Intel Software tools: Advisor, VTune™, ITAC

Presenter: Heinrich Bockhorst Intel FZ Jülich, June 6th 2016
Agenda

• Advisor XE: enable and improve Vectorization

• VTune™ Amplifier XE: single node performance analysis

• Intel Trace Analyzer and Collector (ITAC): MPI analysis
Vectorization Advisor: getting started
Before you analyze
Run GUI or Command Line

Set-up environment

- Linux: source <install-dir>/advixe-vars.sh

Run GUI or Command Line:

- advixe-gui
- advixe-cl –collect survey –project-dir C:\myadvisor mult.exe
Intel Compiler 16 or 17 – enables more data in Survey

Min compilation switches (ICC example):

-O2 -g -xHost

More aggressive optimization: -O3
GUI: Before you analyze

Create Project

- File->New->Project
Analyze what loops you are spending your time in and how they have been vectorized.
Same approach for trip counts

### Trip Counts

<table>
<thead>
<tr>
<th>Median</th>
<th>Min</th>
<th>Max</th>
<th>Call Count</th>
<th>Iteration Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>50</td>
<td>101000000</td>
<td>&lt; 0.0001s</td>
</tr>
<tr>
<td>101</td>
<td>101</td>
<td>101</td>
<td>1000000</td>
<td>&lt; 0.0001s</td>
</tr>
<tr>
<td>1000000</td>
<td>1000000</td>
<td>1</td>
<td>1000000</td>
<td>&lt; 0.0001s</td>
</tr>
</tbody>
</table>

### Loops

<table>
<thead>
<tr>
<th>Loops</th>
<th>Vector Issues</th>
<th>Self Time</th>
<th>Total Time</th>
<th>Trip Counts</th>
<th>Loop Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="loop1.png" alt="loop at Multiply.c:55 in matvec" /></td>
<td><img src="assumed1.png" alt="1 Assumed de..." /></td>
<td>14.030s</td>
<td>14.030s</td>
<td>50 50 50 101000000 &lt; 0.0001s</td>
<td>Scalar</td>
</tr>
<tr>
<td><img src="loop2.png" alt="loop at Multiply.c:44 in matvec" /></td>
<td>![assumed2.png]</td>
<td>0.985s</td>
<td>15.015s</td>
<td>101 101 101 1000000 &lt; 0.0001s</td>
<td>Scalar</td>
</tr>
<tr>
<td><img src="loop3.png" alt="loop at Driver.c:145 in main" /></td>
<td>![assumed3.png]</td>
<td>0.000s</td>
<td>15.035s</td>
<td>1000000 1000000 1000000 1 &lt; 0.0001s</td>
<td>Scalar</td>
</tr>
</tbody>
</table>
Specify loops for deeper analysis
Deeper analysis

Check dependencies

We marked 3 loops for a dependency analysis. Two of the loops had no dependencies. One of the loops has Read-After-Write dependency and can't be vectorized.
Deeper analysis
Memory Access Pattern analysis

1. Survey Target
Explore where to add efficient vectorization and/or threading.

1.1 Find Trip Counts
Find how many iterations are executed.

Mark Loops for Deeper Analysis
Select loops in the Survey result for Correctness and/or Memory Access Patterns analysis.

2.1 Check Correctness
Identify and explore loop-carried dependencies for marked loops. Fix the reported problems.

2.2 Check Memory Access Patterns
Identify and explore complex memory accesses for marked loops. Fix the reported problems.

Stride distribution

Check memory access patterns in your application:

- Summary
- Survey Report
- Refinement Reports
- Animation Report
- Suitability Report

<table>
<thead>
<tr>
<th>Site Location</th>
<th>Loop-Carried Dependencies</th>
<th>Strides Distribution</th>
<th>Access Pattern</th>
<th>Site Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>loop at Driver.c:145 in main</td>
<td>No dependencies found</td>
<td>100% / 0% / 0%</td>
<td>All unit strides</td>
<td>loop_site_6</td>
</tr>
<tr>
<td>loop at Multiply.c:44 in matvec</td>
<td>No dependencies found</td>
<td>85% / 15% / 0%</td>
<td>Mixed strides</td>
<td>loop_site_10</td>
</tr>
<tr>
<td>loop at Multiply.c:55 in matvec</td>
<td>RAW:1</td>
<td>74% / 26% / 0%</td>
<td>Mixed strides</td>
<td>loop_site_8</td>
</tr>
</tbody>
</table>

Memory Access Patterns Report

<table>
<thead>
<tr>
<th>ID</th>
<th>Stride</th>
<th>Type</th>
<th>Source</th>
<th>Nested Function</th>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>P3</td>
<td>3</td>
<td>Parallel site information</td>
<td>Driver.c:145</td>
<td>matrix_vector_multiplication_c.exe</td>
<td></td>
</tr>
<tr>
<td>P9</td>
<td>0</td>
<td>Unit stride</td>
<td>Driver.c:157</td>
<td>matrix_vector_multiplication_c.exe</td>
<td></td>
</tr>
<tr>
<td>P10</td>
<td>0</td>
<td>Unit stride</td>
<td>Multiply.c:39</td>
<td>matvec</td>
<td>matrix_vector_multiplication_c.exe</td>
</tr>
<tr>
<td>P12</td>
<td>0</td>
<td>Unit stride</td>
<td>Multiply.c:44</td>
<td>matvec</td>
<td>matrix_vector_multiplication_c.exe</td>
</tr>
<tr>
<td>P14</td>
<td>0</td>
<td>Unit stride</td>
<td>Multiply.c:45</td>
<td>matvec</td>
<td>matrix_vector_multiplication_c.exe</td>
</tr>
</tbody>
</table>

43-44 int i, j;
45  for (i = 0; i < size1; i++) {
46    b[i] = 0;

Click Collect
VTune™ Amplifier XE: getting started
Creating a Project
GUI Layout

1. Open the project creation dialog by clicking on "Project..." or using the shortcut CTRL+SHIFT+N.

2. Enter the project name "Tachyon1" and set the location "C:\IPS_2013XE_Course\18_Amplifier_Generics\Labs_Windows".

3. Select the "Launch Application" and specify the application path "C:\TEMP\analyze_locks.exe".
Selecting type of data collection

GUI Layout

- All available analysis types
- Different ways to start the analysis
- Helps creating new analysis types
- Copy the command line to clipboard
Summary View

GUI Layout

Clicking on the Summary tab shows a high level summary of the run

Timing for the whole application run

List of 5 Hotspot functions

CPU Usage
Bottom-Up View

GUI Layout

- Menu and Tool bars
- Analysis Type
- Viewpoint currently being used
- Tabs within each result
- Grid area
- Stack Pane
- Filter area
- Timeline area
- Current grouping
Result Analysis

GUI Concepts

Groupings

- Each analysis type has many viewpoints
- Each viewpoint has pre-defined groupings
- Allows you to analyze the data in different hierarchies and granularities
Viewpoints and Groupings

For example, pre-defined groupings can be used to determine load imbalance.
ITAC: getting started
Intel® Trace Analyzer and Collector

ITAC may be applied without touching the program or environment. One way to get a first trace is:

```bash
$ mpirun -trace -n <nprocs> ./test.x
```

Alternatively, just set the preload library and run without the `-trace` flag:

```bash
$ export LD_PRELOAD=libVT.so
$ mpirun -f <hostfile> -n <nprocs> ./test.x
```

this is actually what the flag does internally. This methodology may be applied to situations with complex run scripts not knowing where the mpirun is actually executed.

Note: this does not work for statically linked Intel® MPI (not recommended).
Viewing the trace file

ITAC will generate several files inside the directory where you started mpirun. Just start traceanalyzer in this directory:

```
$ traceanalyzer test.x.stf
```

Alternatively there is a Windows version of traceanalyzer contained in the Linux ICS package.
**ITAC Function Profile**

After starting ITAC a window showing a basic timing profile for MPI and Application will be displayed. Right click on the red **MPI** bar to show the profiling for each used MPI routine:

```plaintext
ungroup MPI
```
ITAC Event Timeline

Most important view of ITAC is the Event Timeline. This shows the temporal development of MPI routines and messages:

Charts -> Event Timeline
ITAC MPI Correctness Checker

Correctness Checker validates MPI correctness. It uses another library but may be started like the ordinary ITAC:

```
$ mpirun -check -n <nprocs> ./test.x
```

or

```
$ export LD_PRELOAD=libVTmc.so
$ mpirun -n <nprocs> ./test.x
```
Vectorization Advisor: Backup
Snapshot concept

User makes a snapshot
Command Line: Intel® Advisor XE

Collect

advixe-cl --collect STEP --project-dir PROJ EXE

Report

advixe-cl --report STEP --project-dir PROJ

Where PROJ is your Intel Advisor XE project

STEP is the specific type of report or collection you are trying to run. Could be: survey, tripcounts, map, dependencies

Need helps? -

advixe-cl --help
advixe-cl --help report
Collecting survey and tripcounts

```
adviXE-cL -collect survey -project-dir ./advi - mult.exe
adviXE-cL -collect tripcounts -project-dir ./advi mult.exe
```

Creating snapshot in command line, e.g:

```
adviXE-cL --snapshot --project-dir ./advi --pack --cache-sources --
cache-binaries -- /tmp/new_snapshot
```

Viewing the results

```
adviXE-gui ./advi
adviXE-cL --report survey --project-dir ./advi
```
Important notes

Reporting (exporting) spreadsheet data to csv/xml/txt: example

advixe-cl --report survey -show-all-columns -format=csv --project-dir ./advi

- Csv file will be created automatically and location will be reported

Enabling experimental features (e.g. FLOPs)

$ export ADVIXE_EXPERIMENTAL=FLOPS
Running Intel Advisor XE on a cluster

```bash
mpirun -n 10 advixe-cl -collect survey --project-dir ./my_proj ./your_app

mpirun -n R advixe-cl -collect survey --project-dir ./my_proj ./your_app : -np 9 ./your_app
```

Intel MPI-specific:

```bash
mpirun -n N -gtool "advixe-cl --collect STEP C:\myadvisor:R,R,R" mult.exe
```

Where STEP is the type of collection, N is the number of processes and R are ranks you want to run.
Running dependency analysis using the command-line

1) First run a survey to get the ID of the loop you want to analyze
2) Then run
   ```
   advixe-cl --collect dependencies
   --mark-up-list ID --project-dir C:\myadvisor mult.exe
   ```
3) To display the results you can use the GUI or the command-line
   ```
   advixe-gui C:\myadvisor
   ```
Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

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.

Copyright © 2015v, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

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