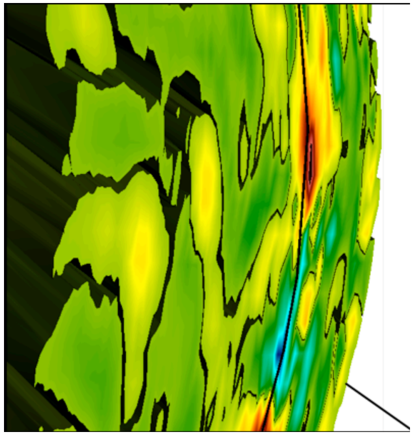
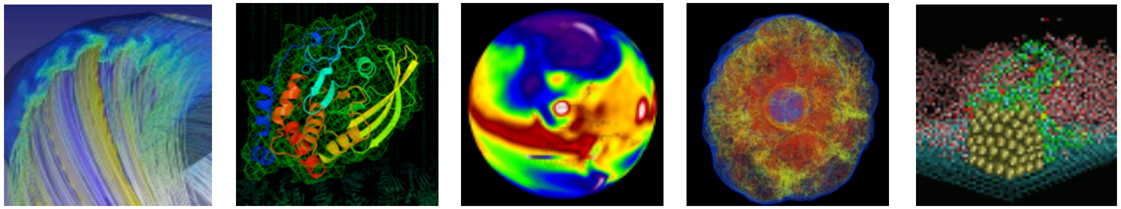
Intel Advisor on Cori

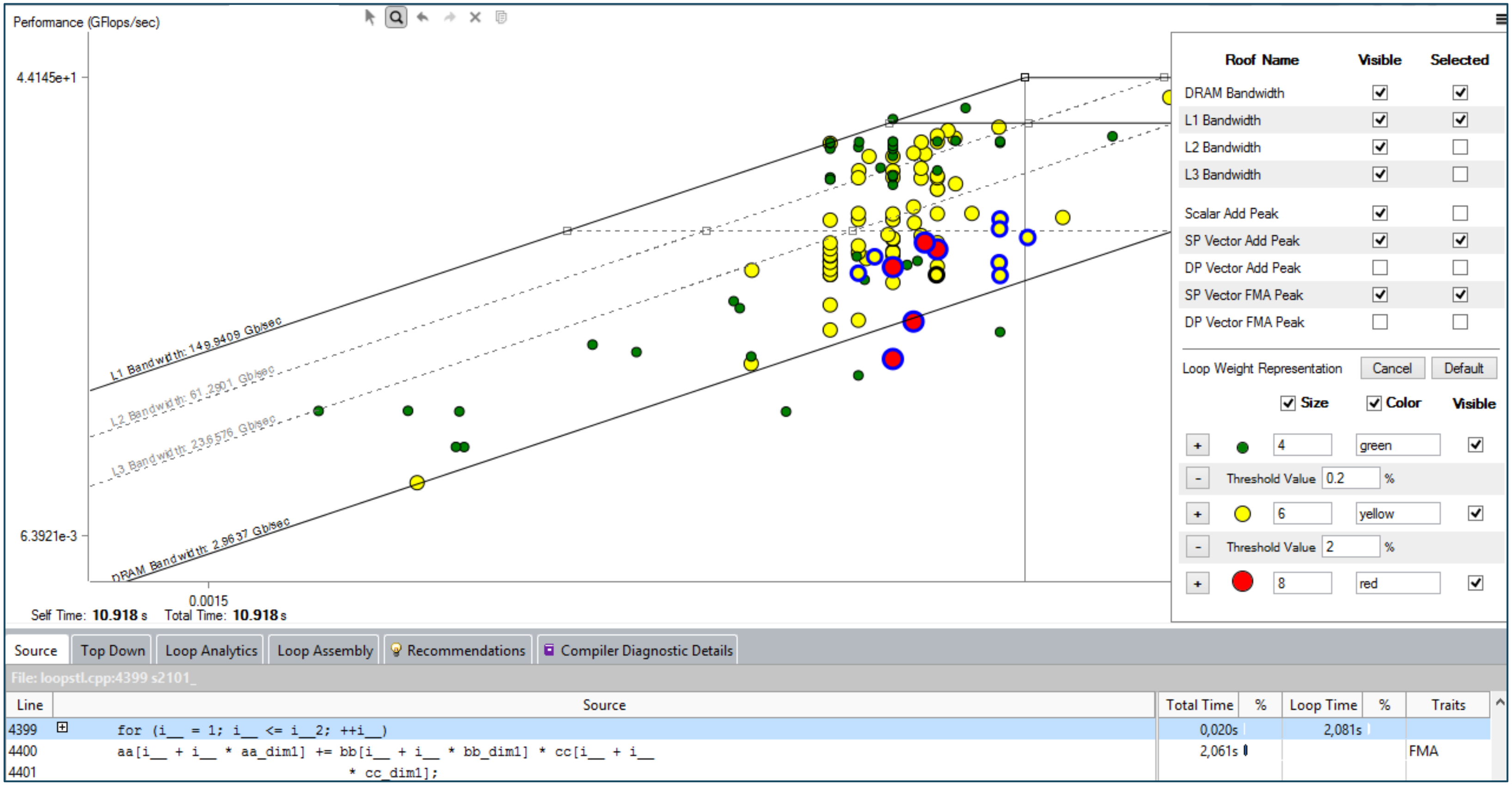




Charlene Yang

Application Performance Group
cjyang@lbl.gov

Intel® Advisor: automatic and refined


Performance (GFlops/sec)
4.4145e+1
6.3921e-3
0.0015
L1 Bandwidth: 149.9409 Gb/sec
L2 Bandwidth: 61.2901 Gb/sec
L3 Bandwidth: 23.6576 Gb/sec
DRAM Bandwidth: 2.9637 Gb/sec

Self Time: **10.918** s Total Time: **10.918** s

Roof Name	Visible	Selected
DRAM Bandwidth	☑	☑
L1 Bandwidth	☑	☑
L2 Bandwidth	☑	☐
L3 Bandwidth	☑	☐
Scalar Add Peak	☑	☐
SP Vector Add Peak	☑	☑
DP Vector Add Peak	☐	☐
SP Vector FMA Peak	☑	☑
DP Vector FMA Peak	☐	☐

Loop Weight Representation Cancel Default

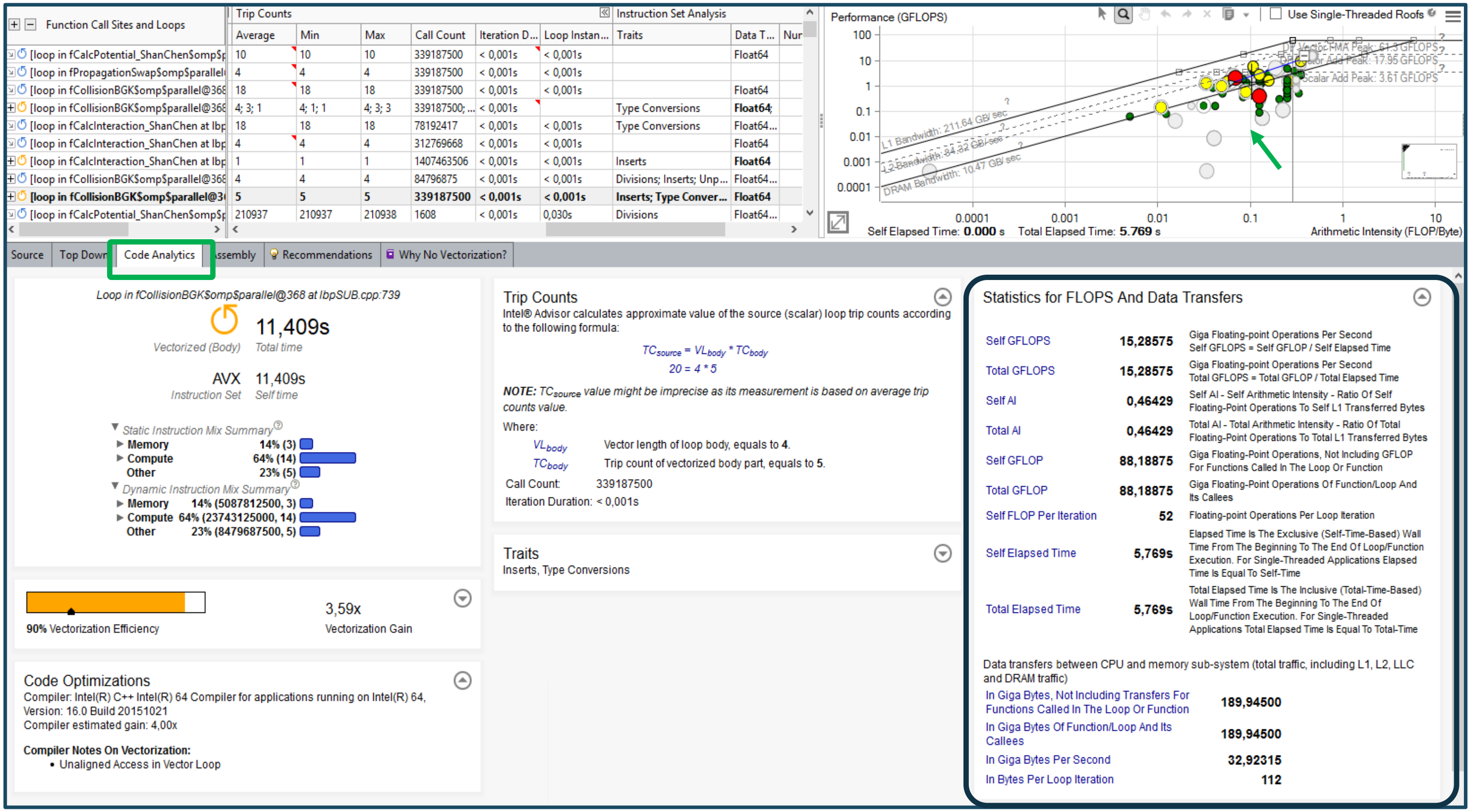
		☑ Size	☑ Color	Visible
+	●	4	green	☑
-	Threshold Value	0.2	%	
+	●	6	yellow	☑
-	Threshold Value	2	%	
+	●	8	red	☑

Source
Top Down
Loop Analytics
Loop Assembly
Recommendations
Compiler Diagnostic Details

File: loopstl.cpp:4399 s2101_

Line	Source	Total Time	%	Loop Time	%	Traits
4399	⊞ `for (i__ = 1; i__ <= i__2; ++i__)`	0,020s		2,081s		
4400	`aa[i__ + i__ * aa_dim1] += bb[i__ + i__ * bb_dim1] * cc[i__ + i__`	2,061s				FMA
4401	`* cc_dim1];`					

Intel® Advisor: code analytics



Function Call Sites and Loops	Trip Counts						Instruction Set Analysis		
	Average	Min	Max	Call Count	Iteration D...	Loop Instan...	Traits	Data T...	Nur
[loop in fCalcPotential_ShanChen$omp$p	10	10	10	339187500	< 0,001s	< 0,001s		Float64	
[loop in fPropagationSwap$omp$parallel	4	4	4	339187500	< 0,001s	< 0,001s			
[loop in fCollisionBGK$omp$parallel@368	18	18	18	339187500	< 0,001s	< 0,001s		Float64	
[loop in fCollisionBGK$omp$parallel@368	4; 3; 1	4; 1; 1	4; 3; 3	339187500; ...	< 0,001s		Type Conversions	**Float64;**	
[loop in fCalcInteraction_ShanChen at lbp	18	18	18	78192417	< 0,001s	< 0,001s	Type Conversions	Float64...	
[loop in fCalcInteraction_ShanChen at lbp	4	4	4	312769668	< 0,001s	< 0,001s		Float64	
[loop in fCalcInteraction_ShanChen at lbp	1	1	1	1407463506	< 0,001s	< 0,001s	Inserts	**Float64**	
[loop in fCollisionBGK$omp$parallel@368	4	4	4	84796875	< 0,001s	< 0,001s	Divisions; Inserts; Unp...	Float64...	
[loop in fCollisionBGK$omp$parallel@3	**5**	**5**	**5**	**339187500**	**< 0,001s**	**< 0,001s**	**Inserts; Type Conver...**	**Float64**	
[loop in fCalcPotential_ShanChen$omp$p	210937	210937	210938	1608	< 0,001s	0,030s	Divisions	Float64...	

Performance (GFLOPS)
☐ Use Single-Threaded Roofs
100
10
1
0.1
0.01
0.001
0.0001
DP Vector FMA Peak: 61.3 GFLOPS
DP Vector Add Peak: 17.95 GFLOPS
Scalar Add Peak: 3.61 GFLOPS
L1 Bandwidth: 211.64 GB/ sec
L2 Bandwidth: 84.32 GB/ sec
DRAM Bandwidth: 10.47 GB/ sec
0.0001
0.001
0.01
0.1
1
10
Arithmetic Intensity (FLOP/Byte)
Source
Top Down
Code Analytics
Assembly
Recommendations
Why No Vectorization?

Self Elapsed Time: **0.000** s Total Elapsed Time: **5.769** s

Loop in fCollisionBGK$omp$parallel@368 at lbpSUB.cpp:739

Vectorized (Body)	11,409s *Total time*
AVX *Instruction Set*	11,409s *Self time*

Static Instruction Mix Summary

Memory	**14% (3)**
Compute	**64% (14)**
Other	**23% (5)**

Dynamic Instruction Mix Summary

Memory	**14% (5087812500, 3)**
Compute	**64% (23743125000, 14)**
Other	**23% (8479687500, 5)**

90% Vectorization Efficiency

3,59x
Vectorization Gain

Code Optimizations

Compiler: Intel(R) C++ Intel(R) 64 Compiler for applications running on Intel(R) 64, Version: 16.0 Build 20151021
Compiler estimated gain: 4,00x

Compiler Notes On Vectorization:

- Unaligned Access in Vector Loop

Trip Counts

Intel® Advisor calculates approximate value of the source (scalar) loop trip counts according to the following formula:

$$TC_{source} = VL_{body} * TC_{body}$$

$$20 = 4 * 5$$

NOTE: *TC_{source} value might be imprecise as its measurement is based on average trip counts value.*

Where:

VL_{body}	Vector length of loop body, equals to 4.
TC_{body}	Trip count of vectorized body part, equals to 5.

Call Count: 339187500
Iteration Duration: < 0,001s

Traits

Inserts, Type Conversions

Statistics for FLOPS And Data Transfers

Self GFLOPS	**15,28575**	Giga Floating-point Operations Per Second Self GFLOPS = Self GFLOP / Self Elapsed Time
Total GFLOPS	**15,28575**	Giga Floating-point Operations Per Second Total GFLOPS = Total GFLOP / Total Elapsed Time
Self AI	**0,46429**	Self AI - Self Arithmetic Intensity - Ratio Of Self Floating-Point Operations To Self L1 Transferred Bytes
Total AI	**0,46429**	Total AI - Total Arithmetic Intensity - Ratio Of Total Floating-Point Operations To Total L1 Transferred Bytes
Self GFLOP	**88,18875**	Giga Floating-Point Operations, Not Including GFLOP For Functions Called In The Loop Or Function
Total GFLOP	**88,18875**	Giga Floating-Point Operations Of Function/Loop And Its Callees
Self FLOP Per Iteration	**52**	Floating-point Operations Per Loop Iteration
Self Elapsed Time	**5,769s**	Elapsed Time Is The Exclusive (Self-Time-Based) Wall Time From The Beginning To The End Of Loop/Function Execution. For Single-Threaded Applications Elapsed Time Is Equal To Self-Time
Total Elapsed Time	**5,769s**	Total Elapsed Time Is The Inclusive (Total-Time-Based) Wall Time From The Beginning To The End Of Loop/Function Execution. For Single-Threaded Applications Total Elapsed Time Is Equal To Total-Time

Data transfers between CPU and memory sub-system (total traffic, including L1, L2, LLC and DRAM traffic)

In Giga Bytes, Not Including Transfers For Functions Called In The Loop Or Function	**189,94500**
In Giga Bytes Of Function/Loop And Its Callees	**189,94500**
In Giga Bytes Per Second	**32,92315**
In Bytes Per Loop Iteration	**112**

Intel® Advisor: 2-step data collection

Roofline : Axis X: **AI** = #**FLOP** / #**Bytes** Axis Y: **FLOP/S** = #**FLOP** (mask aware) / #**Seconds**	Overhead
Step 1: Survey (-collect survey) - Provide **#Seconds** - *Root access not needed* - User mode sampling, non-intrusive.	**1x**
Step 2: FLOPS *(-collect tripcounts –flops)* - Provide #**FLOP,** #**Bytes**, AVX-512 Mask - *Root access not needed* - Precise, instrumentation based, count number of instructions	**3-5x**

Setup on Cori

- **Compile with -g to get debugging info**

```
cc -g -dynamic -openmp -O2 -o mycode.exe mycode.c
```

- **Cache-Aware Roofline Model (CARM)**

```
module load advisor/2018.up1
```

- **Integrated Roofline Model (Cache Simulator)**

```
module load advisor/2018.integrated_roofline.up1
```

- **Incompatible GUI for regular and integrated Advisor**

Run Advisor on Cori

- **Start an interactive session on a KNL node**

```
salloc --qos=interactive -C knl -N 1 -t hh:mm:ss -A <your_account>
```

- **To collect data for roofline, do two collections: survey and tripcounts.**

```
srun -n <num-of-ranks> -c <num_of_cores_per_rank> advixe-cl -v
-collect survey -no-auto-finalize  -project-dir=<same_dir_name>
-data-limit=0 -- <your_executable>
```

```
srun -n <num-of-ranks> -c <num_of_cores_per_rank> advixe-cl -v
-collect tripcounts -flops -no-auto-finalize -project-dir=<same_dir_name>  -
data-limit=0 -- <your_executable>
```

- **Run on the Lustre filesystem $SCRATCH**
- **Finalization is expensive especially on KNL: do it offline!**

Pack/View results

- **Pack results/source file/binary (already packed in demo)**

```
advixe-cl --snapshot --project-dir <same_dir_name> --pack --cache-sources
--cache-binaries -- <target_file_name>
```

- **Load module**

```
module load advisor/2018.integrated_roofline.up1
module show advisor/2018.integrated_roofline.up1
```

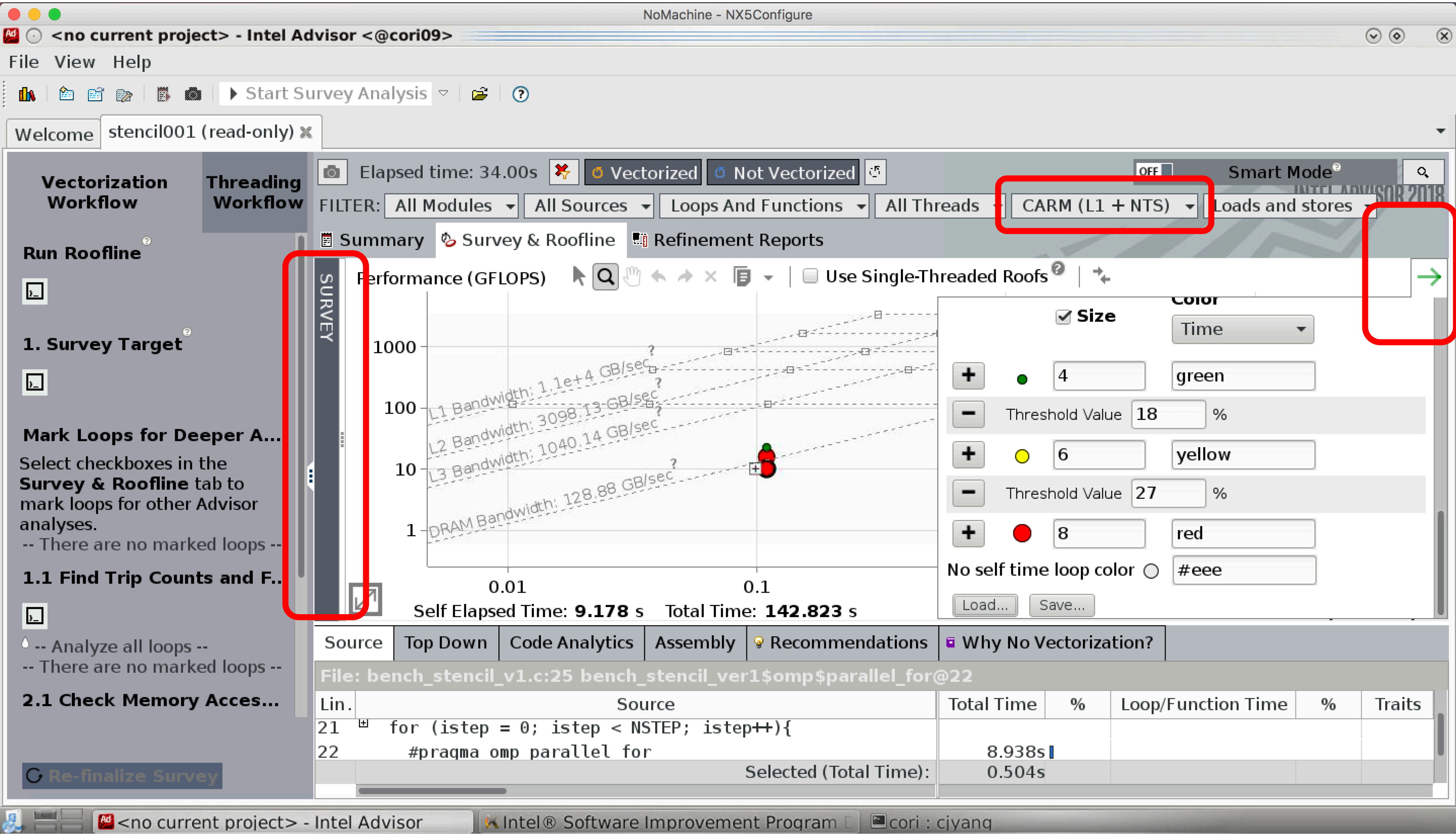
```
setenv ADVISOR_XE_2018_DIR
/global/common/cori/software/intel/advisor_2018.0.2.537542
```

- **Copy file and view results**

```
cp -r $ADVISOR_XE_2018_DIR/ECP-meeting-tutorial/ .
cd ECP-meeting-tutorial/
```

```
advxe-gui stencil.advixeexpz
```

- **or use NX: https://nxcloud01.nersc.gov**



NoMachine - NX5Configure

<no current project> - Intel Advisor <@cori09>

File View Help

Start Survey Analysis

Welcome | stencil001 (read-only)

Vectorization Workflow | Threading Workflow

Run Roofline

1. Survey Target

Mark Loops for Deeper A...

Select checkboxes in the **Survey & Roofline** tab to mark loops for other Advisor analyses.

-- There are no marked loops --

1.1 Find Trip Counts and F...

-- Analyze all loops --

-- There are no marked loops --

2.1 Check Memory Acces...

Re-finalize Survey

Elapsed time: 34.00s | Vectorized | Not Vectorized | OFF Smart Mode

FILTER: All Modules | All Sources | Loops And Functions | All Threads | CARM (L1 + NTS) | Loads and stores

Summary | Survey & Roofline | Refinement Reports

SURVEY

Performance (GFLOPS) | ☐ Use Single-Threaded Roofs

1000
100
10
1
L1 Bandwidth: 1.1e+4 GB/sec
L2 Bandwidth: 3098.13 GB/sec
L3 Bandwidth: 1040.14 GB/sec
DRAM Bandwidth: 128.88 GB/sec
0.01
0.1

Self Elapsed Time: **9.178** s Total Time: **142.823** s

☑ **Size** | **Color**: Time

+	●	4	green
−	Threshold Value	18	%
+	●	6	yellow
−	Threshold Value	27	%
+	●	8	red

No self time loop color ○ #eee

Load... | Save...

Source | Top Down | Code Analytics | Assembly | Recommendations | Why No Vectorization?

File: bench_stencil_v1.c:25 bench_stencil_ver1$omp$parallel_for@22

Lin.	Source	Total Time	%	Loop/Function Time	%	Traits
21	for (istep = 0; istep < NSTEP; istep++){					
22	#pragma omp parallel for	8.938s				
	Selected (Total Time):	0.504s				

<no current project> - Intel Advisor | Intel® Software Improvement Program | cori : cjyang

GPP optimization - cache blocking

Before

```
do my_igp = 1, ngpown ! OpenMP
 do iw = 1 , 3
    do ig = 1, igmax
        load wtilde_array(ig,my_igp) !512KB per row
        load aqsntemp(ig,n1) !512KB per row
        load eps(ig,my_igp) !512KB per row
```

After

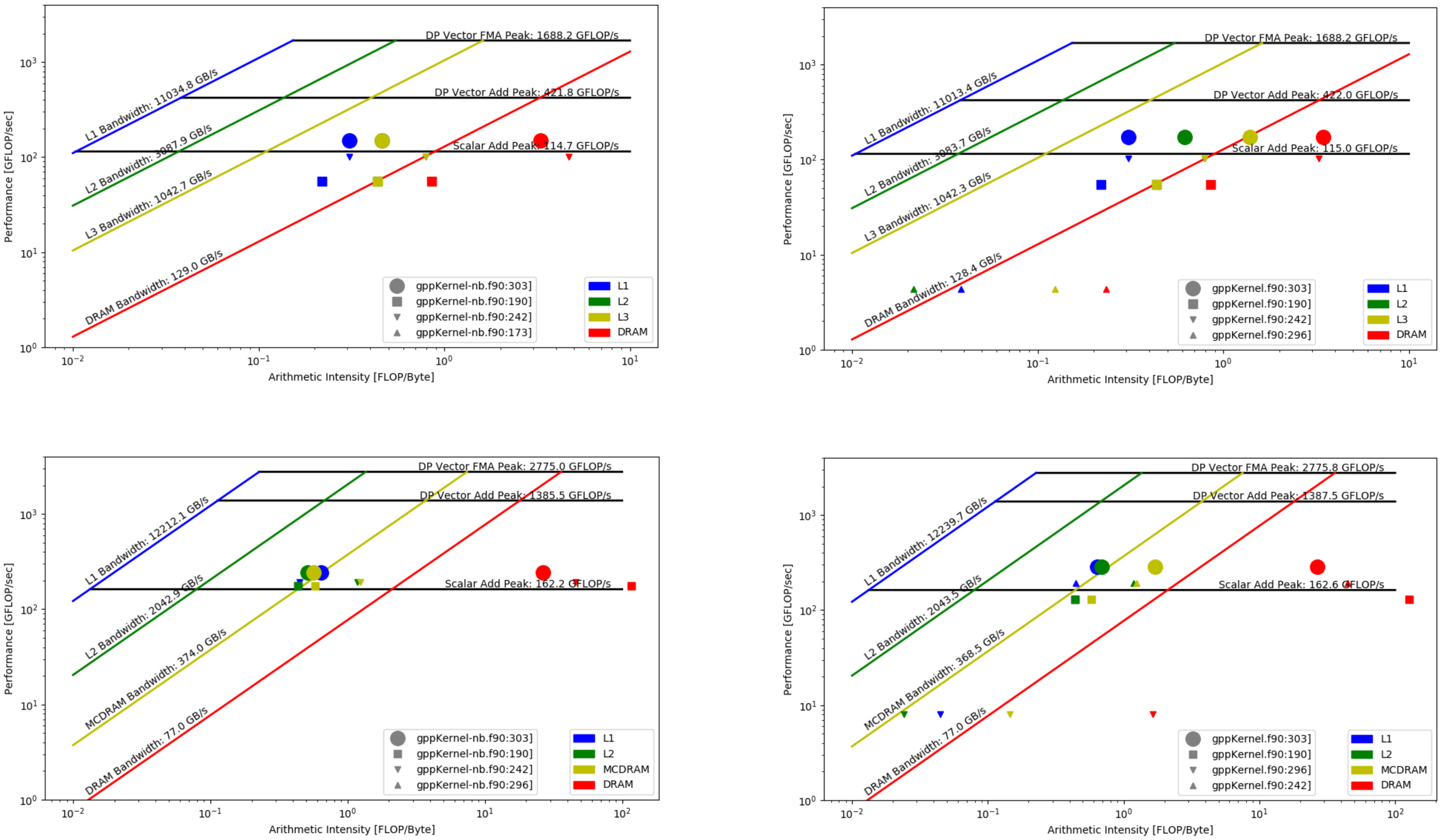
```
do my_igp = 1, ngpown ! OpenMP
 do igbeg = 1, igmax, igblk
    do iw = 1 , 3
        do ig = igbeg, min(igbeg + igblk,igmax)
            load wtilde_array(ig,my_igp)
            load aqsntemp(ig,n1)
            load eps(ig,my_igp)
```

GPP optimization - changes in AI

- **L1 AI and L2 AI remain roughly the same on Haswell and KNL**
- **LLC AI improves by 3x due to fixed trip count 3 in loop *iw***
- **GFLOPS/s improved by 16-18%, less than 3x, because there are divide, shuffle and unpack instructions involved in the innermost loop**

	Haswell		**KNL**	
gppKernel.f90:303	Original	Cache-blocked	Original	Cache-blocked
L1+NTS AI	0.31	0.31	0.64	0.64
L2 AI	0.46	0.62	0.51	0.69
LLC AI	0.46	1.4	0.56	1.71
DRAM AI	3.31	3.44	26.45	26.83
GFLOPS/s	148.57	172.48	242.55	287.28

HSW (top) v.s. KNL (bottom)



NeRSC