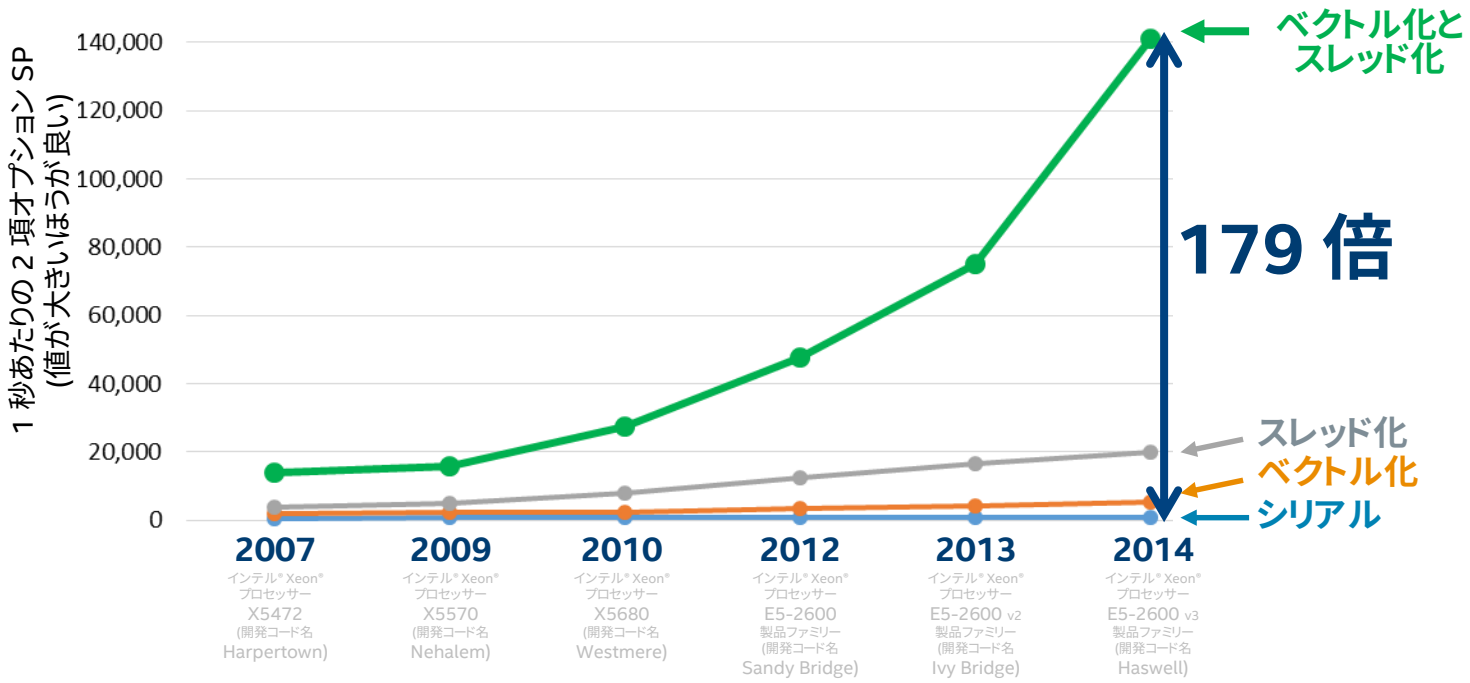
- 1 プロセッサーの性能向上
- 複数タスクの同時実行

## MPI 並列化

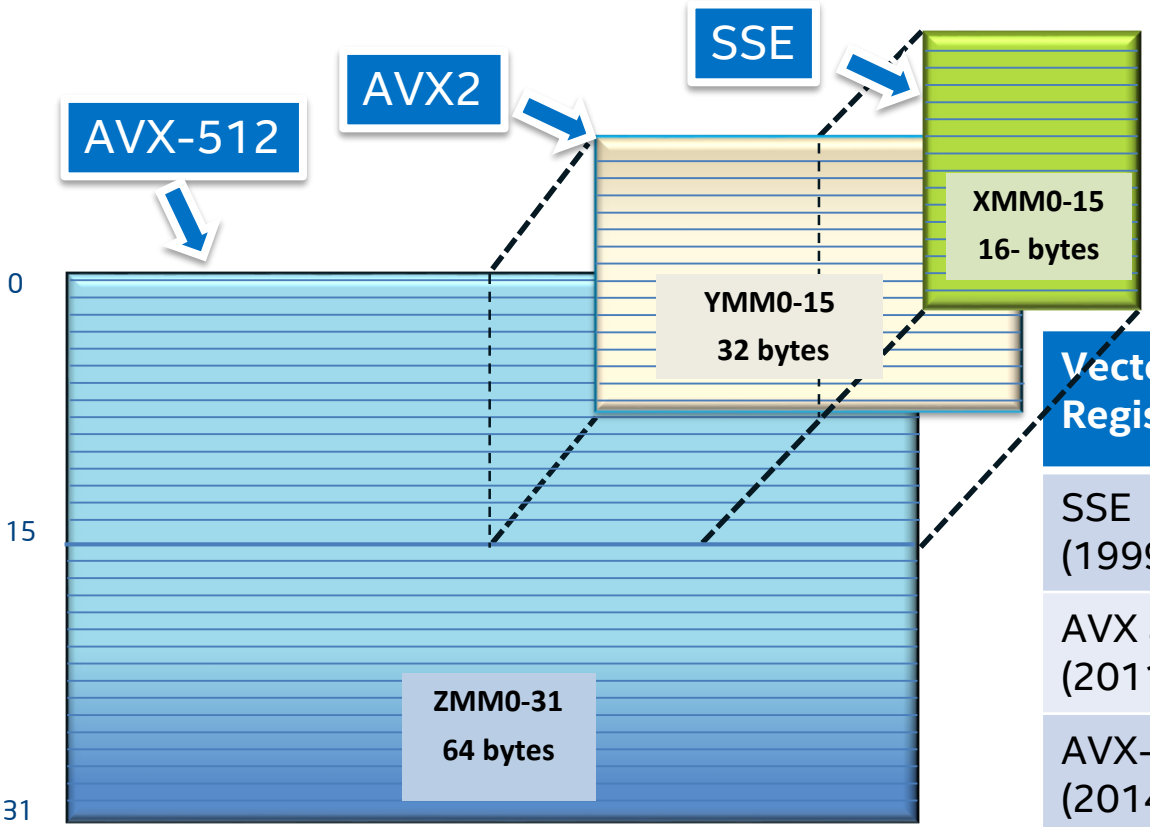
複数マシンを利用

- 複数のマシンを使用
- 複数プロセスの同時実行

# ベクトル化とマルチスレッド化、最適化の効果
## マルチスレッド化 + ベクトル化はより良い効果が得られる



# AVX-512

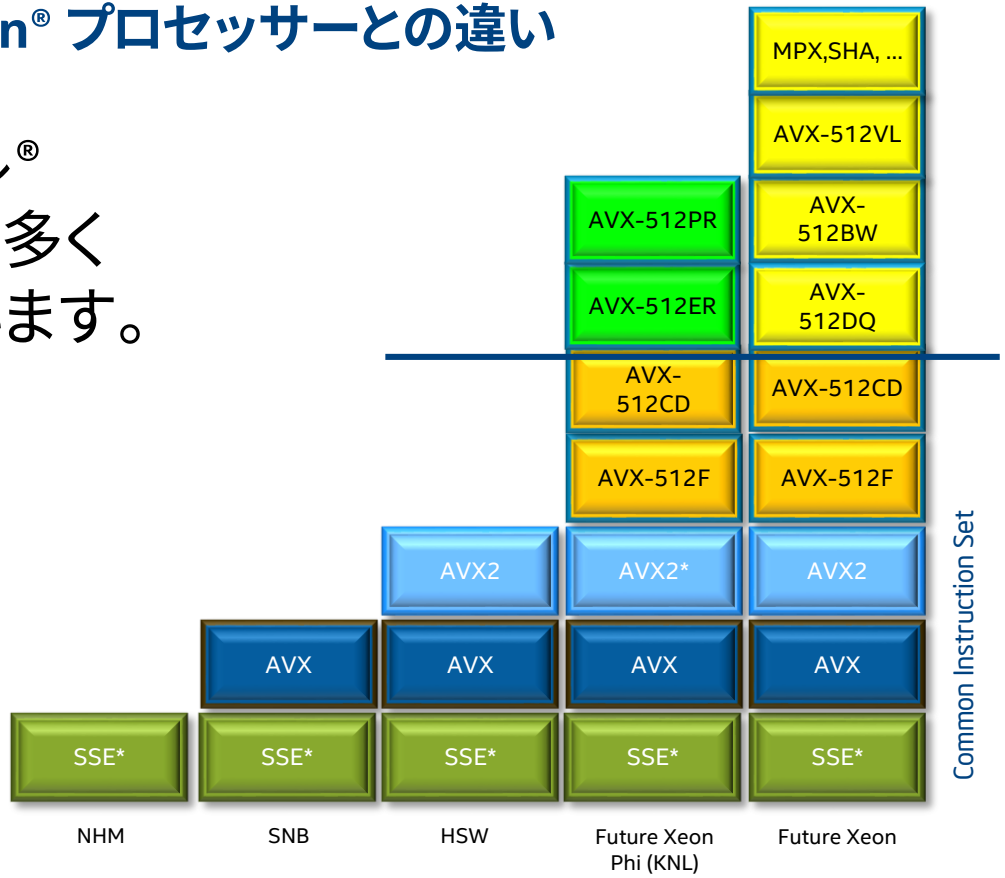



| Vector Registers | IA32 (32bit) | Intel64 (64bit) |
|---|---|---|
| SSE (1999) | 8 x 128bit | 16 x 128bit |
| AVX and AVX-2 (2011 / 2013) | 8 x 256bit | 16 x 256bit |
| AVX-512 (2014 – KNL) | 8 x 512bit | 32 x 512bit |

# AVX-512
## KNL と将来のインテル® Xeon® プロセッサーとの違い

- KNL と将来のインテル® Xeon® プロセッサーは多くの互換命令を有しています。

| コンパイラオプション | ターゲット |
|---|---|
| -xmic-avx512 | KNL のみ |
| -xcore-avx512 | 将来の XEON のみ |
| -xcommon-avx512 | KNL および将来の XEON |
| -mmic | KNC 用 |



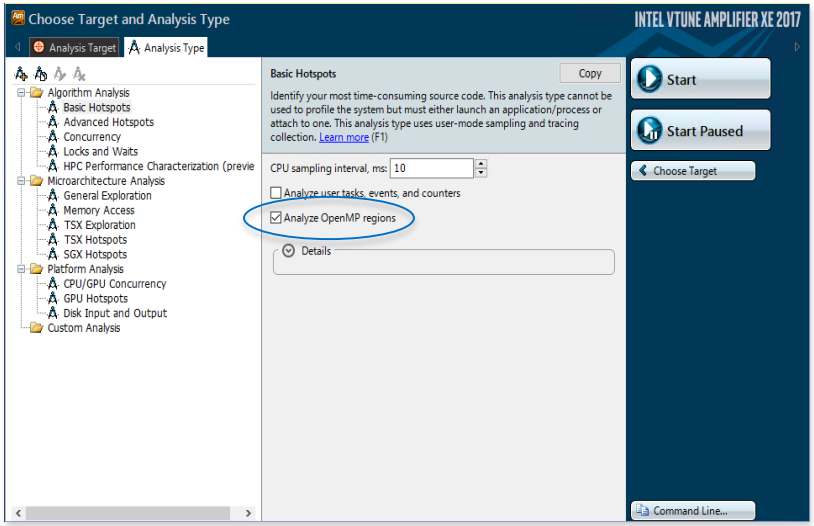
# KNL に向けた準備

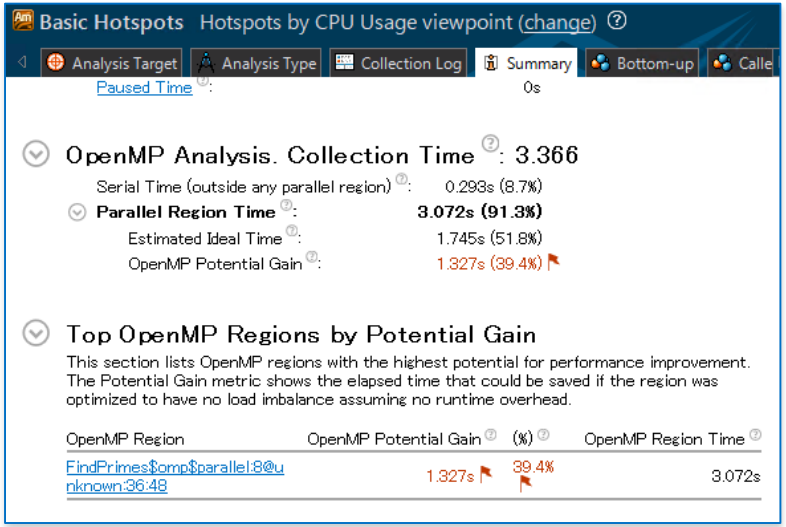
| 作業 | 対応ツール | 実施内容 |
|---|---|---|
| コンパイル/リンク | インテル® コンパイラー | インテル® Xeon® プロセッサーの場合と同じようにコンパイル、実装、解析することができます。<br><br>ただし、512ビットのベクトル化と、高並列性を目指す必要があります。 |
| ベクトル化 | インテル® コンパイラー<br>インテル® Advisor | |
| マルチスレッド化 | インテル® コンパイラー<br>インテル® Advisor | |
| ベクトル/マルチスレッド性能解析 | インテル® Advisor<br>インテル® VTune Amplifier XE | |
| MPI 性能解析 | インテル® Trace Analyzer & Collector | |

インテル® ソフトウェア開発製品は初心者でも上級者に近い最適化を行えるような様々な補助機能を提供します。

# OpenMP* アドバイス機能

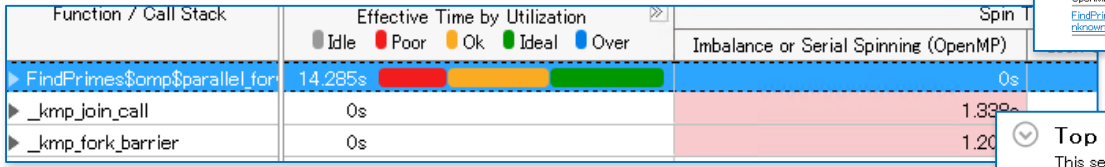
- インテル® VTune™ Amplifier XE の OpenMP* 解析機能を使用することで、OpenMP* を用いたマルチスレッド化のパフォーマンス問題と、改善点を確認することができ、修正した場合のパフォーマンスの向上度合いが表示されます。

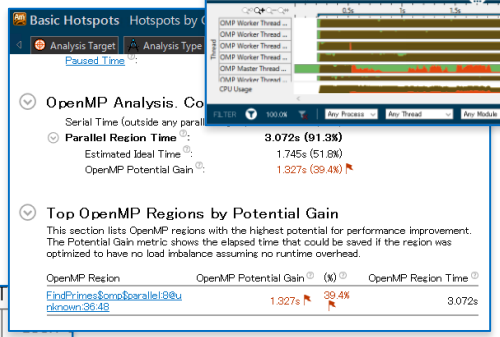


# OpenMP* のパフォーマンス問題

- 仕事の不均一性、ロックなどによるスピン時間と、OpenMP* スレッドの生成、スケジューリング、リダクション、アトミック演算、などのオーバーヘッド時間の問題を特定することができます。

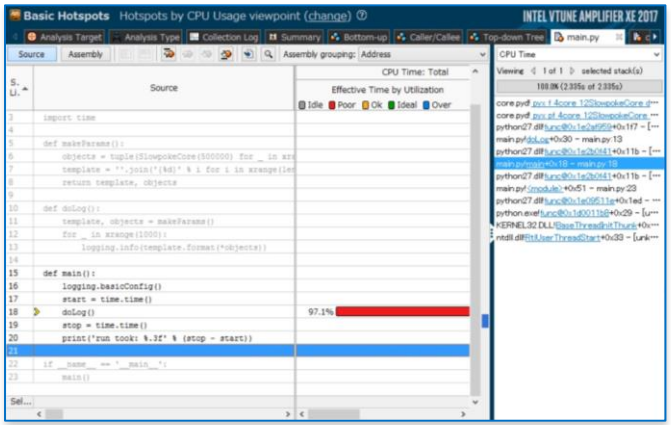



# インテル® Distribution for Python* または Go と性能解析ツール

- パフォーマンスが問題なく発揮されているか、インテル® VTune™ Amplifier XE で素早く確認することができます。

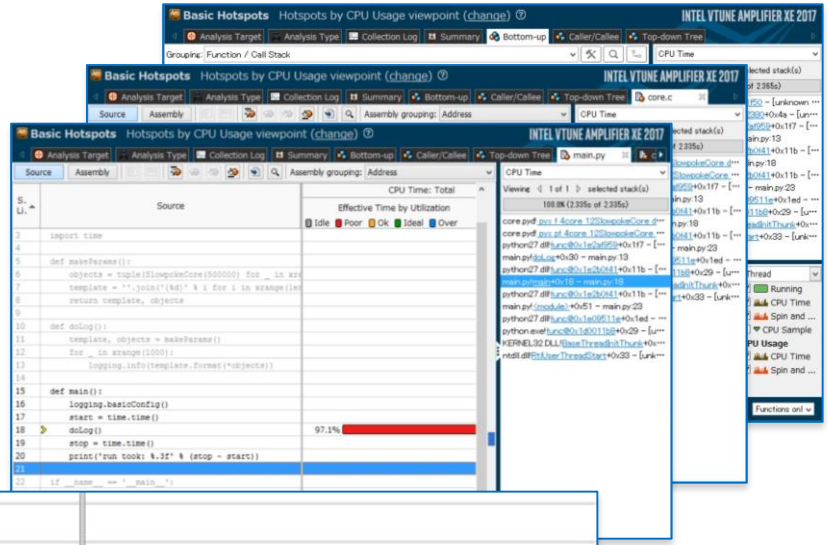
  処理が Python* または Go コード側で行われているのか、C ライブラリーやインテル® MKL に置換されているのかをすぐ確認することができ、Hotspots を即座に発見することができます。




- インテル® Distribution for Python* は通常の Python* コードと使用して内部でインテル® MKL などを呼び出して処理するため、より短時間で演算を行うことができるようになっています。

# インテル® VTune™ Amplifier XE による Python* コードの性能解析

- Python* または Go アプリケーションのパフォーマンス問題を確認することができます。
  また、Cython やインテル® MKL のネイティブ・ライブラリーと、Python* または Go の混在も同様に解析することができます。



# インテル® DAAL

DAAL : (Data Analytics Acceleration Library)

- インテル・プロセッサー上で動作する
  ビッグデータ解析処理を最適化するためのライブラリー

大量のデータを取り扱う
問題分析や意思決定の高速化

| 主な問題をカバー |
| --- |
| 知識発見<br>データマイニング<br>マシンラーニング<br>予測分析<br>AI<br>パターン認識、<br>ニューロ・コンピューティング |

# ビッグデータ (数百テラ～ペタ規模のデータ)

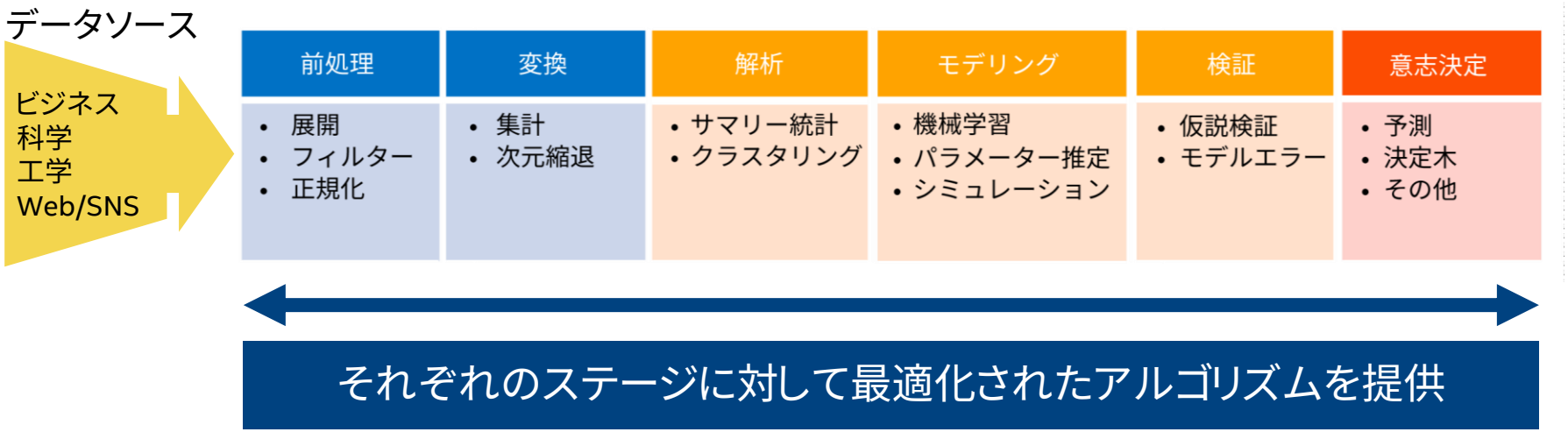
- 一般的に市販されている管理ツールや、
  データベースでは解析、保管が困難なデータ集合体

ビッグデータの特徴
- データサイズが膨大
- データの種類が多い
- データが頻繁に更新される

データ例:
単純なテキスト、画像、動画、音楽、
センサーから送られてくるデータ、
利用者の趣味趣向、金融

# インテル® DAAL

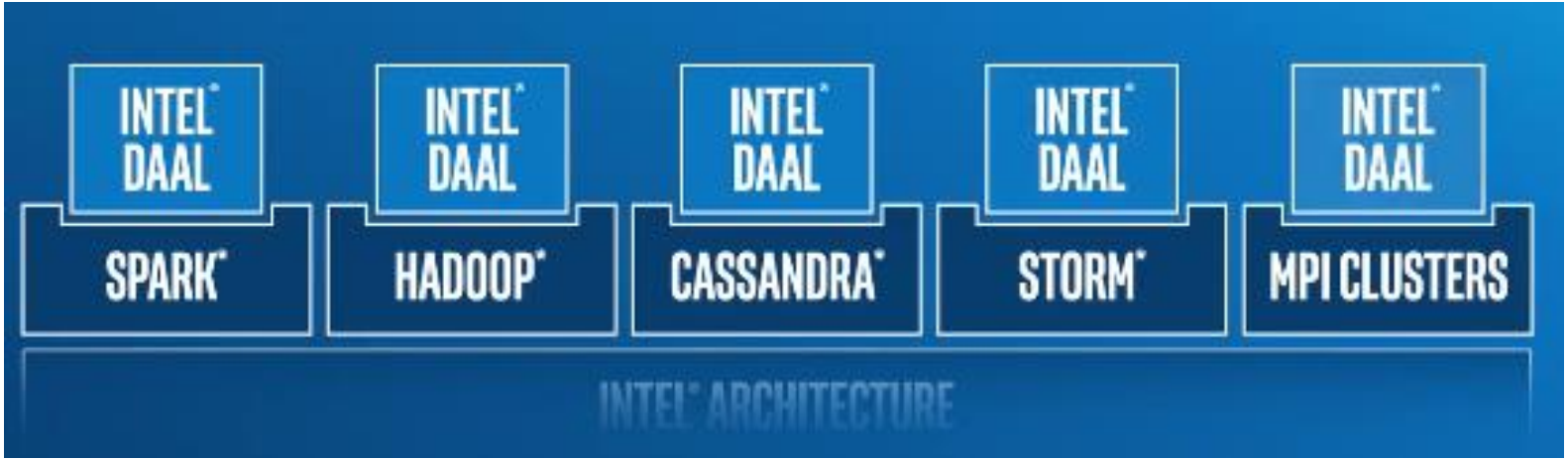
- データ分析で行われる全てのステージをカバー可能



# インテル® DAAL の特徴

- メジャーな解析プラットフォームの解析処理から
  インテル® DAAL のアルゴリズムを接続して使用

  解析処理の開発にかかる時間を短縮

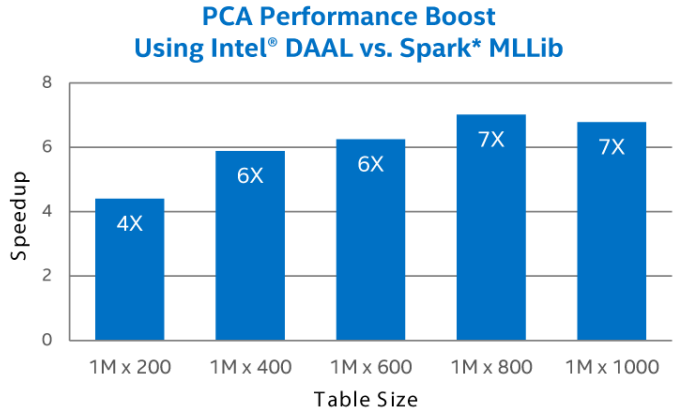


# インテル® DAAL の性能

✓さまざまなインテルプロセッサー向けに最適化済み
✓次世代プロセッサーへの移行を簡単化

対応プロセッサー
- インテル® Atom™ プロセッサー
- インテル® Core™ プロセッサー
- インテル® Xeon® プロセッサー、
- インテル® Xeon Phi™ プロセッサー

DAAL の内部実装は IPP と MKL が提供する関数




Configuration Info - Versions: Intel® Data Analytics Acceleration Library 2016, CDH v5.3.1, Apache Spark* v1.2.0; Hardware: Intel® Xeon® Processor E5-2699 v3, 2 Eighteen-core CPUs (45MB LLC, 2.3GHz), 128GB of RAM per node; Operating System: CentOS 6.6 x86_64. PCA normalized input.

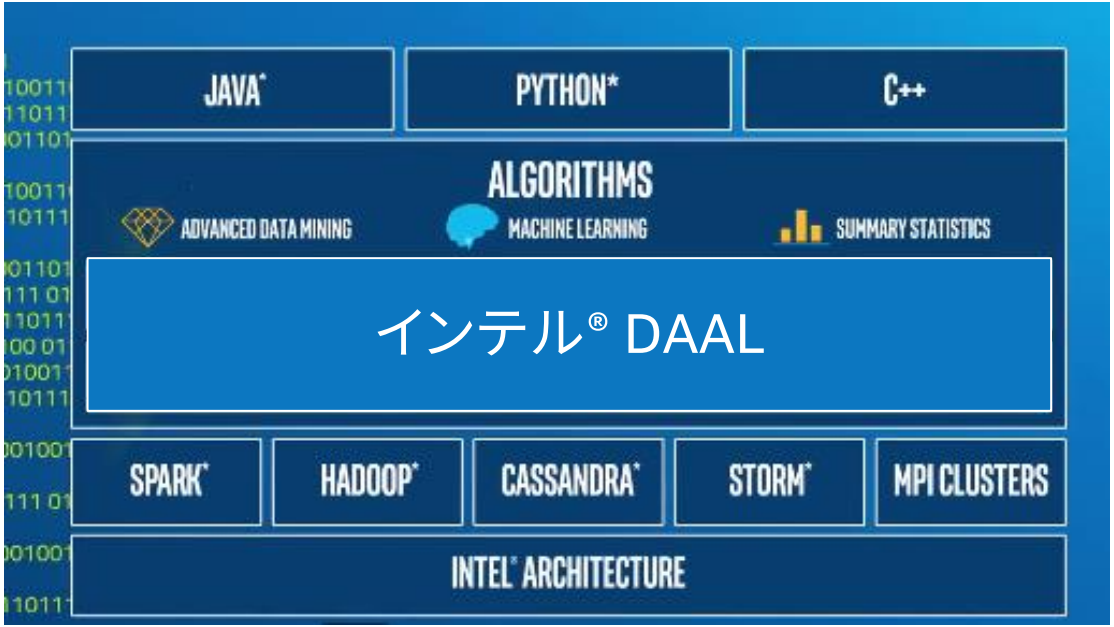
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. *Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation.

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

# インテル® DAAL 対応言語

- Java、Python, C++ 言語に対応

Java などのマネージドコード環境でも、
ネイティブコードの性能が得られます



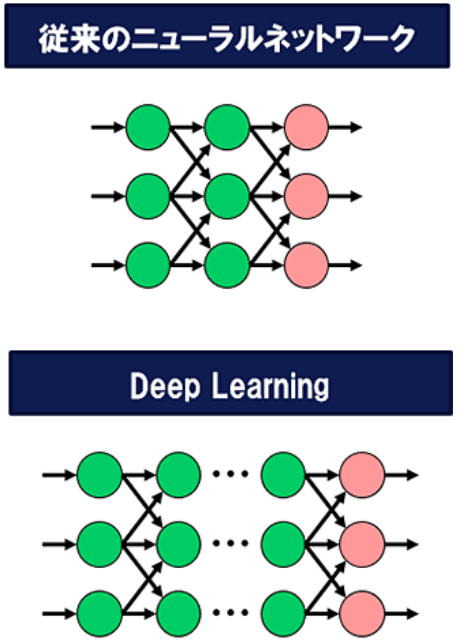
# インテル® MKL に追加された機能 ディープ・ニューラル・ネットワーク (DNN)

- 人間の脳細胞を模倣した構成を持つ機械学習システム
- 幾つかの層に分けられ、それぞれで異なる処理を実装

従来のニューラル・ネットワークでは判断基準を教える必要がある
例：パンダの画像を認識 → 人間がパンダの特徴を教える

DNN ではコンピューター自身が判断基準を学習することができる
例：パンダの画像を認識 → マシン自身がパンダの特徴を学習

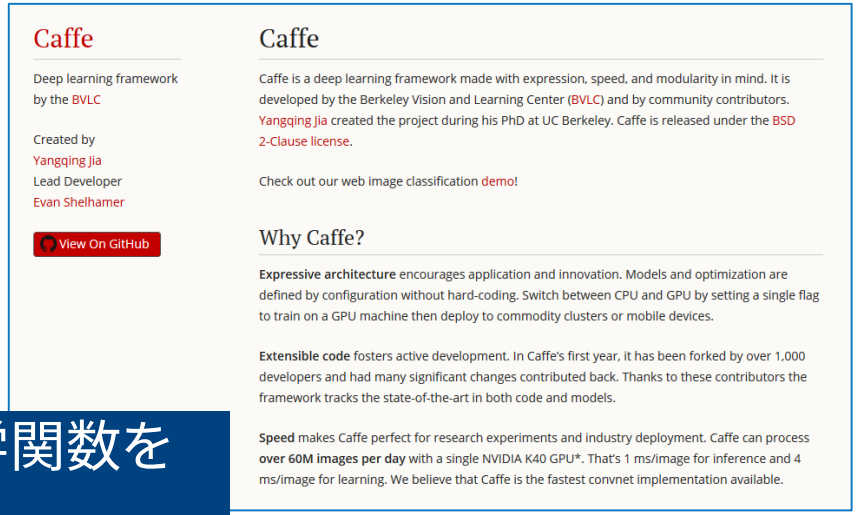
活用例：
日本語翻訳→より「日本語らしい」文章を作成
検索エンジン→より検索意図に近い情報を表示




出典：NTT DATA (本格化する「人工頭脳」のビジネス活用)
(http://www.nttdata.com/jp/ja/insights/trend_keyword/2013110701.html)

# DNN の実装に使用される一般的なフレームワーク

- Caffe：
  ディープラーニング向け
  フレームワーク
- 画像認識に関する
  コミュニティーが活発



Caffe

Deep learning framework by the BVLC

Created by
Yangqing Jia
Lead Developer
Evan Shelhamer

View On GitHub

## Caffe

Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by the Berkeley Vision and Learning Center (BVLC) and by community contributors. Yangqing Jia created the project during his PhD at UC Berkeley. Caffe is released under the BSD 2-Clause license.

Check out our web image classification demo!

### Why Caffe?

**Expressive architecture** encourages application and innovation. Models and optimization are defined by configuration without hard-coding. Switch between CPU and GPU by setting a single flag to train on a GPU machine then deploy to commodity clusters or mobile devices.

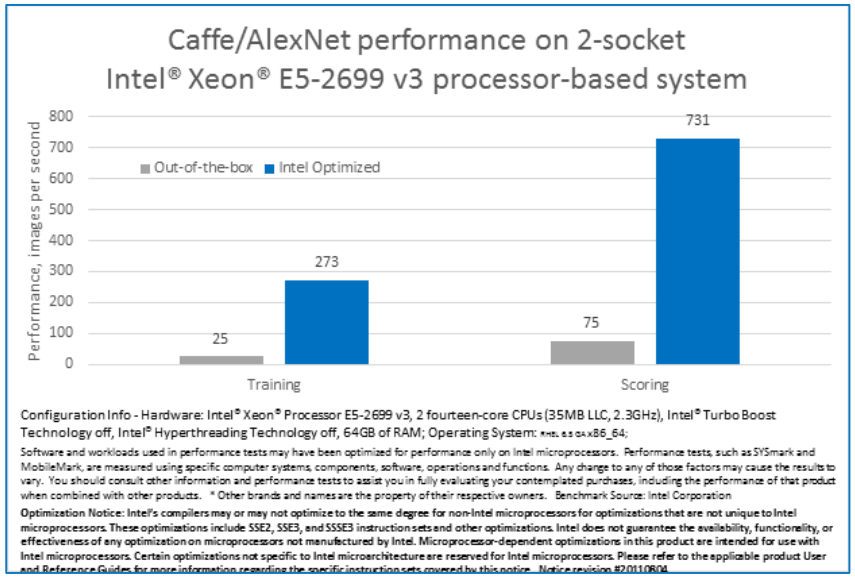
**Extensible code** fosters active development. In Caffe's first year, it has been forked by over 1,000 developers and had many significant changes contributed back. Thanks to these contributors the framework tracks the state-of-the-art in both code and models.

**Speed** makes Caffe perfect for research experiments and industry deployment. Caffe can process **over 60M images per day** with a single NVIDIA K40 GPU*. That's 1 ms/image for inference and 4 ms/image for learning. We believe that Caffe is the fastest convnet implementation available.

Caffe 上で インテル® MKL の数学関数を
利用することが可能

出典：http://caffe.berkeleyvision.org/

# インテル® MKL (DNN) の性能




Configuration Info - Hardware: Intel® Xeon® Processor E5-2699 v3, 2 fourteen-core CPUs (35MB LLC, 2.3GHz), Intel® Turbo Boost Technology off, Intel® Hyperthreading Technology off, 64GB of RAM; Operating System: RHEL 6.5 64x86_64;

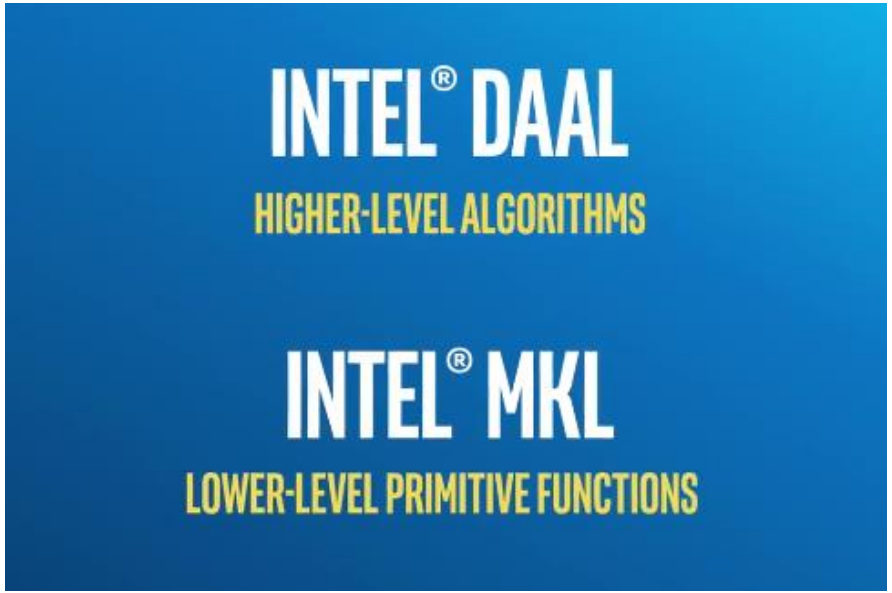
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804

- 2つの処理を高速化
  - ベクトル化と並列化により、学習スピードの最適化
  - 特徴の分類スピードの最適化

AVX 2 以上の命令セットを有するプロセッサーをサポート
(Haswell 以降)

# インテル® DAAL とインテル® MKL の違い



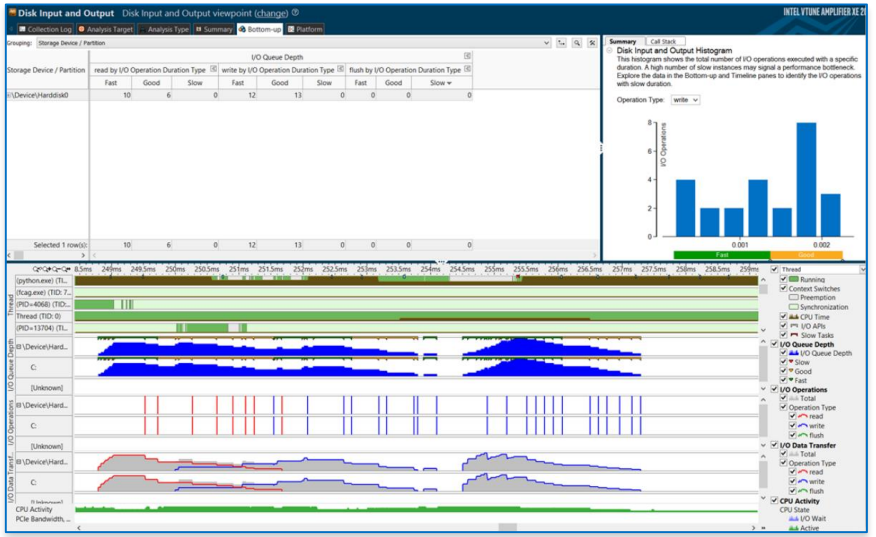

インテル® DAAL：
データ分析に即使用できる
様々なアルゴリズムを提供

インテル® MKL：
ニューラル・ネットワークの
実装を支援する関数群を提供

# ディスクアクセスに関する解析を行う

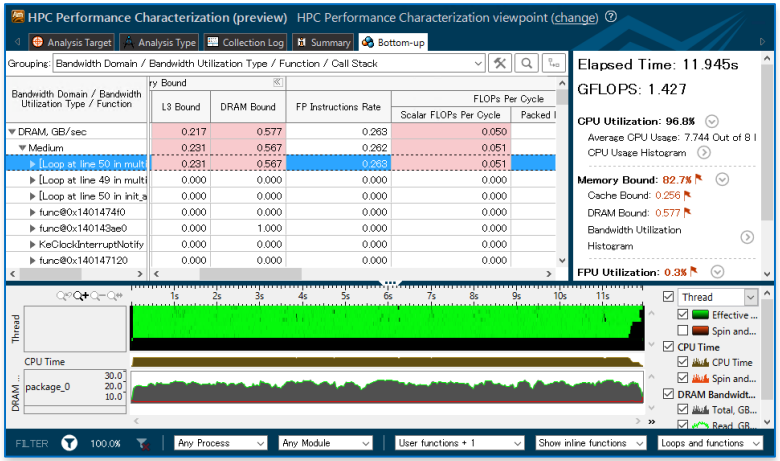
SSD や HDD への読み書きに関する解析を行う Disk Input and Output Analysis が追加されています。この機能を使用することで、読み込み遅延、書き込み遅延の発生を検出し、どのプロセス/モジュール/スレッドが発生原因であるかを素早く特定することができます。

SSD デバイスが複数ある場合や、ドライブが複数ある場合、それらを区別して表示させることができるようになっています。



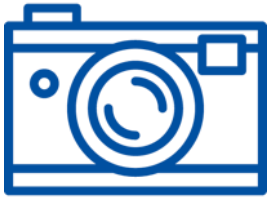
# HPC 向けの新しい解析タイプ

- HPC Performance Characterization Analysis は、HPC 分野で有用な情報である GFLOPs や、関数/ループごとの CPU 使用率や CPU 使用効率、メモリー/キャッシュに関する情報、1 サイクルあたりの FLOPs、ベクトル化状況を確認することができます。



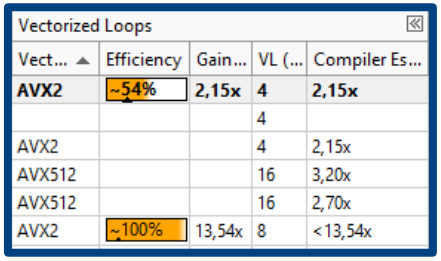

# 詳細な解析をする前のスナップショット

## Step 1 – スナップショット



- Application
- MPI
- Storage

## Step 2 – 詳細な解析

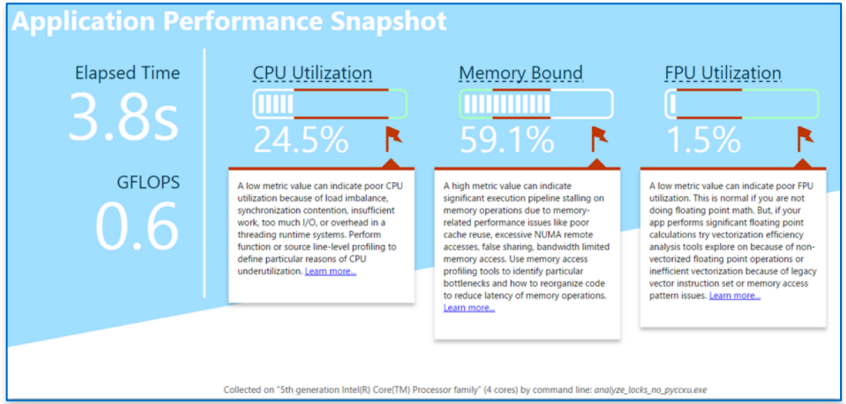



| Intel® VTune™ Amplifier | Intel® Advisor | Intel® Trace Analyzer & Collector |
|---|---|---|
| Application & Storage Profiling & Analysis | Vectorization Optimization & Thread Prototyping | MPI Profiling & Analysis |

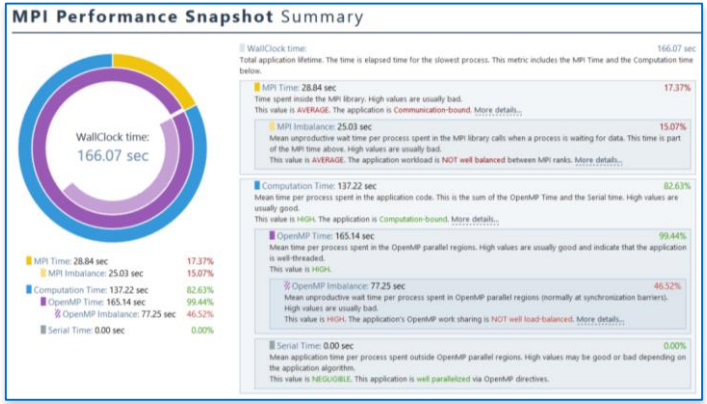
# よりシンプルに全体のパフォーマンスを確認

インテル® Performance Snapshot はアプリケーション全体のパフォーマンスを簡単に表示することができます。

そもそも本格的な解析が必要かを、事前に素早く確認することができます。




インテル® Performance Snapshot
コードの近代化に関する情報を表示

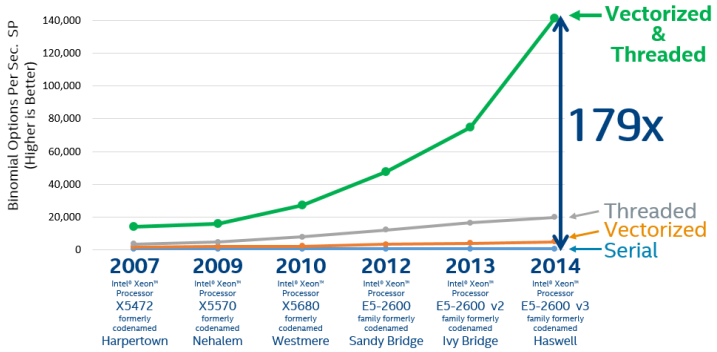



MPI Performance Snapshot
パフォーマンスのスケールを表示

# インテル® AVX512 向けの最適化

- インテル® Xeon Phi™ プロセッサー (開発コード名: Knights Landing) を始めに、今後多くのインテル® AVX512 命令セットをサポートするプロセッサーがリリースされていきます。
- 今日、コードの近代化を行うことでインテル® AVX512 命令セットや多くのコアが搭載されたプロセッサー向けの、将来にも有効な最適化を行うことができます。

必要な作業は
ベクトル化 + マルチスレッド化



# 高速なコードを素早く開発: インテル® Advisor

## 最新プロセッサーで性能を出すためにはベクトル化とマルチスレッド化が必須

- さらに、将来のプロセッサーではより差が顕著に

### ベクトル化でおきる問題:

- インテル® AVX2 を使用したのに速くならない
- そもそもどこをベクトル化すれば良い?
- 最新プロセッサー用の組込み関数を使用する必要がある?
- コンパイラーのベクトル化レポートのどこを見れば良い?

### マルチスレッド化でおきる問題:

- マルチスレッド化したけれど速くならない
- スレッド数を増やしたら性能劣化する
- マルチスレッド化に時間がかかってしまう

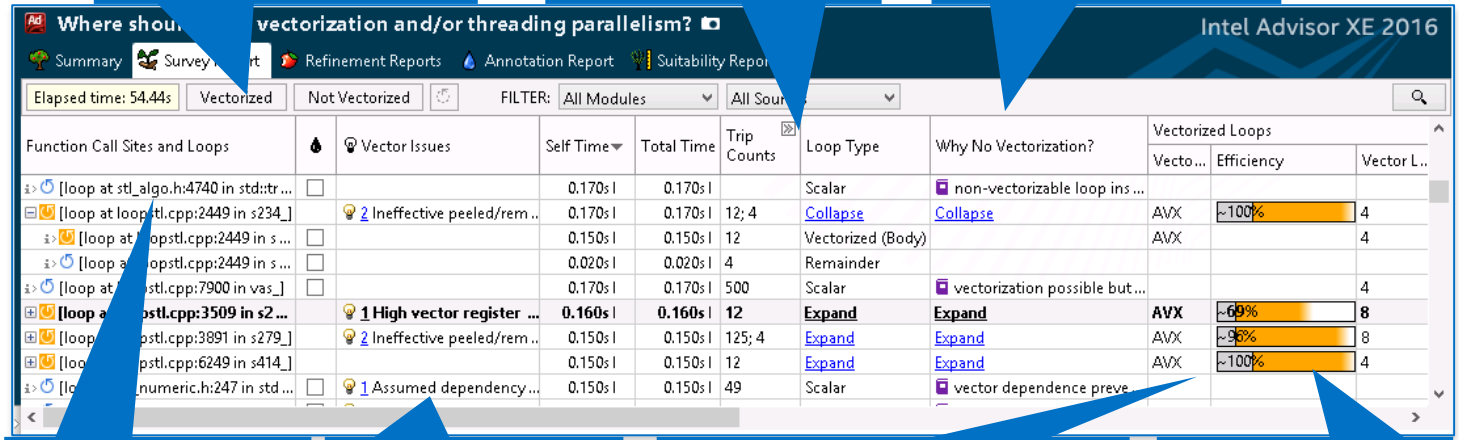
**これらの問題、疑問をインテル® Advisor が解決します**

# 正しいベクトル化のためには正しい情報を

ベクトル化された
ループをフィルター

トリップ
カウントを表示

ベクトル化を
妨げる原因を特定




ホットループ
を表示

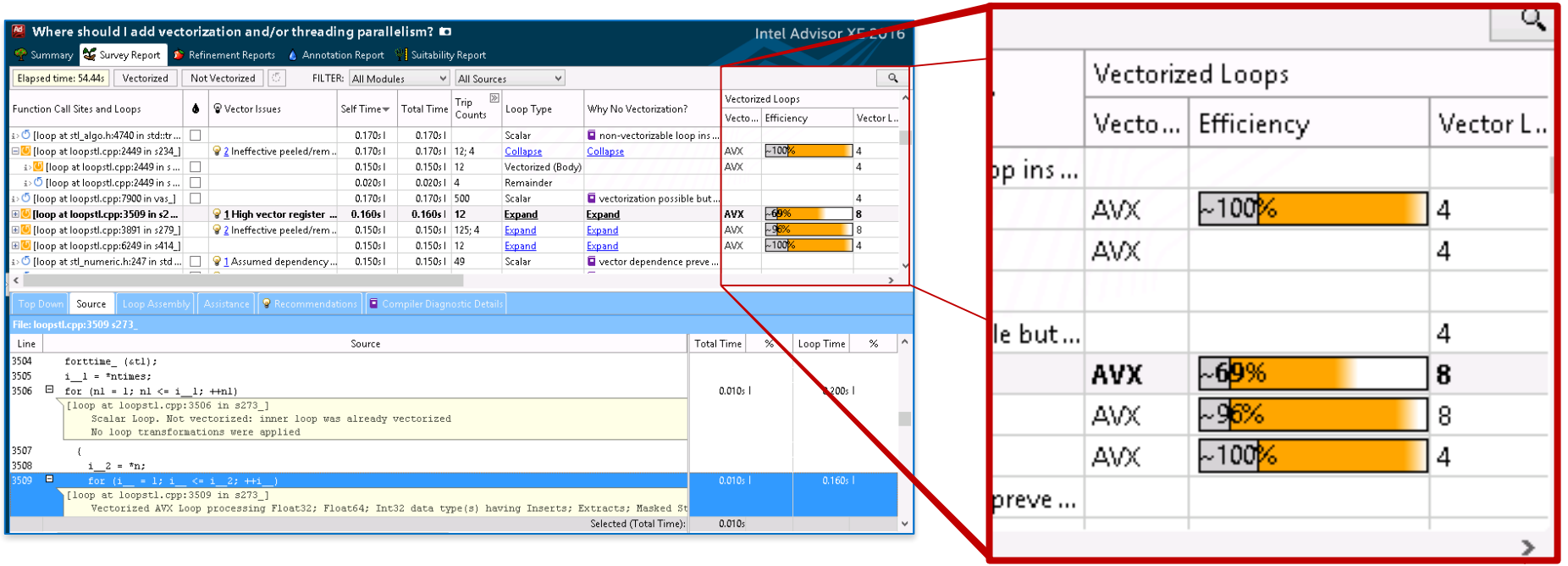
ベクトル化の
問題を表示

ベクトル命令の世代を表示

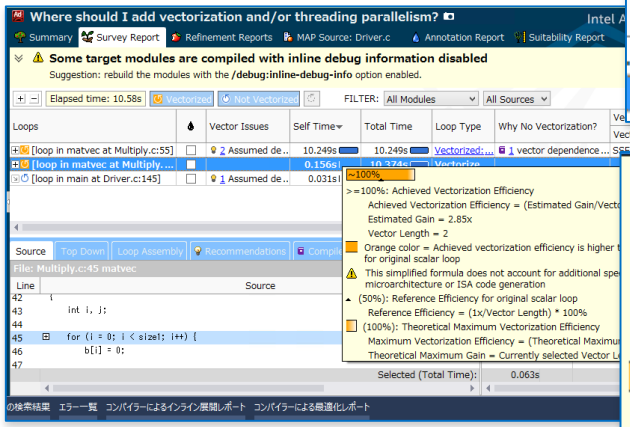
ベクトル化後の
効率を表示

**高速なコードを素早く開発**

# ベクトル化したコードの効率性を インテル® Advisor で評価



# ベクトル化の効率性に関する情報を一度に表示




| Vectorized Loops | | | | Instruction Set... | |
|---|---|---|---|---|---|
| Vector ISA | Efficiency | Gain Estimate | VL (Vector Length) | Traits | Data T... |
| SSE2 | ~100% | 2.85x | 2 | | Float32... |
| | ~71% | 1.42x | 2 | | Float64 |

~100%

>=100%: Achieved Vectorization Efficiency

Achieved Vectorization Efficiency = (Estimated Gain/Vector Length) * 100%

Estimated Gain = 2.85x

Vector Length = 2

Orange color = Achieved vectorization efficiency is higher than reference efficiency for original scalar loop

⚠ This simplified formula does not account for additional speed-ups related to microarchitecture or ISA code generation

▲ (50%): Reference Efficiency for original scalar loop

Reference Efficiency = (1x/Vector Length) * 100%

(100%): Theoretical Maximum Vectorization Efficiency

Maximum Vectorization Efficiency = (Theoretical Maximum Gain/Vector Length) * 100%

Theoretical Maximum Gain = Currently selected Vector Length = 2

# ループのベクトル化

1. ベクトル化可能だがベクトル化できていないループ
   少しの修正でベクトル化できる可能性があります。
2. ベクトル化されているがあまり性能が上がらない
   性能を容易に向上できる可能性があります。
3. ベクトル化されているがデータレイアウトの影響で性能がでない
   高速なデータ参照方法を使用します。
4. ベクトル化されていて、正しく性能を発揮する
   ほかの個所の最適化に進みます。

# ベクトル化できていないループの例

- ループの依存関係の存在
- エイリアスによる依存関係の可能性なのか、実際に依存関係があるか確認します。


```
10        for(i=0; i<NUM; i++){
11            a[i] = i%10;
12        }
13
14        for(j=1; j<NUM; j++){
15            a[j] = a[j-1] + x;
16    }
```

```
1   void addvec(int num, float
*c, float *a, float *b)
2   {
3       int i;
4       for(i=0; i<num; i++){
5           c[i] = a[i] + b[i];
6       }
7   }
```

| Loops | | Vector Issues | Self Time | Total Time | Loop Type | Why No Vectorization? |
|---|---|---|---|---|---|---|
| [loop in fCalcInteraction_ShanChen at lbpFOR... | ☐ | 1 Ineffective peeled/remainder loop(.. | 0.894s | 0.894s | Peeled/Remain... | |
| [loop in fGetSpeedSite at lbpGET.cpp:321] | ☐ | 2 Ineffective peeled/remainder loop(.. | 0.796s | 0.796s | Vectorized (Bo... | |
| **[loop in fPropagationSwap at lbpSUB.cpp:13 ...** | ☐ | **2 Assumed dependency present** | **0.673s** | **1.988s** | **Scalar** | **vector dependence preve...** |

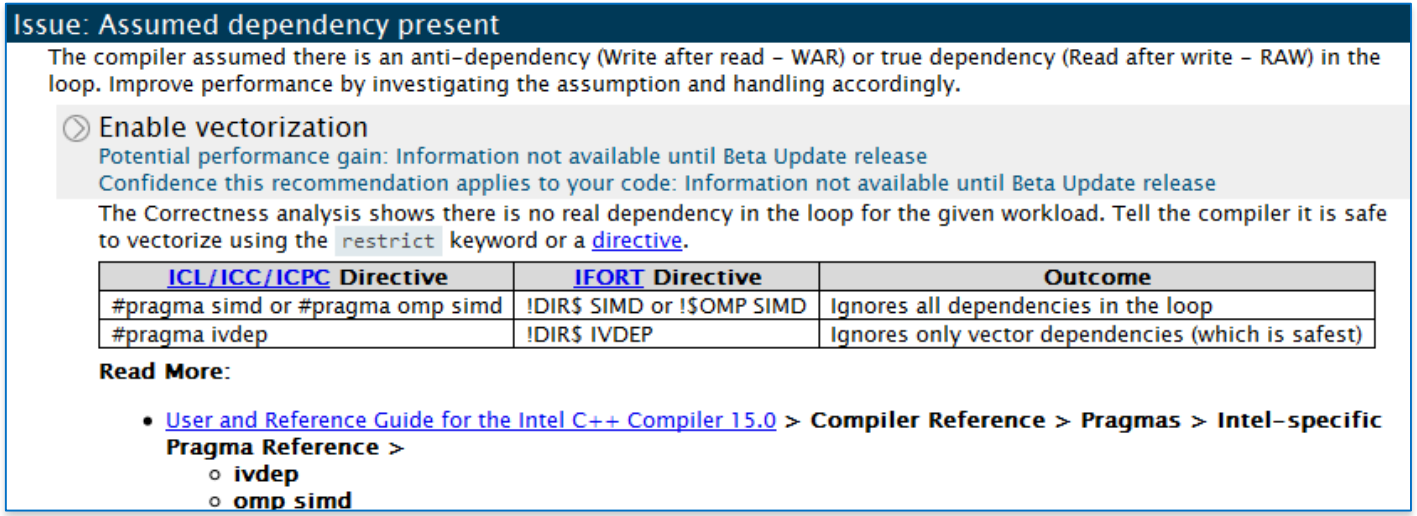
**Issue: Assumed dependency present**

The compiler assumed there is an anti-dependency (Write after read – WAR) or true dependency (Read after write – RAW) in the loop. Improve performance by investigating the assumption and handling accordingly.

Recommendation: Confirm dependency is real — Confidence: Need More Data

There is no confirmation that a real dependency is present in the loop. To confirm: Run a Dependencies analysis.

# インテル® Advisor によるアドバイス機能



**Issue: Assumed dependency present**

The compiler assumed there is an anti-dependency (Write after read – WAR) or true dependency (Read after write – RAW) in the loop. Improve performance by investigating the assumption and handling accordingly.

**Enable vectorization**

Potential performance gain: Information not available until Beta Update release

Confidence this recommendation applies to your code: Information not available until Beta Update release

The Correctness analysis shows there is no real dependency in the loop for the given workload. Tell the compiler it is safe to vectorize using the `restrict` keyword or a directive.

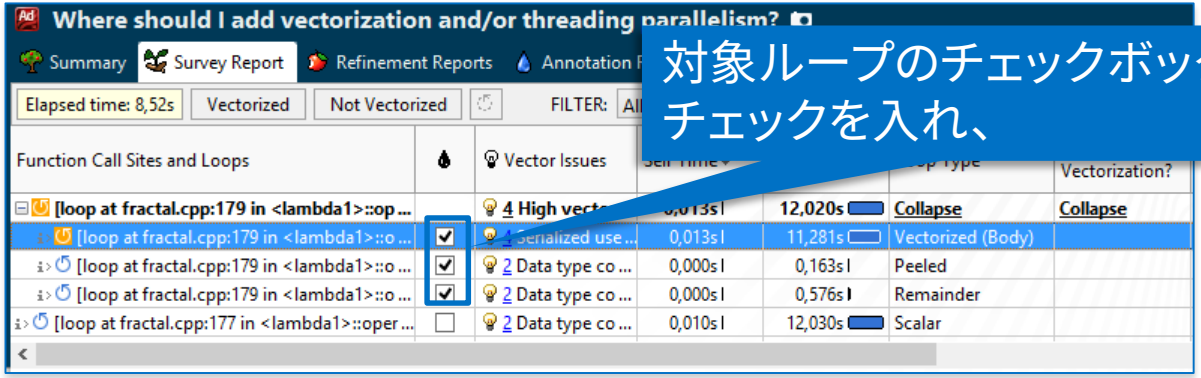
| ICL/ICC/ICPC Directive | IFORT Directive | Outcome |
|---|---|---|
| #pragma simd or #pragma omp simd | !DIR$ SIMD or !$OMP SIMD | Ignores all dependencies in the loop |
| #pragma ivdep | !DIR$ IVDEP | Ignores only vector dependencies (which is safest) |

**Read More:**

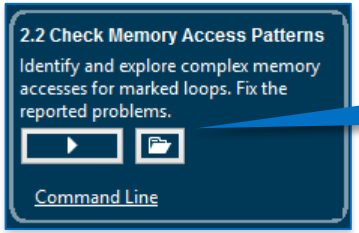
- User and Reference Guide for the Intel C++ Compiler 15.0 > **Compiler Reference > Pragmas > Intel-specific Pragma Reference >**
  - **ivdep**
  - **omp simd**

エイリアスによる依存関係の可能性がベクトル化を妨げている場合、
インテル® Advisor は修正案を提供します。
ここでは #pragma simd や #pragma ivdep の使用を提案されました。

# メモリー・アクセス・パターン解析



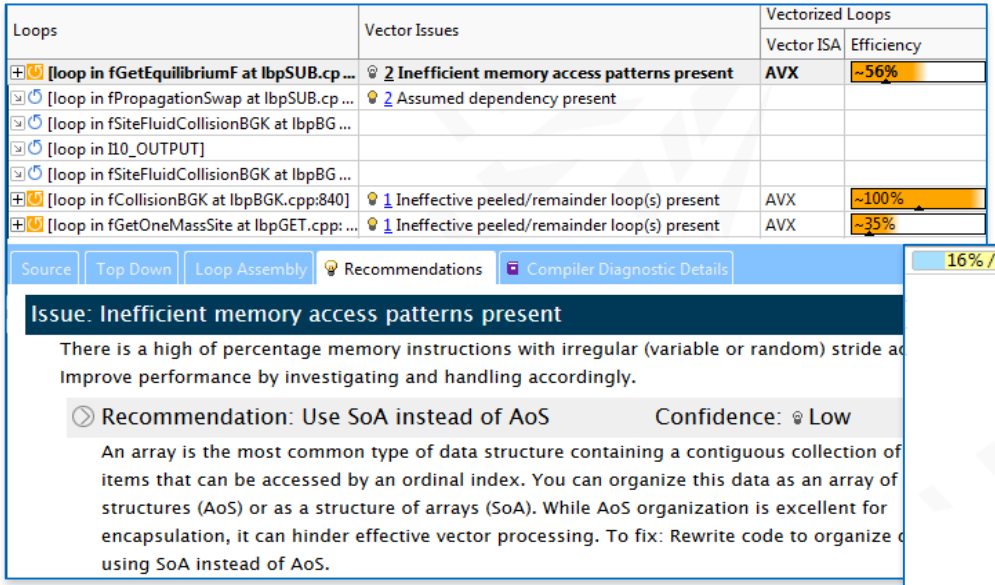

対象ループのチェックボックスにチェックを入れ、

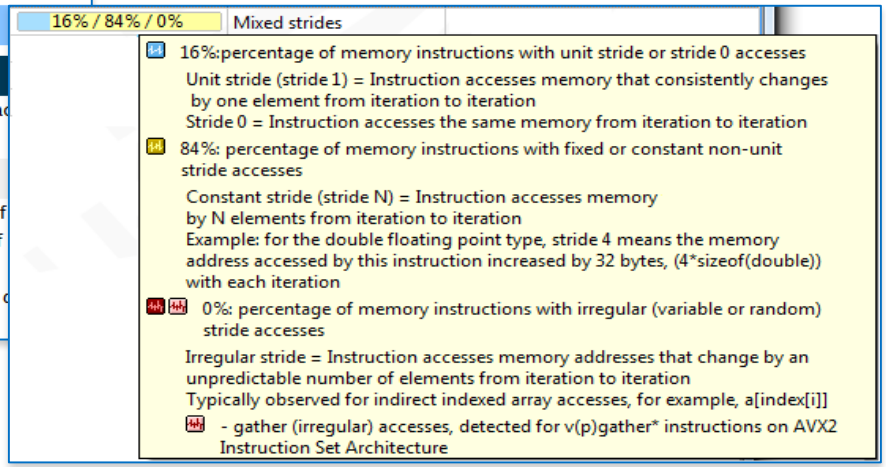



[Check Memory Access Patterns] を使用して、メモリー・アクセス・パターンを解析します。

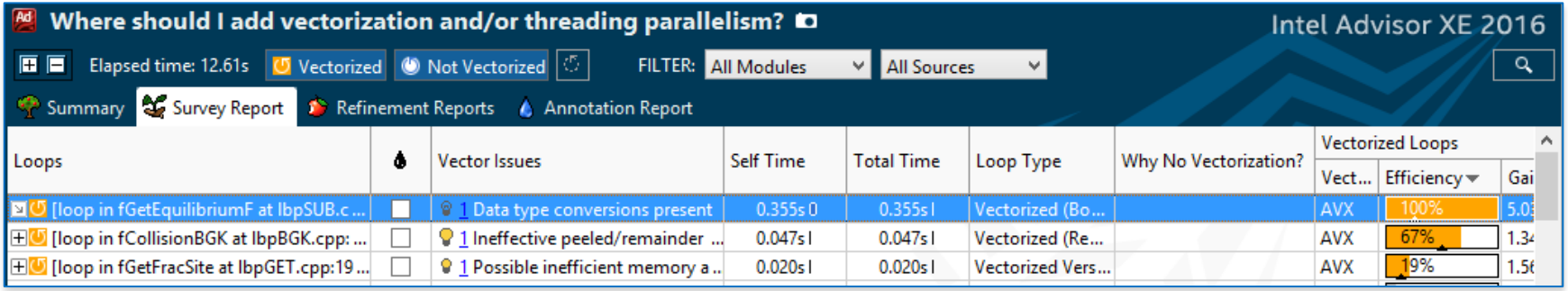
# データレイアウトが問題でベクトル化の性能が発揮できない例

この例では 84% が非ユニットストライドと検出されたため、構造体を Structure of Array に変更することを検討



# ベクトル化個所が性能を発揮しているか、インテル® Advisor を用いて確認




- インテル® AVX でベクトル化され、効率が高く、短時間で処理できるループになりました。
- 数値的、視覚的に、最適化の効果を確認することで作業効率を高めることができます。

# まとめ

- より高度な最適化を短時間で行うことができます。
- 初心者も上級者もツールを使う大きなメリットがあります。
- 数年先まで有効なコードを作ることができるようになります。

# 法務上の注意書きと最適化に関する注意事項

本資料の情報は、現状のまま提供され、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。製品に付属の売買契約書『Intel's Terms and Conditions of Sale』に規定されている場合を除き、インテルはいかなる責任を負うものではなく、またインテル製品の販売や使用に関する明示または黙示の保証 (特定目的への適合性、商品性に関する保証、第三者の特許権、著作権、その他、知的財産権の侵害への保証を含む) をするものではありません。

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ー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。

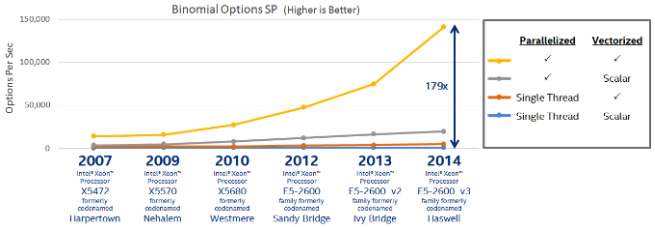


**最適化に関する注意事項**

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ー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ー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ーに関して、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ー依存の最適化は、インテル® マイクロプロセッサー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ー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

# 補足資料: 2 項オプション SP のシステム構成




**最適化に関する注意事項**

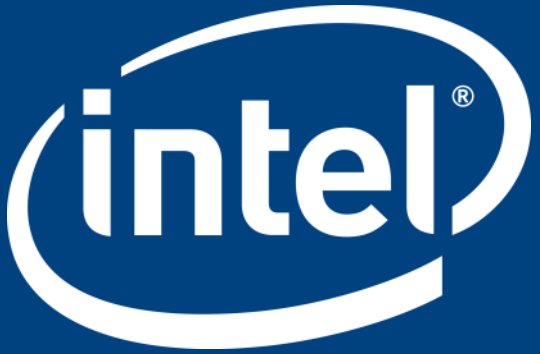
インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテク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ー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

インテル社内での測定値。

システム構成

| プラットフォーム | スケーリングされていないコアクロックの周波数 | コア/ソケット | ソケット数 | L1 データキャッシュ | L1 命令キャッシュ | L2 キャッシュ | L3 キャッシュ | メモリー | メモリー周波数 | メモリーアクセス | H/W プリフェッチ有効 | HT 有効 | ターボ有効 | C ステート | OS | カーネル | コンパイラー・バージョン |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| インテル® Xeon® プロセッサー 5472 | 3GHz | 4 | 2 | 32K | 32K | 12MB | なし | 32GB | 800MHz | UMA | Y | N | N | 無効 | Fedora* 20 | 3.11.10-301.fc20 | icc 14.0.1 |
| インテル® Xeon® プロセッサー X5570 | 2.90GHz | 4 | 2 | 32K | 32K | 256K | 8MB | 48GB | 1333MHz | NUMA | Y | Y | Y | 無効 | Fedora* 20 | 3.11.10-301.fc20 | icc 14.0.1 |
| インテル® Xeon® プロセッサー X5680 | 3.33GHz | 6 | 2 | 32K | 32K | 256K | 12MB | 48MB | 1333MHz | NUMA | Y | Y | Y | 無効 | Fedora* 20 | 3.11.10-301.fc20 | icc 14.0.1 |
| インテル® Xeon® プロセッサー E5-2690 | 2.90GHz | 8 | 2 | 32K | 32K | 256K | 20MB | 64GB | 1600MHz | NUMA | Y | Y | Y | 無効 | Fedora* 20 | 3.11.10-301.fc20 | icc 14.0.1 |
| インテル® Xeon® プロセッサー E5-2697v2 製品ファミリー | 2.70GHz | 12 | 2 | 32K | 32K | 256K | 30MB | 64GB | 1867MHz | NUMA | Y | Y | Y | 無効 | Fedora* 20 | 3.11.10-301.fc20 | icc 14.0.1 |
| 開発コード名 Haswell | 2.20GHz | 14 | 2 | 32K | 32K | 256K | 35MB | 64GB | 2133MHz | NUMA | Y | Y | Y | 無効 | Fedora* 20 | 3.13.5-202.fc20 | icc 14.0.1 |


intel®