Overview of Performance Optimization on Intel® Xeon Phi™

Code Named Knights Landing (KNL)

Intel® Software Development Products

The 2nd Generation Intel® Xeon Phi™ Processor (code named Knights Landing)

Targeted for high performance computing

- **High BW**
  - Integrated memory on package: 490 measured GB/sec*; up to 16 GB capacity
  - Cache or separate NUMA node
- **Cluster Parallelism**
  - Integrated fabric on package (Omni-Path)
  - 2x100 Gbps ports
- **Thread level Parallelism (TLP)**
  - Up to 68 cores X 4 hyper-threads per core = 272 threads (7290 offers 72 cores; premium part)
  - Tiles: 2 cores per tile sharing Cache-Home-Agent for Cache Coherency and 1MB MB L2 cache
- **Data-level Parallelism (DLP)**
  - Introduces AVX-512 ISA
  - Compatible with previous ISA (AVX, SSE, …)
- **Instruction-level Parallelism (ILP)**
  - Out-of-order core
  - Two vector processing units per core
- **Power Efficiency**
  - 215 Watts TDP (7290 is 245 Watts)
  - 2x145 Watts TDP for Xeon Dual socket BDW E5-2697 (2x18 cores)

Performance:
Vector Peak Performance: 3+TF DP, 6+TF SP
Bandwidth: 490 GB/sec Triad Stream Score*

*Using Streaming Stores in Flat Mode

Focus Areas for Optimization

Optimization Focus Areas

Parallelism

Vectorization

Memory BW

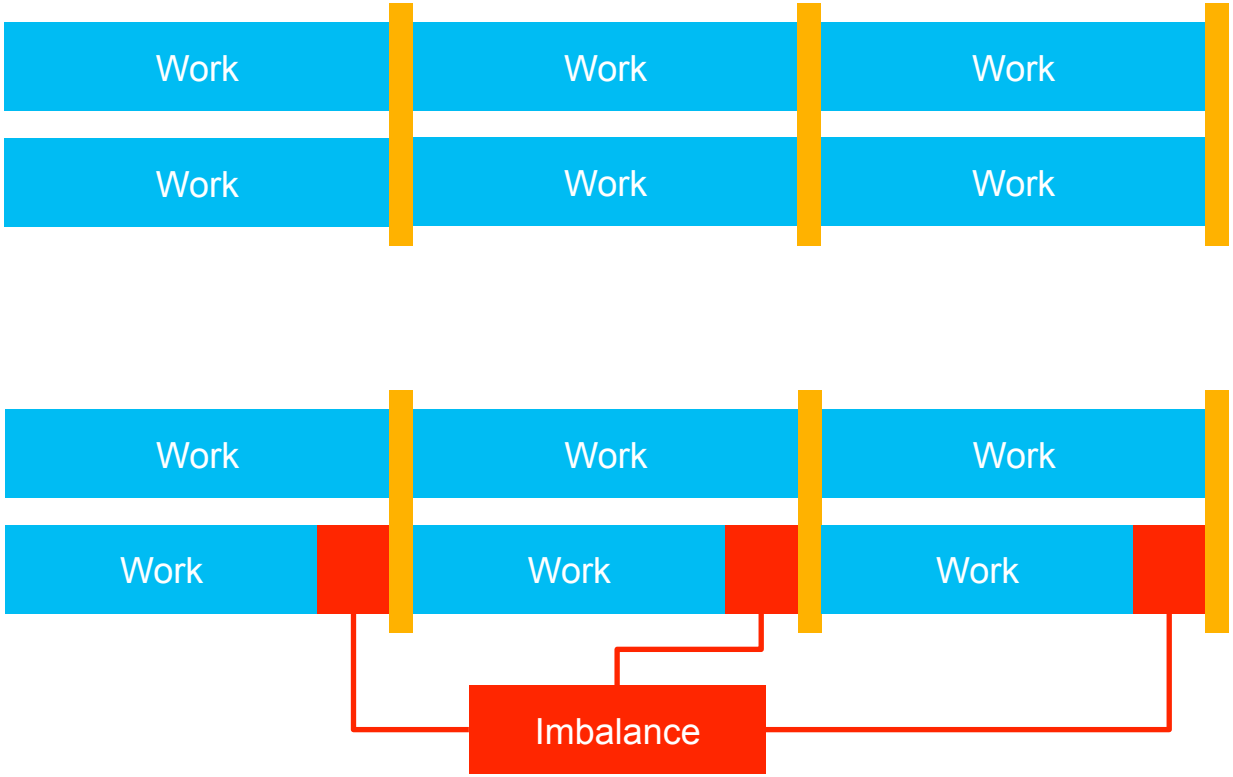
Parallelism on KNL

Multiple Threading Options

- Automatic Parallelism in Intel® Compilers
- OpenMP*
- Intel® Threading Building Blocks
- Threading inside of performance libraries

Also, MPI and MPI+Threading

Defining Imbalance in Parallelism



Vectorization on KNL

AVX-512 vector lanes

Automatic vectorization in compiler

- Sometimes needs help with directives/pragmas

SIMD loops: syntax

#pragma omp simd [*clauses*]

for-loop

!$omp simd [*clauses*]

do-loops

*[***!$omp end simd***]*

Loop has to be in “Canonical loop form”

- as do/for worksharing

SIMD loop clauses

safelen (length)

- Maximum number of iterations that can run concurrently without breaking a dependence
  - in practice, maximum vector length

linear (list[:linear-step])

- The variable value is in relationship with the iteration number
  - $x_i = x_{orig} + i * \text{linear-step}$

aligned (list[:alignment])

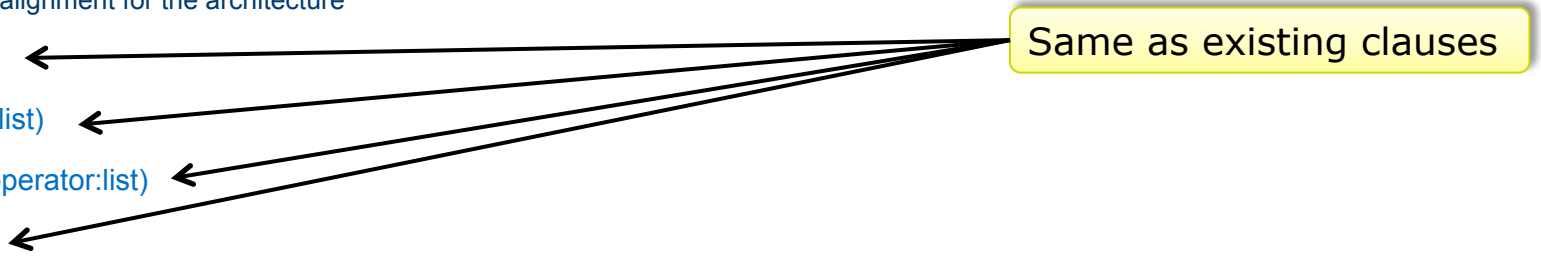
- Specifies that the list items have a given alignment
- Default is alignment for the architecture

private (list)

lastprivate (list)

reduction (operator:list)

collapse (n)



SIMD functions: Syntax

#pragma omp declare simd ***[clauses]***

[#pragma omp declare simd [clauses]]

function definition or declaration

!$omp declare simd (*function-or-procedure-name***)** ***[clauses]***

Instructs the compiler to

- generate a SIMD-enabled version(s) of a given function
- that a SIMD-enabled version of the function is available to use from a SIMD loop

SIMD functions: clauses

simdlen(*length***)**

- generate function to support a given vector length

uniform(*argument-list***)**

- argument has a constant value between the iterations of a given loop

inbranch

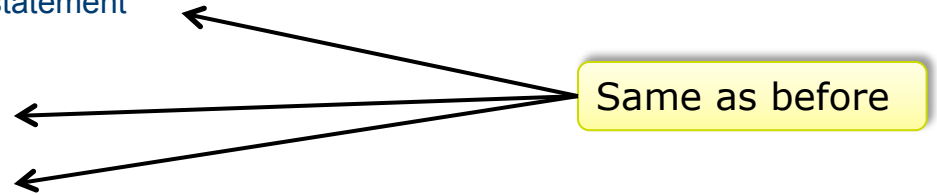
- function always called from inside an if statement

notinbranch

- function never called from inside an if statement

linear(*argument-list[:linear-step]***)**

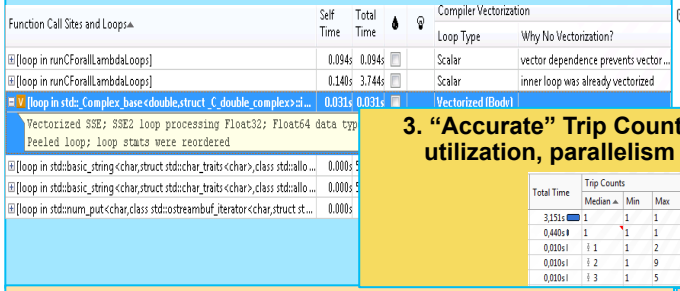
aligned(*argument-list[:alignment]***)**



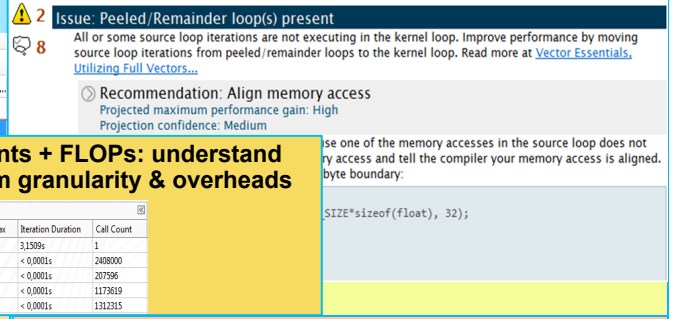
5 Steps to Efficient Vectorization - Vector Advisor

(part of Intel® Advisor, Parallel Studio, Cluster Studio)

1. Compiler diagnostics + Performance Data + SIMD efficiency information

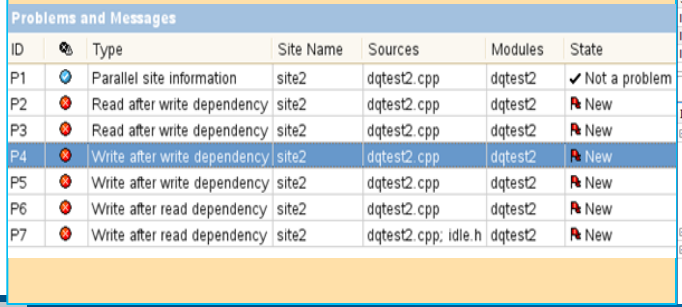


2. Guidance: detect problem and recommend how to fix it

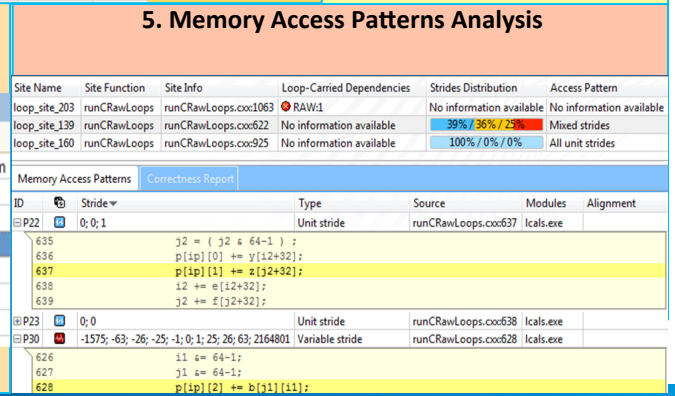


3. "Accurate" Trip Counts + FLOPs: understand utilization, parallelism granularity & overheads

4. Loop-Carried Dependency Analysis



5. Memory Access Patterns Analysis



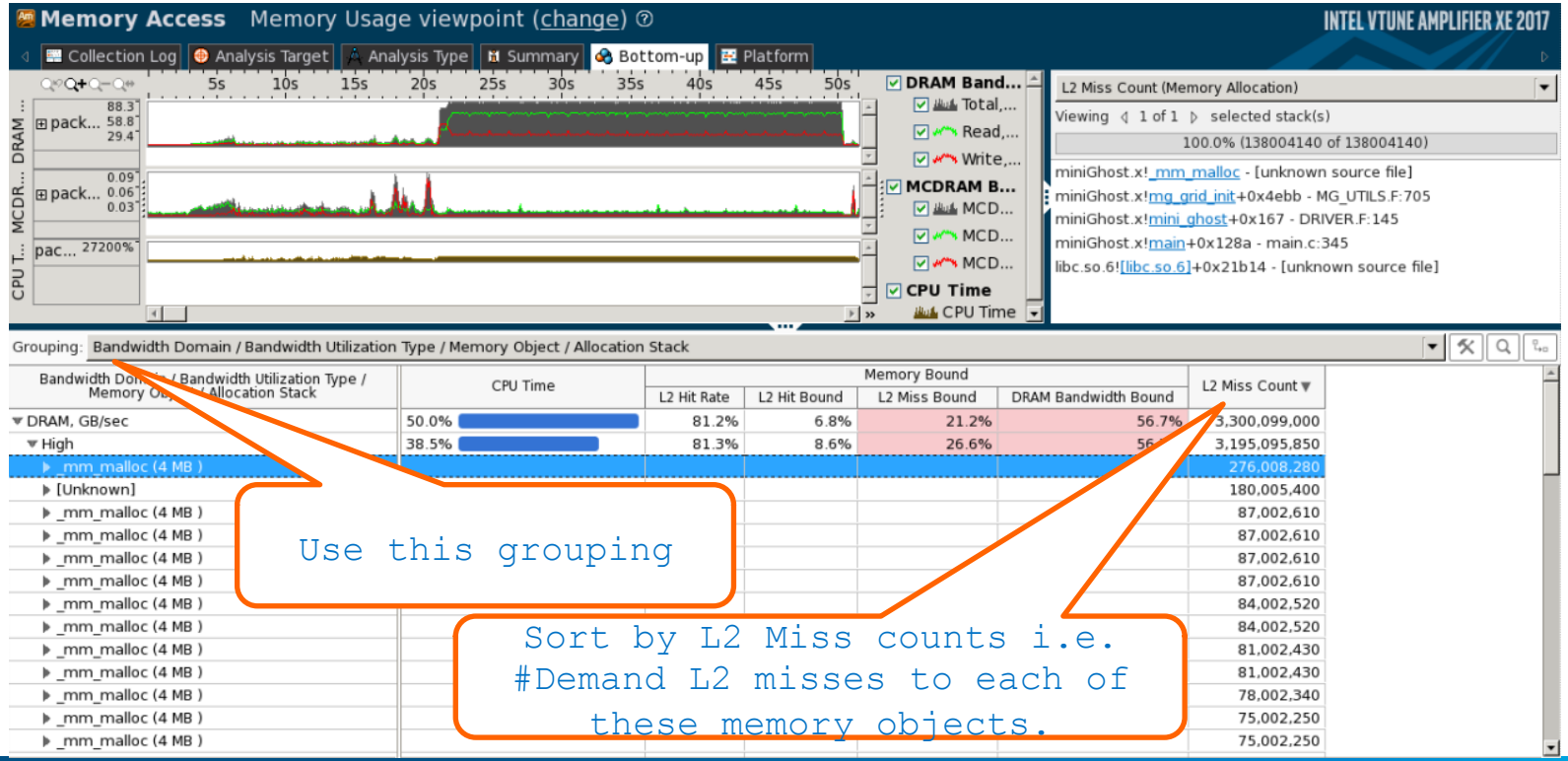
Memory Bandwidth on KNL

High Bandwidth Memory

- Want to maximize utilization
- Find high use memory objects using Intel® VTune™ Amplifier
- Allocate high use memory objects into HBM
  - Memkind library http://memkind.github.io/memkind
    - Also includes AutoHBW
  - Use numactl

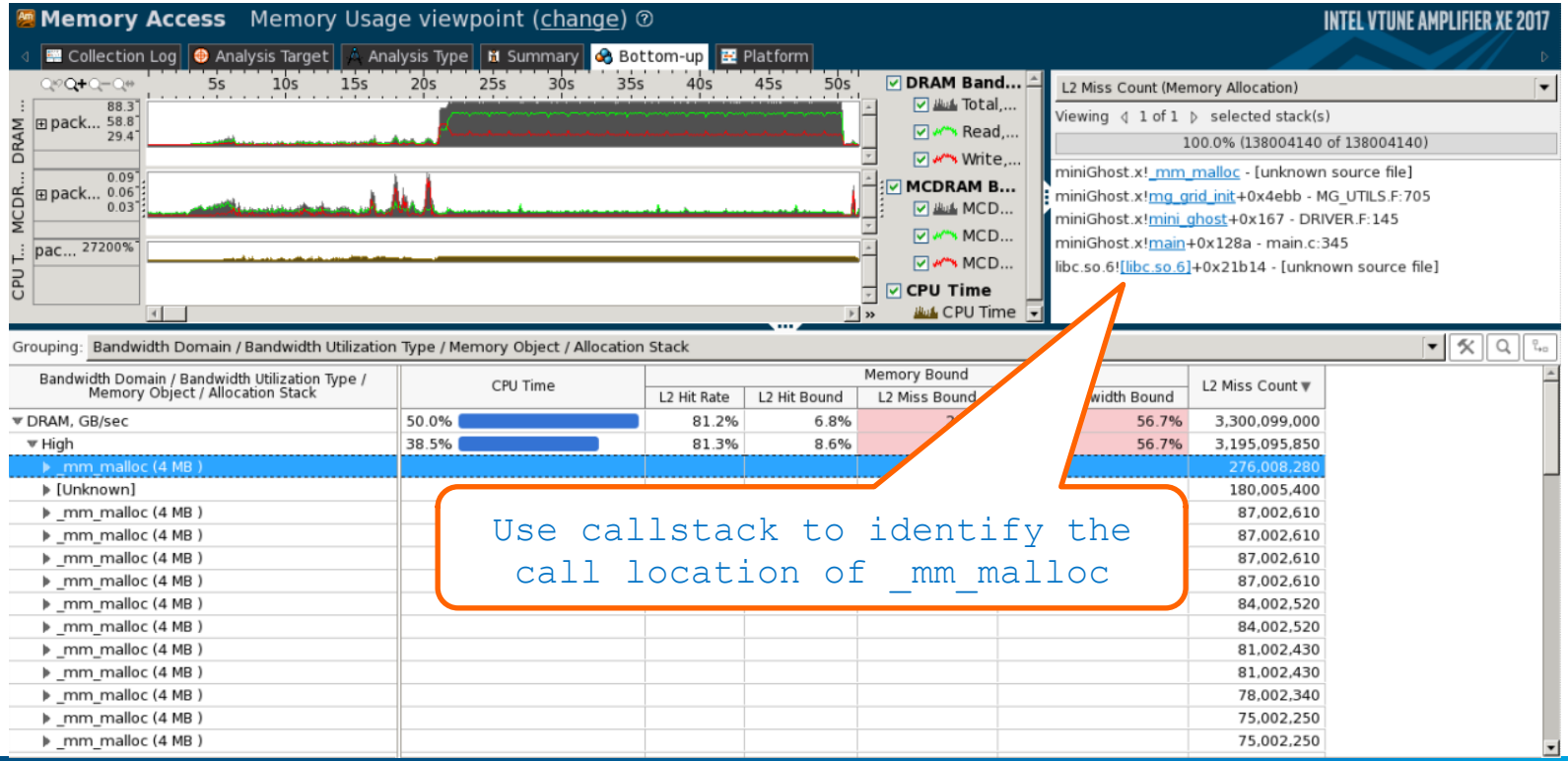
Identifying high bandwidth memory objects (1/3)

Memory object analysis: DDR only



Identifying high bandwidth memory objects (2/3)

Memory object analysis: DDR only



Identifying high bandwidth memory objects (3/3)

MG_UTILS.F

```
685           CALL MG_INIT_GRID ( GRID38, IERR )
686        END IF
687
688        IF ( NUM_VARS > 38 ) THEN
689           ALLOCATE ( GRID39( 0:NX+1, 0:NY+1, 0:NZ+1 ), STAT = IERR )
690           CALL MG_ASSERT ( IERR, 'GRID_INIT: ALLOCATE ( GRID39 )', (NX+2)*(NY+2)*(NZ+2) )
691           CALL MG_INIT_GRID ( GRID39, IERR )
692        END IF
693
694        IF ( NUM_VARS > 39 ) THEN
695           ALLOCATE ( GRID40( 0:NX+1, 0:NY+1, 0:NZ+1 ), STAT = IERR )
696           CALL MG_ASSERT ( IERR, 'GRID_INIT: ALLOCATE ( GRID40 )', (NX+2)*(NY+2)*(NZ+2) )
697           CALL MG_INIT_GRID ( GRID40, IERR )
698        END IF
699
700        IF ( NUM_VARS > 40 ) THEN
701           IERR = -1
702           CALL MG_ASSERT ( IERR, 'GRID_INIT: TOO MANY VARS', NUM_VARS )
703        END IF
704
705        ALLOCATE ( WORK( 0:NX+1, 0:NY+1, 0:NZ+1 ), STAT = IERR )
706        CALL MG_ASSERT ( IERR, 'GRID_INIT: ALLOCATE ( WORK )', (NX+2)*(NY+2)*(NZ+2) )
707
708        RETURN
709
710     END SUBROUTINE MG_GRID_INIT
711
712  !  ================================================================================
713
```

High BW memory object identified is work

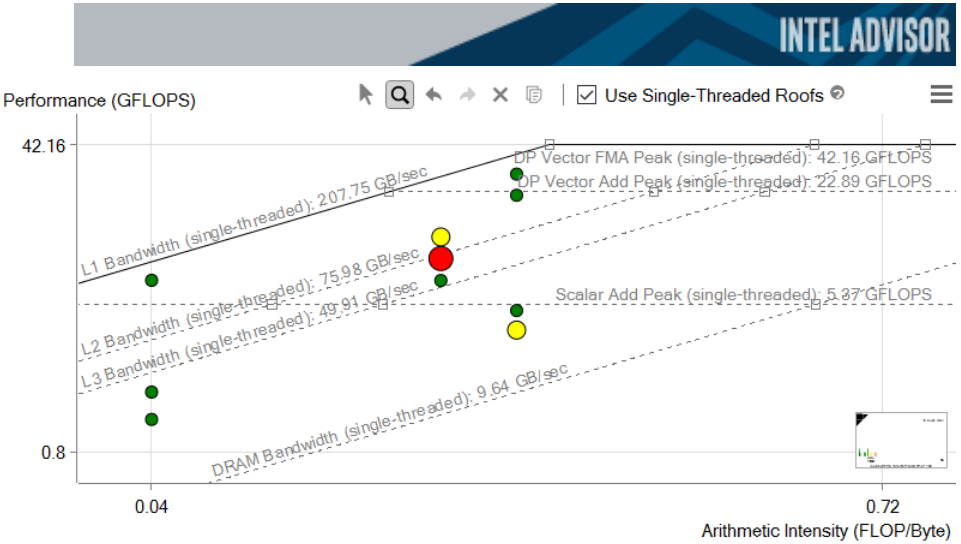
Roofline Analysis Using Intel® Advisor

Find Effective Optimization Strategies

Intel® Advisor: Cache-aware roofline analysis

Roofline Performance Insights

- Highlights poor performing loops
- Shows performance “headroom” for each loop
  - Which can be improved
  - Which are worth improving
- Shows likely causes of bottlenecks
- Suggests next optimization steps



Find Effective Optimization Strategies

Intel® Advisor: Cache-aware roofline analysis

Roofs Show Platform Limits

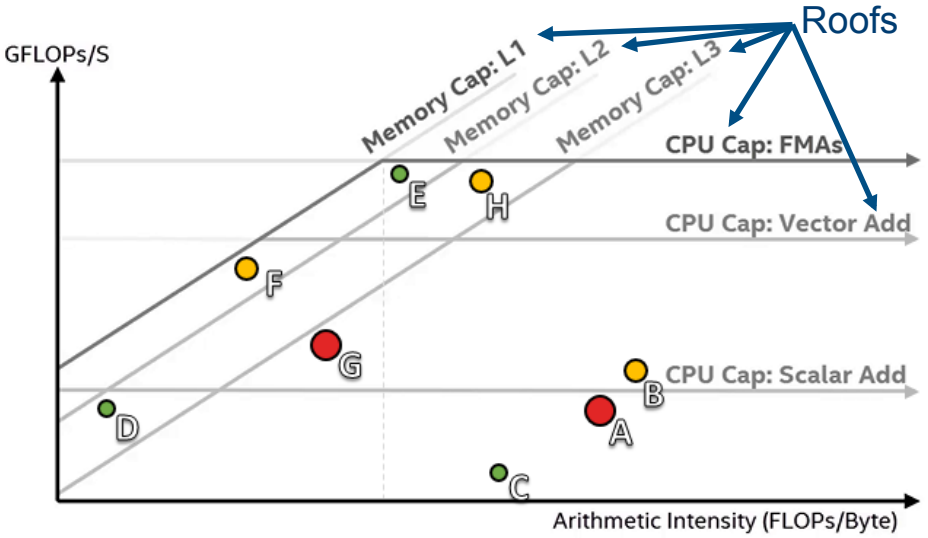
- Memory, cache & compute limits

Dots Are Loops

- Bigger, red dots take more time so optimization has a bigger impact
- Dots farther from a roof have more room for improvement

Higher Dot = Higher GFLOPs/sec

- Optimization moves dots up
- Algorithmic changes move dots horizontally



Which loops should we optimize?

- A and G are the best candidates
- B has room to improve, but will have less impact
- E, C, D, and H are poor candidates

Roofline tutorial video

Create Faster HPC, Cloud, and AI Software

What's New in Intel® Parallel Studio XE 2018 Beta

Get More Performance from New Hardware

- Use fast **AVX-512** instructions on **Intel® Xeon®** and **Xeon Phi™** processors
- Accelerate MPI applications with **Intel® Omni-Path** Architecture support

Discover Untapped Performance Faster

- **Intel® Advisor –** Use Roofline analysis to find high impact, but under optimized loops
- **Application Snapshot –** Get quick answers: Does my hybrid code need optimization?
- **Intel® VTune™ Amplifier** – Profile private clouds with Docker* containers, Java* daemons

Boost Machine Learning Application Performance

- **Intel® Data Analytics Acceleration Library** – Speed machine learning with new optimized algorithms
- **Intel® Distribution for Python*** - Accelerate Python code using fast NumPy/SciPy and scikit-learn packages

Latest Standards and IDEs

- **C++2017** draft parallelizes and vectorizes C++ easily using **Parallel STL***
- Full **Fortran* 2008**, **Fortran 2015** draft
- **OpenMP* 5.0** draft, **Microsoft Visual Studio* 2017**

And much more*…

Register for Beta at: http://intel.ly/intel-parallel-studio-xe-2018-beta

* See Release Notes for the full list with further updates and new features.

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®
experience
what's inside™