HPC User Forum, May 2021

Optimizing NAMD for Intel® AVX-512

W. Michael Brown
Principal Engineer

Reporting on work in collaboration with David Hardy, Senior Research Programmer, University of Illinois at Urbana-Champaign

Molecular Dynamics (MD)

- Simulates the motion of atoms over time
- Essential research tool in pursuit of new therapeutics and options for disease prevention
  - *With current technologies, there are not enough computational resources in the world to satisfy the needs of molecular biophysicists*
- Critical that we continue to push boundaries on time-to-solution in order to expand the capabilities for researchers at many HPC centers
- NAMD, developed at UIUC, is an established MD code popular in biomedical research with numerous contributions to the field

MD Algorithms

- MD implementations are sophisticated and cover a variety of parallel algorithms
  - *N-body, stencil, spectral, machine learning, …*
- The *direct force* component of N-body algorithms is typically dominating in # of mathematical operations
  - *High arithmetic intensity*
  - The subtleties, optimizations, and variants of algorithms for *N-body* are extensive!

Example Direct Force Math

$$U'_R = \sum_i \sum_{j>i} w_0 \frac{C_1 q_i q_j}{r^2} (\mathrm{erfc}(C_2 r) + C_3 C_2 r \exp(-C_2^2 r^2) - s_b) + w_1 \frac{f_{ij}}{r^7}\left(\frac{\alpha_{ij}}{r^6} - \beta_{ij}\right) + w_2 \frac{f_{ij}}{r^7}\left(\frac{\gamma_{ij}}{r^6} - \delta_{ij}\right)$$

$$w_0 = \begin{cases} 0 \text{ if } r > c \\ 1 \text{ otherwise} \end{cases}$$

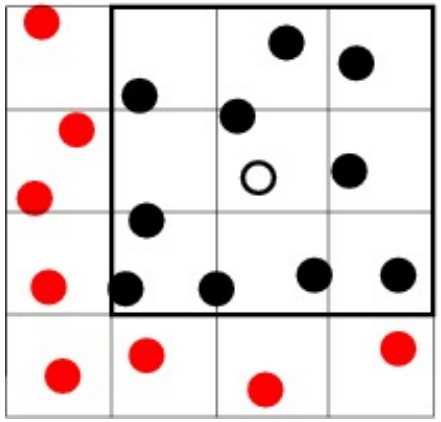
$$w_1 = \begin{cases} w_0 (c - r^2)^2 (c + 2r^2 - 3s) C_4 \text{ if } r > s \\ 1 \text{ otherwise} \end{cases}$$

$$w_2 = \begin{cases} w_0 12 r^2 (c - r^2)(r^2 - s) C_4 \text{ if } r > s \\ 0 \text{ otherwise} \end{cases}$$

where r is interatomic separation, $q_{\cdot}$ is the partial charge on the atoms, $C_{\cdot}$ is a constant, and variables qualified with ij are constants given by atom types indexed by i and j

Direct Force Calculation: $O(N^2)$ -> $O(N)$

- The direct force part of the calculation involves a cutoff distance so only close atoms are evaluated
- A neighbor list build is used to achieve $O(N)$ scaling by tracking neighbors within cutoff
- Example direct force loop structure:



```
foreach atom i
  data_i=load_pos_type_params(i)
  ans_i = {0,0,0}
  foreach neighbor j
    data_j=load_pos_type_params(j)
    distance2=distance_sq(data_i, data_j)
    if (distance2 < cutoff2)
      ans=calculate_force(data_i, data_j)
      ans_i += ans; store_ans(j, -ans)
  store_ans(i, ans_i)
```

Non-contiguous memory access

Intel® Advanced Vector Extensions 512 (AVX-512)

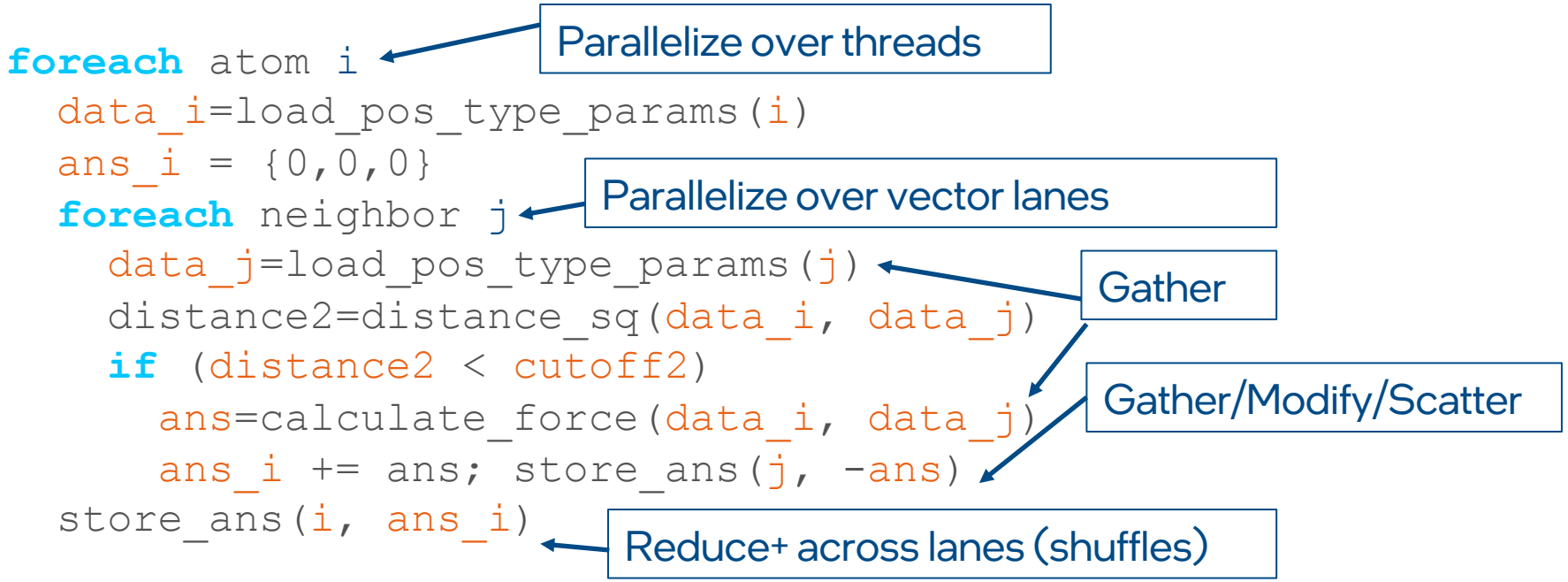
- Intel's most recent vector instruction set architecture (ISA)

- Can significantly reduce instruction counts vs AVX-2:
  - Vector registers are doubled (reduced spilling)
  - Functionality enhancements (instructions such as compress, expand, etc.)
  - Full masking support with fault suppression
  - Support for 512b vector widths in addition to 256b

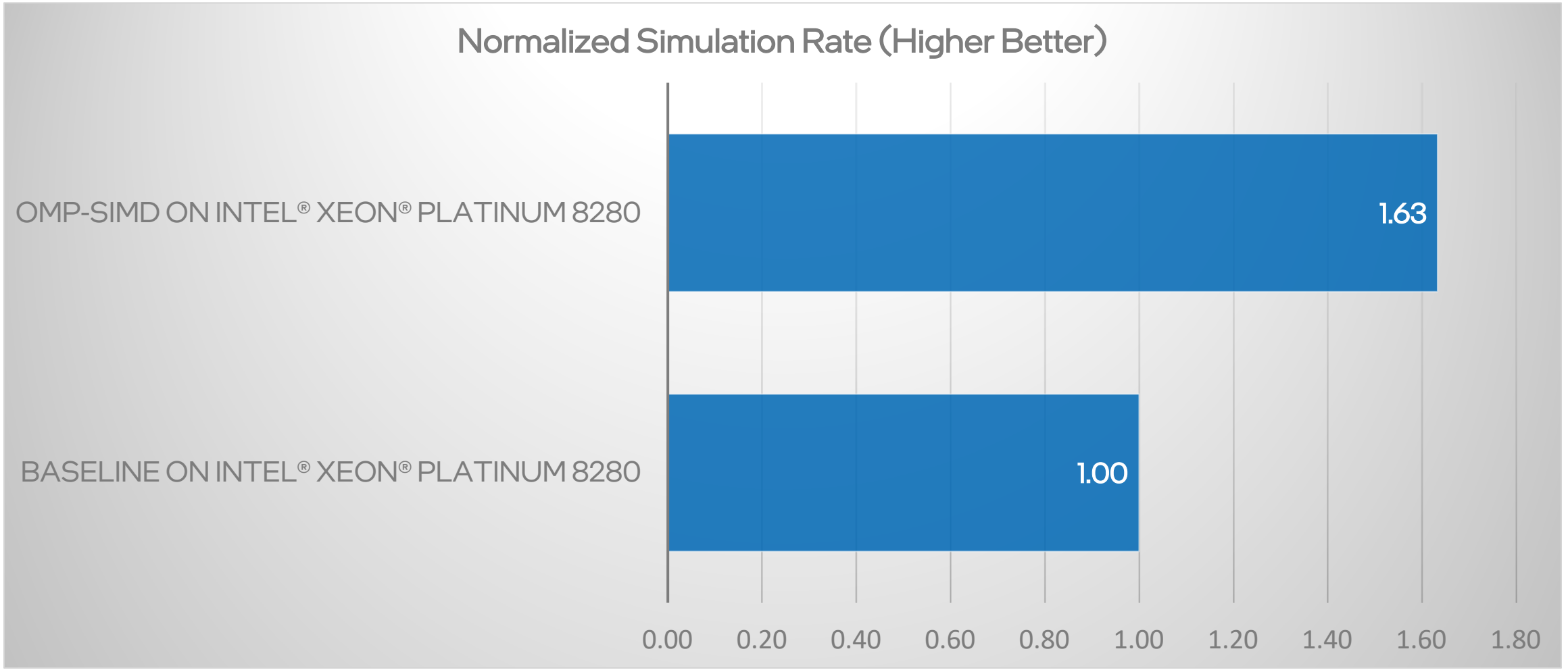
- In my experience, developer productivity is significantly enhanced
  - Better compiler-generated vector code for complex / branchy loops
  - Much simpler lower-level implementations using vector intrinsics

NAMD 2.13 w/ Intel® AVX-512

- Includes optimizations for mixed-precision, data layout, and vectorization
- Inner loop vectorization with compiler-generated vector code via OpenMP SIMD directives *(OMP-SIMD)*



NAMD Satellite Tobacco Mosaic Virus Simulation



Performance varies by use, configuration, and other factors. Configurations: see appendix [1, 2]

3rd Generation Intel® Xeon® Scalable Processors

- Code named "Ice Lake"
- General HPC improvements include:
  - Up to 40 cores / socket with IPC gains from latest µarch
    - *Up to 1.53X geomean HPC performance vs 2nd Gen*[1]
  - 8 DDR4 3200 MT/s memory channels (up to 6TB per proc)
    - *Up to 1.47X Stream Triad vs 2nd Gen*[1]
  - Support for Intel® Optane™ Persistent Memory 200 Series
  - Built-in AI acceleration (Intel® Deep Learning Boost)
  - PCIe Gen4 support with 64 lanes/ socket, 16 GT/s acceleration
  - Intel® Speed Select Technology for granular control

[1]Performance varies by use, configuration, and other factors. Configurations see See [108] at www.intel.com/3gen-xeon-config

Compute Microarchitecture

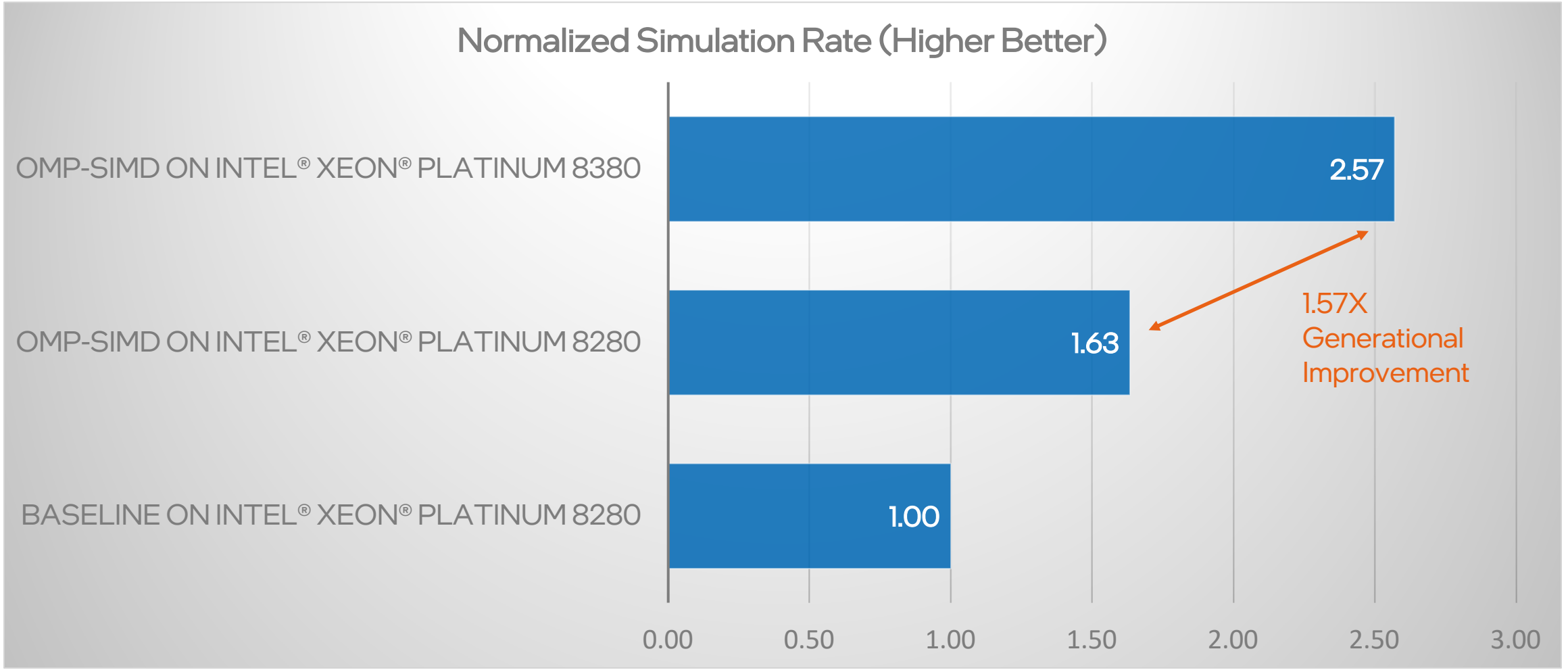
Sunny Cove Core

- Improved Front-end: higher capacity and improved branch predictor
- Wider and deeper machine: wider allocation, larger structures, and execution resources
- Enhancements in TLBs, single thread execution, prefetching
- Larger caches

	Cascade Lake (per core)	Ice Lake (per core)
Out-of-order Window	224	352
In-flight Loads + Stores	72 + 56	128 + 72
Scheduler Entries	97	160
Register Files – Integer + FP	180 + 168	280 + 224
Allocation Queue	64/thread	70/thread 140/1 thread
L1D Cache (KB)	32	48
L2 Unified TLB (STLB)	1.5K	2K
STLB-IG Page support	16	1024 (shared w/ 4K)
STLB-IG Page support	16	1024 (shared w/ 4K)
Mid-level Cache (MB)	1	1.25

NAMD Satellite Tobacco Mosaic Virus Simulation

3rd Generation Intel® Xeon® Scalable Processors



Performance varies by use, configuration, and other factors. Configurations see appendix [1, 2, 3]

Can we do better for AVX-512?

- Reasons for non-ideal vector scaling
  - Predication inefficiency
  - Horizontal Vector Operations
  - Frequency Throttling
  - Non-contiguous memory access

Current gather µarch implementation limited by cache BW

- 2 elements per cycle vs 32 single precision FMAs per cycle
- **Limits scaling from 256b vector widths to 512b**
  - 2X FMAs per cycle, but very minor improvement to gather throughput
  - **NAMD OMP-SIMD implementation is heavily L1-bound w/ 512b vector widths**

Memory divergence can further limit scaling (single cache miss for 1 element)

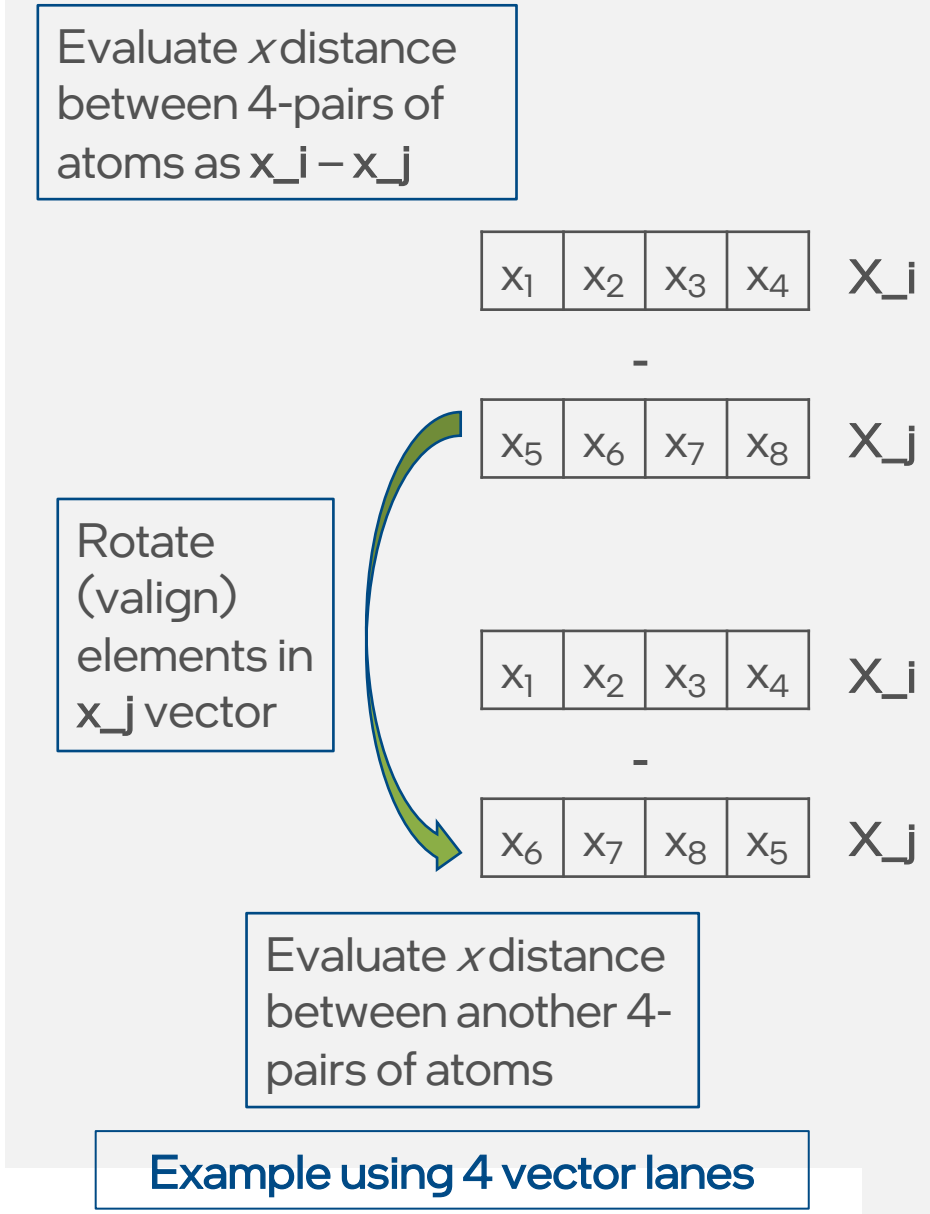
Optimizing Memory Access

- There are known optimizations for memory access in MD
  - E.g., in GROMACS and NAMD CUDA implementations
    - Also improve performance on GPUs , but for different primary reason
    - Previously not implemented for CPU in NAMD

- NAMD Tiles Algorithm
  - Evaluate interactions between clusters of atoms instead of pairs
  - At a high-level, similar to cache-blocking, but for vector registers
    - Block atomic pairwise evaluations into tiles to improve reuse of data in registers …without going back to L1 data-cache as often

Tiles Algorithm in NAMD

- Atoms are spatially sorted with an orthogonal recursive bisection method to cluster atoms based on the vector / SIMD width
- Rather than evaluating atomic interactions in loops at a pair-wise granularity, evaluate interactions in tiles between clusters of atoms
  - Significantly reduces L1 pressure and increases IPC via elimination of gather operations
  - Eliminates some horizontal operations for summing up results with shuffles

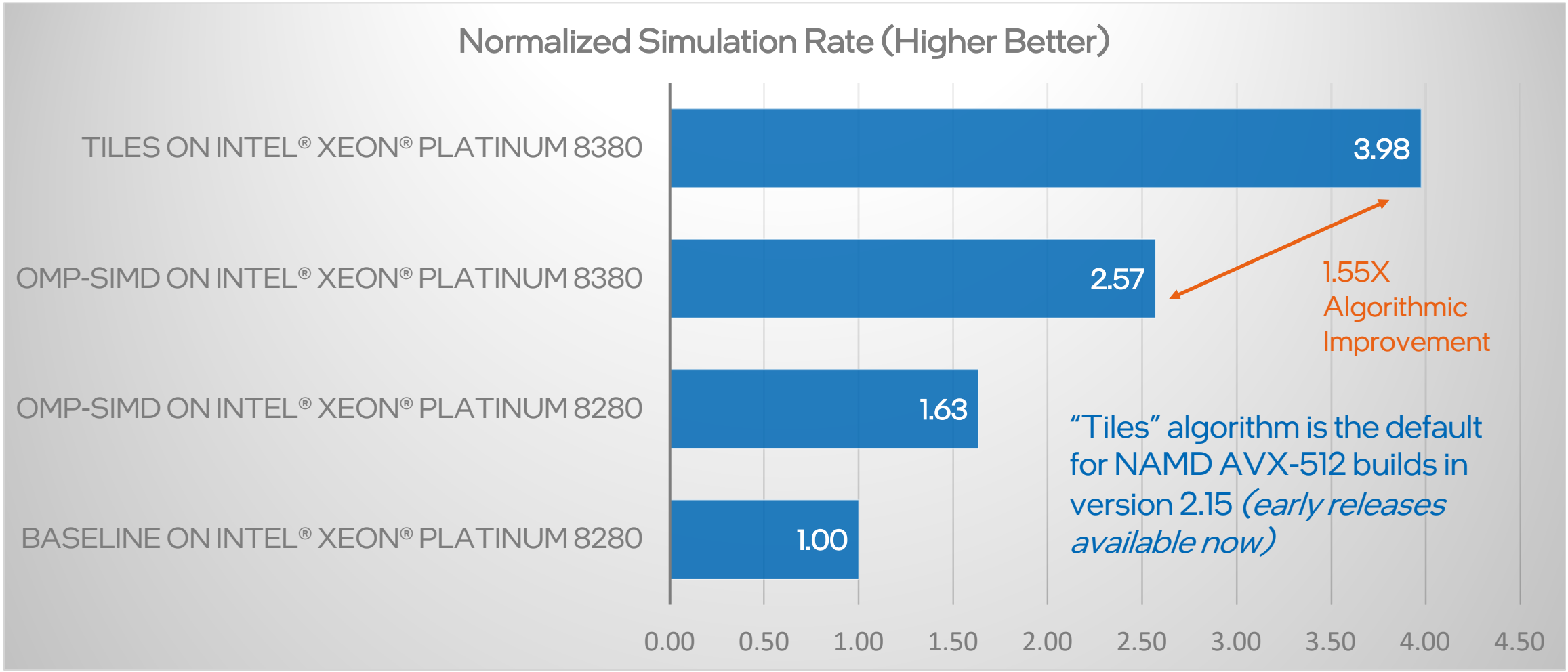


Downsides of the Algorithm

- The neighbor list build for atoms within cutoff is now a list of neighboring *clusters*
  - Many more atom pairs evaluated than necessary
    - ↑ Instruction Count, ↑ Predication inefficiency (masking)
    - Tradeoff depends on the direct force cutoff and arithmetic intensity of the simulation model
      - For atomistic biological simulations with larger cutoff, this type of algorithm can give significant performance upside

NAMD Satellite Tobacco Mosaic Virus Simulation

Tiles on 3rd Generation Intel® Xeon® Scalable Processors



Performance varies by use, configuration, and other factors. Configurations see appendix [1, 2, 3, 4]

Differences from NAMD GPU Implementation

- More extensive use of explicit math versus interpolation to further reduce L1 pressure and increase numerical accuracy
- The algorithm is integrated into the CPU load-balancing protocols (as opposed to thread aggregation w/ offload for GPU scheduling)
- Increased use of FP64 vs FP32 for accumulation
- Handling of modified/excluded interactions integrated into algorithm
- A hybrid Tiles/Pair algorithm allows some interactions to be calculated in traditional pair-wise vector loops (↓ predication inefficiency)
- Minor differences for using explicit vector mask registers w/ AVX-512

Programming Model

- Concept of a vector is tightly coupled with the Tiles algorithm
  - OpenMP SIMD has evolved as a model where the developer writes scalar code that is vectorized by the compiler – very unnatural for this type of algorithm
- Current implementation uses AVX-512 vector intrinsics
  - Much simpler with AVX-512 versus AVX-2 due to ISA enhancements
- Data Parallel C++, a component of Intel® oneAPI offers an alternative
  - Abstraction where workers correspond to vector lanes allowing tighter control over the vector algorithm with a high-level programming model
  - Portability across accelerator targets (AVX-512, GPU, FPGA, …)

The Theoretical and Computational Biophysics Group at UIUC has established a oneAPI academic Center of Excellence (CoE) focusing on standards-based cross-architecture programming, preparing NAMD for the exa-scale Aurora supercomputer.

Summary

- Intel continues to provide significant advances in products for HPC research critical for improving health care and quality of life
- The ISA enhancements in AVX-512 offer significant improvements in developer productivity and performance on Intel® platforms
  - As with most modern platforms, optimizing memory access can be important to allow software to best exploit the high computational throughput available.
- 3rd Generation Intel® Scalable Processors show big gains for HPC
- oneAPI tools are free and represents a significant advance for development across a variety of platforms
  - *This is not a promise of performance portability, but rather a consistent abstraction and language for vector algorithms, data management, and accelerator offload along with a consistent API for optimized libraries*

Performance Disclaimers

- Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex
- Intel optimizations, for Intel compilers or other products, may not optimize to the same degree for non-Intel products.
- Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.
- Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries and brands may be claimed as the property of others.

Appendix - Configurations

1. Baseline on Intel® Xeon® Platinum 8280 - App Version: NAMD 2.13; Build notes: Tools: Intel MKL, Intel C Compiler 2020u1, Intel MPI 2019u7, Intel Threading Building Blocks 2019u4; threads/core: 2; Turbo: used; Build knobs: -ip -fp-model fast=2 -no-prec-div -qoverride-limits -qopenmp-simd -O3 -xCORE-AVX512 -qopt-zmm-usage=high -DNAMD_KNL. 1-node, 2x Intel Xeon Platinum 8280 (28C/2.7GHz, 205W TDP) processor on Intel Software Development Platform with 192GB (12 slots/ 16GB/ 2933) total DDR4 memory, ucode 0x500002c, HT on, Turbo on, CentOS Linux CentOS Linux 7.7.1908 3.10.0-1127.13.1.el7.crt1.x86_64, 1x Intel_SSDSC2BA80 .Tested by Intel Jul 30th, 2020
2. OMP-SIMD on Intel® Xeon® Platinum 8280 - App Version: NAMD 2.15-Alpha1; Build notes: Tools: Intel MKL, Intel C Compiler 2020u4, Intel MPI 2019u8, Intel Threading Building Blocks 2020u4; threads/core: 2; Turbo: used; Build knobs: -ip -fp-model fast=2 -no-prec-div -qoverride-limits -qopenmp-simd -O3 -xCORE-AVX512 -qopt-zmm-usage=high -DNAMD_KNL. 1-node, 2x Intel Xeon Platinum 8280 (28C/2.7GHz, 205W TDP) processor on Intel Software Development Platform with 192GB (12 slots/ 16GB/ 2933) total DDR4 memory, ucode 0x4002f01, HT on, Turbo on, CentOS Linux 8.3.2011, 4.18.0-240.1.1.el8_3.crt1.x86_64, 1x Intel_SSDSC2KG48 .Tested by Intel between February 1, 2021 to February 20, 2021
3. OMP-SIMD on Intel® Xeon® Platinum 8380 - App Version: NAMD 2.15-Alpha1; Build notes: Tools: Intel MKL, Intel C Compiler 2020u4, Intel MPI 2019u8, Intel Threading Building Blocks 2020u4; threads/core: 2; Turbo: used; Build knobs: -ip -fp-model fast=2 -no-prec-div -qoverride-limits -qopenmp-simd -O3 -xCORE-AVX512 -qopt-zmm-usage=high -DNAMD_KNL. 1-node, 2x Intel Xeon Platinum 8380 (40C/2.3GHz, 270W TDP) processor on Intel Software Development Platform with 256 GB (16 slots/ 16GB/ 3200) total DDR4 memory, ucode 0x261, HT on, Turbo on, CentOS Linux 8.3.2011, 4.18.0-240.1.1.el8_3.crt1.x86_64, 1x Intel_SSDSC2KG96, Tested by Intel between March 12, 2021 and March 29, 2021
4. Tiles on Intel® Xeon® Platinum 8380 - App Version: NAMD 2.15-Alpha1; Build notes: Tools: Intel MKL, Intel C Compiler 2020u4, Intel MPI 2019u8, Intel Threading Building Blocks 2020u4; threads/core: 2; Turbo: used; Build knobs: -ip -fp-model fast=2 -no-prec-div -qoverride-limits -qopenmp-simd -O3 -xCORE-AVX512 -qopt-zmm-usage=high -DNAMD_AVXTILES. 1-node, 2x Intel Xeon Platinum 8380 (40C/2.3GHz, 270W TDP) processor on Intel Software Development Platform with 256 GB (16 slots/ 16GB/ 3200) total DDR4 memory, ucode 0x261, HT on, Turbo on, CentOS Linux 8.3.2011, 4.18.0-240.1.1.el8_3.crt1.x86_64, 1x Intel_SSDSC2KG96, Tested by Intel between March 12, 2021 and March 29, 2021.


intel