インテル® Advisor 2017 ツールによるベクトル化のクイック解析

この記事は、インテル® デベロッパー・ゾーンに公開されている「Quick Analysis of Vectorization Using the Intel® Advisor 2017 Tool」の日本語参考訳です。

この記事では、以前の記事 (英語) で使用したループの例を用いて、引き続きインテル® Xeon Phi™ プロセッサー上でのベクトル化について調査します。インテル® Advisor 2017 のコマンドライン・インターフェイスを使用して、ループのパフォーマンスの初期解析を行い、コード内の hotspot の概要を得る方法を紹介します。その後、インテル® Advisor 2017 のグラフィカル・ユーザー・インターフェイス (GUI) でより詳細な解析を行います。

はじめに

インテルは、ソフトウェア開発者の生産性を向上し、インテル® プロセッサーを最大限に活用できるように支援するため、各種ソフトウェア製品を提供しています。その 1 つであるインテル® Parallel Studio XE は、コンパイラーと解析ツール群を備えており、インテル製ハードウェア上で動作するアプリケーションの開発、解析、最適化を支援します。

この記事では、インテル® Parallel Studio XE に含まれる解析ツールの 1 つであるインテル® Advisor 2017 を使用して、アプリケーションを解析し、コードのベクトル化を向上するためのアドバイスを得ます。

ベクトル化が重要な理由とインテル® Advisor を使用する利点

ベクトルレベルの並列処理により、ソフトウェアは、ベクトルレジスターや SIMD (Single Instruction Multiple Data) 命令などの特別なハードウェア機能を利用できます。インテル® Xeon Phi™ プロセッサーなどの新しいインテル® プロセッサーでは、512 ビット幅のベクトルレジスターとインテル® アドバンスト・ベクトル・エクステンション 512 (インテル® AVX-512) 命令セット・アーキテクチャー (ISA) を利用できます。そのため、各コアで 2 つのベクトル・プロセシング・ユニット (VPU) を使用して、各 VPU で 16 の単精度 (32 ビット) または 8 つの倍精度 (64 ビット) 浮動小数点数を処理することが可能です。

最新のプロセッサーのパフォーマンスを最大限に引き出すには、マルチコアを利用するようにコードをスレッド化する必要があります。ベクトル化とスレッド化を併用することで、相乗効果により、それぞれを単独で行うよりもコードを高速化できます。

インテル® Advisor は、アプリケーションを解析し、ベクトル化の範囲を知らせるだけでなく、さらなるベクトル化と現在のベクトル化の効率向上の可能性も示します。

インテル® Advisor は、任意のコンパイラーでコンパイルされたアプリケーションを解析できますが、インテル® コンパイラーを使用すると、コンパイラーにより生成されたレポートの情報を使用できるため、さらに効果的です。

インテル® Advisor の使用法

インテル® Advisor を最も効率良く利用するには、GUI を使用します。GUI から、インテル® Advisor がコードを解析して収集したすべての情報と推奨事項にアクセスできます。詳細は、https://www.isus.jp/intel-advisor-xe/ を参照してください。ドキュメント、トレーニング資料、サンプルコード、製品サポート、インテル® Advisor コミュニティー・フォーラムに関する情報を入手できます。

インテル® Advisor には、コマンドライン・インターフェイス (CLI) もあります。リモートホストで作業したり、スクリプトを使用するなどして解析タスクを自動化しやすい方法で情報を生成できます。

インテル® Xeon Phi™ プロセッサーを搭載した Linux* ベースのマシンで操作する場合、特定の解析ワークフローに応じて、GUI と CLI を使い分ける必要があります。一部のケースでは、パフォーマンス・サマリーの概要を取得したり、ワークフロー解析の初期フェーズで CLI が役立ちます。インテル® Advisor for Linux* の CLI に関する詳細は、https://software.intel.com/en-us/intel-advisor-2017-user-guide-linux-command-line-interface-reference (英語) を参照してください。

次のセクションでは、Linux* ベースのマシンでインテル® Advisor の CLI を使用して初期パフォーマンス解析を実行する手順を説明します。このクイック解析により、アプリケーションのパフォーマンス・ボトルネックと初期の最適化で注目すべき場所に関する情報が得られます。また、この手順を利用して、テストと結果報告を自動化することができます。

この解析は、最初のステップとして実行されることを想定しているため、限られた情報のみ提供します。
インテル® Advisor により提供される完全な情報と機能は、GUI と CLI を併用することで利用できます。

インテル® Xeon Phi™ プロセッサー上でインテル® Advisor を使用する: Survey Analysis (調査解析) の実行

ここでは、以前の記事 (英語) で使用したインテル® AVX-512 ISA を利用してベクトル化を向上するサンプルコードを使って説明します。ソースコードに関する詳細は、以前の記事を参照してください。サンプルコードは、こちらからダウンロードできます。

このサンプルコードを次のハードウェアで実行します。

プロセッサー: インテル® Xeon Phi™ プロセッサー 7250 (1.40GHz)
コア数: 68
スレッド数: 272

最初に、インテル® Xeon Phi™ プロセッサー上で実行する最適化された実行ファイルを生成します。インテル® Advisor が情報を抽出できる実行ファイルを生成するため、いくつかのオプションを指定してアプリケーションをコンパイルします。`-xMIC-AVX512` オプションは、必ず指定しなければなりません。このオプションは、インテル® Xeon Phi™ プロセッサーでサポートされるインテル® AVX-512 のすべてのサブセットを使用できるようにします (Zhang, 2016)。デバッグ情報とシンボルを生成するため `-g` と、実行ファイルを最適化するため `-O3` も指定します。`-O3` の代わりに `-O2` を使用することもできます。

```
$ icpc Histogram_Example.cpp -g -O3 -restrict -xMIC-AVX512 -o run512
-lopencv_highgui -lopencv_core -lopencv_imgproc
```

ここでは、アプリケーションで使用されるポインターがエイリアスされないことをコンパイラーに知らせるため、`-restrict` オプションも指定しています。また、このアプリケーションは、OpenCV* ライブラリー (www.opencv.org (英語)) を使用してディスクからイメージを読み取るため、OpenCV* ライブラリーにリンクしています。ダウンロードしたサンプルコードには、`Makefile` ファイルが含まれています。この `Makefile` ファイルを使用して、インテル® Advisor 向けの実行ファイルを生成できます。

次に、インテル® Advisor の CLI バージョンを実行します。調査解析は、コードのベクトル化の状況と hotspot を特定するのに必要な情報が得られるため、解析の開始点として役立ちます。

```
$ advixe-cl -collect survey -project-dir ./AdvProj-Example-AVX512 -search-dir
all:=./src -- ./run512 image01.jpg
```

上記のコマンドは、インテル® Advisor を実行し、プロジェクト・ディレクトリー AdvProj-Example-AVX512 を作成します。インテル® Advisor は、このディレクトリー内に解析結果を含む e000 ディレクトリーを作成します。e000 ディレクトリーには、次のものが含まれます。

```
$ ls AdvProj-Example-AVX512/e000/
e000.advixeexp  hs000  loop_hashes.def
$
```

hs000 ディレクトリーには、調査解析の結果が生成されます。

次のステップでは、インテル® Advisor で実行した調査解析の結果を確認します。CLI を使用してレポートを生成します。-collect オプションを -report オプションに変更して、データ収集と同じプロジェクト・ディレクトリーを参照していることを確認します。次のコマンドは、プロジェクト・ディレクトリー以下の結果ディレクトリーに含まれる調査データから調査レポートを生成します。

```
$ advixe-cl -report survey -project-dir ./AdvProj-Example-AVX512 -format=text
-report-output=./REPORTS/survey-AVX512.txt
```

上記のコマンドは、REPORTS サブディレクトリーに survey-AVX512.txt という名前のレポートを作成します。このレポートは、列形式になっており、複数の列が含まれているため、コンソール上では読みづらいかもしれません。-filter オプションを指定すると、表示される列数を制限できます (インテル® Advisor の現在のバージョンでは、調査レポートでのみこのオプションがサポートされています)。

xml 形式でレポートを作成することもできます。その場合、-format オプションの値を text から xml に変更します。

```
$ advixe-cl -report survey -project-dir ./AdvProj-Example-AVX512 -format=xml
-report-output=./REPORTS/survey-AVX512.xml
```

xml 形式では、レポートファイル内の情報が 1 列にまとめられるため、小さな画面では、xml 形式のレポートのほうが読みやすいでしょう。以下に、レポートの一部を示します。

```
(…)
</function_call_site_or_loop>
  <function_call_site_or_loop ID="4"
   Function_Call_Sites_and_Loops="[child]-[loop in main at
Histogram_Example.cpp:107]"
                               Self_Time="0.060s"
                               Total_Time="0.120s"
                               Type="Vectorized (Body)"
                               Why_No_Vectorization=""
                               Vector_ISA="AVX512"
                               Compiler_Estimated_Gain="3.37x"
                               Trip_Counts_Average=""
                               Trip_Counts_Min=""
                               Trip_Counts_Max=""
                               Trip_Counts_Call_Count=""
                               Transformations=""
                               Source_Location="Histogram_Example.cpp:107"
                               Module="run512">
  (…)
```

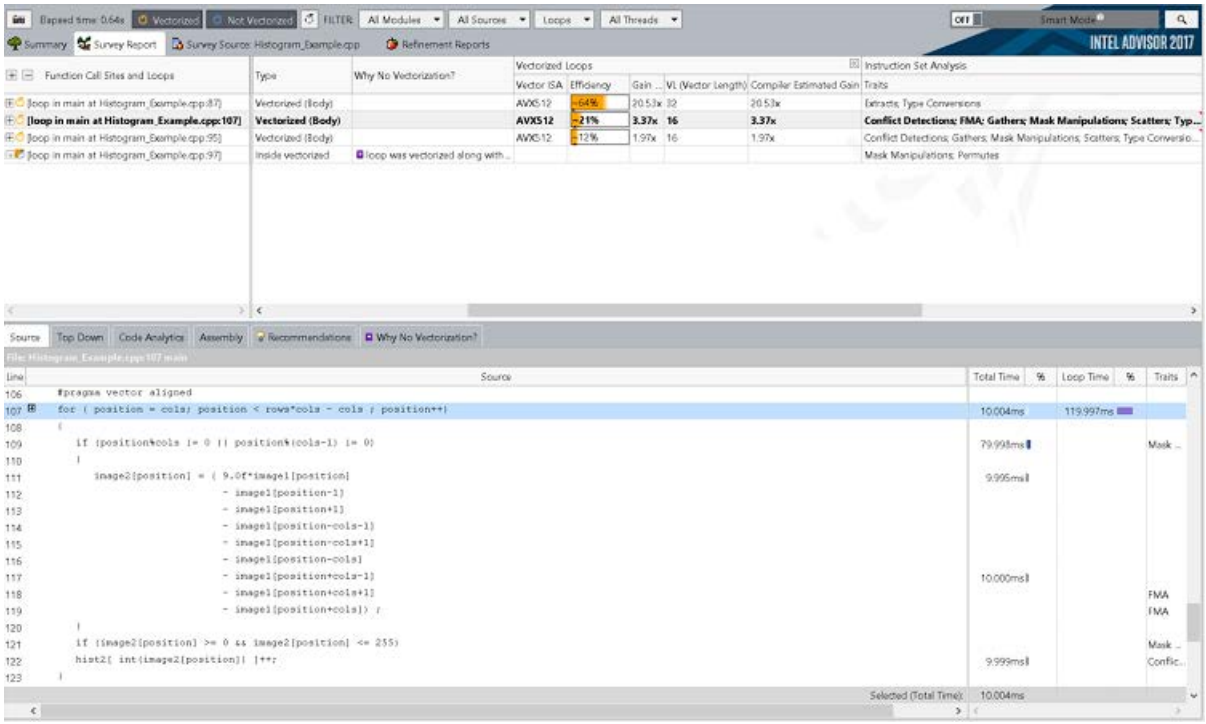
```
  </function_call_site_or_loop>
  <function_call_site_or_loop ID="8" name="[loop in main at
Histogram_Example.cpp:87]"
                              Self_Time="0.030s"
                              Total_Time="0.030s"
                              Type="Vectorized (Body; [Remainder])"
                              Why_No_Vectorization="1 vectorization possible
but seems inefficient.Use vector always directive or -vec-threshold0 to
override "
                              Vector_ISA="AVX512"
                              Compiler_Estimated_Gain="20.53x"
                              Trip_Counts_Average=""
                              Trip_Counts_Min=""
                              Trip_Counts_Max=""
                              Trip_Counts_Call_Count=""
                              Transformations=""
                              Source_Location="Histogram_Example.cpp:87"
                              Module="run512">
  </function_call_site_or_loop>
  <function_call_site_or_loop ID="1"
   Function_Call_Sites_and_Loops="[child]-[loop in main at
Histogram_Example.cpp:87]"
                              Self_Time="0.030s"
                              Total_Time="0.030s"
                              Type="Vectorized (Body)"
                              Why_No_Vectorization=""
                              Vector_ISA="AVX512"
                              Compiler_Estimated_Gain="20.53x"
                              Trip_Counts_Average=""
                              Trip_Counts_Min=""
                              Trip_Counts_Max=""
                              Trip_Counts_Call_Count=""
                              Transformations=""
                              Source_Location="Histogram_Example.cpp:87"
                              Module="run512">
```

前述のとおり、インテル® Advisor の調査解析は、アプリケーションに含まれるループのパフォーマンスの概要を生成します。例えば、上記の例の最初のコードブロックは、ソースコードの 107 行目のループがインテル® AVX-512 ISA を使用してベクトル化されたこと、および (スカラーバージョンと比較した) ループのパフォーマンス向上の推定値とタイミング情報を示しています。2 つ目と 3 つ目のコードブロックからは、ソースコードの 87 行目のループのパフォーマンスの概要が得られます。ループ本体はベクトル化されましたが、リマインダー・ループはベクトル化されなかったことが分かります。

また、個々のループにループ ID が割り当てられていることが分かります。これは、インテル® Advisor が将来の解析でループを追跡できるようにするためのラベル付けです (例えば、上記のパフォーマンスの概要を確認後、コマンドラインでループ ID を指定して特定のループの詳細情報を生成することができます)。

このセクションでは、インテル® Xeon Phi™ プロセッサー上でベクトル化のクイック解析を実行して表示する方法を示しました。この手順に従って、最小限の労力でベクトル化の基本情報を簡単に表示することができます。(最適化プロセスの異なるステージでこれらの解析を複数回実行する場合) 最適化ステップの概要を示す図表を作成することもできます。特徴やアセンブリー・コードなどの詳細な解析情報が必要な場合は、別のマシンで (プロジェクト・フォルダーをそのマシンにコピーするか、ネットワークを介してそのマシンにアクセスして) GUI から、インテル® Advisor によって提供されるすべての情報を利用します。

例えば、上記の調査解析を実行すると、インテル® Advisor の GUI は次のようになります。CLI レポートに含まれる情報に加えて、GUI では、特徴、ソースおよびアセンブリー・コードなどのその他の情報も確認できます。



詳細な情報の収集

Survey（調査）解析でパフォーマンスの概要を確認したら、その他のオプションを使用して特定の情報をレポートに追加できます。例えば、Tripcounts（トリップカウント）解析を実行すると、ループの反復回数が得られます。

この情報をプロジェクトに追加するには、調査解析を行ったプロジェクトで tripcounts 解析を実行します。

```
$ advixe-cl -collect tripcounts -project-dir ./AdvProj-Example-AVX512
-search-dir all:=./src -- ./run512 image01.jpg
```

そして、tripcounts レポートを生成します。

```
$ advixe-cl -report tripcounts -project-dir ./AdvProj-Example-AVX512
-format=xml -report-output=./REPORTS/tripcounts-AVX512.xml
```

xml レポートにループの反復回数に関する情報 (Trip_Counts フィールド) が追加されます。調査レポートからの情報は保持されます。以下は、レポートの一部 (最初の最も時間のかかるループのみ) を抜粋したものです。

```
(…)
  </function_call_site_or_loop>
  <function_call_site_or_loop ID="4"
   Function_Call_Sites_and_Loops="[child]-[loop in main at
Histogram_Example.cpp:107]"
                                Self_Time="0.070s"
                                Total_Time="0.120s"
                                Type="Vectorized (Body)"
                                Why_No_Vectorization=""
```

```
                                Vector_ISA="AVX512"
                                Compiler_Estimated_Gain="3.37x"
                                Trip_Counts_Average="761670"
                                Trip_Counts_Min="761670"
                                Trip_Counts_Max="761670"
                                Trip_Counts_Call_Count="1"
                                Transformations=""
                                Source_Location="Histogram_Example.cpp:107"
                                Module="run512">
```

同様の方法で、その他のレポートを生成し、ループに関する有用な情報を得ることができます。利用可能な収集とレポートの種類は、インテル® Advisor のコマンドライン・オプション `-help collect` と `-help report` で確認できます。

```
$ advixe-cl -help collect
Intel(R) Advisor Command Line Tool
Copyright (C) 2009-2016 Intel Corporation.All rights reserved.

-c, -collect=<string>         Collect specified data.Specifying --search-dir
                              when collecting data is strongly recommended.

Usage: advixe-cl -collect=<string> [-action-option] [-global-option] [--]
        <target> [<target options>]

        <string> is one of the following analysis types to perform on
<target>:

            survey        - Explore where to add efficient vectorization
and/or threading.
            dependencies  - Identify and explore loop-carried dependencies
for marked loops.
            map           - Identify and explore complex memory accesses for
marked loops.
            suitability   - Analyze the annotated program to check its
predicted parallel performance.
            tripcounts    - Find how many iterations are executed.

$ advixe-cl -help report
Intel(R) Advisor Command Line Tool
Copyright (C) 2009-2016 Intel Corporation.All rights reserved.

-R, -report=<string>          Report the results that were previously
gathered.

Generates a formatted data report with the specified type and action options.

 Usage: advixe-cl -report=<string> [-action-option] [-global-option] [--]
        <target> [<target options>]

        <string> is the list of available reports:

            survey        - shows results of the survey analysis
            annotations   - lists the annotations in the sources
            dependencies  - shows possible dependencies
            hotspots      -
            issues        -
```

```
map           - reports memory access patterns
suitability   - shows possible performance gains
summary       - shows the collection summary
threads       - shows the list of threads
top-down      - shows the report in a top-down view
tripcounts    - shows survey report with tripcounts data added
```

例えば、ソースコードのメモリー・アクセス・パターンの詳細を得るには、map オプションを使用して Memory Access Patterns (メモリー・アクセス・パターン: MAP) 解析を実行します。

```
$ advixe-cl -collect map -project-dir ./AdvProj-Example-AVX512 -search-dir
all:=./src -- ./run512 image01.jpg

$ advixe-cl -report map -project-dir ./AdvProj-Example-AVX512 -format=xml
-report-output=./REPORTS/map-AVX512.xml
```

上記のすべてのケースでは、プロジェクト・ディレクトリー (この例では AdvProj-Example-AVX512) に GUI で完全な解析を実行するのに必要な情報が含まれています。GUI を使用する場合は、前述のように、プロジェクト・ディレクトリーをワークステーションやラップトップなどの別のマシンにコピーして (または、ファイルシステムを利用してそのマシンにアクセスして)、そのマシンでインテル® Advisor の GUI を実行します。

まとめ

この記事では、インテル® Advisor 2017 を使用してベクトル化のパフォーマンスを素早く簡単に調査する方法を示しました。インテル® Advisor の CLI を利用することで、最初にクイック解析を実行し、結果をインテル® Xeon Phi™ プロセッサーのテキストウィンドウで確認できます。そして、インテル® Advisor の GUI インターフェイスを利用して、コードに関するより詳細な情報を取得できます。

この手順は、ソースコードの最適化を複数回行った後、パフォーマンスに関する情報をまとめるのにも役立ちます。Unix* スクリプトなどを使用することで、異なるレポートから情報を収集し、素早く図表にまとめることができます。

参考文献 (英語)

Zhang, B. (2016). "Guide to Automatic Vectorization With Intel AVX-512 Instructions in Knights Landing Processors."

コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。