OPENMP* ANALYSIS IN INTEL® VTUNE™ AMPLIFIER XE

TALKING TO A USER ABOUT OPENMP* PARALLELIZATION EFFICIENCY IN THE LANGUAGE A PROGRAM WAS WRITTEN IN

DMITRY PROHOROV, VTUNE HPC LEAD

Agenda

- VTune Amplifier XE OpenMP* Analysis: answering on customers' questions about performance in the same language a program was written in
  - Concepts, metrics and technology inside
  - VTune Amplifier XE OpenMP Analysis Workflow
- OpenMP analysis for hybrid MPI + OpenMP applications
- Summary

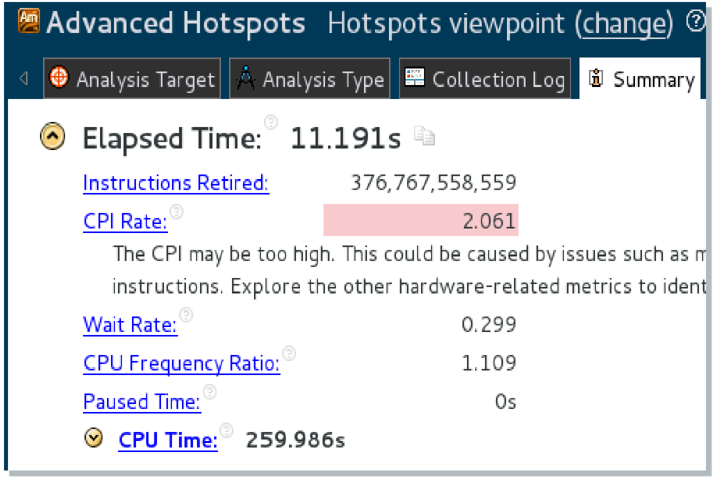
Typical customer questions on parallelization efficiency of OpenMP* applications

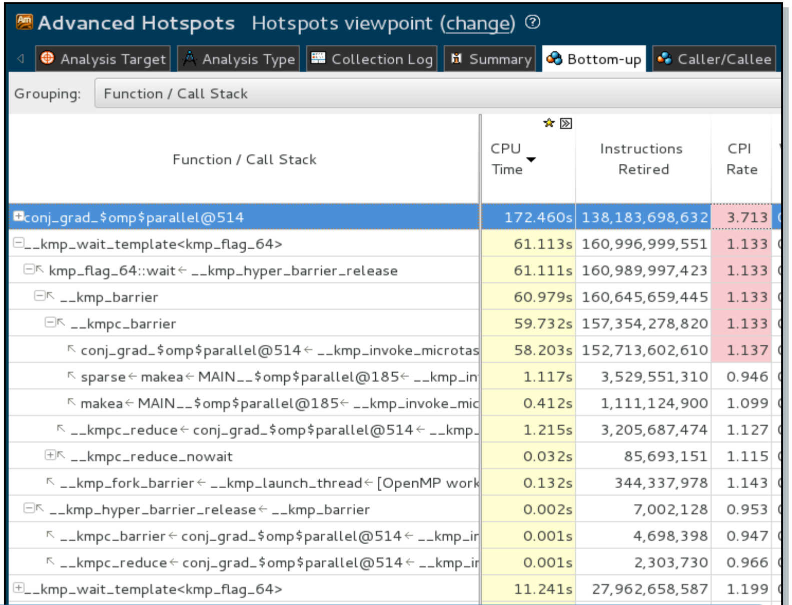
- “I put pragmas but why my speed up is far from linear?
  - Parallelization inefficiency
- “I ran my app on a system with bigger number of cores but it does not run as efficient as on smaller number”
  - Scalability issues

Decomposing the questions

- Is serial time of my application significant to prevent scaling?
- How efficient is my OpenMP parallelization?
  - If not, how much gain can be achieved if I invest in fighting with the inefficiencies?
- What OpenMP regions/loops/barriers are worth to tune?
  - What are the particular problems with them?

If Performance Information is OpenMP "Unaware"..



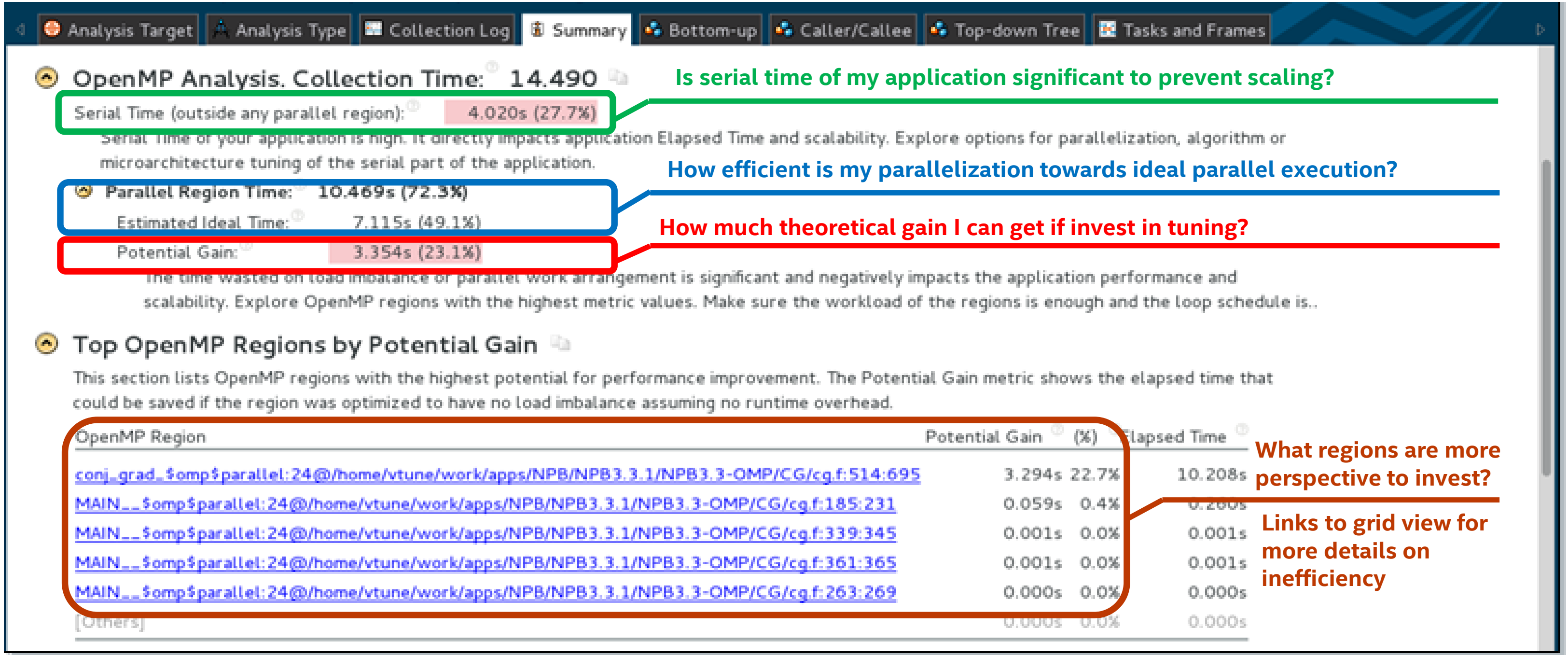


OpenMP
Difficult
runtime
that this

The questions are tied to OpenMP program structure – #pragmas
Answers should be given the same way to be understandable and actionable

Optim

VTune Amplifier XE OpenMP* Analysis: answering on customers' questions about performance in the same language a program was written in
Overview on summary pane



Is serial time of my application significant to prevent scaling?

How efficient is my parallelization towards ideal parallel execution?

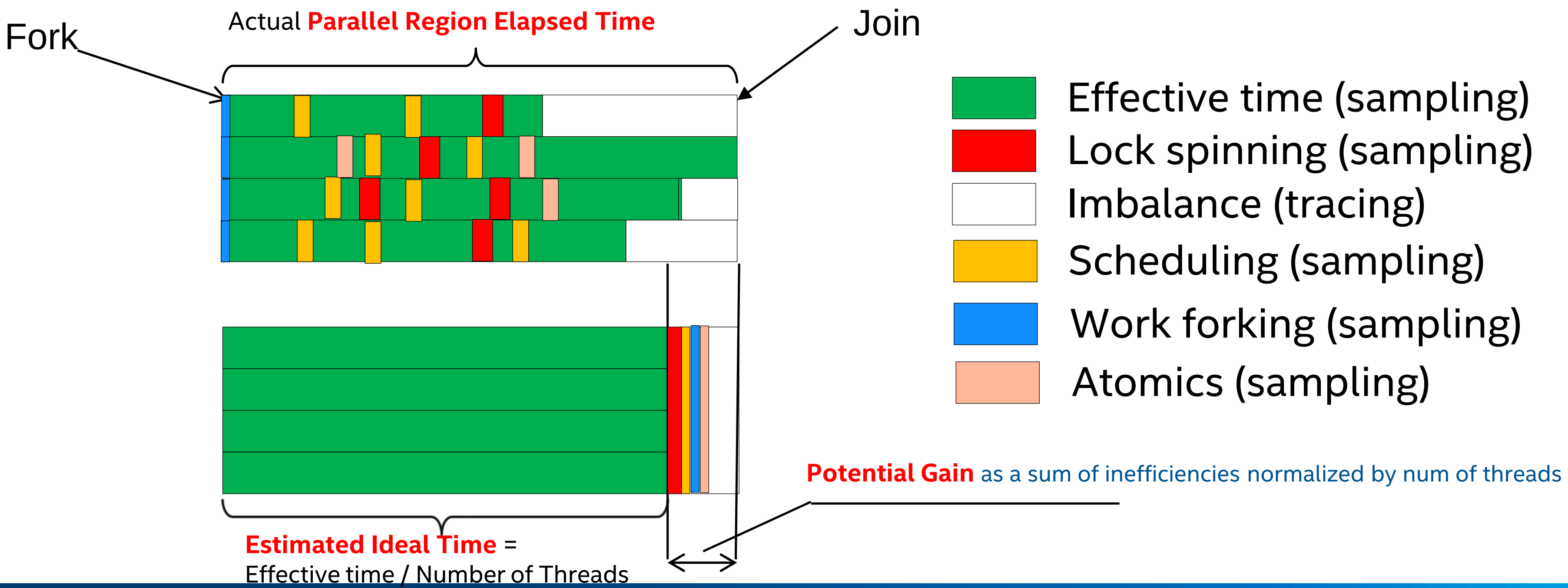
How much theoretical gain I can get if invest in tuning?

What regions are more perspective to invest?

Links to grid view for more details on inefficiency

Key of OpenMP awareness in VTune – Region based views and metrics

Definition of Region **Potential Gain** (elapsed time metric)



Technology under VTune Amplifier XE OpenMP Analysis

Tracing of OpenMP constructions to provide region/work sharing context and precise imbalance on barriers

- Provided to VTune by Intel OpenMP Runtime under profiling
  - Fork-Join points of parallel regions with number of working threads (Intel Compiler 14 and later)
  - OpenMP construct barrier points with imbalance info and OpenMP loop metadata
  - -parallel-source-info=2 compiler option to embed source file name to a region name
- Looking at transition to OMPT, working with John M.-C. on interface enrichments for low overhead analysis

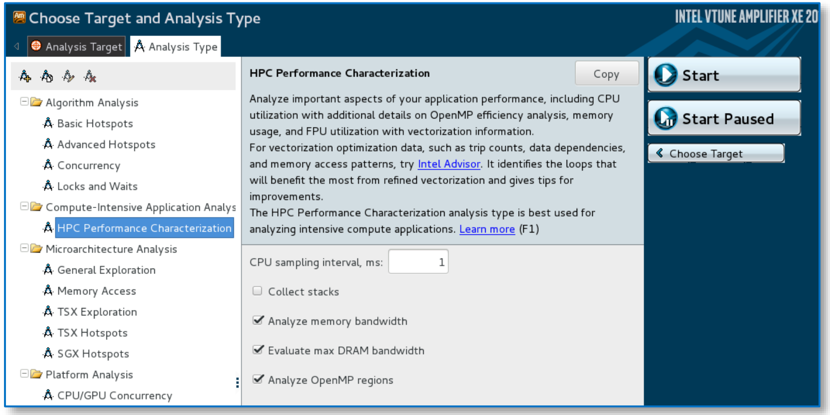
Sampling to define and classify CPU time - user's code and OpenMP RTL work

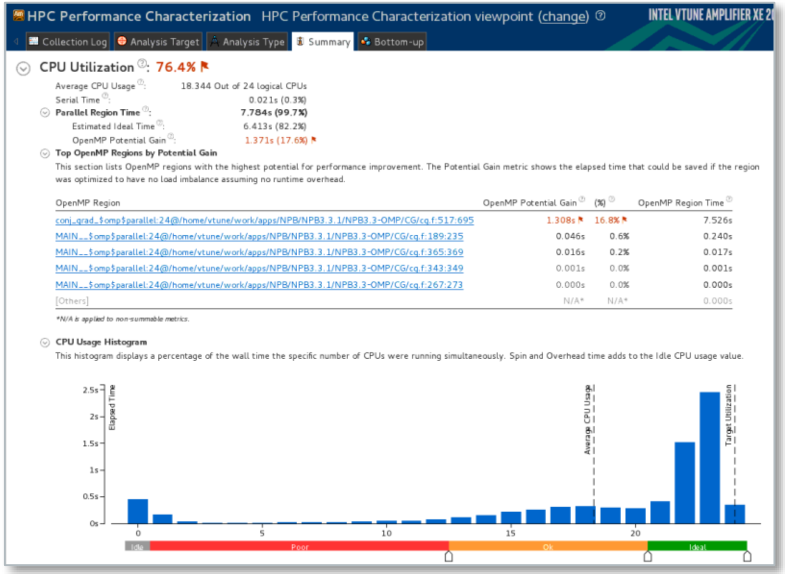
  - Classification: Locking, Scheduling, Work Forking

VTune Amplifier XE OpenMP Analysis Workflow

Start with HPC Performance Characterization analysis

Explore CPU Utilization aspect metrics related to OpenMP in summary, grid, source view





CL: *>amplxe-cl –collect hpc-performance <my_app>*

Per Region Details in grid view: inefficiencies in **wall time** - classification and issue highlighting

Advanced Hotspots Hotspots viewpoint (change) Intel VTune Amplifier XE 20

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Platform

Imbalance on a loop barrier

Grouping: OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack

OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack	OpenMP Potential Gain: Imbalance	Lock Con..	Creation	Schedu...	Redu...	Atomi...	Other	Elapsed Time	Number of OpenMP threads	Ins... Cou..	OpenMP Loop Schedule Type	OpenMP Loop Chunk	Avg OpenMP Loop Iteration Count
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	3.944s	0s	0.000s	0.002s	0.000s	0s	0.094s	11.095s	24	76			
conj_grad_omploop_barrier_segment@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:580	3.725s	0s	0s	0.000s	0s	0s	0.006s	10.445s	24		Static	3125	75,000
conj_grad_omploop_barrier_segment@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:683	0.149s	0s	0s	0s	0s	0s	0.004s	0.418s	24		Static	3125	75,000
conj_grad_omploop_barrier_segment@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:625	0.033s	0s	0s	0.002s	0.000s	0s	0.002s	0.068s	24		Static	3125	75,000
conj_grad_omploop_barrier_segment@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:650	0.015s	0s	0s	0.000s	0s	0s	0.001s	0.064s	24		Static	3125	75,000
conj_grad_omploop_barrier_segment@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:664	0.014s	0s	0s	0.000s	0s	0s	0.001s	0.079s	24		Static	3125	75,000

Advanced Hotspots Hotspots viewpoint (change) Intel VTune Amplifier XE 2015

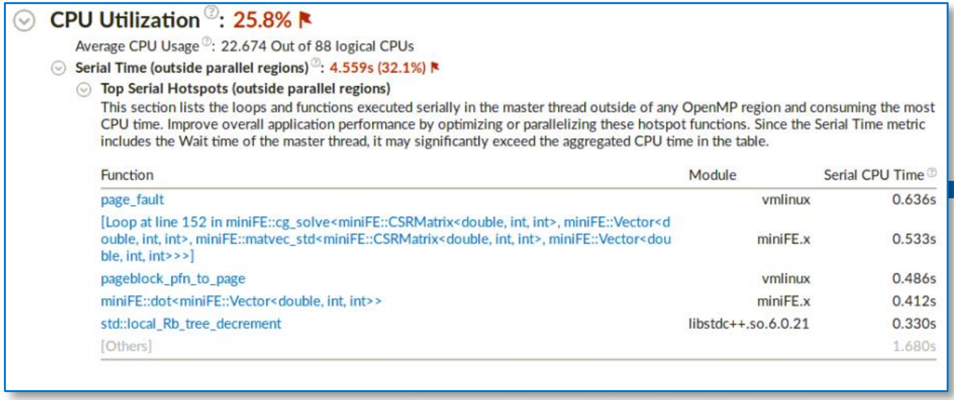
Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Tasks and Frames

Grouping: OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack

OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack	OpenMP Potential Gain: Imba...	Lock Con...	Cre...	Scheduling	Red..	Oth.	OpenMP Potential Gain (% of Collection T...): Imba... (%)	Lock Cont...	Cre.. (%)	Scheduli... (%)	Red.. (%)	Oth.. (%)	Elap... Time	Nu. of Ope. thr..	Ins. Co.	Ope. Loo.. Chu.	Open... Loop Sched... Type
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	0.20...	0.0...	0.0...	3.127s	0.0..	0.0.	1.7%	0.0%	0.0%	25.9%	0.0%	0.0%	11.7...	24	76		
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:572:580	0.00...	0.0...	0s	3.125s	0s	0.0.	0.1%	0.0%	0.0%	25.9%	0.0%	0.0%	11.1...	24		1	Dynamic
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:675:683	0.12...	0s	0s	0s	0s	0.0.	1.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.41...	24		312.	Static
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:621:625	0.02...	0s	0s	0.001s	0.0..	0.0.	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.07...	24		312.	Static
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:637:650	0.02...	0s	0s	0.000s	0.0..	0.0.	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.07...	24		312.	Static

Dynamic scheduling overhead on a parallel loop

Details in Grid View: **Serial** Time **Hotspots**



CPU Utilization: 25.8%

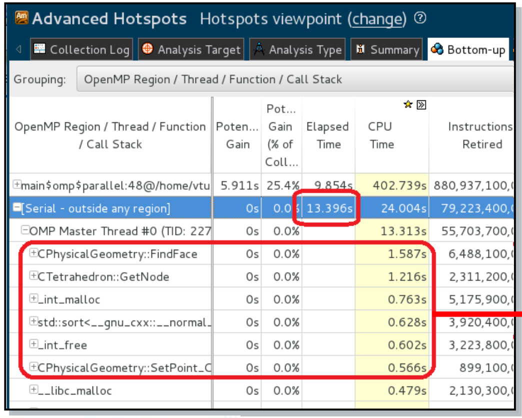
Average CPU Usage: 22.674 Out of 88 logical CPUs

Serial Time (outside parallel regions): 4.559s (32.1%)

Top Serial Hotspots (outside parallel regions)

This section lists the loops and functions executed serially in the master thread outside of any OpenMP region and consuming the most CPU time. Improve overall application performance by optimizing or parallelizing these hotspot functions. Since the Serial Time metric includes the Wait time of the master thread, it may significantly exceed the aggregated CPU time in the table.

Function	Module	Serial CPU Time
page_fault	vmlinux	0.636s
[Loop at line 152 in miniFE::cg_solve<miniFE::CSRMatrix<double, int, int>, miniFE::Vector<double, int, int>, miniFE::matvec_std<miniFE::CSRMatrix<double, int, int>, miniFE::Vector<double, int, int>>>]	miniFE.x	0.533s
pageblock_pfn_to_page	vmlinux	0.486s
miniFE::dot<miniFE::Vector<double, int, int>>	miniFE.x	0.412s
std::local_Rb_tree_decrement	libstdc++.so.6.0.21	0.330s
[Others]		1.680s



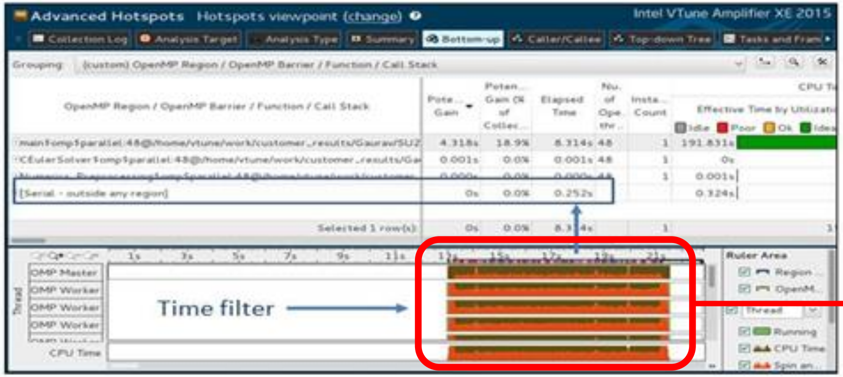
Advanced Hotspots Hotspots viewpoint (change)

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up

Grouping: OpenMP Region / Thread / Function / Call Stack

OpenMP Region / Thread / Function / Call Stack	Poten... Gain	Pot... Gain (% of Coll...	Elapsed Time	CPU Time	Instructions Retired
mainompparallel:48@/home/vtu	5.911s	25.4%	9.854s	402.739s	880,937,100,
[Serial - outside any region]	0s	0.0%	13.396s	24.004s	79,223,400,
OMP Master Thread #0 (TID: 227	0s	0.0%		13.313s	55,703,700,
CPhysicalGeometry::FindFace	0s	0.0%		1.587s	6,488,100,
CTetrahedron::GetNode	0s	0.0%		1.216s	2,311,200,
_int_malloc	0s	0.0%		0.763s	5,175,900,
std::sort<__gnu_cxx::__normal_	0s	0.0%		0.628s	3,920,400,
_int_free	0s	0.0%		0.602s	3,223,800,
CPhysicalGeometry::SetPoint_C	0s	0.0%		0.566s	899,100,
__libc_malloc	0s	0.0%		0.479s	2,130,300,

Serial hotspots under Master Thread

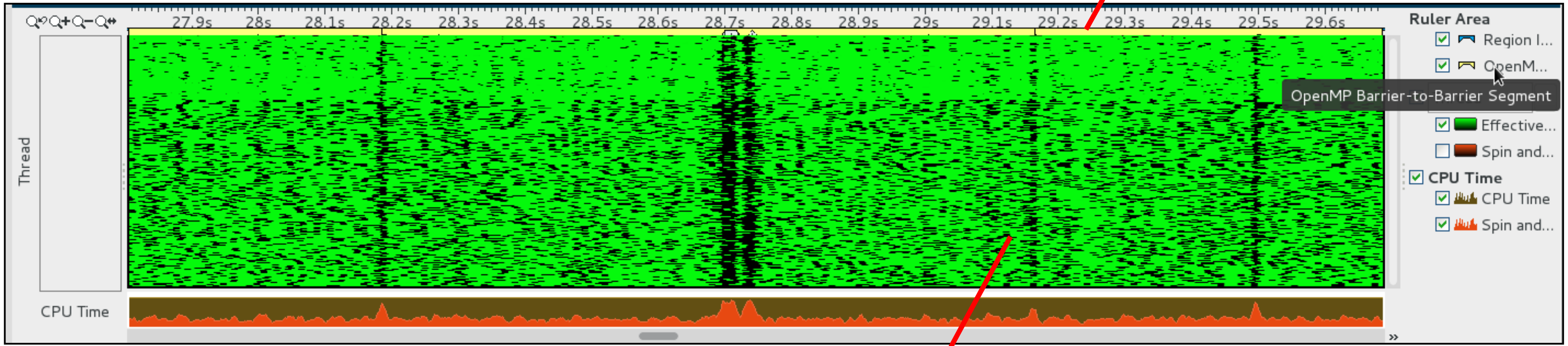


Time Filter to exclude initialization phase

Details on Scalable **Timeline**

Super tiny timeline display mode – a bird-eye's view having all data without scrolling

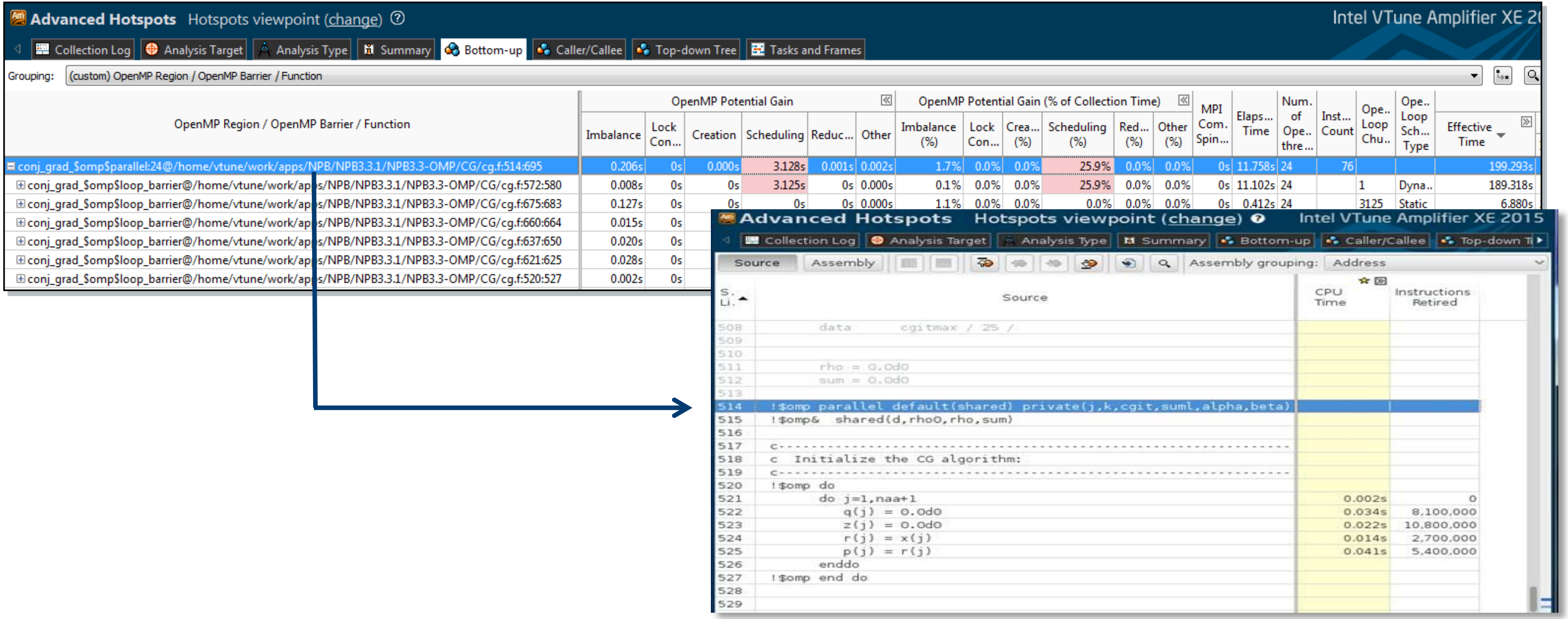
Region frames on the ruler



More green color – more efficient multithreaded execution

Intel® Xeon Phi™ profiling result with 288 threads

Details for a Region at **source** file **level**



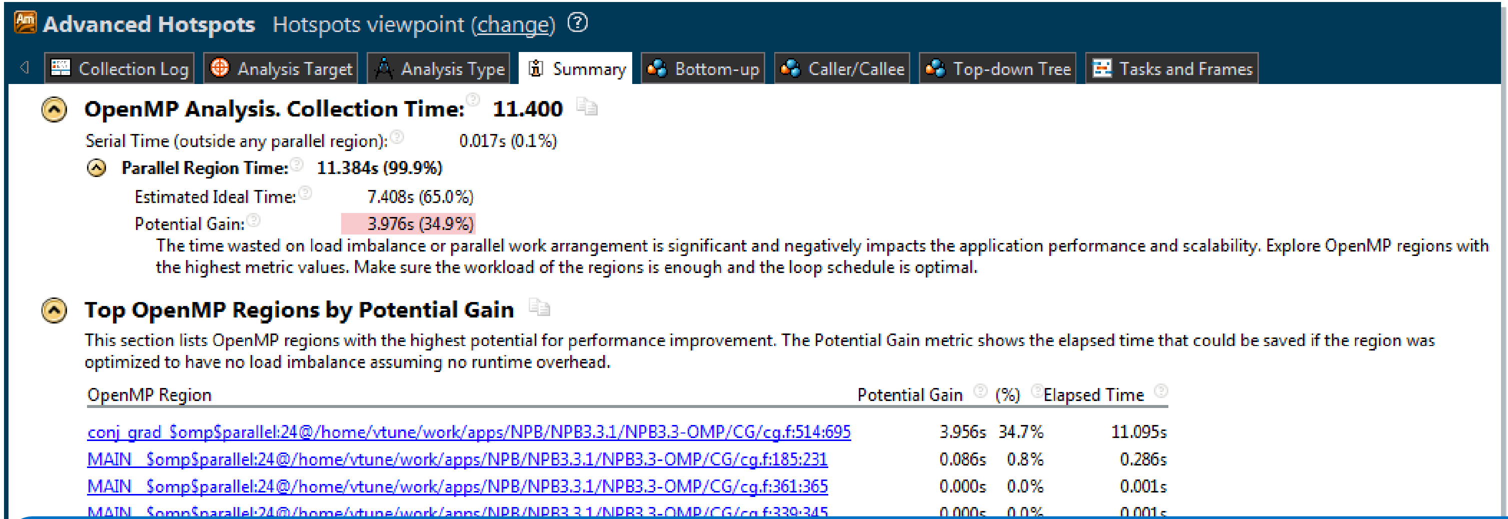
Summary

- VTune Amplifier XE OpenMP analysis answers on customer's questions about performance on the language of OpenMP constructs
- The analysis is well-scalable for many-core systems with good balance of tracing and sampling collection technologies
- The OpenMP analysis is "MPI-aware" that is helpful for inner-node hybrid MPI + OpenMP application tuning
- The full feature set is available in VTune Amplifier XE 2018 with Intel OpenMP and Intel MPI runtimes as a part of Intel® Parallel Studio XE 2018

Back-up

A Use Case: NPB CG imbalance improvement

- Step 1: Profiling original application – NPB CG (Class B)



There is a region with promising potential gain – go to Grid View for more details

A Use Case: NPB CG imbalance improvement

- Step 1: Profiling original application – NPB CG (Class B)
  - There are barriers in region – use experimental “/OpenMP Region/OpenMP Barrier..” grouping

Advanced Hotspots Hotspots viewpoint (change) Intel VTune Amplifier XE

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Tasks and Frames

Grouping: (custom) OpenMP Region / OpenMP Barrier / Function / Call Stack

OpenMP Region / OpenMP Barrier / Function / Call Stack	Potential Gain	Poten... Gain (% of Collec...	Elaps... Time	Ope.. Loop Chu..	Ope.. Loop Sch... Type	CPU Time: Effective Time	CPU Time: Spin Time: Imbalance ...	CPU Time: Spin Time: Lock Contention...	CPU Time: Spin Time: Other	CPU Time: Overhead Ti: Creation...	CPU Time: Overhead Ti: Scheduling ...
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	3.956s	34.7%	11.095s			172.963s	92.007s	0s	0.212s	0.001s	0.048s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:572:580	3.734s	32.8%	10.445s	3125	Static	163.287s	86.109s	0s	0.198s	0s	0.002s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:675:683	0.150s	1.3%	0.418s	3125	Static	6.528s	3.451s	0s	0.007s	0s	0s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:621:625	0.036s	0.3%	0.068s	3125	Static	0.446s	1.128s	0s	0.004s	0s	0.042s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:637:650	0.016s	0.1%	0.064s	3125	Static	1.044s	0.527s	0s	0.001s	0s	0.001s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:660:664	0.014s	0.1%	0.079s	3125	Static	1.420s	0.570s	0s	0s	0s	0.001s
conj_grad_ompbarrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:555	0.004s	0.0%	0.009s			0.056s	0.119s	0s	0.001s	0s	0s

- Imbalance on omp loop in cg.f, lines: 572 - 580, schedule is static

```
571
572   !$omp do
573            do j=1,lastrow-firstrow+1
574               suml = 0.d0
575               do k=rowstr(j),rowstr(j+1)-1
576                  suml = suml + a(k)*p(colidx(k))
577               enddo
578               q(j) = suml
579            enddo
580   !$omp end do
```

A Use Case: NPB CG imbalance improvement

- Step 2: Trying dynamic scheduling omp do schedule (dynamic)

Advanced Hotspots Hotspots viewpoint (change)

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Tasks and Frames

OpenMP Analysis. Collection Time: 12.063

Serial Time (outside any parallel region): 0.018s (0.1%)

Parallel Region Time: 12.045s (99.9%)

Estimated Ideal Time: 8.515s (70.6%)

Potential Gain: 3.530s (29.3%)

The time wasted on load imbalance or parallel work arrangement is significant and negatively impacts the application performance and scalability. Explore OpenMP regions with the highest metric values. Make sure the workload of the regions is enough and the loop schedule is optimal.

Top OpenMP Regions by Potential Gain

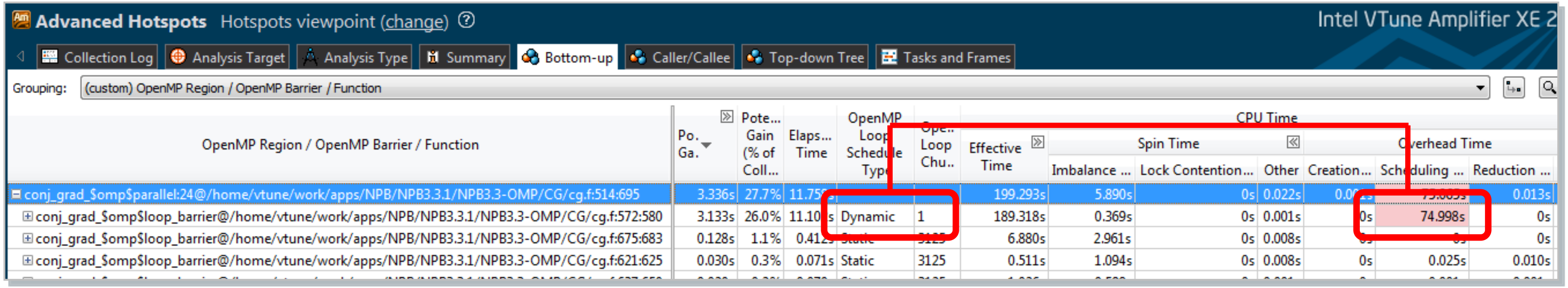
This section lists OpenMP regions with the highest potential for performance improvement. The Potential Gain metric shows the elapsed time that could be saved if the region was optimized to have no load imbalance assuming no runtime overhead.

OpenMP Region	Potential Gain	(%)	Elapsed Time
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	3.336s	27.7%	11.758s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:185:231	0.075s	0.6%	0.285s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:339:345	0.000s	0.0%	0.001s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:361:365	0.000s	0.0%	0.001s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:263:269	0.000s	0.0%	0.000s
[Others]	0.000s	0.0%	0.000s

Elapsed time increased – no improvement
Go to Grid View for details

A Use Case: NPB CG imbalance improvement

- Step 2: Trying dynamic scheduling “omp do schedule (dynamic)”



OpenMP Region / OpenMP Barrier / Function	Po. Ga.	Pote... Gain (% of Coll...	Elaps... Time	OpenMP Loop Schedule Type	Ope.. Loop Chu..	CPU Time: Effective Time	Spin Time: Imbalance ...	Spin Time: Lock Contention...	Spin Time: Other	Overhead Time: Creation...	Overhead Time: Scheduling ...	Overhead Time: Reduction ...
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	3.336s	27.7%	[illegible]			199.293s	5.890s	0s	0.022s	[illegible]	[illegible]	0.013s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:572:580	3.133s	26.0%	[illegible]	Dynamic	1	189.318s	0.369s	0s	0.001s	0s	74.998s	0s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:675:683	0.128s	1.1%	[illegible]	[illegible]	[illegible]	6.880s	2.961s	0s	0.008s	[illegible]	[illegible]	0s
conj_grad_omploop_barrier@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:621:625	0.030s	0.3%	0.071s	Static	3125	0.511s	1.094s	0s	0.008s	0s	0.025s	0.010s

Default chunk size is 1 and it led to scheduling overhead
Let’s try bigger chunk size

A Use Case: NPB CG imbalance improvement

- Step 3: Trying dynamic schedule with chunk 20

Advanced Hotspots Hotspots viewpoint (change)

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Tasks and Frames

OpenMP Analysis. Collection Time: 9.878

Serial Time (outside any parallel region): 0.017s (0.2%)

Parallel Region Time: 9.861s (99.8%)

Estimated Ideal Time: 9.406s (95.2%)

Potential Gain: 0.455s (4.6%)

Top OpenMP Regions by Potential Gain

This section lists OpenMP regions with the highest potential for performance improvement. The Potential Gain metric shows the elapsed time that could be saved if the region was optimized to have no load imbalance assuming no runtime overhead.

OpenMP Region	Potential Gain	(%)	Elapsed Time
conj_grad_ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	0.262s	2.7%	9.568s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:185:231	0.091s	0.9%	0.291s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:361:365	0.001s	0.0%	0.001s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:339:345	0.000s	0.0%	0.001s
MAIN__ompparallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:263:269	0.000s	0.0%	0.000s
[Others]	0.000s	0.0%	0.000s

Improved original elapsed time ~15%, eliminated imbalance

Back-up

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.

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

intel®