Intel® VTune™ Profiler
Application Performance Snapshot

Dmitry Tarakanov

Software Technical Consulting Engineer

Notices & Disclaimers

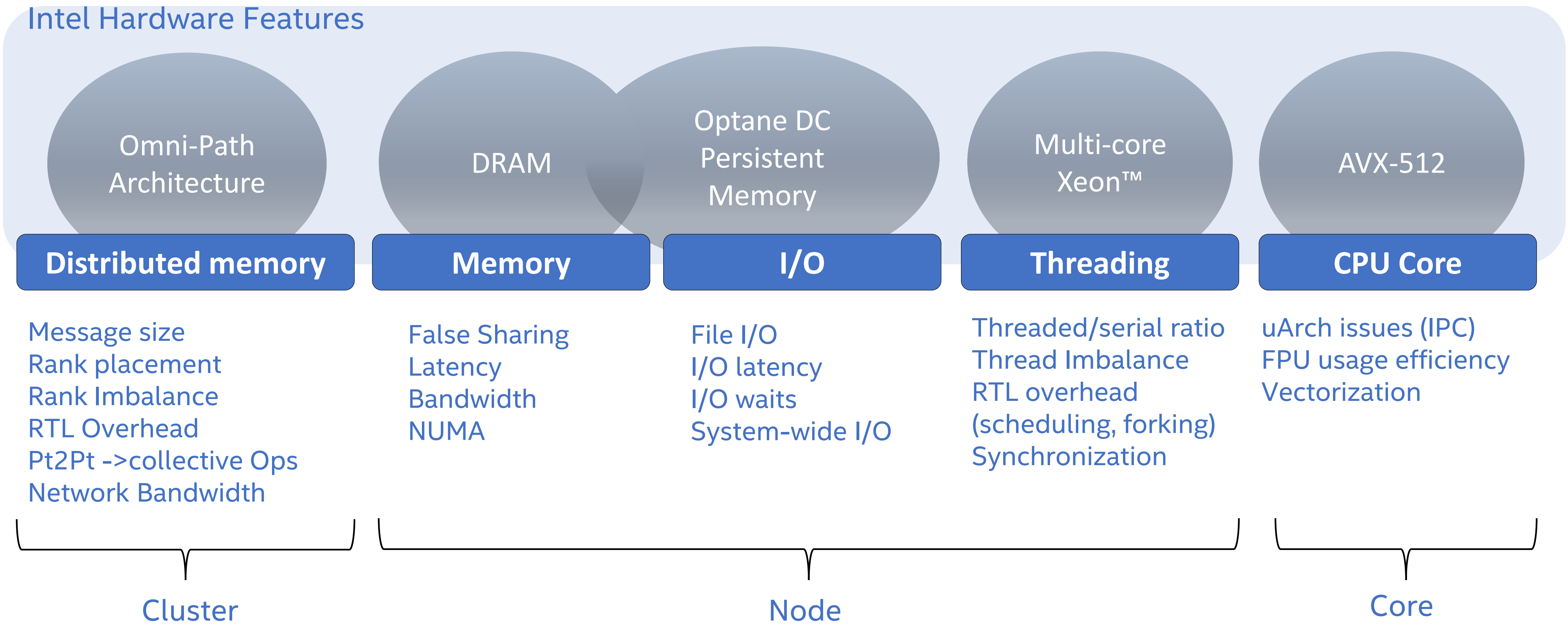
Performance varies by use, configuration, and other factors. Learn more at www.Intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See configuration disclosure for details.

Your costs and results may vary.

Intel technologies may require enabled hardware, software or service activation.

Aspects of HPC/Throughput Application Performance



Intel Hardware Features

Omni-Path Architecture

DRAM

Optane DC Persistent Memory

Multi-core Xeon™

AVX-512

Distributed memory

Message size
Rank placement
Rank Imbalance
RTL Overhead
Pt2Pt ->collective Ops
Network Bandwidth

Memory

False Sharing
Latency
Bandwidth
NUMA

I/O

File I/O
I/O latency
I/O waits
System-wide I/O

Threading

Threaded/serial ratio
Thread Imbalance
RTL overhead
(scheduling, forking)
Synchronization

CPU Core

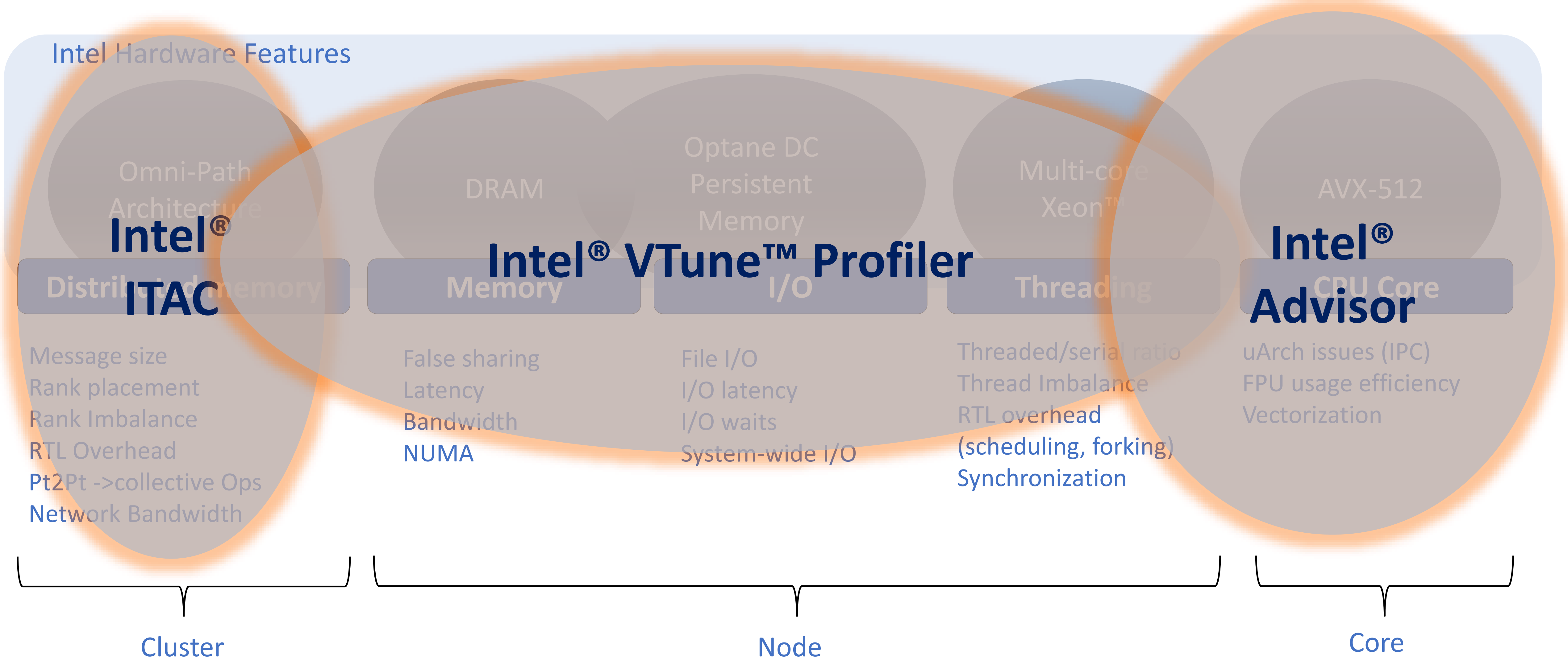
uArch issues (IPC)
FPU usage efficiency
Vectorization

Cluster

Node

Core

Intel Tools covering the Aspects



Intel Hardware Features

Intel® ITAC

Intel® VTune™ Profiler

Intel® Advisor

Omni-Path Architecture

Distributed memory

Message size
Rank placement
Rank Imbalance
RTL Overhead
Pt2Pt ->collective Ops
Network Bandwidth

DRAM

Memory

False sharing
Latency
Bandwidth
NUMA

Optane DC Persistent Memory

I/O

File I/O
I/O latency
I/O waits
System-wide I/O

Multi-core Xeon™

Threading

Threaded/serial ratio
Thread Imbalance
RTL overhead
(scheduling, forking)
Synchronization

AVX-512

CPU Core

uArch issues (IPC)
FPU usage efficiency
Vectorization

Cluster

Node

Core

Before diving into a particular tool …

- How to assess that I have **potential in performance** tuning?
- **Which tool** should I use first?
- What to use on **large scale** avoiding being overwhelmed with huge trace size, post processing time and collection overhead?
- How to **quickly** evaluate environment settings or incremental code changes?
- **Answer:**

  Use VTune Profiler’s Application Performance Snapshot

Application Performance Snapshot at a glance (1/2)

- High-level **overview** of application performance
  - Detailed reports on MPI statistics
- Primary optimization areas and **next steps** in analysis with deep tools
- **Easy** to install, run, explore results with CL or HTML reports
  - No driver installation required working through perf
  - If SEP driver is available - will be additional advantage
- Application Performance Snapshot comes bundled with all installations of VTune Profiler on Linux* OS.
  - Standalone VTune Profiler download
  - As part of the Intel® oneAPI Base Toolkit
  - As part of the Intel® oneAPI System Bring-Up Toolkit

Application Performance Snapshot at a glance (2/2)

- **Low** collection overhead – 1-3%*
    - HW counters – counting mode only, no overtime
    - MPI and OpenMP tracing - trace aggregation in runtime, no overtime
        - Trace levels to collect more MPI details (potentially for cost of overhead)
    - Ability to choose either tracing or HW counting in the case of interest in particular metric subset and avoid overhead (--collection-mode option)
- **Scales** to large jobs
    - Tested and worked on 64K ranks
    - Trace size on default statistics level ~ 4Kb per rank

* MPI app startup on KNL/KNM in the condition of large number of ranks per node might have fixed time slowdown

APS workflow

Setup Environment

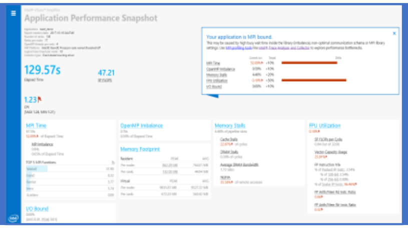
- “>source /opt/intel/oneapi/vtune/latest/apsvars.sh”

Run Application

- >aps <application and args>
- MPI: >mpirun <mpi options> aps <application and args>

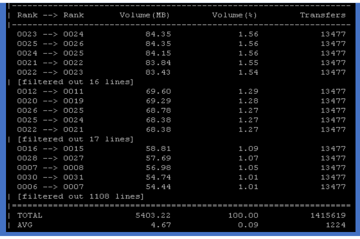
Generate summary report for result_folder

- >aps --report <result_folder>



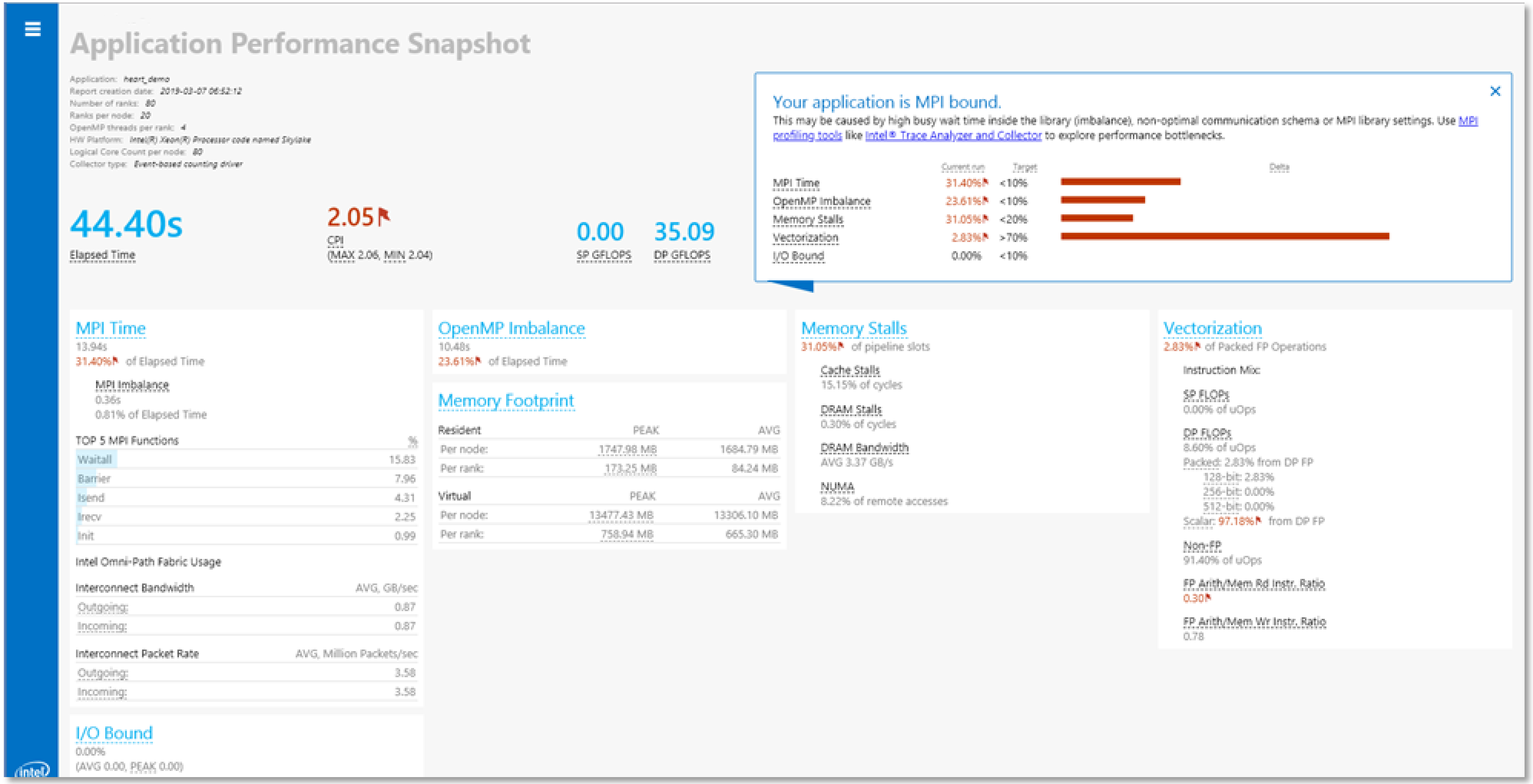
Generate CL reports with detailed MPI statistics for a result_folder

- aps-report --<option> <result_folder>



```
|-----------------------------------------------------------------------
| Rank --> Rank        Volume(MB)          Volume(%)           Transfers
|-----------------------------------------------------------------------
  0023 --> 0024             84.35               1.56               13477
  0025 --> 0026             84.35               1.56               13477
  0024 --> 0025             84.15               1.56               13477
  0021 --> 0022             83.84               1.55               13477
  0022 --> 0023             83.43               1.54               13477
| [filtered out 16 lines]
  0012 --> 0011             69.60               1.29               13477
  0020 --> 0019             69.29               1.28               13477
  0026 --> 0025             68.78               1.27               13477
  0025 --> 0024             68.38               1.27               13477
  0022 --> 0021             68.38               1.27               13477
| [filtered out 17 lines]
  0016 --> 0015             58.81               1.09               13477
  0028 --> 0027             57.69               1.07               13477
  0007 --> 0008             56.98               1.05               13477
  0030 --> 0031             54.74               1.01               13477
  0006 --> 0007             54.44               1.01               13477
| [filtered out 1108 lines]
|=======================================================================
| TOTAL                   5403.22             100.00             1415619
| AVG                        4.67               0.09                1224
```

APS HTML Report



Application Performance Snapshot

Application: *heart_demo*
Report creation date: *2019-03-07 06:52:12*
Number of ranks: *80*
Ranks per node: *20*
OpenMP threads per rank: *4*
HW Platform: *Intel(R) Xeon(R) Processor code named Skylake*
Logical Core Count per node: *80*
Collector type: *Event-based counting driver*

44.40s
Elapsed Time

2.05
CPI
(MAX 2.06, MIN 2.04)

0.00
SP GFLOPS

35.09
DP GFLOPS

Your application is MPI bound.

This may be caused by high busy wait time inside the library (imbalance), non-optimal communication schema or MPI library settings. Use MPI profiling tools like Intel® Trace Analyzer and Collector to explore performance bottlenecks.

	Current run	Target	Delta
MPI Time	31.40%	<10%	
OpenMP Imbalance	23.61%	<10%	
Memory Stalls	31.05%	<20%	
Vectorization	2.83%	>70%	
I/O Bound	0.00%	<10%	

MPI Time

13.94s
31.40% of Elapsed Time

MPI Imbalance
0.36s
0.81% of Elapsed Time

TOP 5 MPI Functions	%
Waitall	15.83
Barrier	7.96
Isend	4.31
Irecv	2.25
Init	0.99

Intel Omni-Path Fabric Usage

Interconnect Bandwidth	AVG, GB/sec
Outgoing:	0.87
Incoming:	0.87

Interconnect Packet Rate	AVG, Million Packets/sec
Outgoing:	3.58
Incoming:	3.58

I/O Bound

0.00%
(AVG 0.00, PEAK 0.00)

OpenMP Imbalance

10.48s
23.61% of Elapsed Time

Memory Footprint

Resident	PEAK	AVG
Per node:	1747.98 MB	1684.79 MB
Per rank:	173.25 MB	84.24 MB

Virtual	PEAK	AVG
Per node:	13477.43 MB	13306.10 MB
Per rank:	758.94 MB	665.30 MB

Memory Stalls

31.05% of pipeline slots

Cache Stalls
15.15% of cycles

DRAM Stalls
0.30% of cycles

DRAM Bandwidth
AVG 3.37 GB/s

NUMA
8.22% of remote accesses

Vectorization

2.83% of Packed FP Operations

Instruction Mix

SP FLOPs
0.00% of uOps

DP FLOPs
8.60% of uOps
Packed: 2.83% from DP FP
128-bit: 2.83%
256-bit: 0.00%
512-bit: 0.00%
Scalar: 97.18% from DP FP

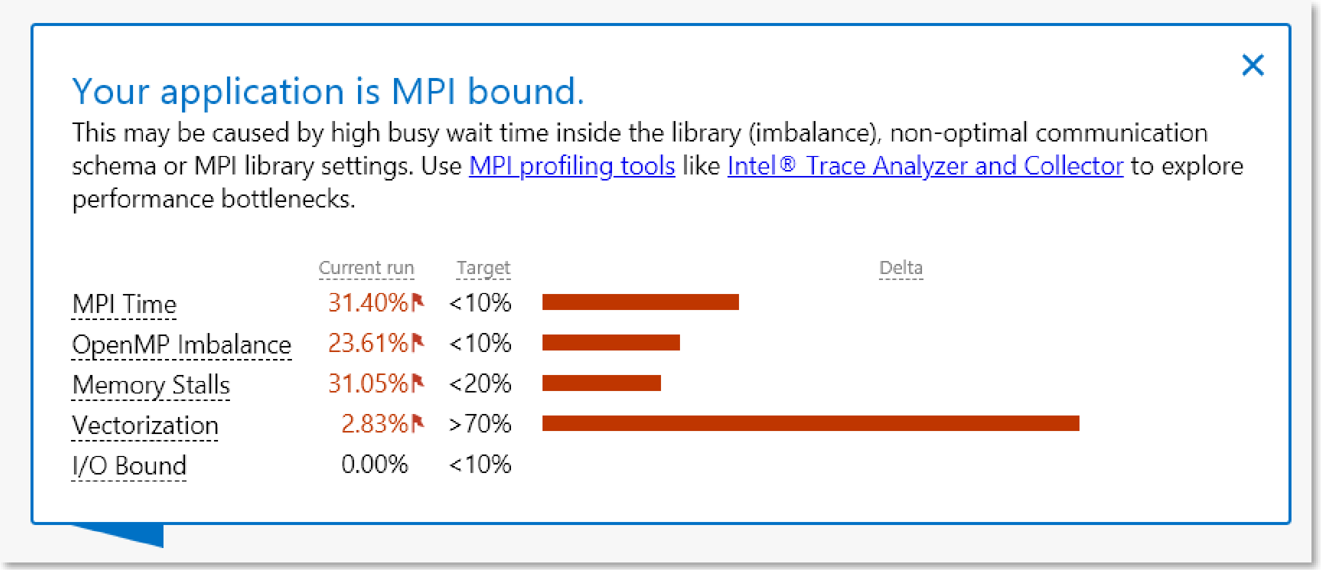
Non-FP
91.40% of uOps

FP Arith/Mem Rd Instr. Ratio
0.30

FP Arith/Mem Wr Instr. Ratio
0.78

APS HTML Report Breakdown - Overview

- Overview shows all areas and relative impact on code performance
- Provides recommendation for next step in performance analysis
- “X” collapses the summary, removing the flags (objective numbers only)



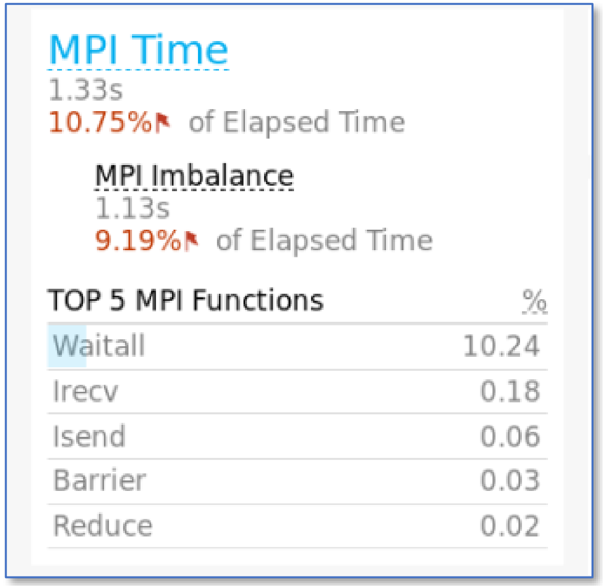
Your application is MPI bound.

This may be caused by high busy wait time inside the library (imbalance), non-optimal communication schema or MPI library settings. Use MPI profiling tools like Intel® Trace Analyzer and Collector to explore performance bottlenecks.

	Current run	Target	Delta
MPI Time	31.40%⚑	<10%	
OpenMP Imbalance	23.61%⚑	<10%	
Memory Stalls	31.05%⚑	<20%	
Vectorization	2.83%⚑	>70%	
I/O Bound	0.00%	<10%	

APS HTML Report Breakdown – Parallel Runtimes

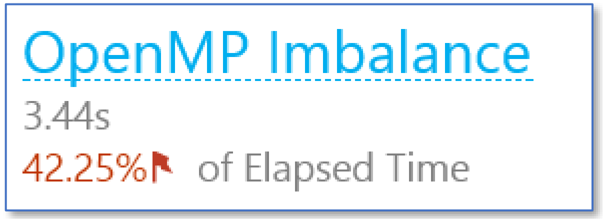
- MPI Time
  - How much time was spent in MPI calls
  - Averaged by ranks with % of Elapsed time
  - Available for MPICH-based MPI and OpenMPI
- MPI Imbalance
  - Unproductive time spent in MPI library waiting for data
    - Switched off by default
    - Available for Intel MPI with APS_IMBALANCE_TYPE=1
    - Over supported MPISs with APS_IMBALANCE_TYPE=2
- OpenMP Imbalance
  - Time spent at OpenMP Synchronization Barriers normalized by number of threads
  - Available for Intel OpenMP
- Serial time
  - Time spend outside OpenMP regions
  - Available for Intel OpenMP, shared memory applications only



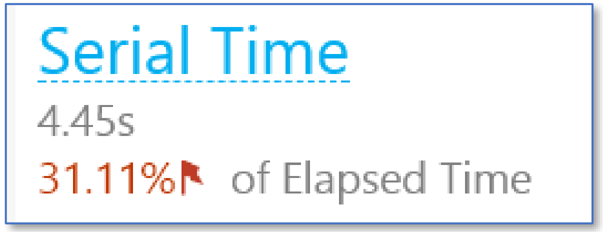
MPI Time
1.33s
10.75% of Elapsed Time

MPI Imbalance
1.13s
9.19% of Elapsed Time

TOP 5 MPI Functions	%
Waitall	10.24
Irecv	0.18
Isend	0.06
Barrier	0.03
Reduce	0.02



OpenMP Imbalance
3.44s
42.25% of Elapsed Time

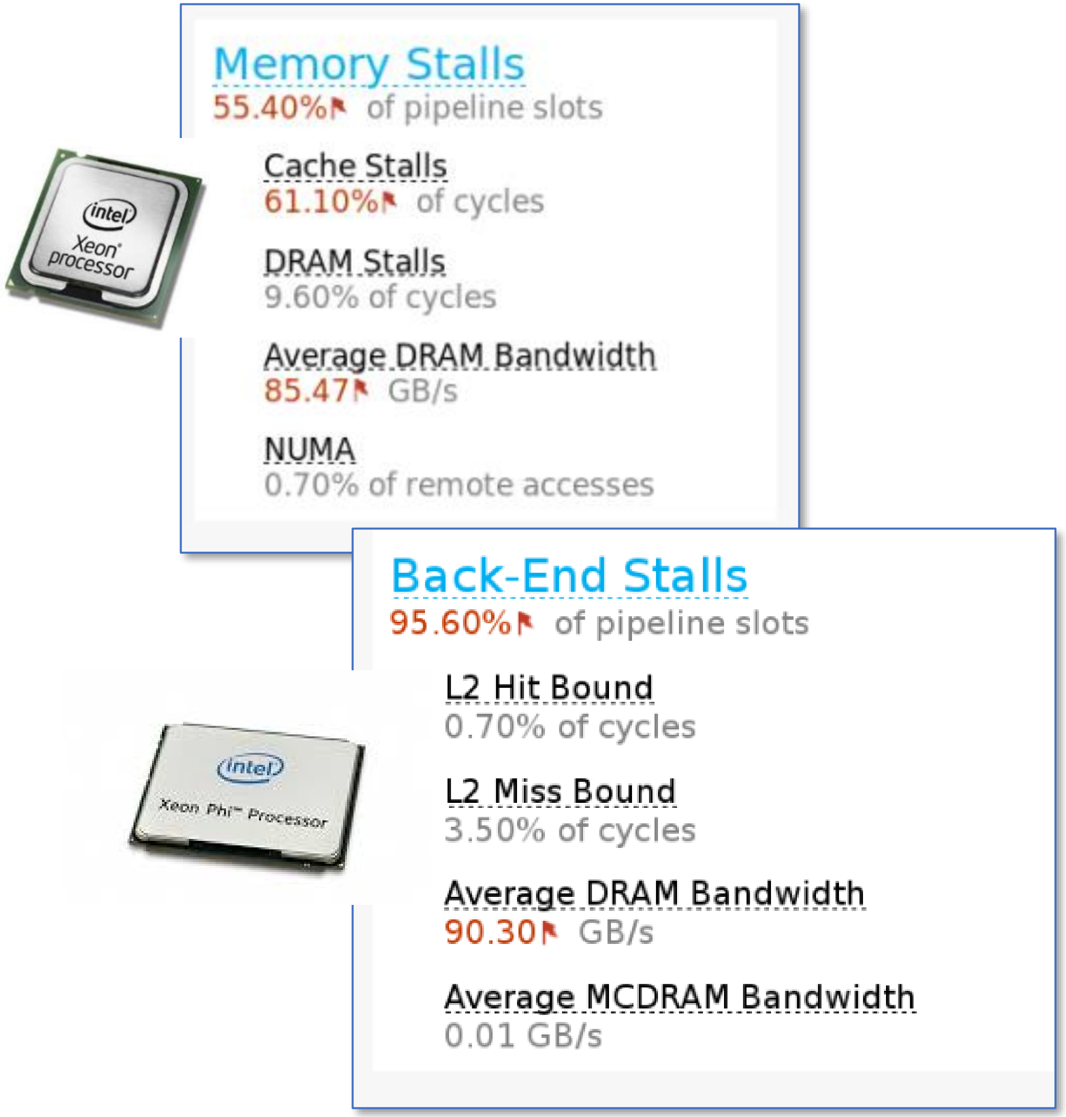


Serial Time
4.45s
31.11% of Elapsed Time

APS HTML Report Breakdown – Memory Access

- Memory stalls measurement with breakdown by cache and DRAM
- Average DRAM Bandwidth*
- NUMA ratio
- Xeon Phi (KNL/KNM):
  - back-end stalls with L2-demand access efficiency
  - Average DRAM AND MCDRAM Bandwidth*

*Average DRAM and MCDRAM bandwidth collection is available with Intel driver or perf system wide monitoring enabled on a system



Memory Stalls
55.40% of pipeline slots

Cache Stalls
61.10% of cycles

DRAM Stalls
9.60% of cycles

Average DRAM Bandwidth
85.47 GB/s

NUMA
0.70% of remote accesses

Back-End Stalls
95.60% of pipeline slots

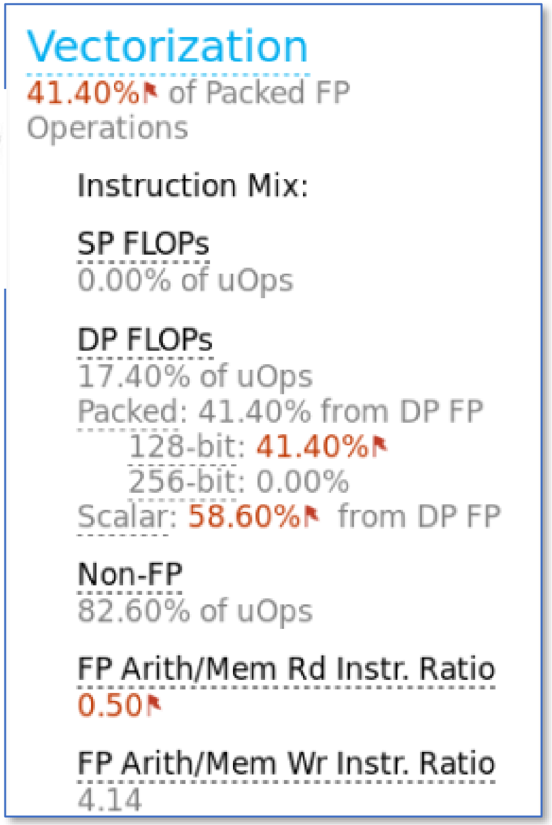
L2 Hit Bound
0.70% of cycles

L2 Miss Bound
3.50% of cycles

Average DRAM Bandwidth
90.30 GB/s

Average MCDRAM Bandwidth
0.01 GB/s

APS HTML Report Breakdown – Vectorization



Vectorization
41.40% of Packed FP Operations

Instruction Mix:

SP FLOPs
0.00% of uOps

DP FLOPs
17.40% of uOps
Packed: 41.40% from DP FP
128-bit: 41.40%
256-bit: 0.00%
Scalar: 58.60% from DP FP

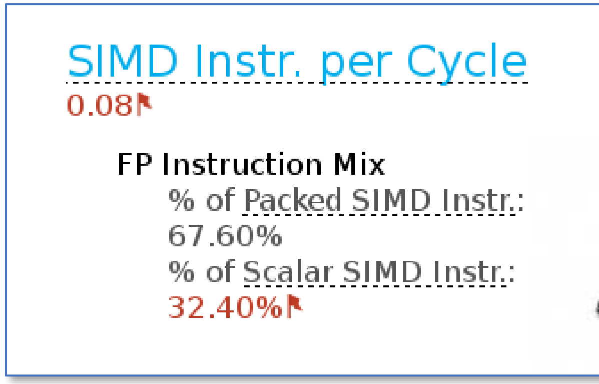
Non-FP
82.60% of uOps

FP Arith/Mem Rd Instr. Ratio
0.50

FP Arith/Mem Wr Instr. Ratio
4.14

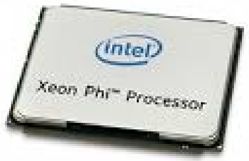
- Vectorization efficiency based on HW-event statistics with
  - Breakdown by vector/scalar instructions
  - Floating point vs memory instruction ratio

- SIMD Instr. per Cycle
  - Scalar vs. vectorized instructions

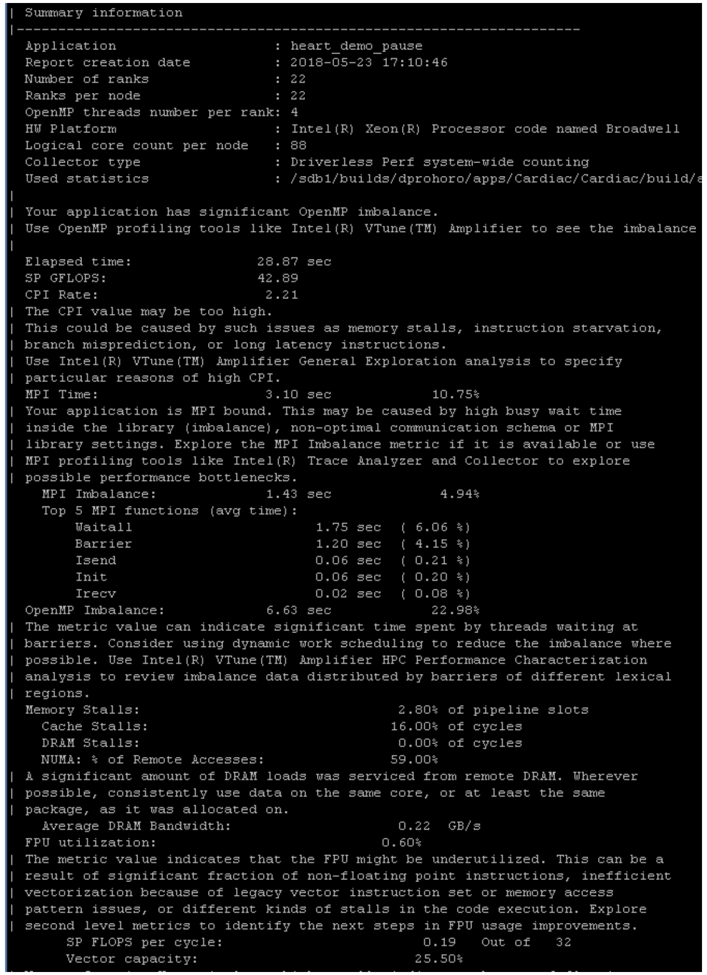


SIMD Instr. per Cycle
0.08

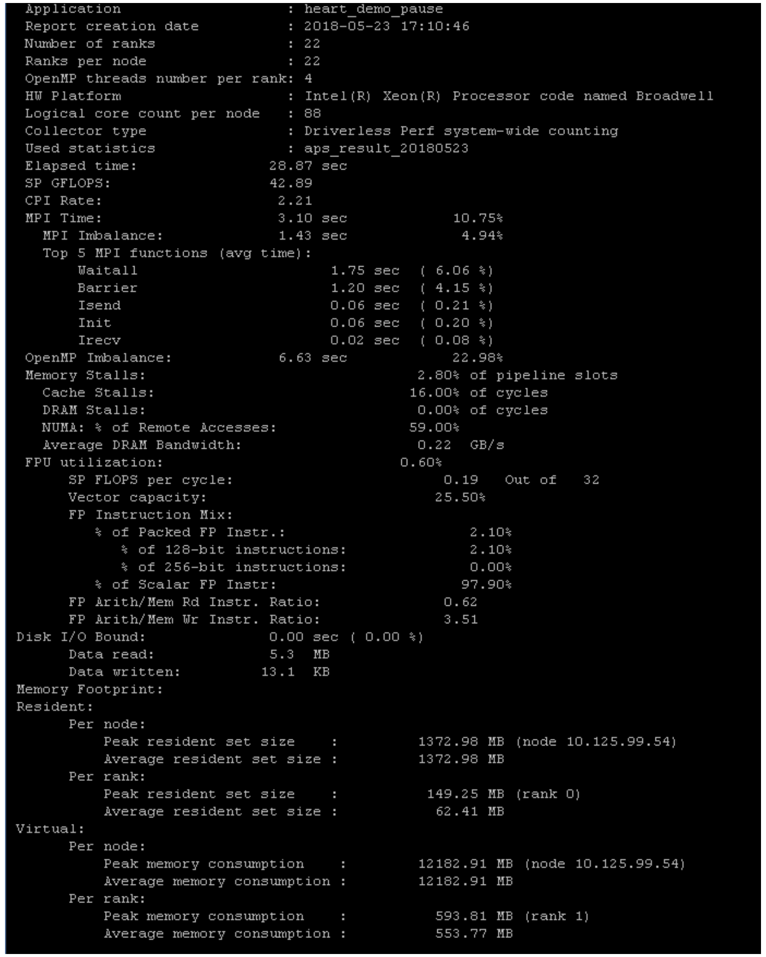
FP Instruction Mix
% of Packed SIMD Instr.:
67.60%
% of Scalar SIMD Instr.:
32.40%



APS Command Line Reports - Summary



```
| Summary information
|--------------------------------------------------------------------
  Application                   : heart_demo_pause
  Report creation date          : 2018-05-23 17:10:46
  Number of ranks               : 22
  Ranks per node                : 22
  OpenMP threads number per rank: 4
  HW Platform                   : Intel(R) Xeon(R) Processor code named Broadwell
  Logical core count per node   : 88
  Collector type                : Driverless Perf system-wide counting
  Used statistics               : /sdb1/builds/dprohoro/apps/Cardiac/Cardiac/build/a
|
| Your application has significant OpenMP imbalance.
| Use OpenMP profiling tools like Intel(R) VTune(TM) Amplifier to see the imbalance
|
  Elapsed time:               28.87 sec
  SP GFLOPS:                  42.89
  CPI Rate:                    2.21
| The CPI value may be too high.
| This could be caused by such issues as memory stalls, instruction starvation,
| branch misprediction, or long latency instructions.
| Use Intel(R) VTune(TM) Amplifier General Exploration analysis to specify
| particular reasons of high CPI.
  MPI Time:                    3.10 sec            10.75%
| Your application is MPI bound. This may be caused by high busy wait time
| inside the library (imbalance), non-optimal communication schema or MPI
| library settings. Explore the MPI Imbalance metric if it is available or use
| MPI profiling tools like Intel(R) Trace Analyzer and Collector to explore
| possible performance bottlenecks.
    MPI Imbalance:             1.43 sec             4.94%
    Top 5 MPI functions (avg time):
        Waitall                      1.75 sec  ( 6.06 %)
        Barrier                      1.20 sec  ( 4.15 %)
        Isend                        0.06 sec  ( 0.21 %)
        Init                         0.06 sec  ( 0.20 %)
        Irecv                        0.02 sec  ( 0.08 %)
  OpenMP Imbalance:            6.63 sec            22.98%
| The metric value can indicate significant time spent by threads waiting at
| barriers. Consider using dynamic work scheduling to reduce the imbalance where
| possible. Use Intel(R) VTune(TM) Amplifier HPC Performance Characterization
| analysis to review imbalance data distributed by barriers of different lexical
| regions.
  Memory Stalls:                                  2.80% of pipeline slots
    Cache Stalls:                                16.00% of cycles
    DRAM Stalls:                                  0.00% of cycles
    NUMA: % of Remote Accesses:                  59.00%
| A significant amount of DRAM loads was serviced from remote DRAM. Wherever
| possible, consistently use data on the same core, or at least the same
| package, as it was allocated on.
    Average DRAM Bandwidth:                       0.22  GB/s
  FPU utilization:                              0.60%
| The metric value indicates that the FPU might be underutilized. This can be a
| result of significant fraction of non-floating point instructions, inefficient
| vectorization because of legacy vector instruction set or memory access
| pattern issues, or different kinds of stalls in the code execution. Explore
| second level metrics to identify the next steps in FPU usage improvements.
       SP FLOPS per cycle:                           0.19   Out of   32
       Vector capacity:                             25.50%
```



```
  Application                   : heart_demo_pause
  Report creation date          : 2018-05-23 17:10:46
  Number of ranks               : 22
  Ranks per node                : 22
  OpenMP threads number per rank: 4
  HW Platform                   : Intel(R) Xeon(R) Processor code named Broadwell
  Logical core count per node   : 88
  Collector type                : Driverless Perf system-wide counting
  Used statistics               : aps_result_20180523
  Elapsed time:               28.87 sec
  SP GFLOPS:                  42.89
  CPI Rate:                    2.21
  MPI Time:                    3.10 sec            10.75%
    MPI Imbalance:             1.43 sec             4.94%
    Top 5 MPI functions (avg time):
        Waitall                      1.75 sec  ( 6.06 %)
        Barrier                      1.20 sec  ( 4.15 %)
        Isend                        0.06 sec  ( 0.21 %)
        Init                         0.06 sec  ( 0.20 %)
        Irecv                        0.02 sec  ( 0.08 %)
  OpenMP Imbalance:            6.63 sec            22.98%
  Memory Stalls:                                  2.80% of pipeline slots
    Cache Stalls:                                16.00% of cycles
    DRAM Stalls:                                  0.00% of cycles
    NUMA: % of Remote Accesses:                  59.00%
    Average DRAM Bandwidth:                       0.22  GB/s
  FPU utilization:                              0.60%
       SP FLOPS per cycle:                           0.19   Out of   32
       Vector capacity:                             25.50%
       FP Instruction Mix:
          % of Packed FP Instr.:                        2.10%
             % of 128-bit instructions:                 2.10%
             % of 256-bit instructions:                 0.00%
          % of Scalar FP Instr:                        97.90%
       FP Arith/Mem Rd Instr. Ratio:                 0.62
       FP Arith/Mem Wr Instr. Ratio:                 3.51
 Disk I/O Bound:               0.00 sec ( 0.00 %)
       Data read:              5.3   MB
       Data written:          13.1   KB
 Memory Footprint:
 Resident:
       Per node:
           Peak resident set size    :         1372.98 MB (node 10.125.99.54)
           Average resident set size :         1372.98 MB
       Per rank:
           Peak resident set size    :          149.25 MB (rank 0)
           Average resident set size :           62.41 MB
 Virtual:
       Per node:
           Peak memory consumption    :        12182.91 MB (node 10.125.99.54)
           Average memory consumption :        12182.91 MB
       Per rank:
           Peak memory consumption    :          593.81 MB (rank 1)
           Average memory consumption :          553.77 MB
```

Tip:

>aps –report=<my_result_dir> | grep –v “|”
eliminating verbose descriptions

APS Command Line Reports – Detailed MPI statistics

aps-report [keys] [options] <result>

[keys] – what to show

--functions
--mpi-time-per-rank
--message-sizes
--transfers-per-communication
--transfers-per-rank
--node-to-node
--transfers-per-function
--communicators-list

[options] – how to show

--rank
--comm-id
--details
--communicators
--volume-threshold
--time-threshold
--number-of-lines
--no-filters
--communicators-list
--format

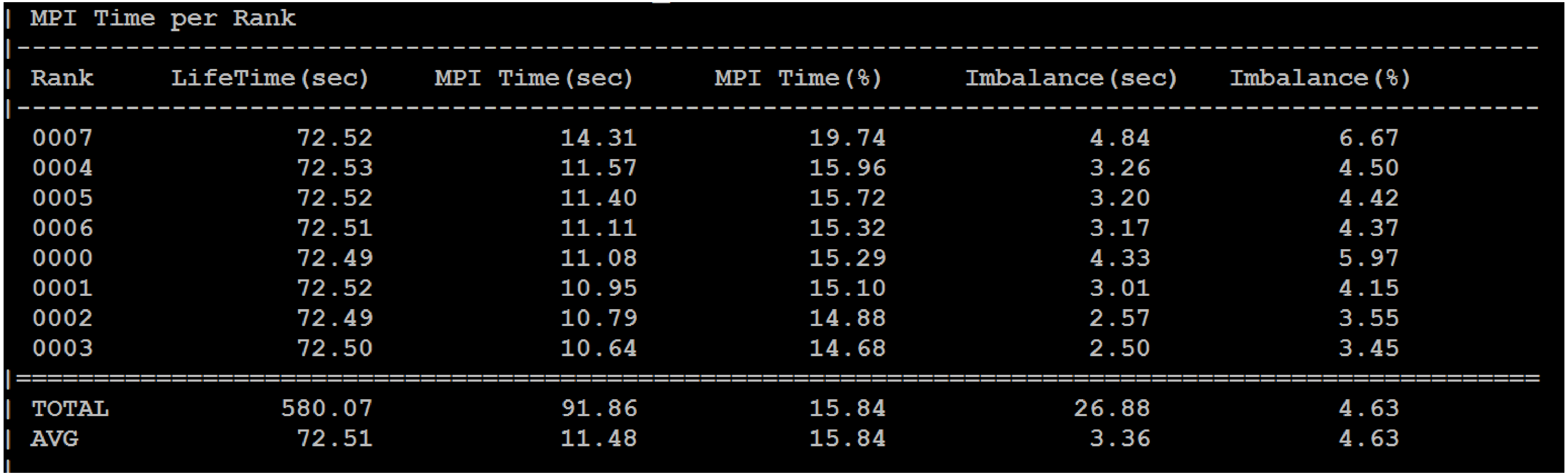
See descriptions with >aps-report command

Please note: some reports are available with non-default MPS_STAT_LEVEL=1

APS Command Line Reports – Detailed MPI statistics (1/4) Report examples

- MPI Time per rank

  >aps-report --mpi-time-per-rank <result>

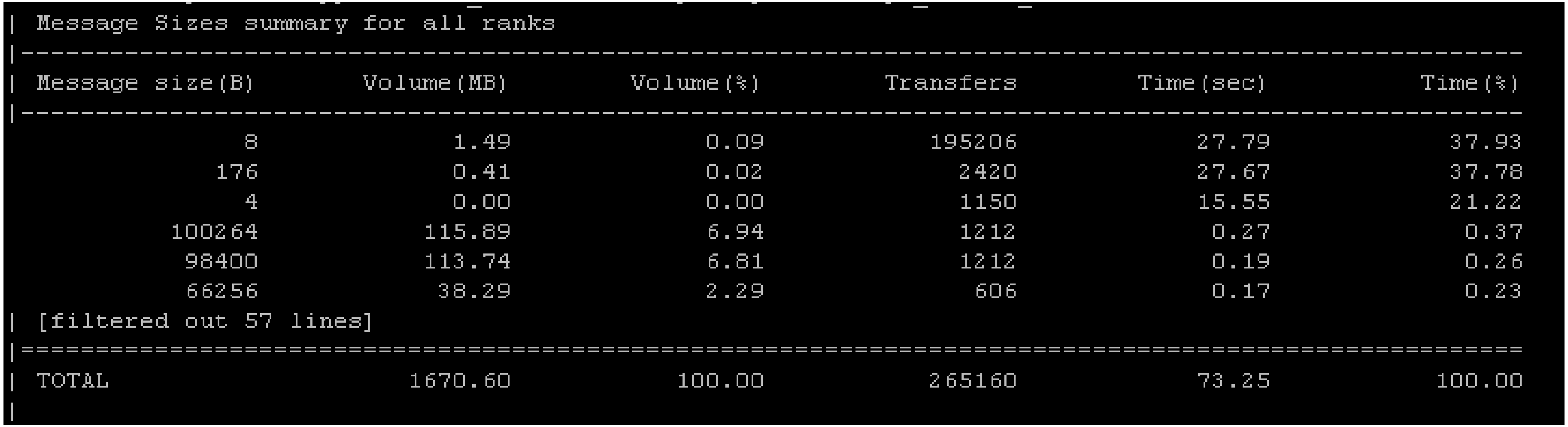


```
| MPI Time per Rank
|----------------------------------------------------------------------------------
| Rank      LifeTime(sec)     MPI Time(sec)       MPI Time(%)       Imbalance(sec)    Imbalance(%)
|----------------------------------------------------------------------------------
  0007              72.52            14.31              19.74                 4.84            6.67
  0004              72.53            11.57              15.96                 3.26            4.50
  0005              72.52            11.40              15.72                 3.20            4.42
  0006              72.51            11.11              15.32                 3.17            4.37
  0000              72.49            11.08              15.29                 4.33            5.97
  0001              72.52            10.95              15.10                 3.01            4.15
  0002              72.49            10.79              14.88                 2.57            3.55
  0003              72.50            10.64              14.68                 2.50            3.45
|==================================================================================
| TOTAL            580.07            91.86              15.84                26.88            4.63
| AVG               72.51            11.48              15.84                 3.36            4.63
|
```

APS Command Line Reports – Detailed MPI statistics (2/4)

- Message Size Summary by all ranks

  >aps-report --message-sizes <result>



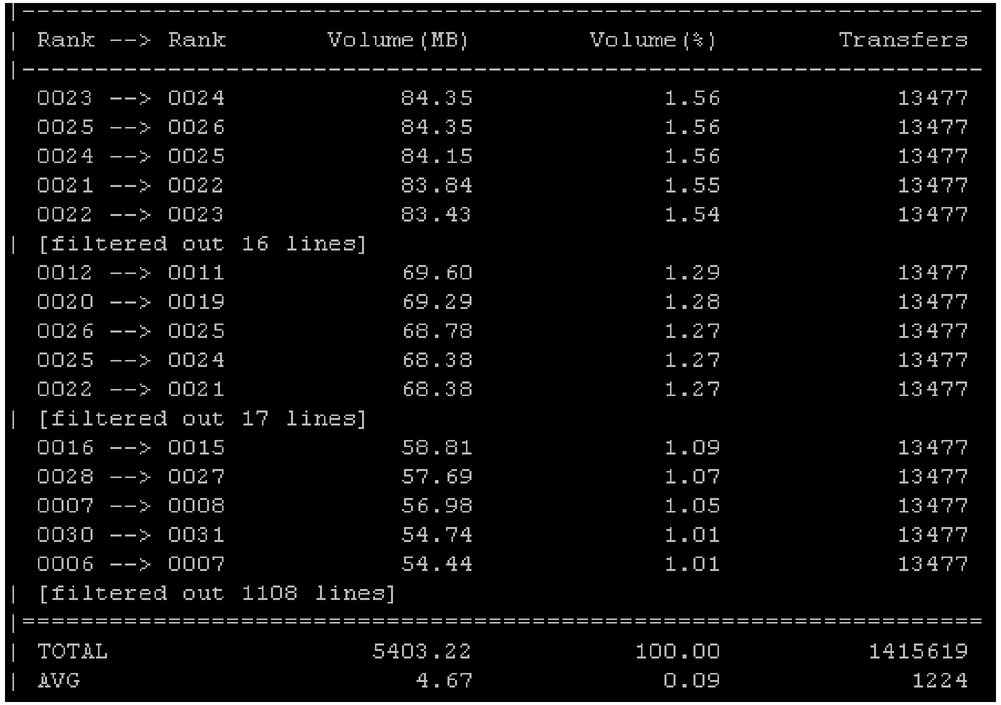
```
| Message Sizes summary for all ranks
|-------------------------------------------------------------------------------------------------------
| Message size(B)        Volume(MB)        Volume(%)         Transfers         Time(sec)           Time(%)
|-------------------------------------------------------------------------------------------------------
                8              1.49             0.09            195206             27.79             37.93
              176              0.41             0.02              2420             27.67             37.78
                4              0.00             0.00              1150             15.55             21.22
           100264            115.89             6.94              1212              0.27              0.37
            98400            113.74             6.81              1212              0.19              0.26
            66256             38.29             2.29               606              0.17              0.23
| [filtered out 57 lines]
|=======================================================================================================
| TOTAL                     1670.60           100.00            265160             73.25            100.00
|
```

APS Command Line Reports – Detailed MPI statistics (3/4)

- Data Transfers for Rank-to-Rank Communication

  >aps-report --transfers-per-communication <result>

  Requires setting MPS_STAT_LEVEL=4 before collection launch

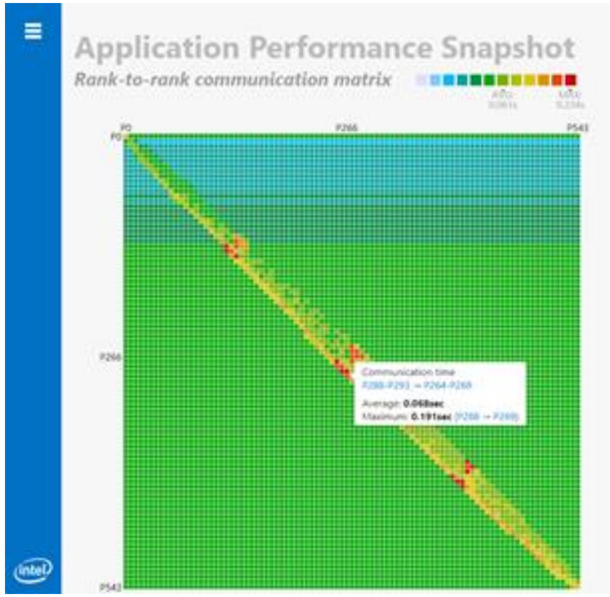


```
|-----------------------------------------------------------------------
| Rank --> Rank         Volume(MB)          Volume(%)          Transfers
|-----------------------------------------------------------------------
  0023 --> 0024              84.35               1.56              13477
  0025 --> 0026              84.35               1.56              13477
  0024 --> 0025              84.15               1.56              13477
  0021 --> 0022              83.84               1.55              13477
  0022 --> 0023              83.43               1.54              13477
| [filtered out 16 lines]
  0012 --> 0011              69.60               1.29              13477
  0020 --> 0019              69.29               1.28              13477
  0026 --> 0025              68.78               1.27              13477
  0025 --> 0024              68.38               1.27              13477
  0022 --> 0021              68.38               1.27              13477
| [filtered out 17 lines]
  0016 --> 0015              58.81               1.09              13477
  0028 --> 0027              57.69               1.07              13477
  0007 --> 0008              56.98               1.05              13477
  0030 --> 0031              54.74               1.01              13477
  0006 --> 0007              54.44               1.01              13477
| [filtered out 1108 lines]
|=======================================================================
| TOTAL                    5403.22             100.00            1415619
| AVG                         4.67               0.09               1224
```

APS Command Line Reports – Detailed MPI statistics (4/4)

- Data Transfers for Rank-to-Rank Communication – UI representation

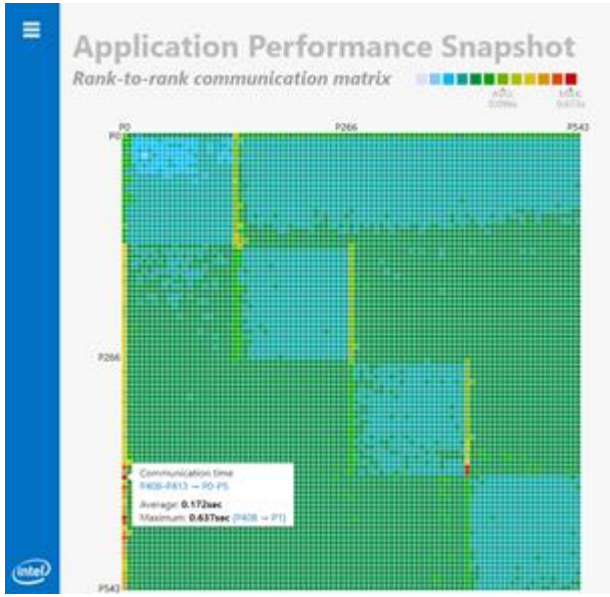
  >aps-report --transfers-per-communication --format=html <result>



Application Performance Snapshot

Rank-to-rank communication matrix

Communication time
P288-P293 → P264-P269
Average: **0.068sec**
Maximum: **0.191sec** (P288 → P269)



Application Performance Snapshot

Rank-to-rank communication matrix

Communication time
P408-P413 → P0-P5
Average: **0.172sec**
Maximum: **0.637sec** (P408 → P1)

use "-v" to generate the chart by volume

Requires setting MPS_STAT_LEVEL=4 before collection

Collection Control API

- To measure a particular application phase or exclude initialization/finalization phases use:

  MPI:

  - Pause: MPI_Pcontrol(0)
  - Resume: MPI_Pcontrol(1)

  MPI or Shared memory applications:

  - Pause: __itt_pause()
  - Resume: __itt_resume()
    - See how to configure the build of your application to use itt API

Tip: use aps “-start-paused” option allows to start application without profiling and skip initialization phase

Data collection selection to reduce overhead

- Use --collection-mode option to limit collection either by MPI or OpenMP tracing or HW-counters

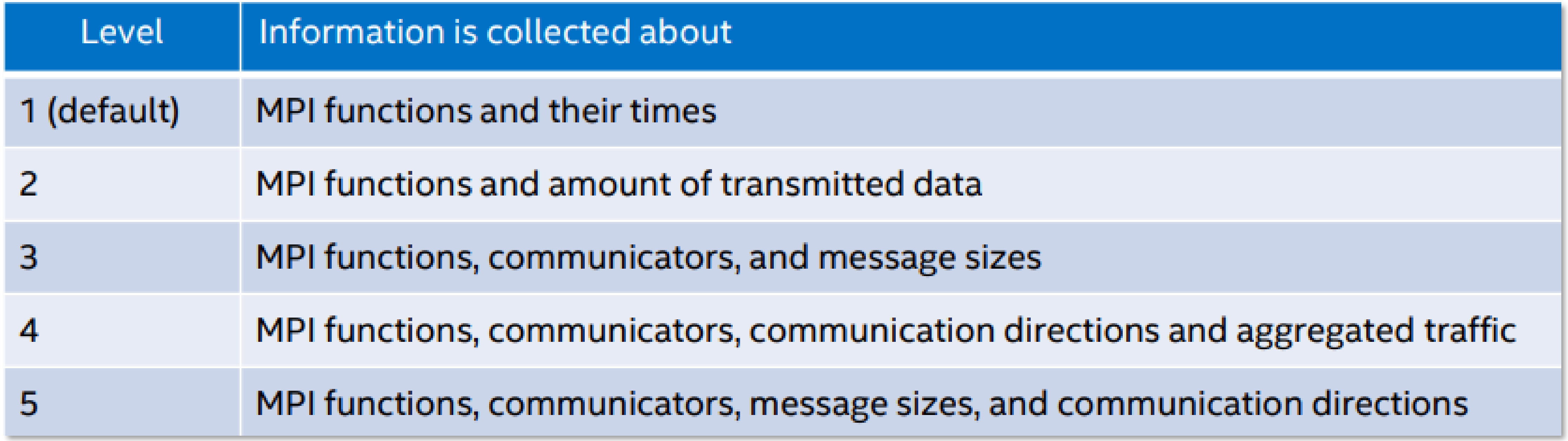
  - Use case: interest in MPI statistics only

    ```
    >mpirun –n 512 –ppn 24 aps --collection-mode=mpi <my_MPI_app>
    ```

    In this case APS will not collect HW counters – less overhead - so Memory Stalls and FLOPS/FPU Utilization will not be available in reports

Reducing collected data for MPI tracing

- >exprort MPS_STAT_LEVEL <Level>



Level	Information is collected about
1 (default)	MPI functions and their times
2	MPI functions and amount of transmitted data
3	MPI functions, communicators, and message sizes
4	MPI functions, communicators, communication directions and aggregated traffic
5	MPI functions, communicators, message sizes, and communication directions

Summary

Intel® VTune™ Profiler's Application Performance Snapshot is:

- Your entry point for HPC application performance analysis
- Simple and well-structured command line and HTML reports
- Clear next steps for tuning with connection to detailed performance tools
- Tool-of-choice of MPI efficiency analysis at scale