



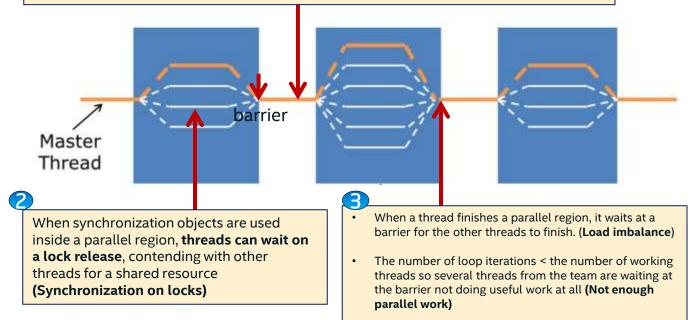
Intel[®] VTune Amplifier XE Analysis of OpenMP applications

Kirill Rogozhin



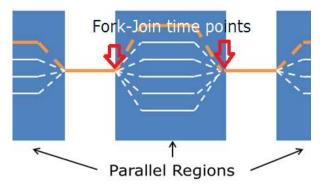
Major reasons why working threads wait

When the master thread is executing a serial region, the worker threads are in the OpenMP runtime waiting for the next parallel region



VTune Amplifier XE OpenMP Analysis

- Tracing of OpenMP is used to provide region/work sharing context
 - Provided to VTune by Intel OpenMP Runtime:
 - Fork-Join time points of parallel regions with number of working threads
 - Overhead of tracing can be substantial- used carefully per region instance on region fork-join points



- Sampling to determine different kinds of overhead, synchronization spinning etc.
 - Any type of VTune analysis that support CPU time calculation (such as hotspots, advancedhotspots with or without stacks, etc.)
 - With Hotspot Viewpoint selected



VTune Amplifier XE OpenMP Analysis

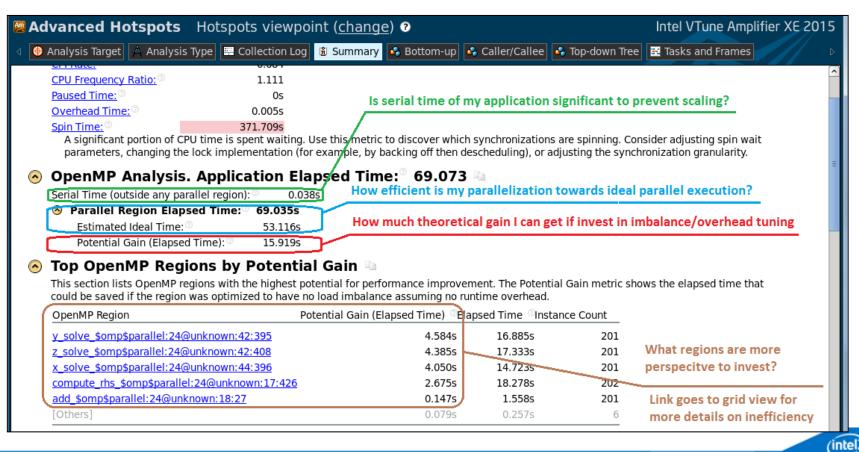
Enhancing OpenMP analysis with a set of metrics to answer the following questions:

- Is serial time of my application significant to prevent scaling?
- How efficient is my OpenMP parallelization?
- How much gain I can take if invest in reducing load imbalance/overhead?
- What regions are more perspective to invest?

Metrics are based on elapsed time improvement possibilities on application wall clock time



VTune Amplifier XE OpenMP Analysis



Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

Definition of metrics

Serial time: time spent by the application outside any OpenMP* region in the master thread during collection: Elapsed time - Σ [Elapsed time of all Parallel regions]

Effective CPU time of a Parallel Region Instance:

([CPU time] - [Spin Time] - [Overhead Time])

where CPU, Spin and Overhead time aggregated by threads in the Region instance

Estimated Ideal time of a Region Instance:

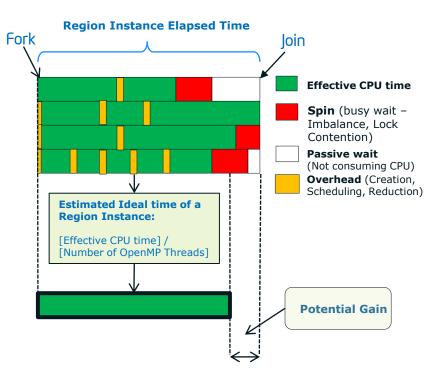
[Effective CPU time] / [Number of Threads]

Potential Gain of a Parallel Region Instance:

[Region Instance Elapsed Time] – [Estimated Ideal Time of the Region Instance]

Potential Gain of a Region: Σ [Potential Gain of all instances of a Region]

Potential Gain of a Program: ∑[Potential Gain of all Regions]



Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

OpenMP region patterns in VTune Amplifier

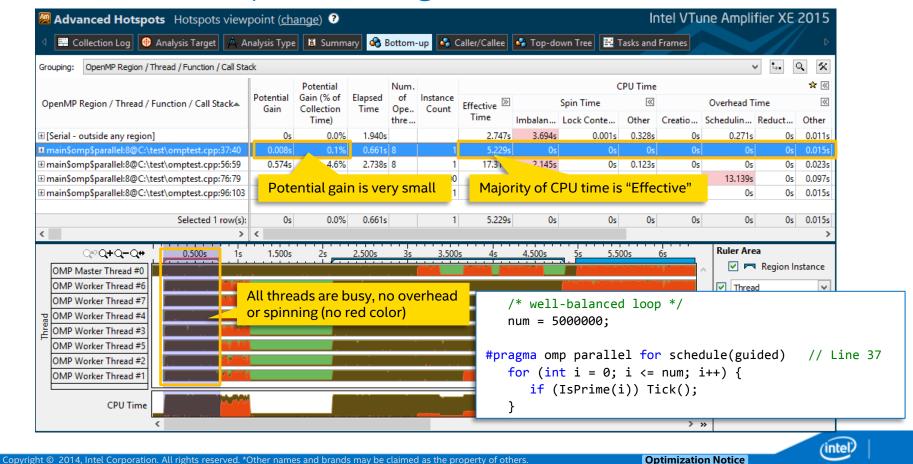
- Serial region
- Well-balanced region
- Imbalanced region
- Region with runtime overhead
- Region with synchronization objects

Serial region

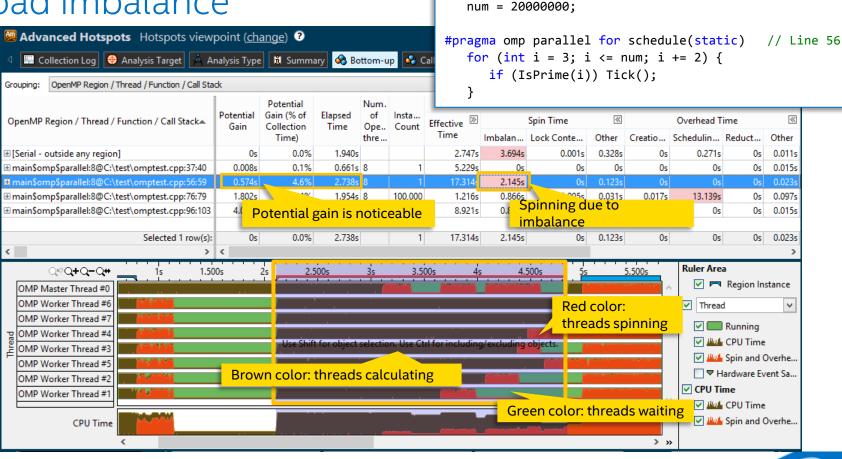
Advanced Hotspots Hotspots view	/point (<u>ch</u>	ange) ?							Ir	itel VTu	ne Ampli [.]	fier XE	2015
🖣 🔛 Collection Log \varTheta Analysis Target 🔥 A	Analysis Type	e 🕅 Summa	ary 🔗 B	ottom-u	ip 🔹 C	Caller/Callee	🔩 Top-do	wn Tree 🛛 🔣 T	asks and	Frames			
Grouping: OpenMP Region / Thread / Function / Call St	ack											/ L+= (2
		Potential		Num. of Ope				CF	U Time +				* /
OpenMP Region / Thread / Function / Call Stack	Potential Gain	Gain (% of Collection	Elapsed Time		Insta Count	Effective 🚿		Spin Time	\ll		Overhead Time		
		Time)	mine	thre		Time		Lock Conte	Other	Creatio	Schedulin	Reduct	Oth
⊞ main\$omp\$parallel:8@C:\test\omptest.cpp:96:	•/-III-		C		1	8.921s	0.836s	19.959s	3.247s	0s	0s	0s	0.01
⊞ main\$omp\$parallel:8@C:\test\omptest.cpp:56:		ck time o	r serial	regio	on 1	17.314s	2.145s	0s	0.123s	0s	0s	0s	0.02
∃main\$omp\$parallel:8@C:\test\omptest.cpp:76:79	1.802s	14.4%		s 8	100,000	1.216s	0.866s	0.005s	0.031s	0.017s	13.139s	0s	0.09
∃[Serial - outside any region]	0s	0.0%	1.940	s		2.747s	3.694s	0.001s	0.328s	0s	0.271s	0s	0.01
OMP Master Thread #0 (TID: 6528)	0s	0.0%				1.880s	0s	0.001s	0.002s	0s	0.040s	0s	0.0
⊞ main Dight c	lick cho	ose "Filt	or in by	ردمام	ction"	1.865s	CP	U time of	serial	region	0s		
			еппр	Sele	CUOIT	0s	CI	o time or	scriat	i cgion	0.040s		
	Os	0.0%				Os			0s				0.00
kmp_allocate_team Selected 1 row(s)	0s : 0s	0.0%				0s 1.880s			0.002s				0.00
< >	< .	0.070				110003	03	0.0013	0.0023	03	0.0403	03	>
Q© Q+ Q−Q# _1s _2s		4s	5s	6	s	7s 8	s 9s	10s	11s	12s	Ruler	Агеа	
OMP Master Thread #0						And Services And	المعادية ومعدد المتعاد	Martin Topical Street Street			₩_	🗖 Regi	on I
OMP Worker Thread #6	N	laster th	read is	com	outing	to Manual Arrows							V
OMP Worker Thread #7					U U		in the second second	- Contraction (March March 19	()weather the states) The	- New International	- I I	nreau	¥
E OMP Worker Thread #4						101.1.1.10.00	here a state of the second					Runn	ing
CMP Worker Thread #4 OMP Worker Thread #3						and a strength of the	a sector and the sector of the sector back and the sector back and the sector back and the sector and the sector of the					Hul- CPU	Time
											_	And Spin	and
CPU Time						W						♥ Hardwa	re E
											> » 🗹 CF	'U Time	

Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

Well-balanced parallel region



Load imbalance



/* imbalanced loop */

Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

Optimization Notice

intel

Runtime overhead

Advanced Hotspots Hotspots viewpoint (change) ?

🞬 Collection Log \varTheta Analysis Target 🛕 Analysis Type 📓 Summary 🚱 Bottom-up

<pre>/* overhead loop</pre>	*/
num = 50000000;	
value=2000;	

}

#pragma omp parallel for schedule (dynamic) // Line 71
for (int i = 3; i <= num; i += 2) {
 if (IsPrime(value)) Tick();</pre>

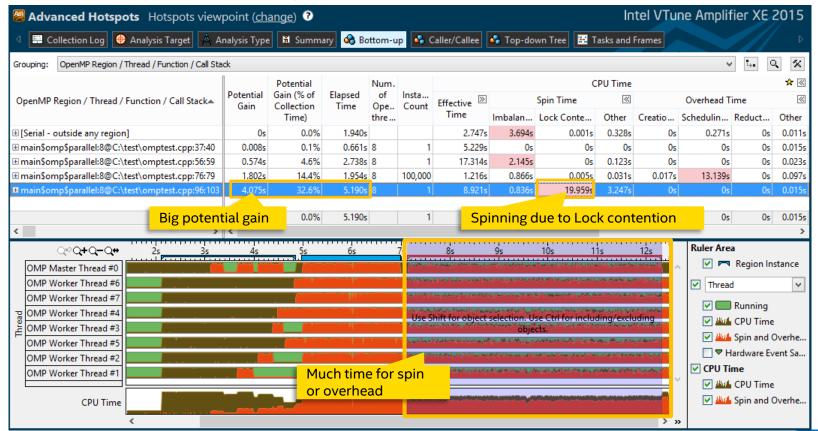
Grouping: OpenMP Region / Function / Call Stack											~	ц. Q	Χ.
		Pote Gain		Num		CPU Time+							☆ 🛛
OpenMP Region / Function / Call Stack	/ Function / Call Stack		Elap Time	of Open	Insta Count	»»		Spin Time	«		Overhead Tir	ne	
		(% of Coll		thre	count		Imbalanc	Lock Conte	Other	Creatio	Schedulin	Red	Other
main\$omp\$parallel:8@C:\test\omptest.cpp:89:96	4.048s	37.5%	5.150s	8	1	8.815s	0.822s	19.894s	3.252s	0s	0s	0s	0.016
⊞main\$omp\$parallel:8@C:\test\omptest.cpp:53:56	0.489s	4.5%	2.634s	8	1	17.158s	1.845s	Os	0.107s	0s	0s	0s	0.021
∃[Serial - outside any region]	0s	0.0%	1.892s			2.713s	3.483s	Os	0.344s	0s	0s	0s	0.002s
⊞main\$omp\$parallel:8@C:\test\omptest.cpp:36:39	0.008s	0.1%	0.661s	8	1	5.225s	Os	Os	Os	0s	Os	0s	0.014
main\$omp\$parallel:8@C:\test\omptest.cpp:71:74	0.421s	3.9%	0.455s	8	1	0.273s	0s	0s	0s	Os	3.351s	0s	0:
Reference and the state of the								Runti	me sch	eduling	overh	ead	
Q≈Q+Q−Q+ <u>4</u> 5	4.500s		, ,	5s		5.324s 5.500s		бs	6.	500s	Ruler Ar	ea	^
OMP Master Thread #0	alma da alma - a	<u> </u>				Region I							
OMP Worker Thread #7				بالإمادين		Region Instan					Thre	ad	¥
ਲ੍ਹ OMP Worker Thread #6		1	1.00.00	*** ~		Start: 4.889s D							
فَ OMP Worker Thread #3			time	for sp	nin and	OpenMP Regi		np\$parallel:8@C:' od	\test\ompte	est.cpp:/i:/		Runnii	-
OMP Worker Thread #4						and whether an	AND A REAL PROPERTY OF	enel ku Marenderan	active Weblew	An Aberlik of A		CPU Time	
OMP Worker Thread #5	0	<mark>r ove</mark>	mea	a				which and the particular	NUMBER OF STREET	AN WARA			
CPU Time	· · · · · ·		-							and the second second		■ ■ Hardware	
		"									CPU 1		
<										>	» 🔽 👑	🖌 CPU T	ime 🗸

(intel)

Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

Optimization Notice

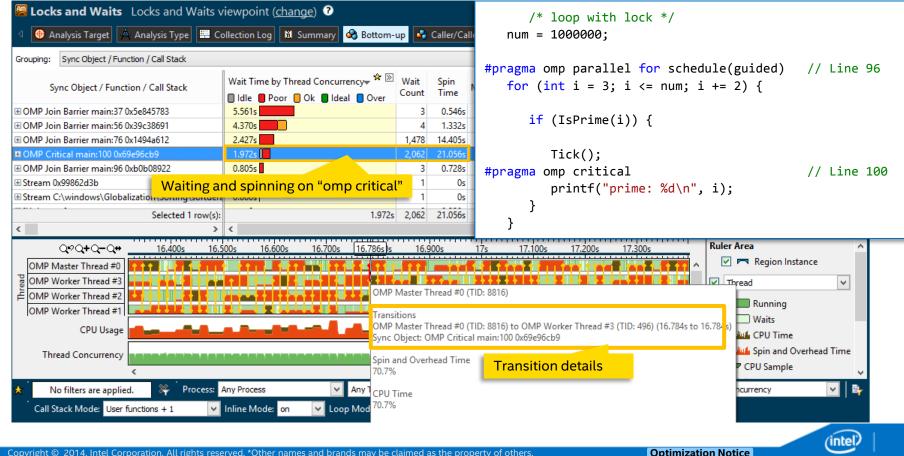
Synchronization objects



Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

Optimization Notice

Synchronization objects – Locks & Waits analysis



Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

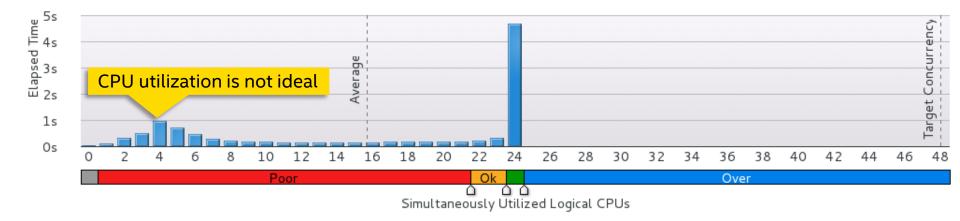
NASA Parallel Benchmark optimization

Setup

- CPU: Intel[®] Xeon[®] processor E5-2697 v2 @ 2.70GHz, 24 cores/48 threads.
- OS: RHEL 7.0 x64
- Compiler: Intel[®] Parallel Studio XE Composer Edition 2015 update 2
- Workload: NPB 3.3.1, "CG Conjugate Gradient, irregular memory access and communication" module, class B.

📀 CPU Usage Histogram 🖻

This histogram displays a percentage of the wall time the specific number of CPUs were running simultaneously. Spin and Overhead time adds to the Idle CPU usage value.



inte

📀 OpenMP Analysis. Collection Time: 🖱 11.400 🗈

Serial Time (outside any parallel region):⁽²⁾ 0.017s (0.1%)

O Parallel Region Time:[®] 11.384s (99.9%)

Estimated Ideal Time: 😳

Potential Gain: ⁽²⁾

7.408s (65.0%) 3.975s (34.9%)

Potential gain is 34.9%

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..

🖻 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 💿 (%	6) [©] Elapsed Time [©]
conj_grad_\$omp\$parallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:514:695	3.958s 34	11.095s
MAIN\$omp\$parallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:185:231	0.086s (0.8% 0.286s
MAIN\$omp\$parallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:361:365	0.000s (0.0% 0.001s
MAIN\$omp\$parallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:339:345	0.000s (0.0% 0.001s
MAIN\$omp\$parallel:24@/home/vtune/work/apps/NPB/NPB3.3.1/NPB3.3-OMP/CG/cg.f:263:269	0.000s (0.000s
[Others]	0.000s (0.000s

Advanced Hotspots Hotspots viewpoint (<u>change</u>)										
\vee 🕮 Collection Log 🕹 Analysis Target 🖄 Analysis Type 🛍 Summary 🔗 Bottom-up 🗳 Caller/Callee 🗳 Top-down Tree 🖽 Tasks and Frames 🖡 cg.f										
Grouping: OpenMP Region / Function / Call Stack										
		Poten		Number		CPU	Time		*	
OpenMP Region / Function / Call Stack	Poten Gain	Gain (% of	Elapsed Time	of OpenMP	Instance Count	Effective Time by Utilizati	on S	5pin [™]	Overhead 📎	
		Colle		threads		📕 Idle 📕 Poor 📙 Ok 📕 Idea	al 🗌 Ov	Time	Time	
<pre>Bconj_grad_\$omp\$parallel:24@/home/vtune/work/apps</pre>	3.958s	34.7%	11.095s	24	76	172.969s	9	92.159s	0.138s	
⊡MAIN\$omp\$parallel:24@/home/vtune/work/apps/N	0.086s	0.8%	0.286s	24	1	4.819s		906s	0s	
€[Serial - outside any region]	Os	0.0%	0.017s			0.045s	Big s	pin ti	me	
⊕MAIN\$omp\$parallel:24@/home/vtune/work/apps/N	0.000s	0.0%	0.001s	24	75	0.004s		0.015s	0.001s	



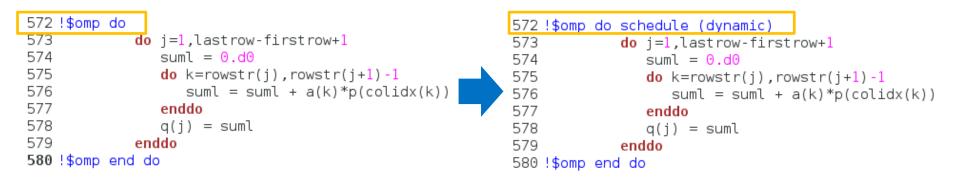
```
514 !$omp parallel default(shared) private(j,k,cgit,suml,alpha,beta)
515 !$omp& shared(d,rho0,rho,sum)
516
517 c-----
518 c Initialize the CG algorithm:
519 c - - - -
520 !$omp do
                       Many "omp do" in the same parallel region
521
       do j=1,naa+1
522
         q(i) = 0.000
523
         z(i) = 0.000
524
         r(i) = x(i)
525
         p(i) = r(i)
526
       enddo
527 !$omp end do
528
529
                  _____
531 c
    rho = r.r
532 c
    Now, obtain the norm of r: First, sum squares of r elements locally...
534 !$omp do reduction(+:rho)
535
       do j=1, lastcol-firstcol+1
536
       rho = rho + r(j)*r(j)
537
       enddo
538 !$omp end do
539
```

(inte

Grouping: (custom) OpenMP Region / OpenMP Barrier Type / OpenMP Barrier / Function / Call Stack											
OpenMP Region / OpenMP		× N		OpenMP	OpenMP	CPU Time					
Barrier Type / OpenMP Barrier / Function / Call	Poten Gain	Elapsed Time			Loop Schedule	Effective Time	⊗ Spi	Spin Time √			
Stack			threads	Chunk	Туре	📕 Idle 📕 Poor 📘	(Imbalance	Lock C	Other		
⊡conj_grad_\$omp\$paralle		11.095c		C	tatic ccho	172.969s	91.947s	Os	0.212s		
ELoop barriers	Per-Da		akdown	5	tatic sche		91.825s	Os	0.211s		
	_ 3.734s	10.445s	24	3125	Static	163.287s	86.096s	Os	0.198s		
⊕conj_grad_\$omp\$loop	- Hotte	st loop is	s on line 5	572	Static	6.528s	149s	Os	0.007s		
⊕conj_grad_\$omp\$loop	_ 0.036s	0.068s	24	3125	Static	0.450s	Spin due to	o imbalar	nce _{04s}		

572	!\$omp	do		
573			do	j=1,lastrow-firstrow+1
574				suml = 0.d0
575				<pre>do k=rowstr(j),rowstr(j+1)-1</pre>
576				suml = suml + a(k)*p(colidx(k))
577				enddo
578				q(j) = suml
579			end	obt
580	!\$omp	enc	d do	

Changed to dynamic scheduling



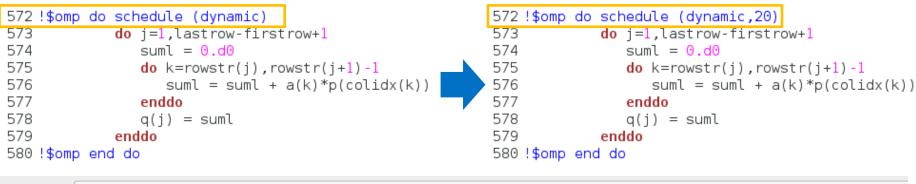
Grouping: (custom) OpenMP	Region / Op	enMP Bari	rier Type /	OpenMP Ba	arrier / Function / Ca	all Stack						
OpenMP Region / OpenMP	2	(OpenMP	OpenMP CPU Time							
Barrier Type / OpenMP Barrier	Poten Gain	Elapsed Time	OpenMP Loop	Loop Schedule	Effective Time	Spin 🔊	Overhead Time			≪		
/ Function / Call Stack		Chunk	Туре	📕 Idle 📕 Poor 📘	Time	Creation	Scheduling	Reduction	Other			
⊡conj_grad_\$omp\$parallel:24@	onj_grad_\$omp\$parallel:24@ Elapsed time ii		e incre	ased	199.298s	5.866s	0.001s	75.051s	0.012s	0.083s		
□Loop barriers		405			199.272s	5.709s	0.001s	75.021s	0.012s	0.071s		
± conj_grad_\$omp\$loop_bar	3.133s	11.102s	1	Dynamic	189.320s	0.368s	Os	74.990s	Os	0.009s		
±conj_grad_\$omp\$loop_bar	0.128s			Static	6.8805	9696	05		05	05		
⊡conj_grad_\$omp\$loop_bar	0.031s	0. C	iunk siz	<mark>e is only</mark>	1Spin	is fixe	a now _{Os}	New prob	olem: sche	eduling		

Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

'inte

Default chunk size is 1

Set chunk size to 20

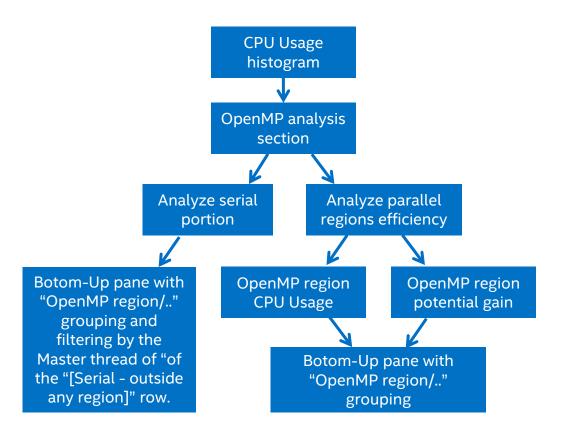


Grouping: (custom) OpenMP Region / OpenMP Barrier Type / OpenMP Barrier / Function / Call Stack

OpenMP Region / OpenMP Barrier	D	Potential		Number	OpenMP	OpenMP	CPU Tin	ie	*
Type / OpenMP Barrier / Function / Call Stack	Potenti Gain	Gain (% of Collection	Elapsed Time	of OpenMP threads	Loop Chunk	Jeneda	Effective Time by	Spin 🔊	Over 🔊
Cart Stack		Time)					📕 Idle 📕 Poor 📙 Ok	Time	Time
⊂conj_grad_\$omp\$parallel:24@/ho	0.264s	2.7%	9.568s	24			220.930s	6.135s	1.061s
□Loop barriers	0.258s	2.6%	9.557s	24			220.904s	5.982s	1.019s
⊡conj_grad_\$omp\$loop_barrier(0.119s	1.2%	0.406s	24	3125	Static	6.963s	2.769s	0.001s
➡conj_grad_\$omp\$loop_barrier(0.077s	0.8%	8.928s	24	20	Dynamic	210.848s	1.083s	0.893s
⊡conj_grad_\$omp\$loop_barrier(0.3%		24	3125	Static	0.538s	1.010s	0.083s
⊕conj_grad_\$omp\$loop_barri <mark>k</mark>	Elapsed tir	ne 8.928	s vs 10.	<mark>445s</mark>	3125	Static	1.365s	0.537s	0.022s

Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

General OpenMP analysis workflow





Copyright © 2014, Intel Corporation. All rights reserved. *Other names and brands may be claimed as the property of others.

Optimization Notice



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 © 2014, 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

Optimization Notice

24