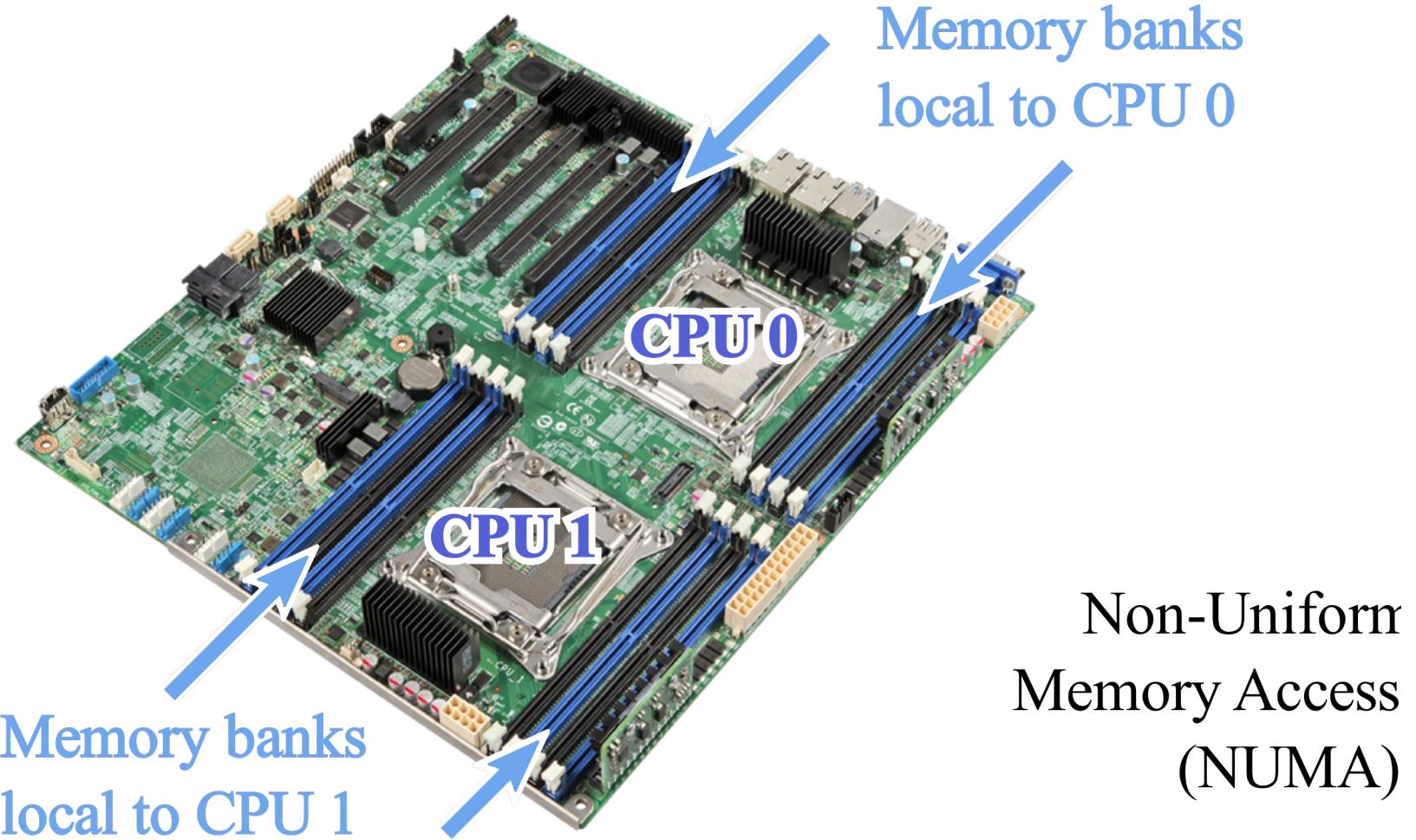


Achieving High Bandwidth with Intel Parallel Studio XE

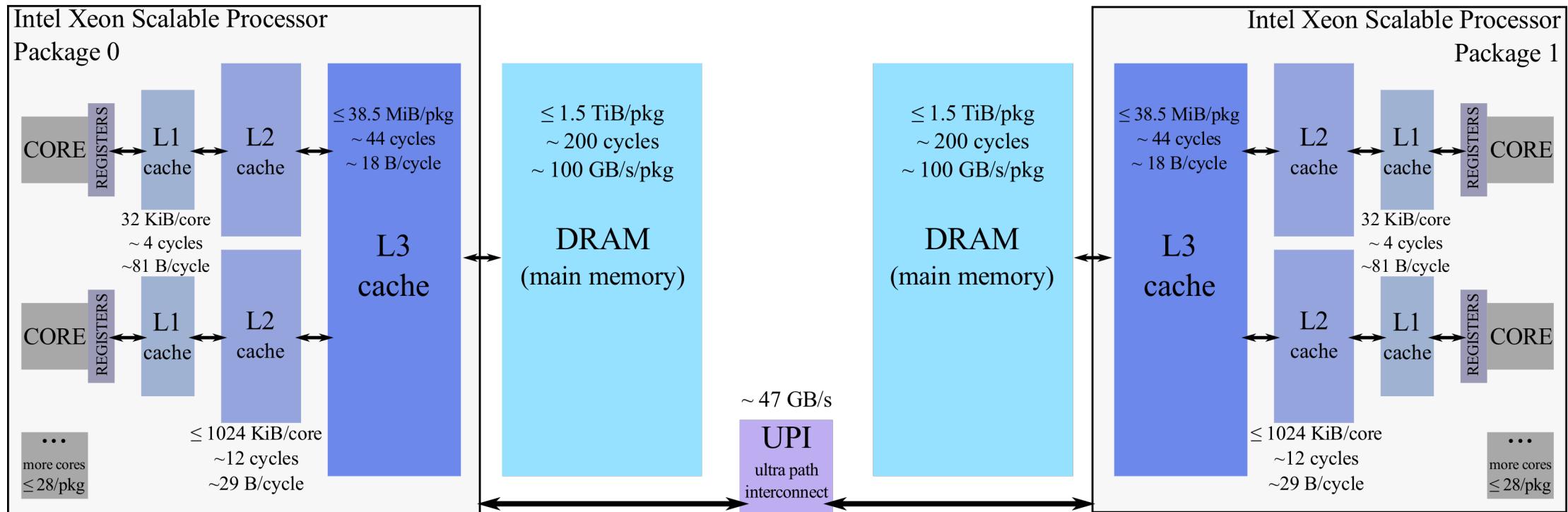
Stanford University, ME344 — Summer 2018

Andrey Vladimirov, Colfax International

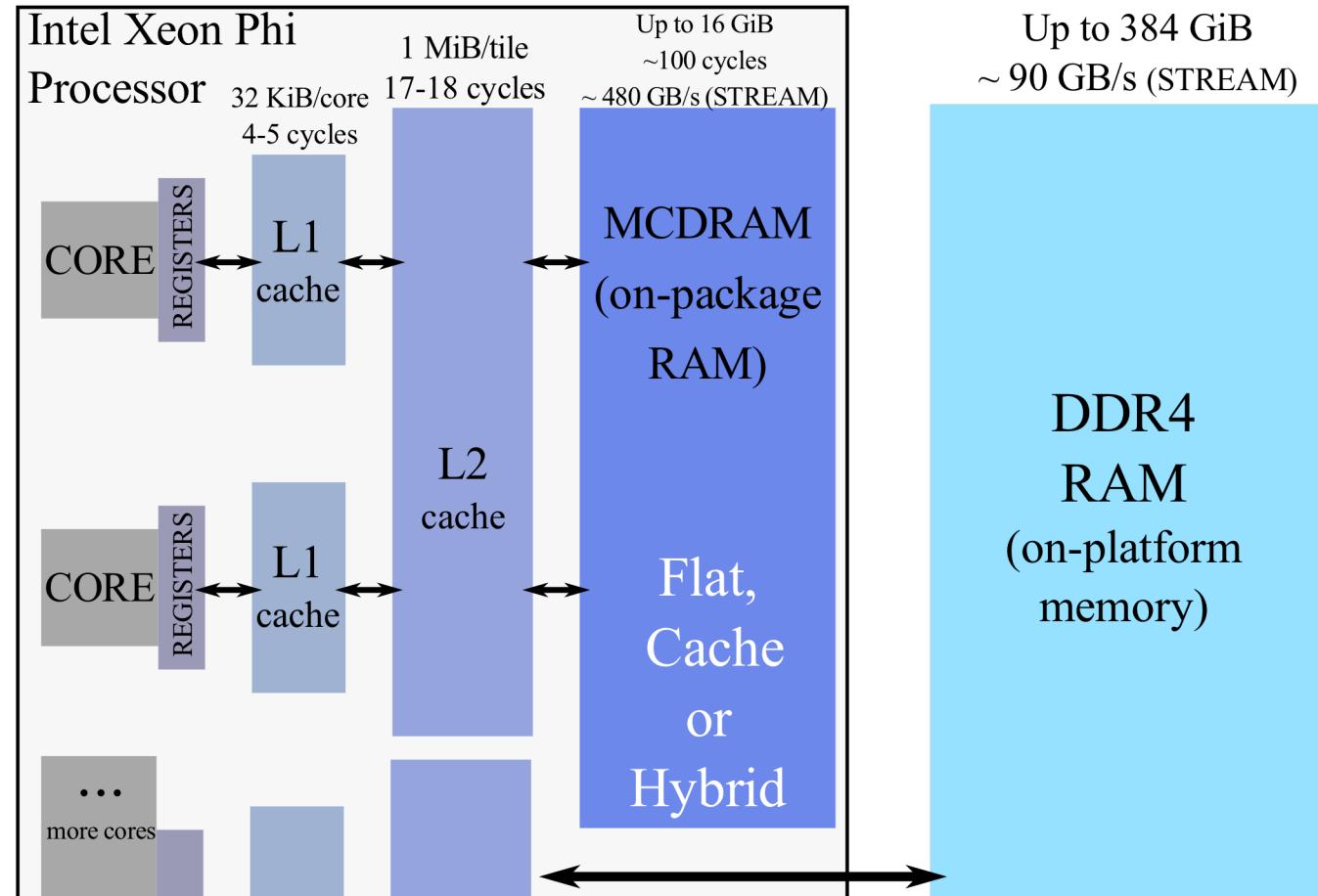
NUMA and Data Access Locality



Memory Hierarchy, Intel Xeon CPU



Memory Hierarchy, Intel Xeon Phi CPU

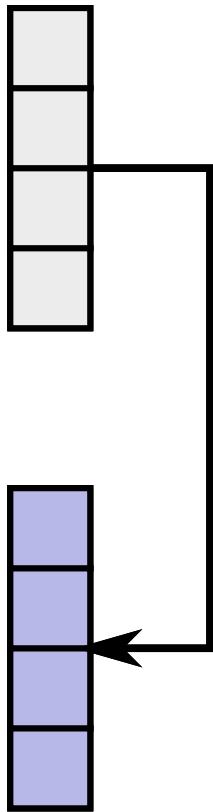


Access Online Materials

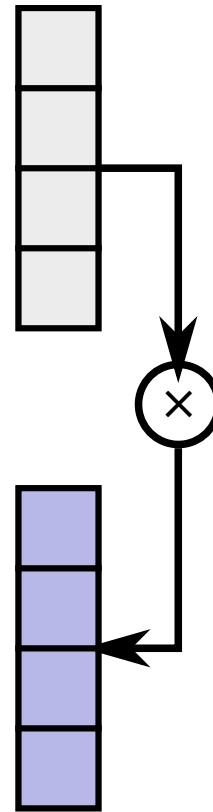
[mc2learning.com/course/
parallelism-and-memory-bandwidth-in-a-stencil-code-c-lin/](https://mc2learning.com/course/parallelism-and-memory-bandwidth-in-a-stencil-code-c-lin/)

STREAM Benchmark

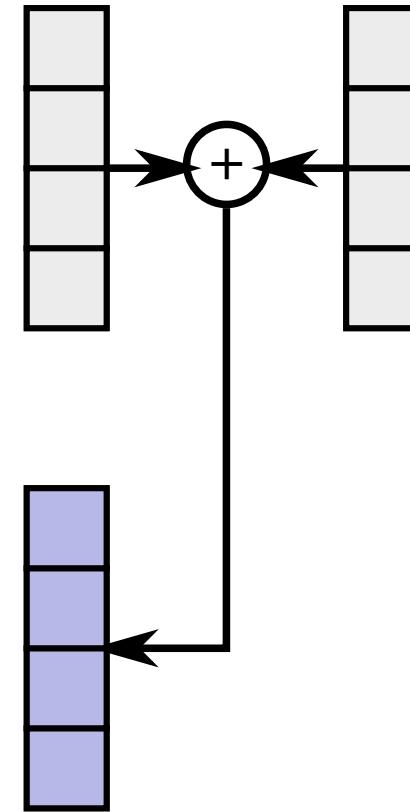
Copy:
 $c[j] = a[j]$



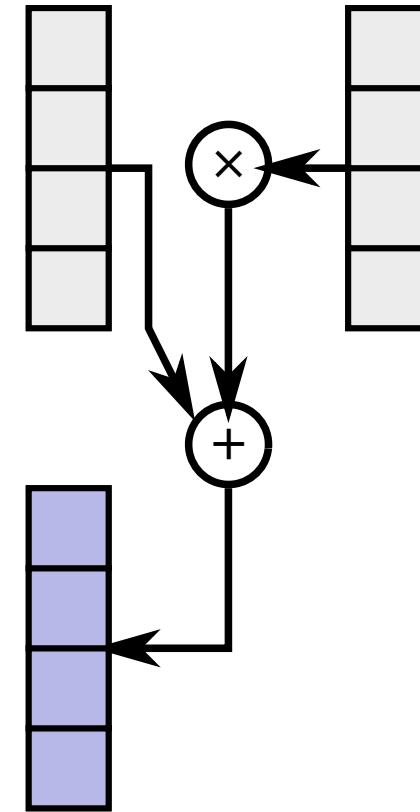
Scale:
 $b[j] = s \cdot a[j]$



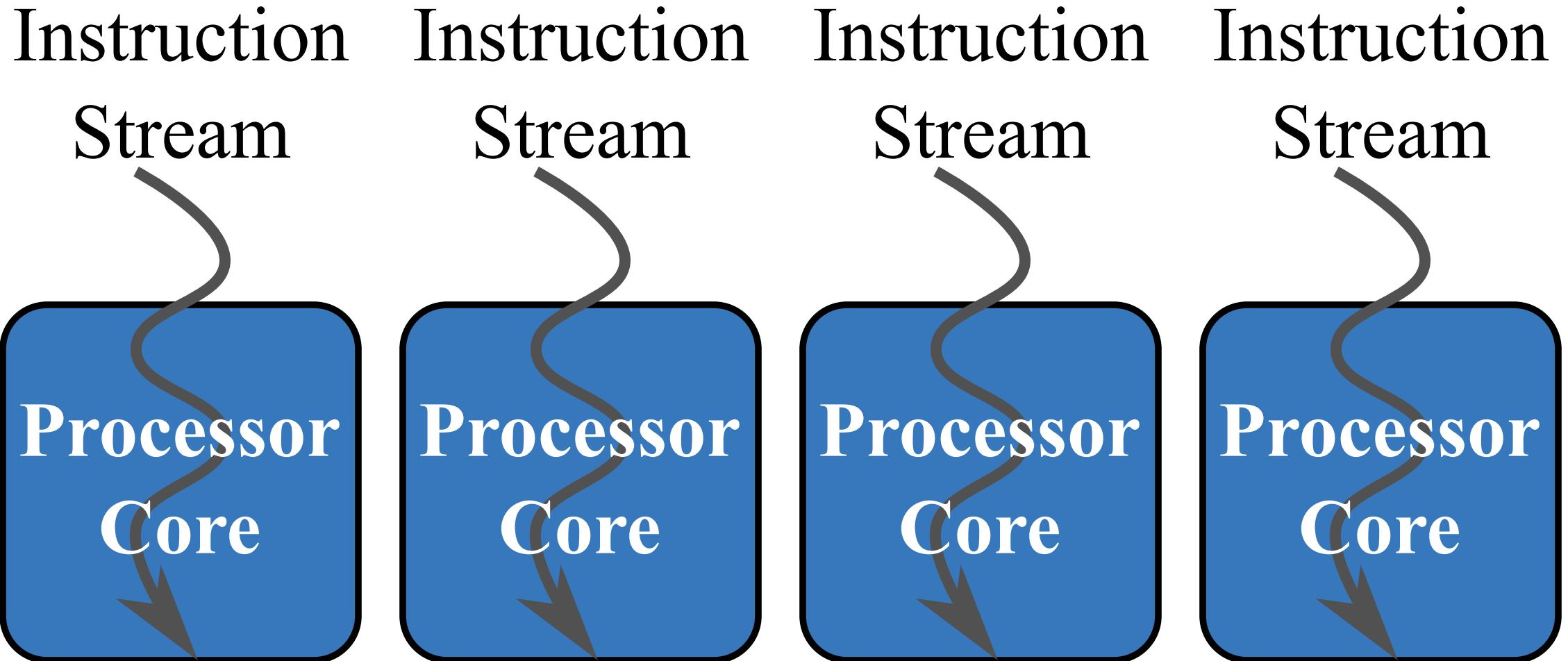
Add:
 $c[j] = a[j] + b[j]$



Triad:
 $a[j] = b[j] + s \cdot c[j]$



Memory Controllers and Threads

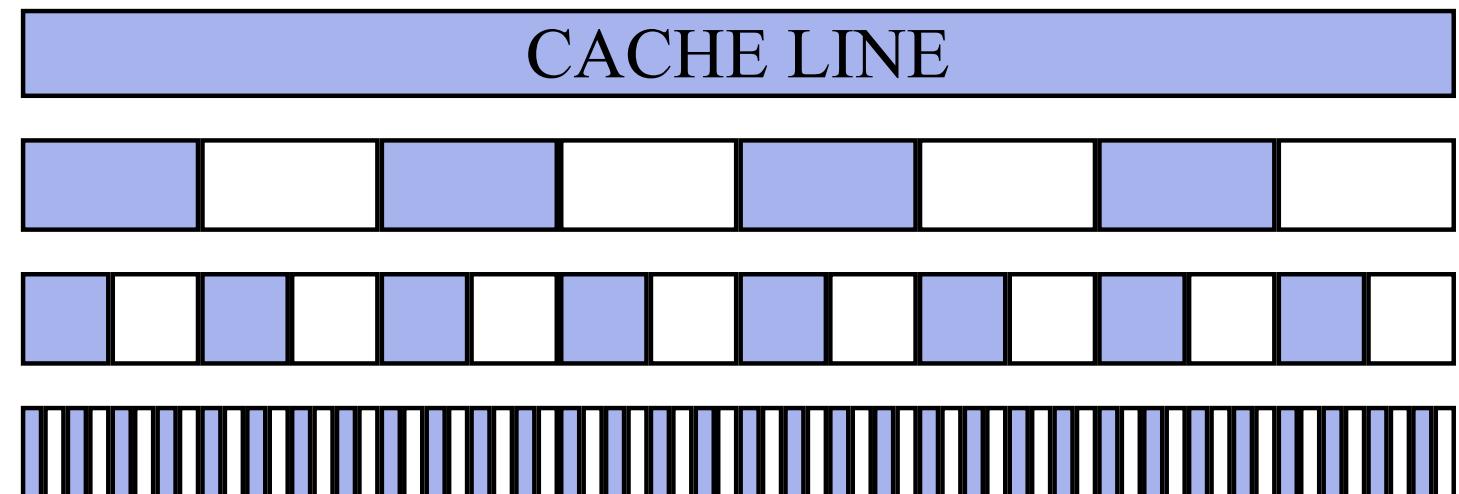


Cache Lines and Vectorization

8 double precision values

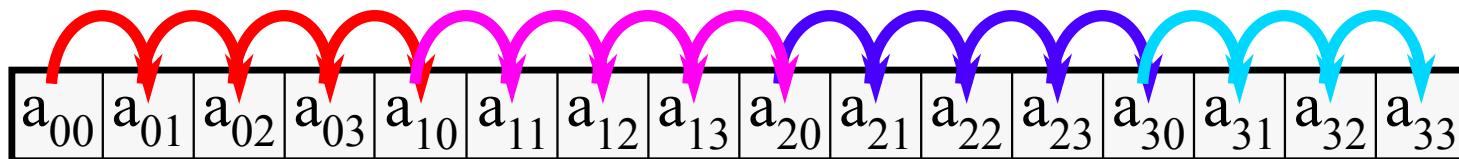
16 single precision values

64 bytes

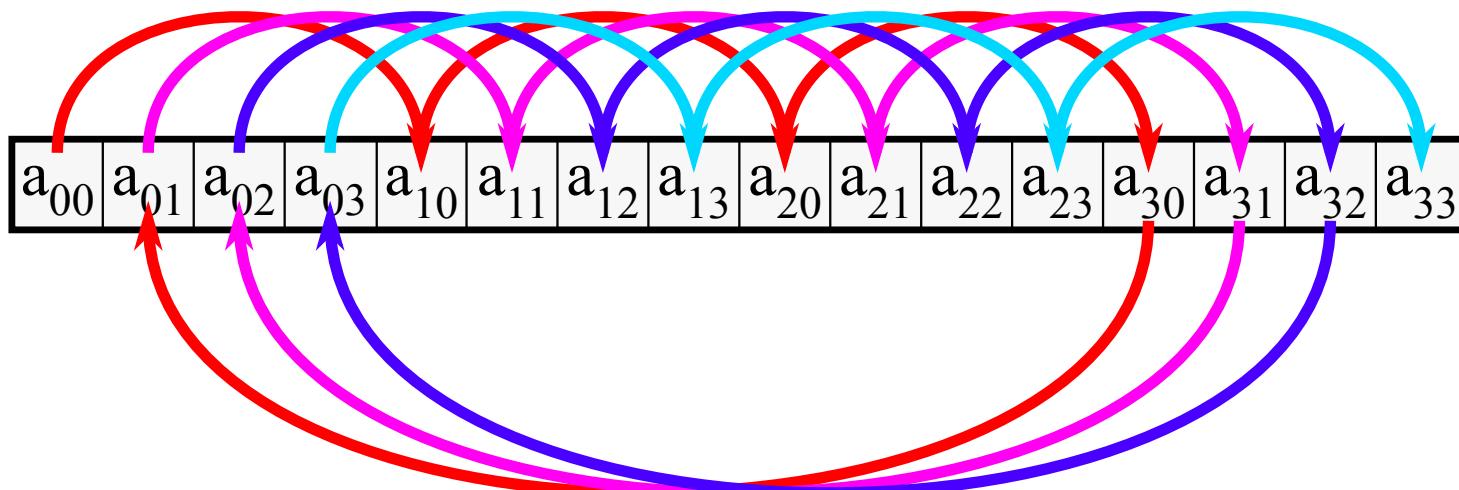


Sequential Access

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

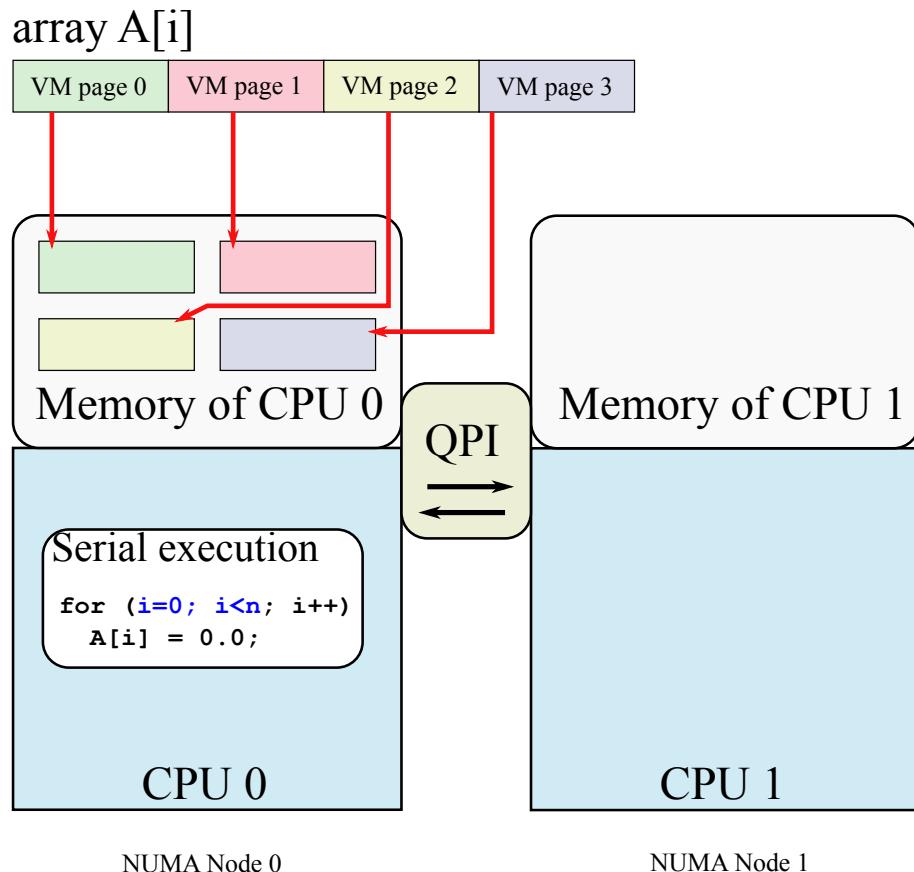


a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

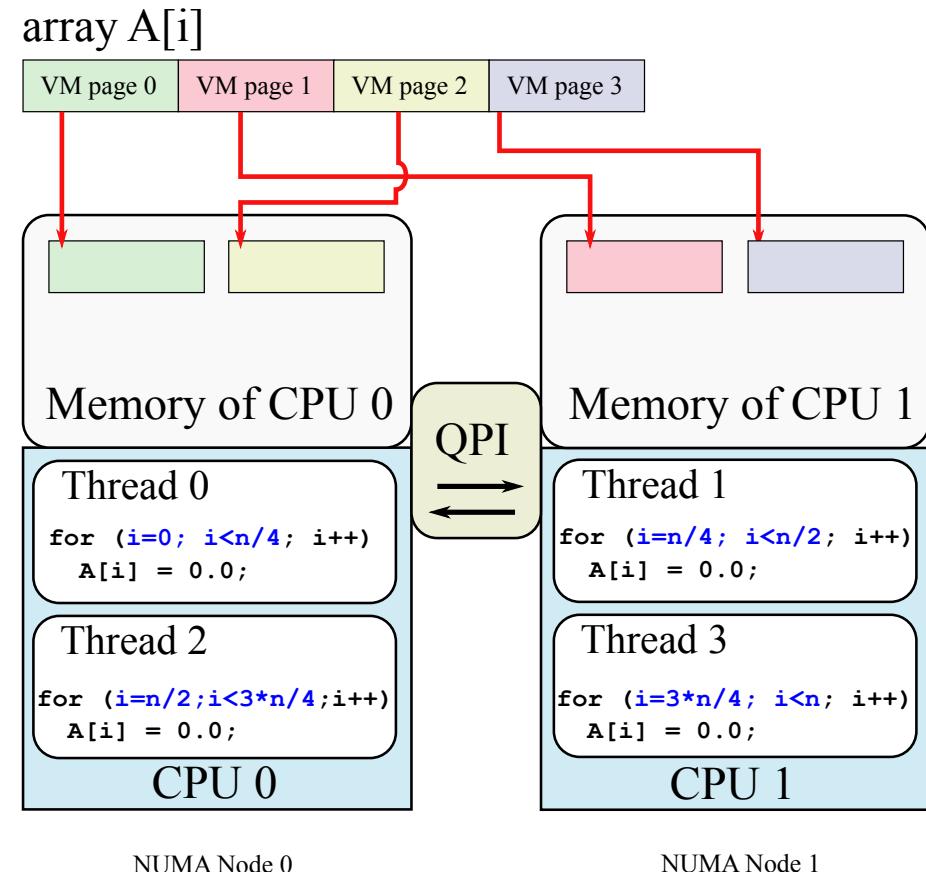


First Touch Policy and Array Initialization

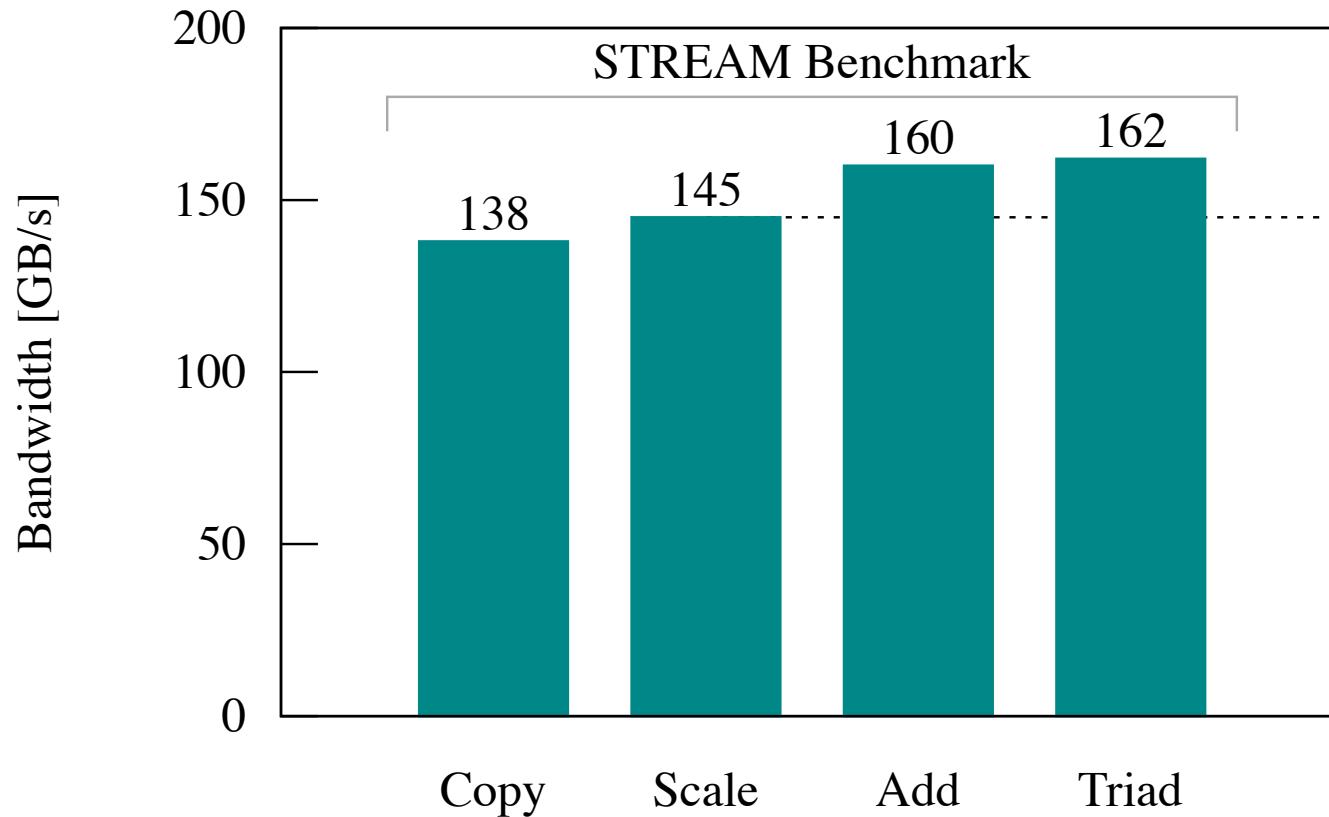
Poor First-Touch Allocation



Good First-Touch Allocation

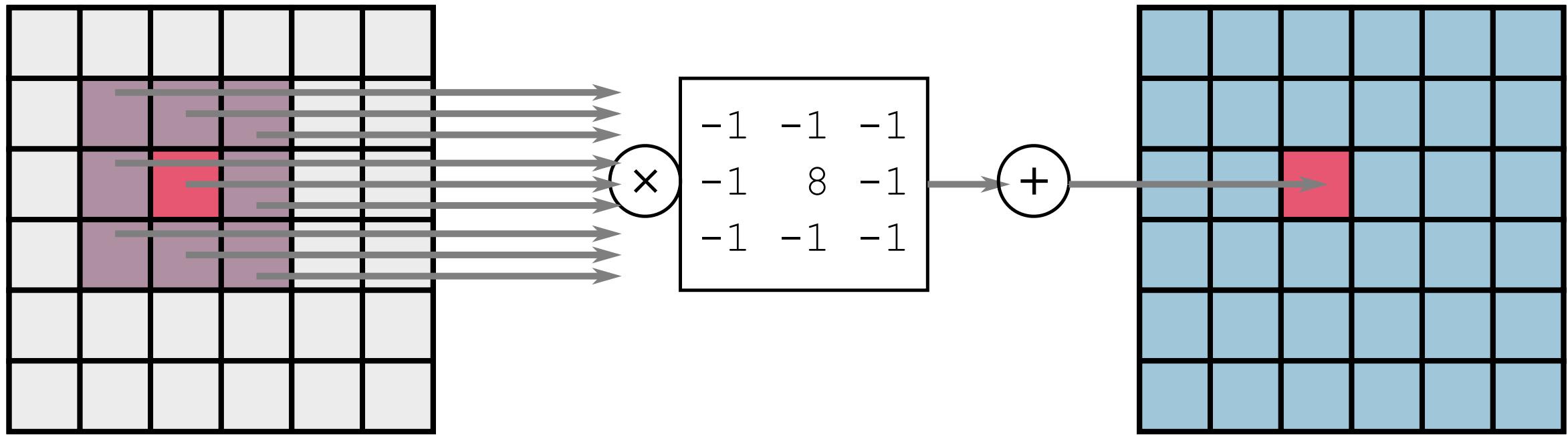


STREAM Optimization

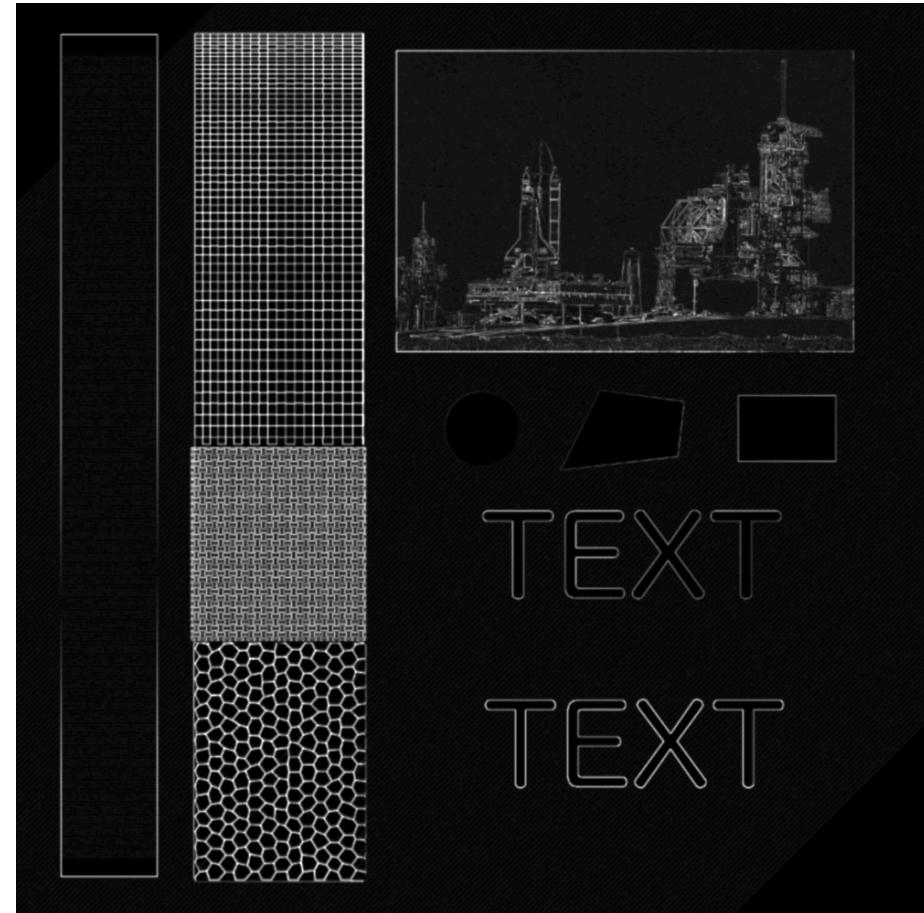
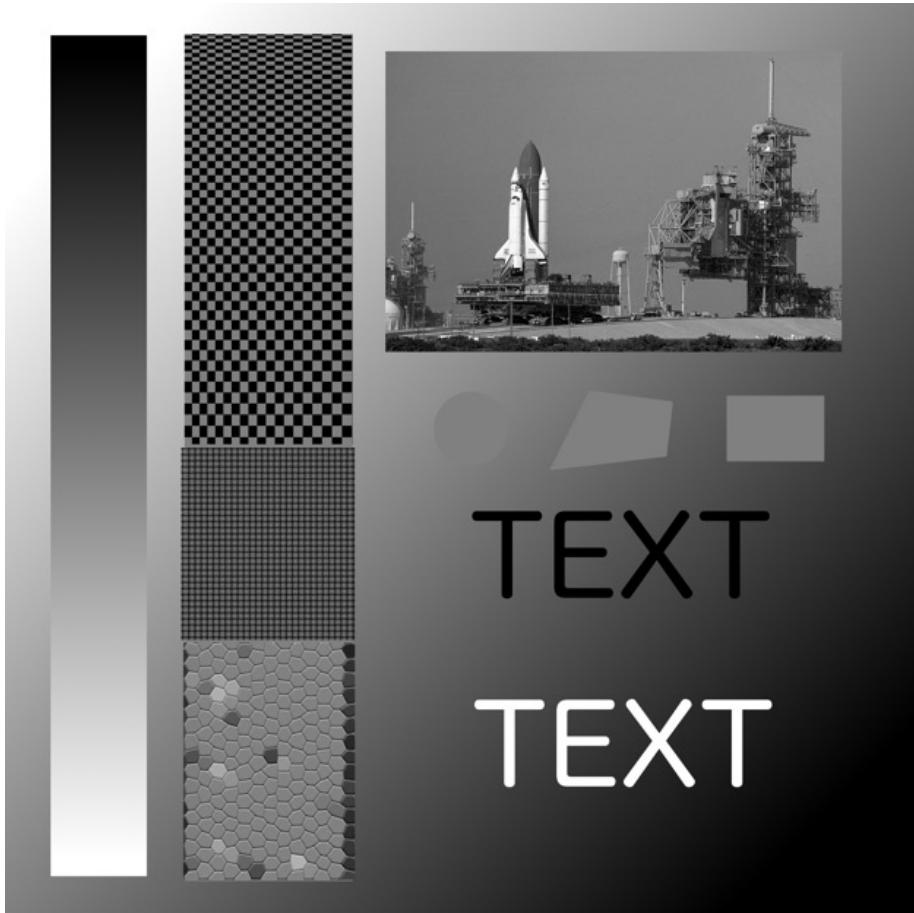


- Multithreading
- Vector ISA
- Thread affinity
- Streaming Stores

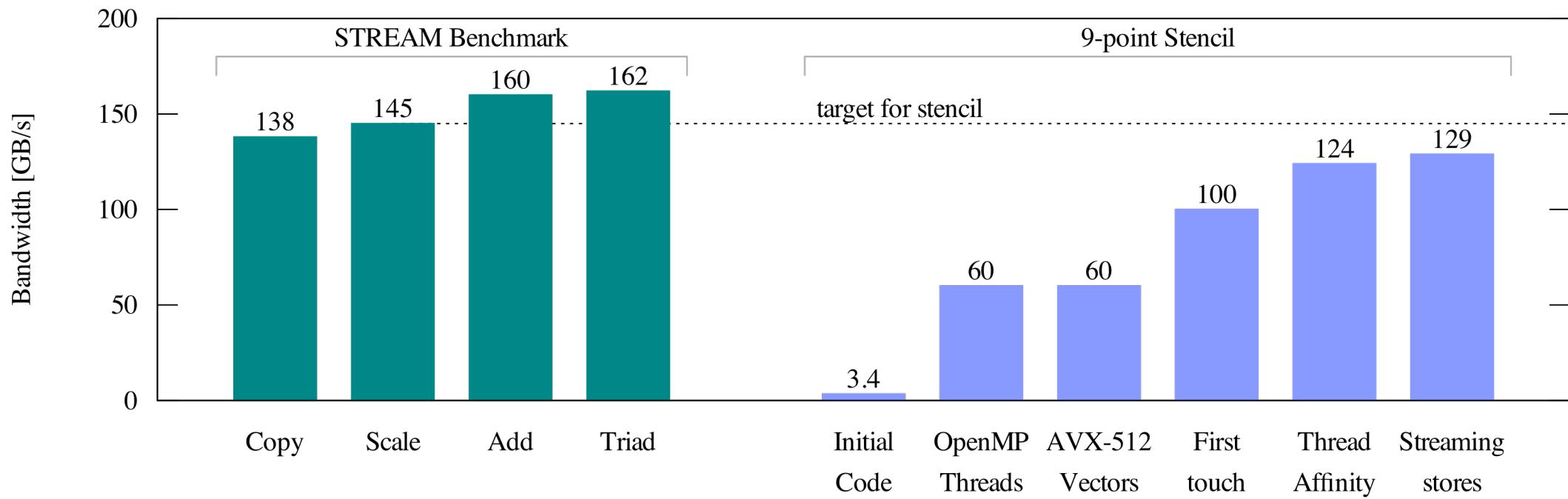
9-Point Stencil



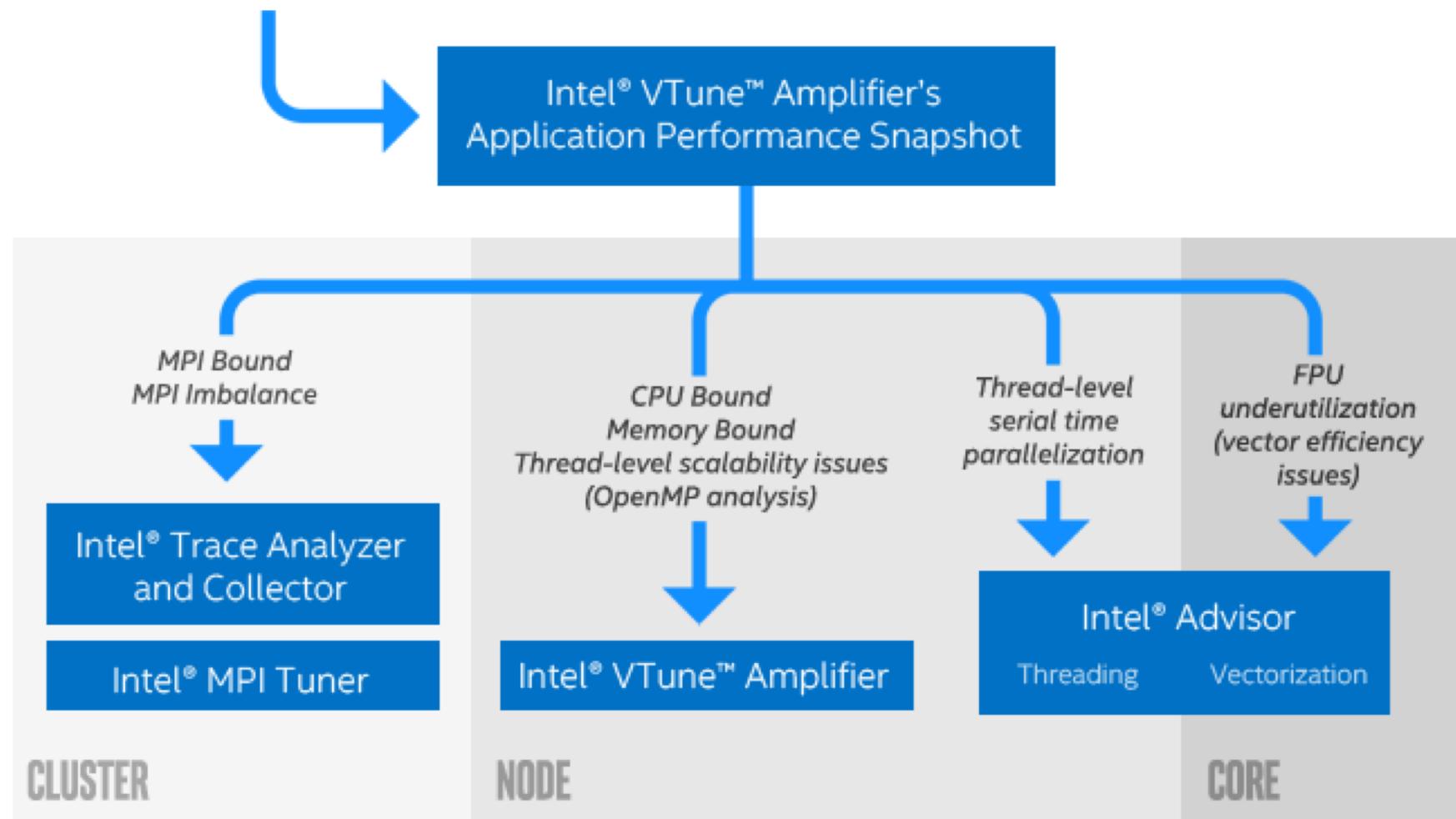
Edge Detection



Stencil Code Optimization



Intel Parallel Studio XE Tools



Intel VTune Amplifier — Application Performance Snapshot

Application Performance Snapshot

Application: *stencil*
Report creation date: 2018-02-23 15:56:01
OpenMP threads: 24
HW Platform: Intel(R) Xeon(R) Processor code named Skylake
Logical Core Count per node: 24

31.77s
Elapsed Time

125.11
SP FLOPS

3.24  CPI

Serial Time
13.72% of Elapsed Time
(4.35s)

OpenMP Imbalance
6.45% of Elapsed Time
(2.05s)

Memory Footprint
Resident total: 317.80 MB
Virtual total: 1936.58 MB

Your application is memory bound.
Use [memory access analysis tools](#) like [Intel® VTune™ Amplifier](#) for a detailed metric breakdown by memory hierarchy, memory bandwidth, and correlation by memory objects.

	Current run	Target	Delta
Serial Time	13.72%	<15%	
OpenMP Imbalance	6.45%	<10%	
Memory Stalls	68.80% 	<20%	
FPU Utilization	3.20% 	>50%	

Memory Stalls
68.80%  of pipeline slots

Cache Stalls
27.30%  of cycles

DRAM Stalls
44.10%  of cycles

NUMA
15.40% of remote accesses

FPU Utilization
3.20% 

SP FLOPs per Cycle
2.06 Out of 64.00

Vector Capacity Usage
100.00%

FP Instruction Mix

% of Packed FP Instr.:	100.00%
% of 128-bit:	0.00%
% of 256-bit:	0.00%
% of 512-bit:	100.00%
% of Scalar FP Instr.:	0.00%

FP Arith/Mem Rd Instr. Ratio
1.10

FP Arith/Mem Wr Instr. Ratio
9.68

Intel Advisor

Summary Survey & Roofline Refinement Reports Annotation Report Suitability Report

Maximum Program Gain For All Sites: 0.14x

Target System: CPU Threading Model: Intel TBB CPU Count: 8

Site Label	Source Location	Impact to Program Gain	Combined Site Metrics, All Instances			Site Instance Metrics, Parallel Time
			Total Serial Time	Total Parallel Time	Site Gain	
"columnloop"	stencil.cc:15	0.14x	42.02s	292.95s	0.14x	0.005s

Site Performance Scalability **Site Details**

Scalability of Maximum Site Gain

Maximum Site Gain

CPU Count

Loop Iterations (Tasks) Modeling

Avg. Number of Iterations (Tasks): 5998

Avg. Iteration (Task) Duration: < 0.001s

0.008x	0.008x
0.040x	0.040x
0.200x	0.200x
1x (5998)	1x (< 0.001s)
5x	5x
25x	25x
125x	125x

Runtime Modeling

Type of Change Gain Benefit if Enabled

- Reduce [Site Overhead](#)
- Reduce [Task Overhead](#) +1.20x
- Reduce [Lock Overhead](#)
- Reduce [Lock Contention](#)
- Enable [Task Chunking](#) +4.60x

Apply

83.7% Load Imbalance: 245.14s

99.9% Runtime Overhead: 292.65s

0.0% Lock Contention: 0s

Total Parallel Time: 292.95s

Warning

⚠ Current tasks are too fine-grain, and not effective for multi-threading. Suggestion: Increase task granularity/duration, reduce task overhead, or consider vectorization.

Intel VTune Amplifier

