HLRN 05.11.2020

Intel® Application Performance Snapshot (APS)

Dr. Heinrich Bockhorst

Notices & Disclaimers

Intel technologies may require enabled hardware, software or service activation. Learn more at intel.com or from the OEM or retailer.

Your costs and results may vary.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

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. https://software.intel.com/en-us/articles/optimization-notice

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. See backup for configuration details. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.

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. No product or component can be absolutely secure.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

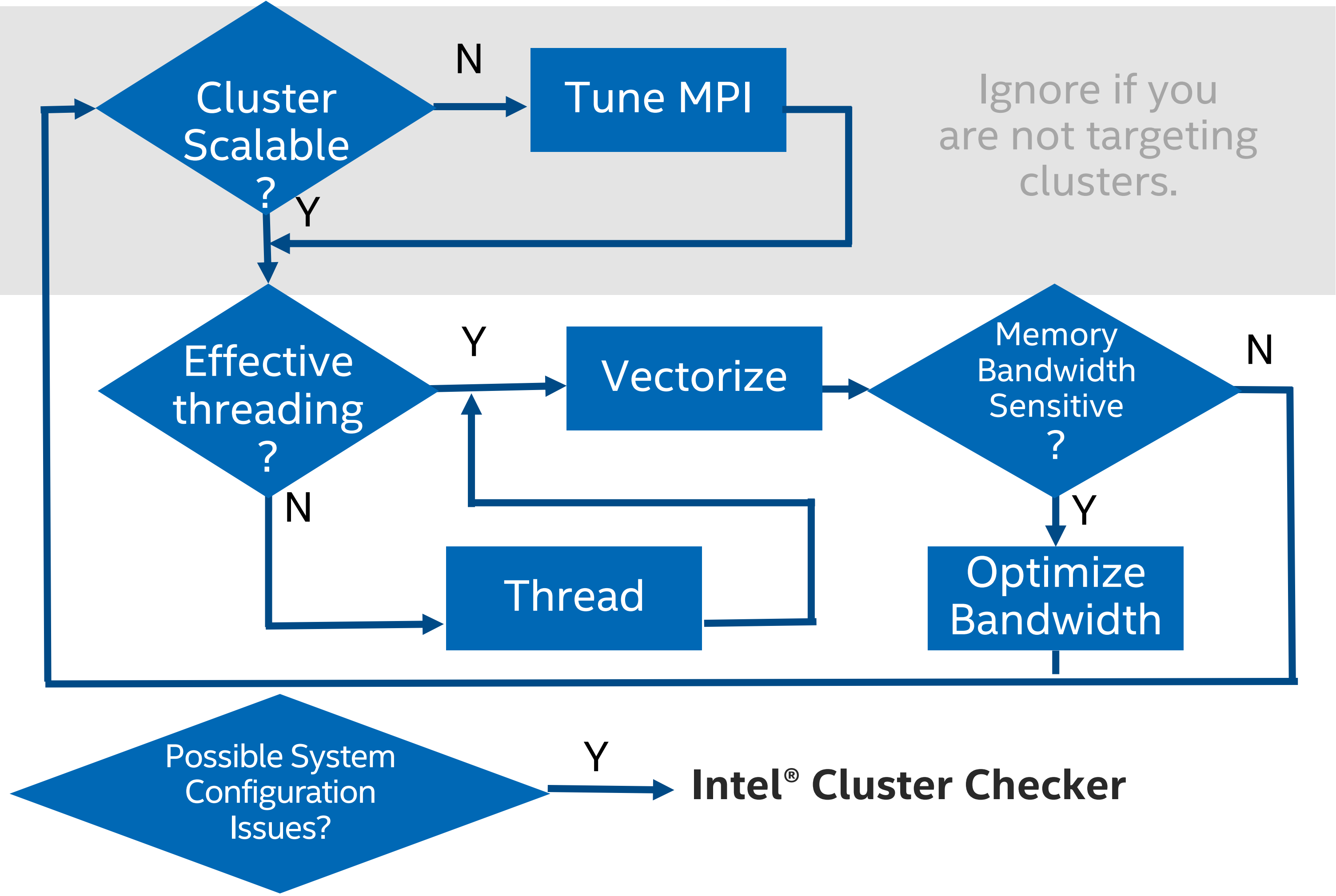
Agenda

- Which tool should I use?
- APS – first step of analysis
- Parallel Runtimes overview (MPI, OpenMP)
- Bandwidth and Memory Analysis
- Vectorization
- Detailed MPI statistics

Which tool should I use?

Optimizing Performance on Parallel Hardware

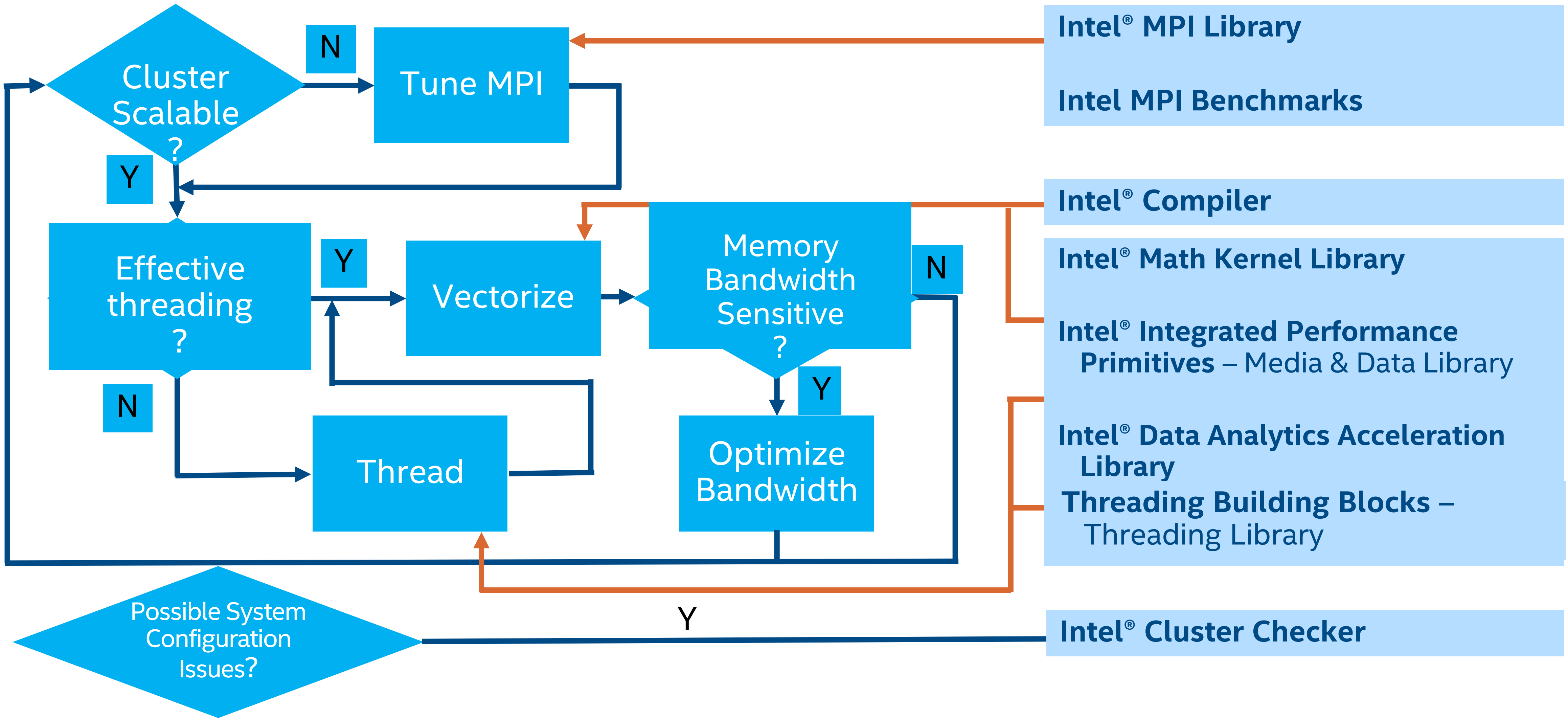
Intel® Parallel Studio XE—It's an iterative process...


Cluster Scalable ?
N
Tune MPI
Y
Ignore if you are not targeting clusters.
Effective threading ?
Y
Vectorize
Memory Bandwidth Sensitive ?
N
N
Thread
Y
Optimize Bandwidth
Possible System Configuration Issues?
Y

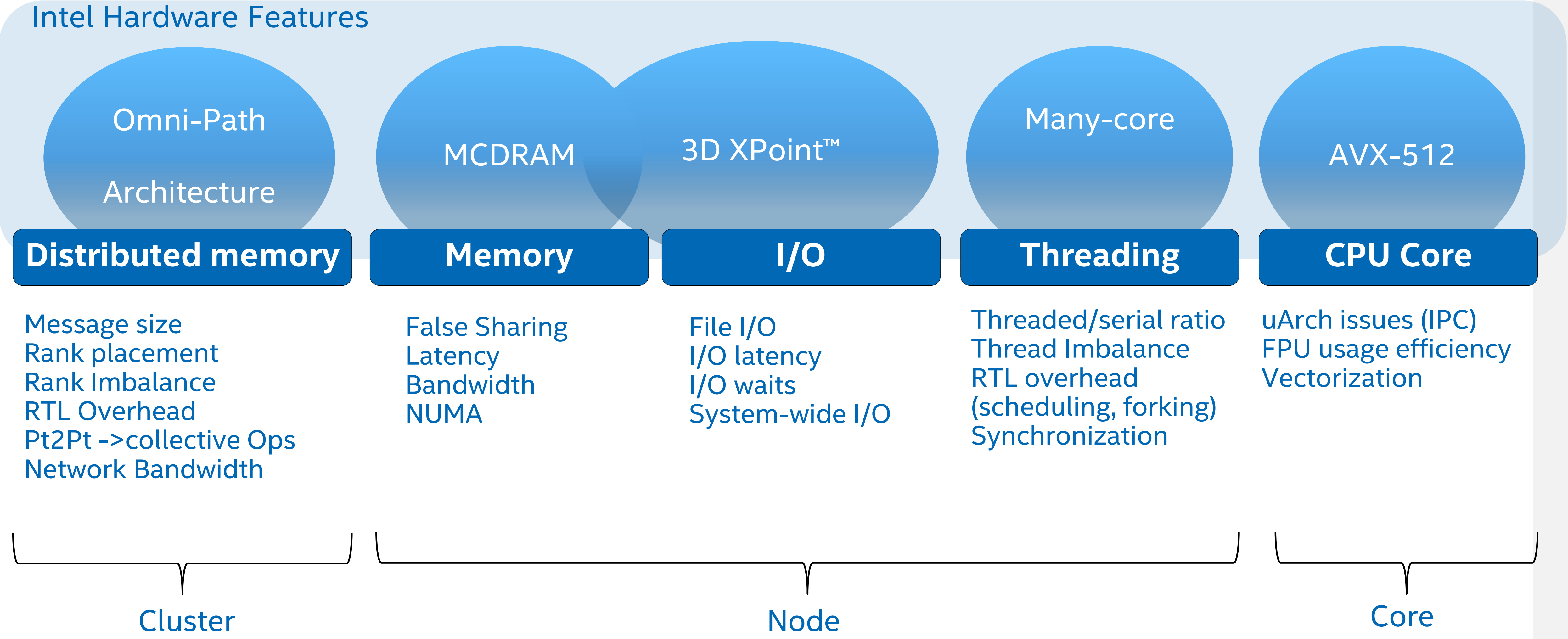
Intel® Cluster Checker

Tools for High Performance Implementation

Intel® Parallel Studio XE


Cluster Scalable ?
N
Tune MPI
Y
Effective threading ?
Y
Vectorize
Memory Bandwidth Sensitive ?
N
Y
N
Thread
Optimize Bandwidth
Possible System Configuration Issues?
Y
Intel® MPI Library
Intel MPI Benchmarks
Intel® Compiler
Intel® Math Kernel Library
Intel® Integrated Performance Primitives – Media & Data Library
Intel® Data Analytics Acceleration Library
Threading Building Blocks – Threading Library
Intel® Cluster Checker

ASPECTS OF HPC/THROUGHPUT APPLICATION PERFORMANCE



Intel Hardware Features

Omni-Path Architecture

MCDRAM

3D XPoint™

Many-core

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

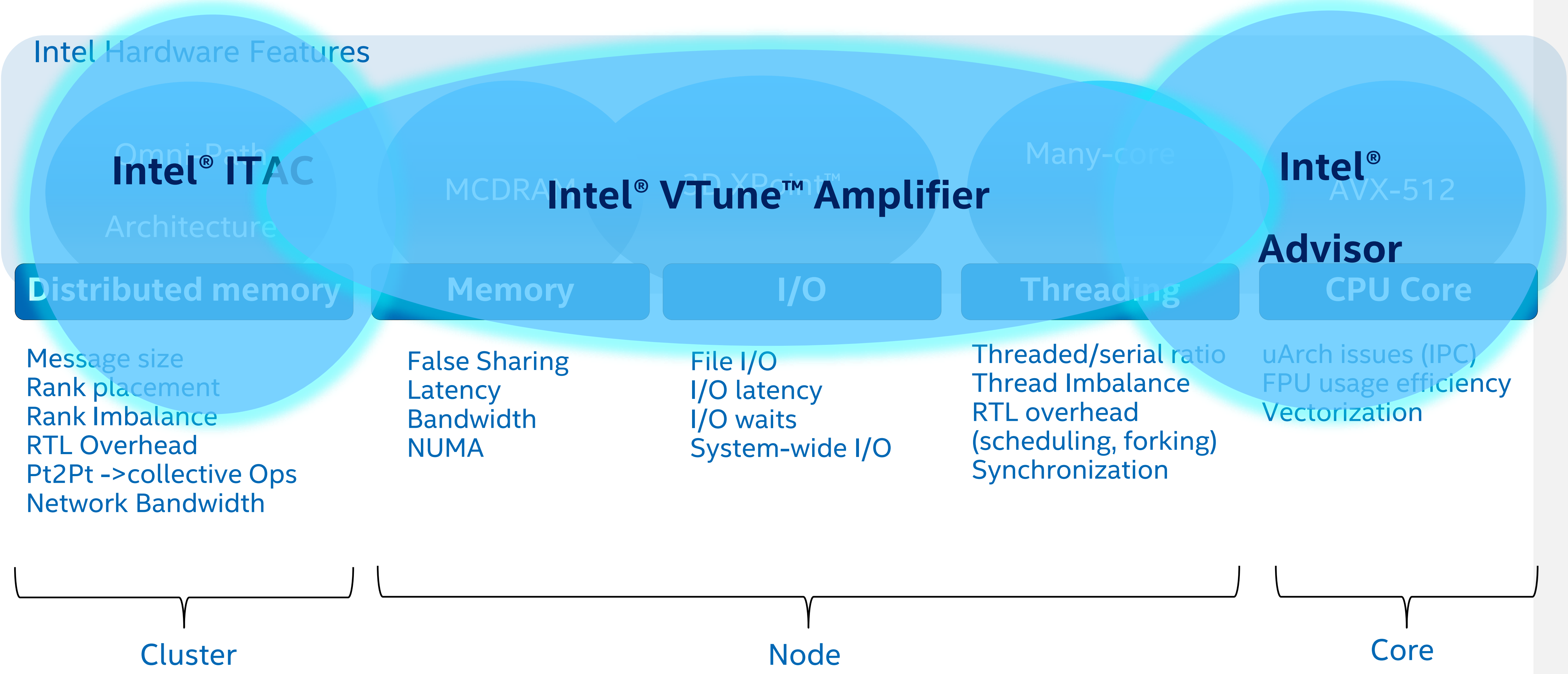
- uArch issues (IPC)
- FPU usage efficiency
- Vectorization

Cluster

Node

Core

ASPECTS OF HPC/THROUGHPUT APPLICATION PERFORMANCE



Intel Hardware Features

Omni-Path Architecture

MCDRAM

3D XPoint™

Many-core

AVX-512

Intel® ITAC

Intel® VTune™ Amplifier

Intel® Advisor

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

Before dive to a particular tool..

- How to assess easily any potential in performance tuning?
- What to use on big scale not be overwhelmed with huge trace size, post processing time and collection overhead?
- How to quickly evaluate environment settings or incremental code changes?
- Which tool should I use first?

- **Answer: try Application Performance Snapshot (APS)**

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
- Part of Intel® Parallel Studio XE, VTune Amplifier standalone

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 Usage

Setup Environment

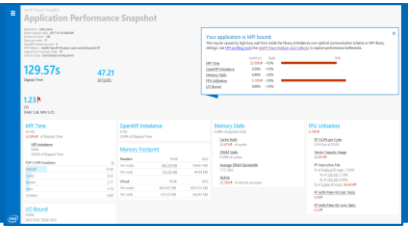
- module load vtune

Run Application

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

Generate Report on Result Folder

- > aps –report <result folder>



Application Performance Snapshot

129.57s

Elapsed Time

47.21

SP GFLOPS

1.23

CPI

Your application is MPI bound.

MPI Time

OpenMP Imbalance

Memory Stalls

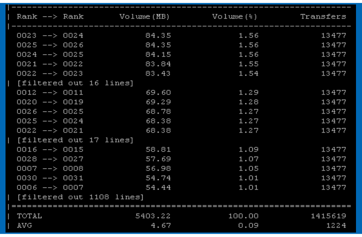
FPU Utilization

Memory Footprint

I/O Bound

Generate CL reports with detailed MPI statistics on Result Folder

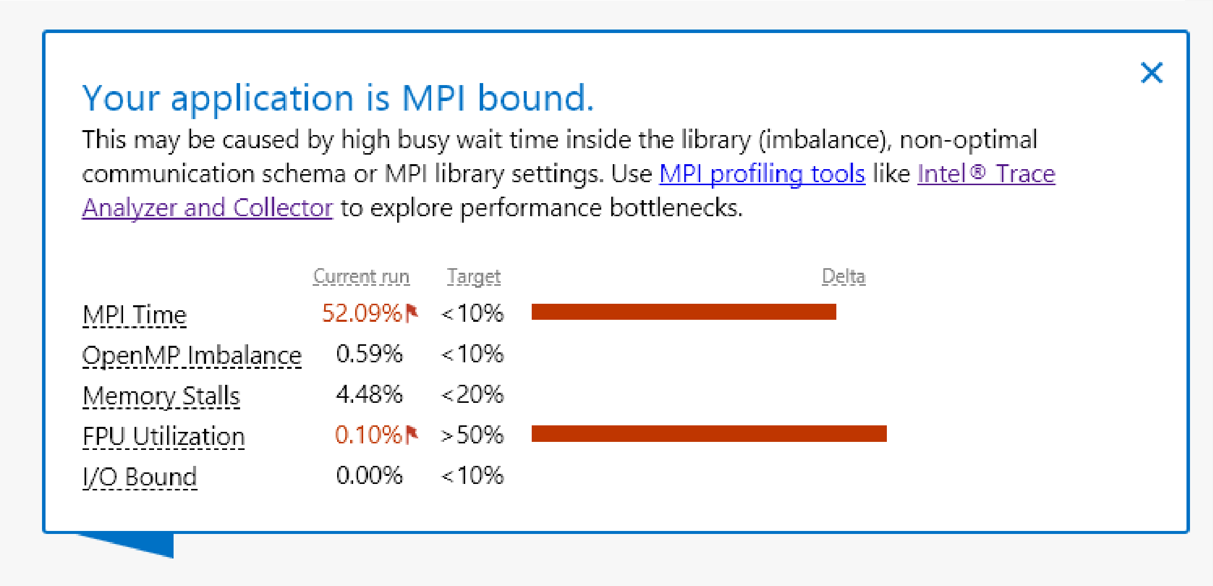
- aps-report –<option> <result folder>



```
|------------------------------------------------------------------
| 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 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)



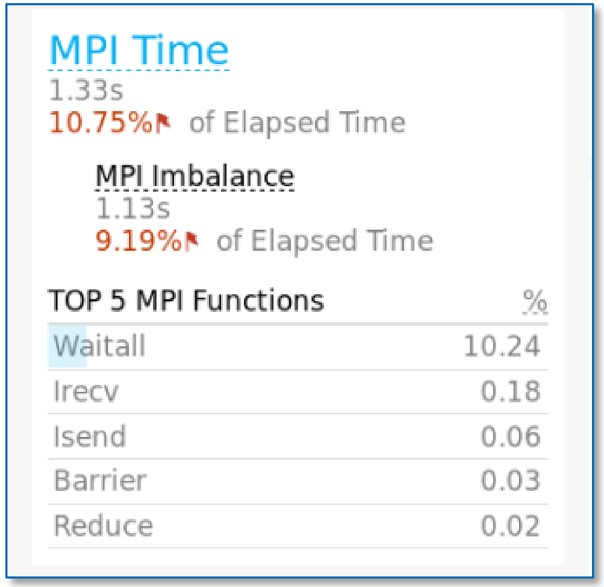
Your application is MPI bound.

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	52.09% ⚑	<10%	▬▬▬▬▬
OpenMP Imbalance	0.59%	<10%	
Memory Stalls	4.48%	<20%	
FPU Utilization	0.10% ⚑	>50%	▬▬▬▬▬▬
I/O Bound	0.00%	<10%	

APS HTML Report Breakdown – Parallel Runtimes

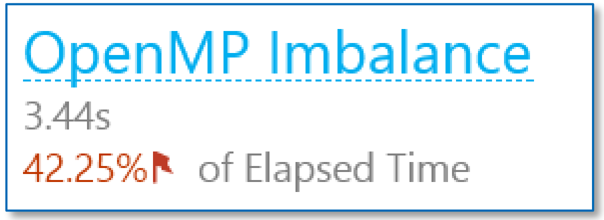
- MPI Time
  - Averaged by ranks with % of Elapsed time
  - Available for MPICH-based MPIs
- MPI Imbalance
  - Unproductive time spent in MPI library waiting for data
  - Available for Intel MPI
- OpenMP Imbalance
  - Time spent at OpenMP Synchronization Barriers normalized by number of threads (Intel OpenMP)
- Serial time
  - Time spend outside OpenMP regions (Intel OpenMP)



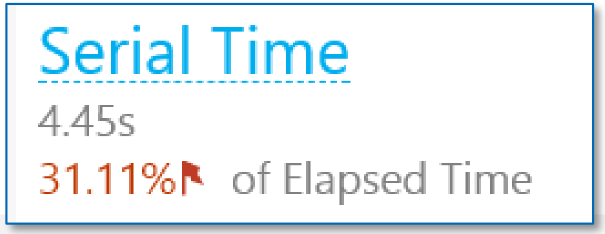
MPI Time
1.33s
10.75% of Elapsed Time

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

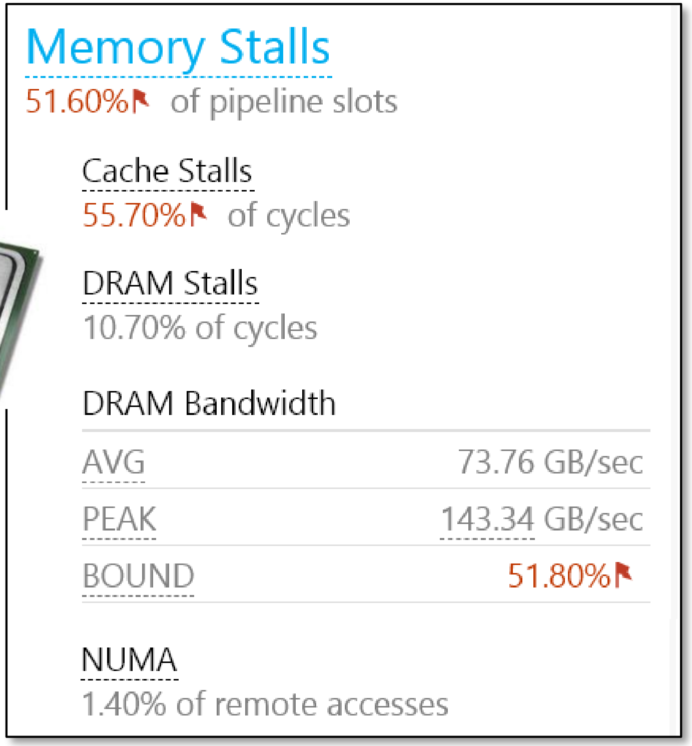


Serial Time
4.45s
31.11% of Elapsed Time

APS HTML Report Breakdown – Memory Access

- Memory stalls measurement with breakdown by cache and DRAM
- Average, Pick, Bound DRAM and Persistent Memory Bandwidth*
- NUMA ratio

*Available with Intel driver or Linux Perf system wide monitoring enabled on a system



Memory Stalls
51.60%⚑ of pipeline slots

Cache Stalls
55.70%⚑ of cycles

DRAM Stalls
10.70% of cycles

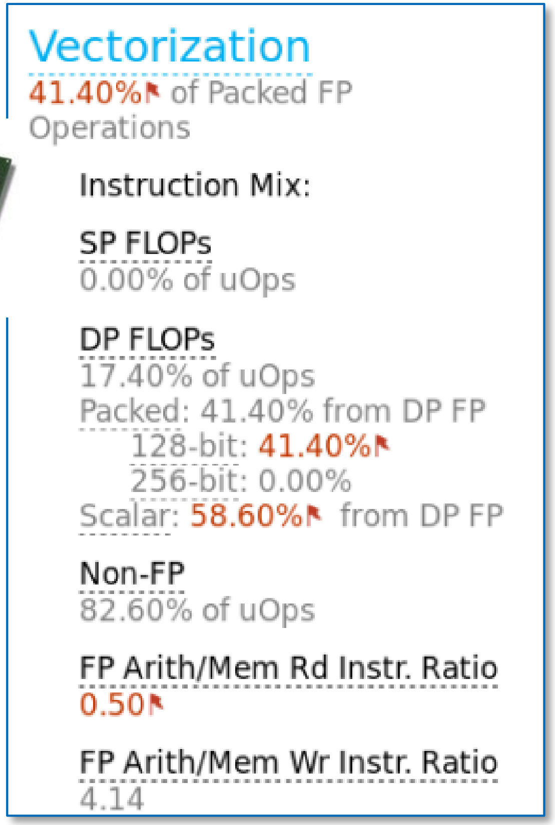
DRAM Bandwidth

AVG	73.76 GB/sec
PEAK	143.34 GB/sec
BOUND	51.80%⚑

NUMA
1.40% of remote accesses

APS HTML Report Breakdown – vectorization

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



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

APS Command Line Reports – Detailed MPI statistics

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

[keys] – what to show

--functions

--mpi_time_per_rank

--collop_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

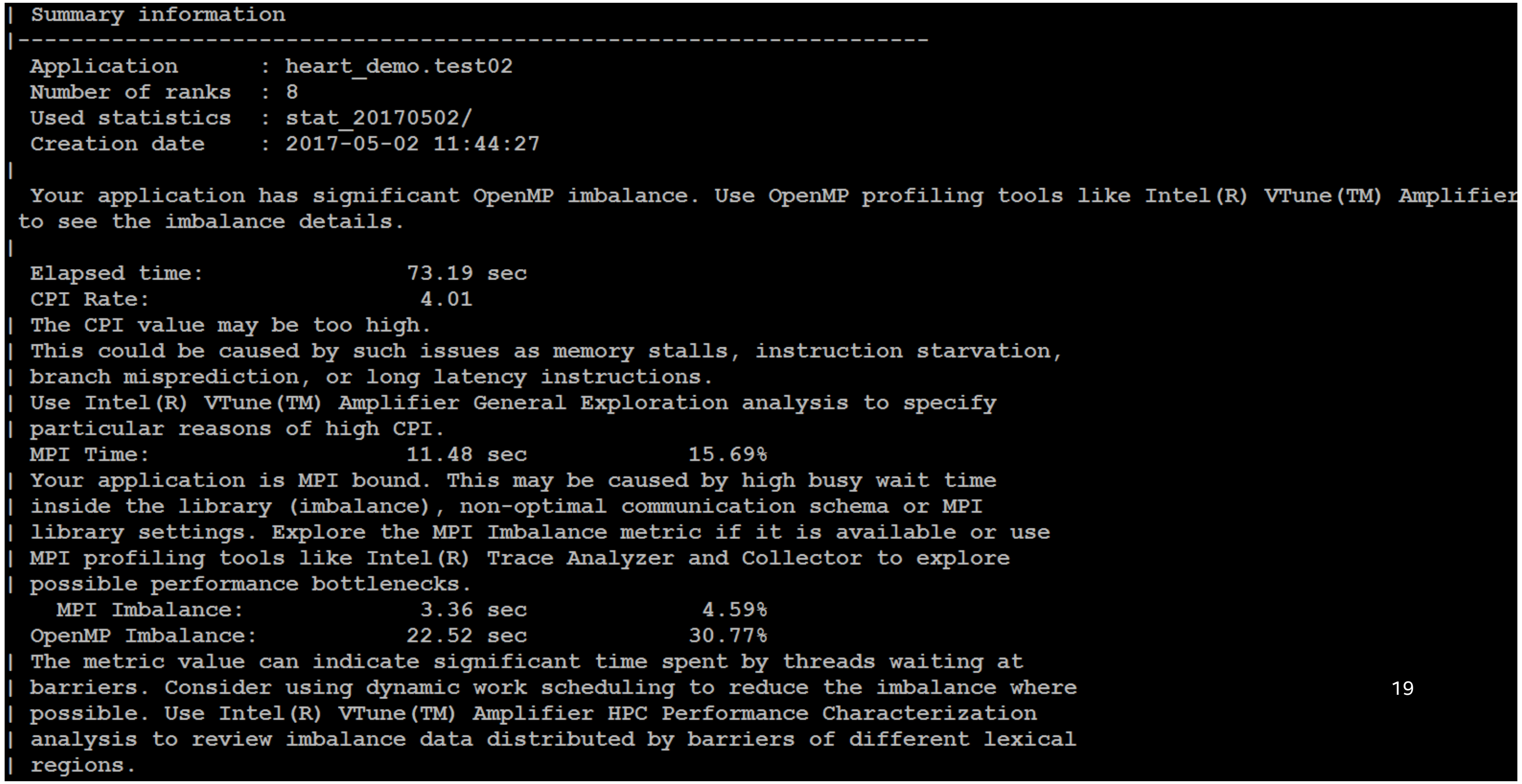
--communicators_list

--format

Please note: some reports

are available with non-default MPS_STAT_LEVEL=1

APS Command Line Reports - Summary



```
| Summary information
|--------------------------------------------------------------------
  Application       : heart_demo.test02
  Number of ranks   : 8
  Used statistics   : stat_20170502/
  Creation date     : 2017-05-02 11:44:27
|
  Your application has significant OpenMP imbalance. Use OpenMP profiling tools like Intel(R) VTune(TM) Amplifier
 to see the imbalance details.
|
  Elapsed time:                 73.19 sec
  CPI Rate:                      4.01
| 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:                     11.48 sec              15.69%
| 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:               3.36 sec               4.59%
  OpenMP Imbalance:             22.52 sec              30.77%
| 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.
```

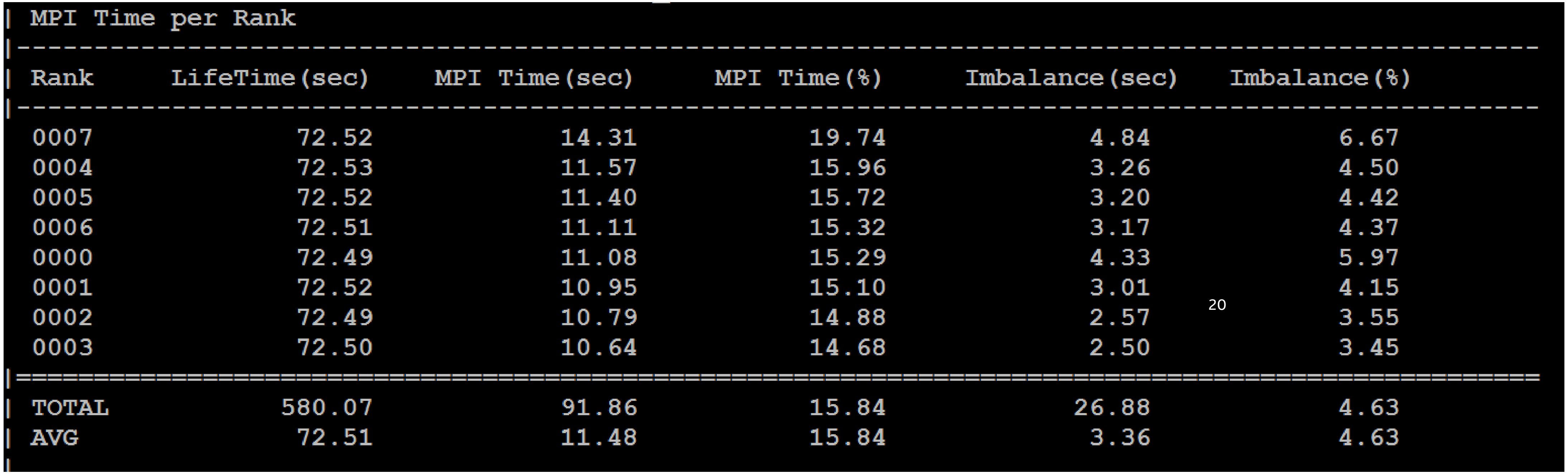
Tip:

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

eliminating verbose descriptions

APS Command Line Reports – Advanced MPI statistics (1/4)

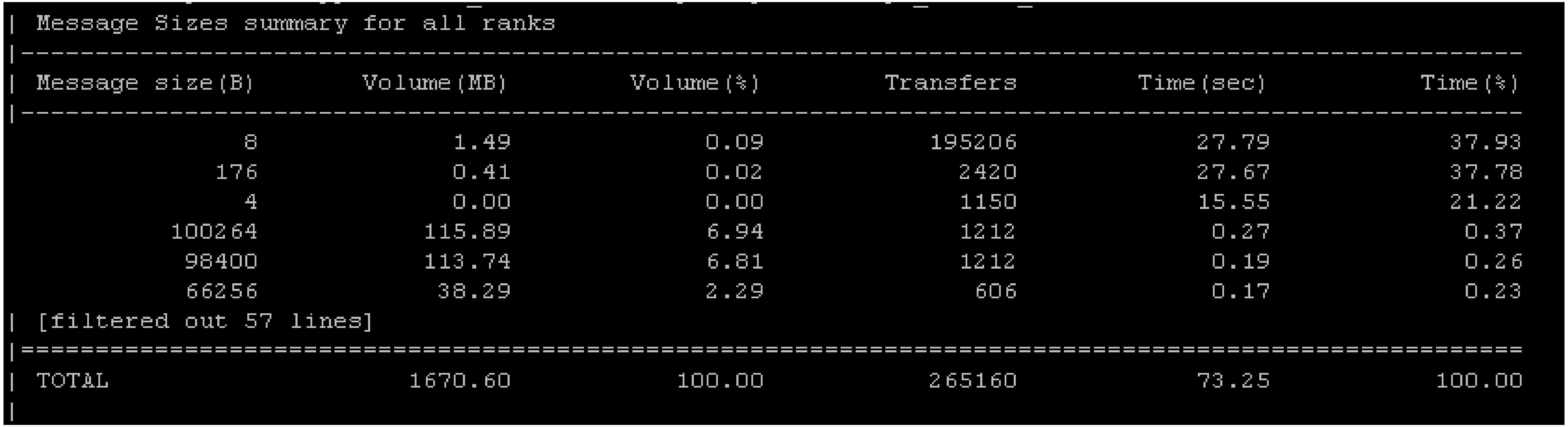
- MPI Time per rank
  - aps-report –t <result>



```
| 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 – Advanced MPI statistics (2/4)

- Message Size Summary by all ranks
  - aps-report –m <result>



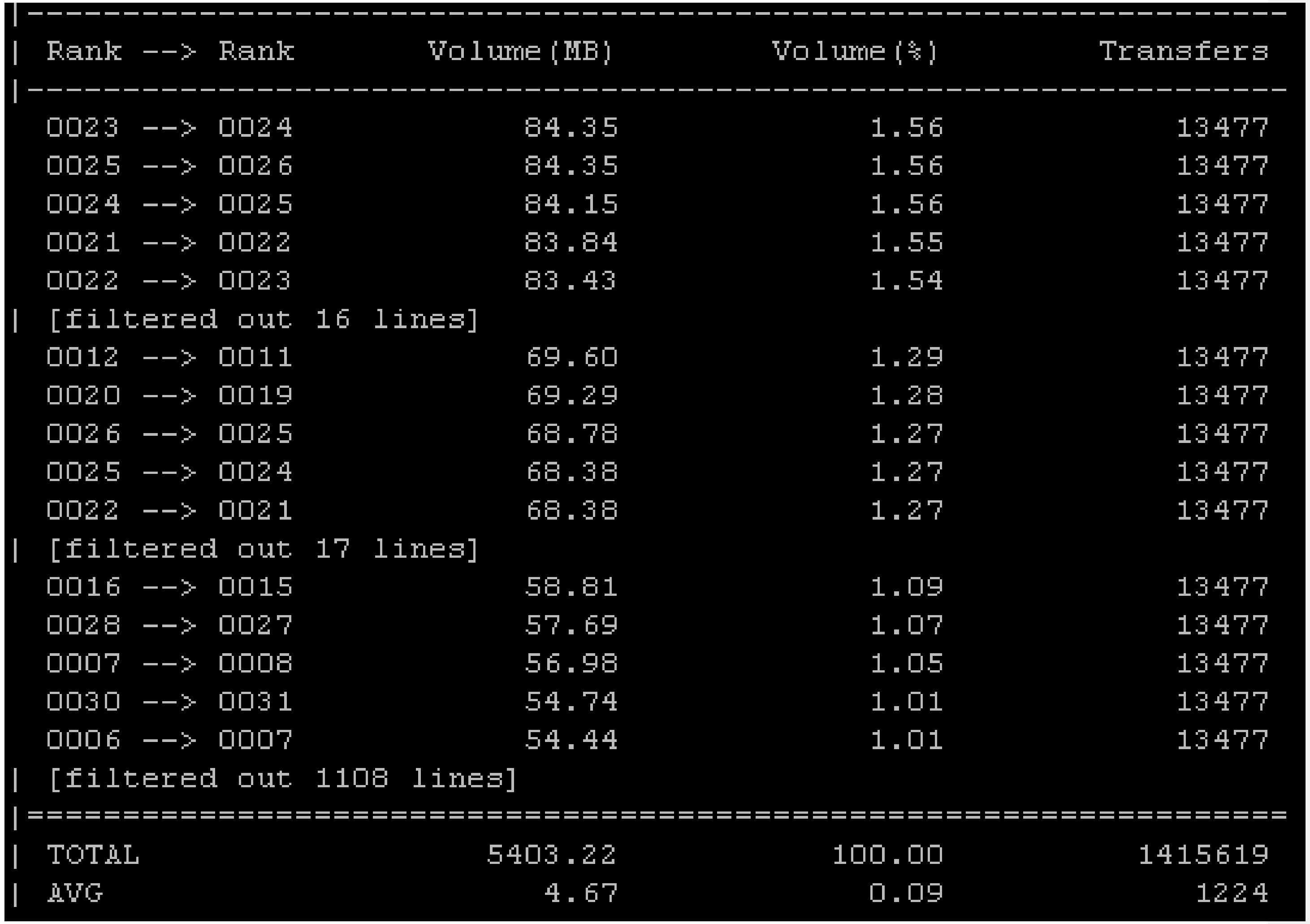
```
| Message Sizes summary for all ranks
|--------------------------------------------------------------------------------------------------
| Message size(B)       Volume(MB)        Volume(%)        Transfers        Time(sec)        Time(%)
|--------------------------------------------------------------------------------------------------
                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 – Advanced MPI statistics (3/4)

- Data Transfers for Rank-to-Rank Communication
  - aps-report –x <result>

And many others – check

- aps-report -help

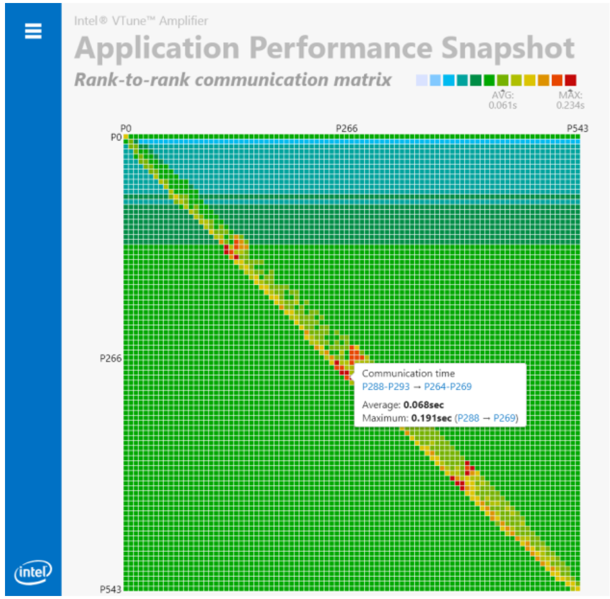


```
|------------------------------------------------------------------
| Rank --> Rank       Volume(MB)           Volume(%)          Transfers
|------------------------------------------------------------------
  0023 --> 0024            84.35                1.56              13477
  0025 --> 0026            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
```

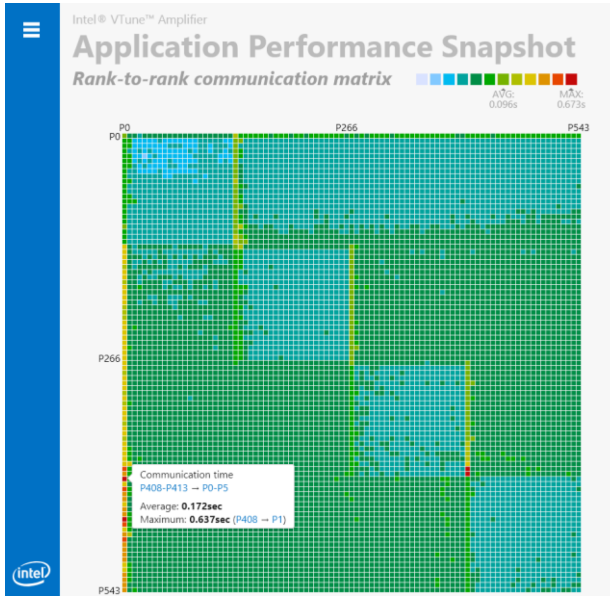
APS Command Line Reports – Detailed MPI statistics (4/4)

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

>aps-report –-transfers_per_communication --format=html <result>


Intel® VTune™ Amplifier
Application Performance Snapshot
Rank-to-rank communication matrix
AVG: 0.061s
MAX: 0.234s
P0
P266
P543
P0
P266
P543

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


Intel® VTune™ Amplifier
Application Performance Snapshot
Rank-to-rank communication matrix
AVG: 0.096s
MAX: 0.673s
P0
P266
P543
P0
P266
P543

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

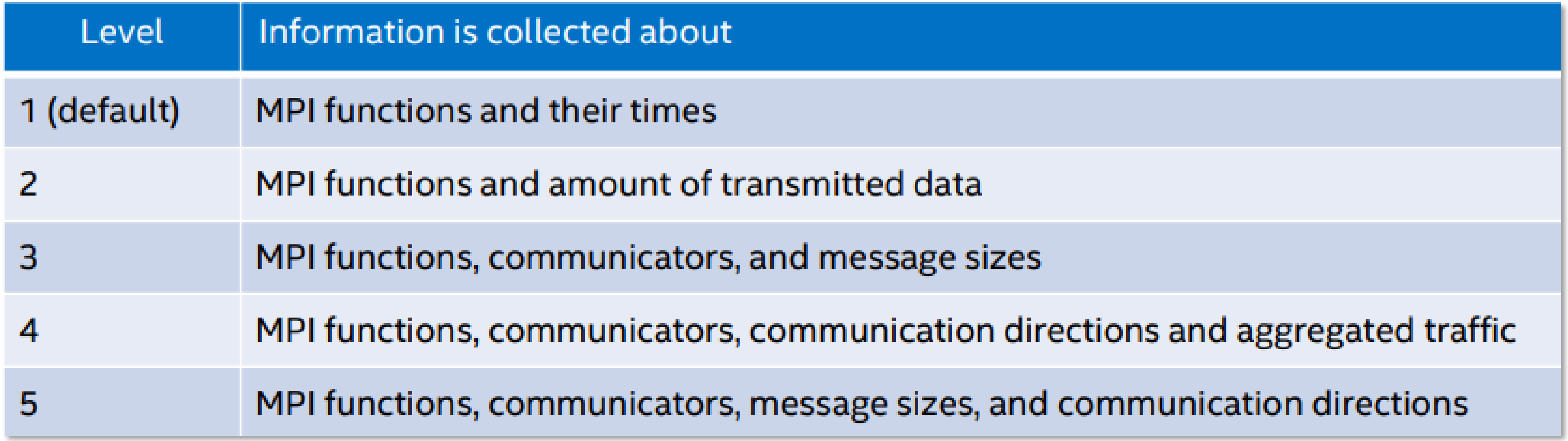
- Use –collection-mode option to limit collection either by MPI or OpenMP tracing or HW-counters
  - Use case: interest in MPI statistics only

```
>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

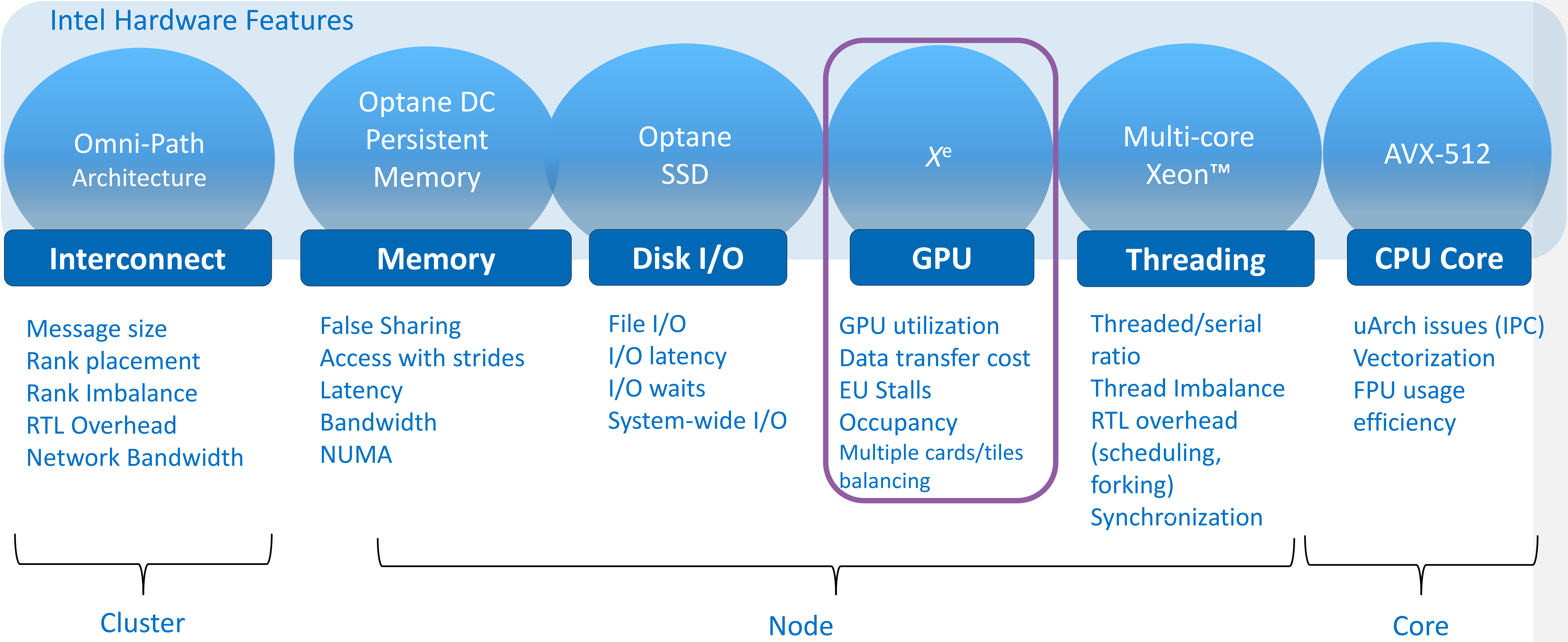
Reducing collected data for MPI tracing

- >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

ASPECTS OF HPC/THROUGHPUT APPLICATION PERFORMANCE



Intel Hardware Features

Omni-Path Architecture

Interconnect

Message size
Rank placement
Rank Imbalance
RTL Overhead
Network Bandwidth

Optane DC Persistent Memory

Memory

False Sharing
Access with strides
Latency
Bandwidth
NUMA

Optane SSD

Disk I/O

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

X^e

GPU

GPU utilization
Data transfer cost
EU Stalls
Occupancy
Multiple cards/tiles balancing

Multi-core Xeon™

Threading

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

AVX-512

CPU Core

uArch issues (IPC)
Vectorization
FPU usage efficiency

Cluster

Node

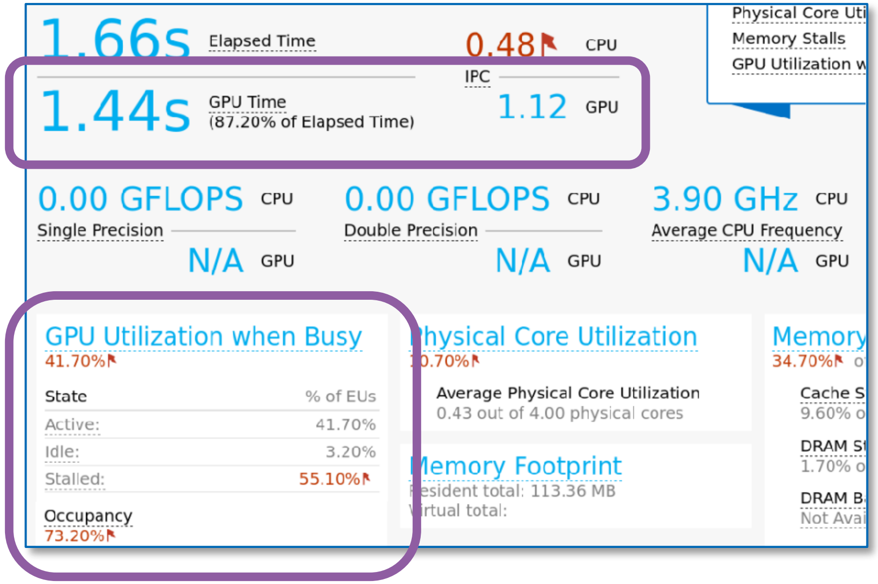
Core

GPU compute efficiency metrics in APS

- What
  - GPU Time
  - GPU IPC
  - GPU Utilization when busy with breakdown by active, stalled and idle Eus
  - Occupancy

How

- Based on MD API
- Metrics aggregated by a node
- Averaged by nodes in summary reports for MPI multi-node applications



1.66s Elapsed Time

1.44s GPU Time (87.20% of Elapsed Time)

IPC: 0.48 CPU, 1.12 GPU

0.00 GFLOPS CPU, N/A GPU — Single Precision

0.00 GFLOPS CPU, N/A GPU — Double Precision

3.90 GHz CPU, N/A GPU — Average CPU Frequency

Physical Core Uti
Memory Stalls
GPU Utilization w

GPU Utilization when Busy
41.70%

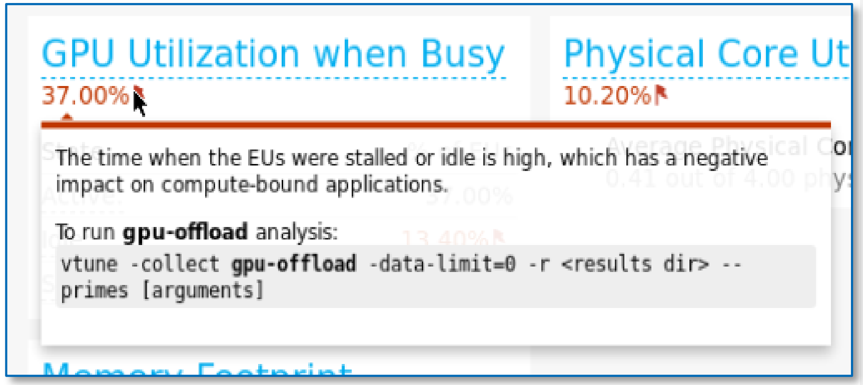
State	% of EUs
Active:	41.70%
Idle:	3.20%
Stalled:	55.10%

Occupancy
73.20%

hysical Core Utilization
0.70%
Average Physical Core Utilization
0.43 out of 4.00 physical cores

Memory
34.70% o
Cache S
9.60% o
DRAM St
1.70% o
DRAM B
Not Avai

emory Footprint
esident total: 113.36 MB
irtual total:



GPU Utilization when Busy
37.00%

Physical Core Ut
10.20%

The time when the EUs were stalled or idle is high, which has a negative impact on compute-bound applications.

To run **gpu-offload** analysis:

```
vtune -collect gpu-offload -data-limit=0 -r <results dir> --
primes [arguments]
```

Summary

- Intel® VTune™ Amplifier's Application Performance Snapshot is:
- Your entry point for HPC application performance analysis
- A part of Parallel Studio XE or VTune
- 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

intel®