ALCF HANDS-ON HPC WORKSHOP

Argonne NATIONAL LABORATORY

# PATH TO EXASCALE MATERIAL SIMULATION


**YE LUO**
Argonne National Laboratory

Oct 10th 2023 Argonne National Laboratory

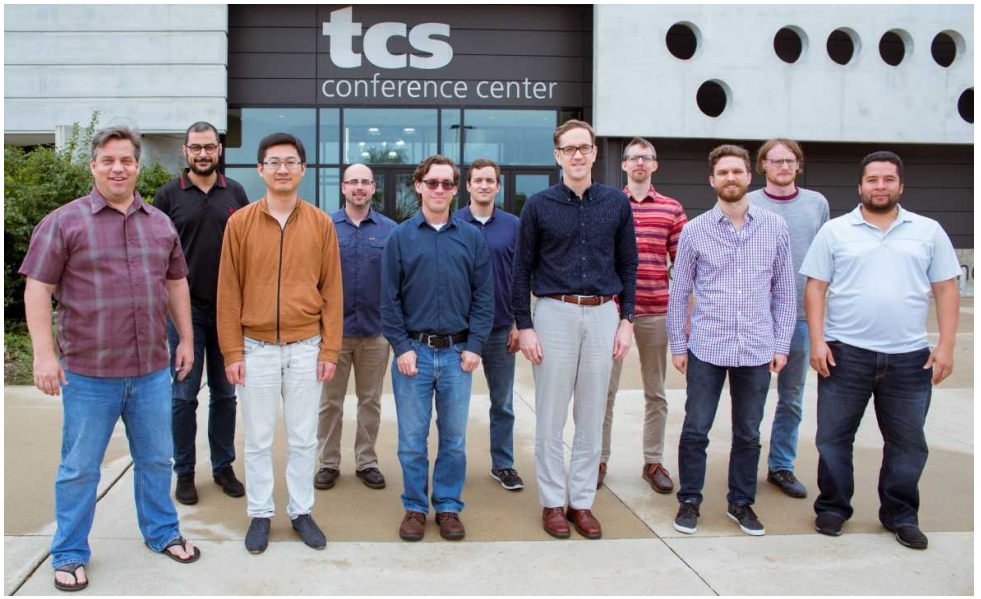
# OUTLINE

- Quantum Monte Carlo intro
- A High-Performance Design for Hierarchical Parallelism (HiPar 22)
- A highly efficient delayed update algorithm for evaluating Slater determinants (APS March Meeting 23)
- Performance benchmark.

# ACKNOWLEDGEMENT

## Exascale Computing Project : application development


- This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.

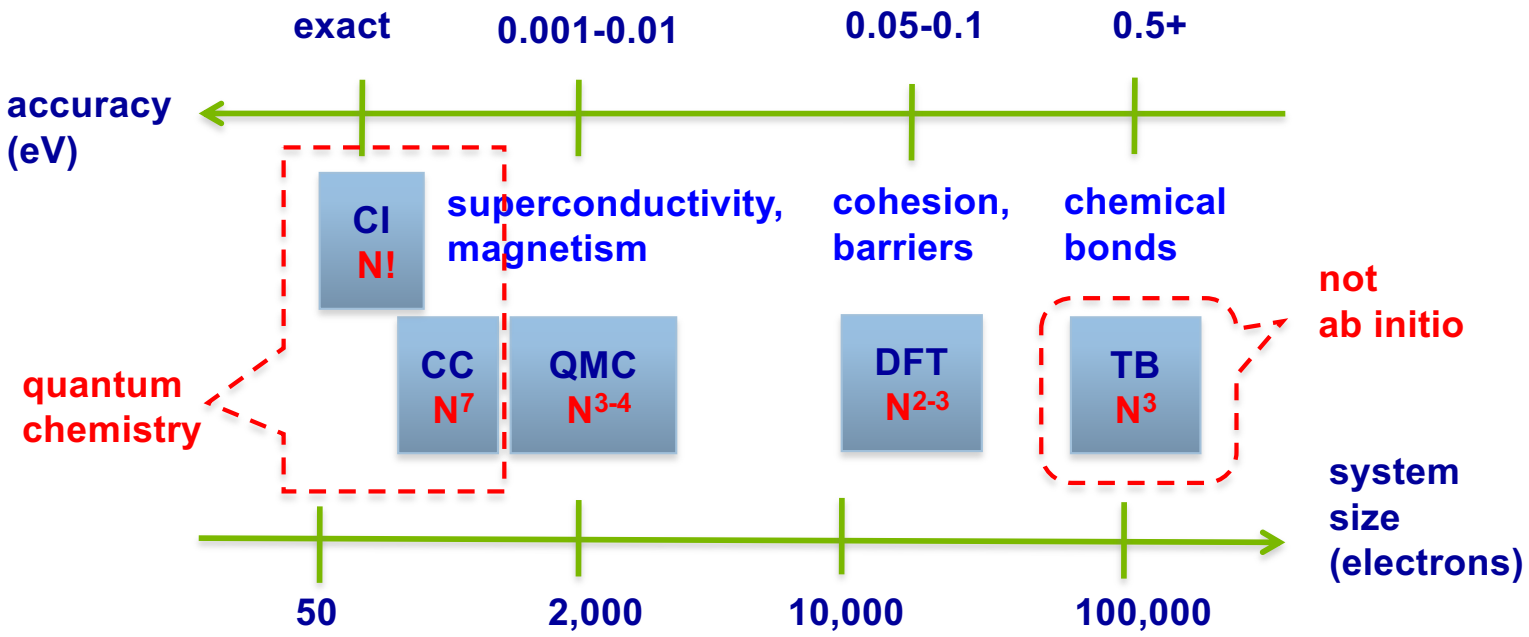



# ELECTRONIC STRUCTURE METHODS

## QMC can be the new sweet spot

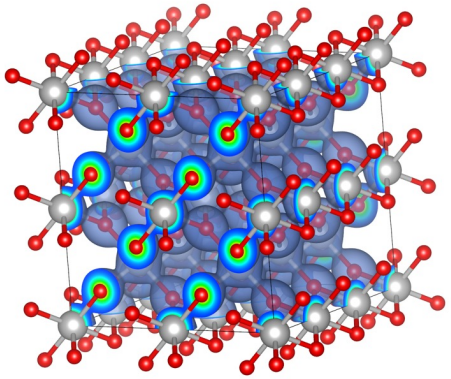
Time scale: picosecond = $10^{-12}$ seconds

Length scale: 10 nm = $10^{-8}$ meters

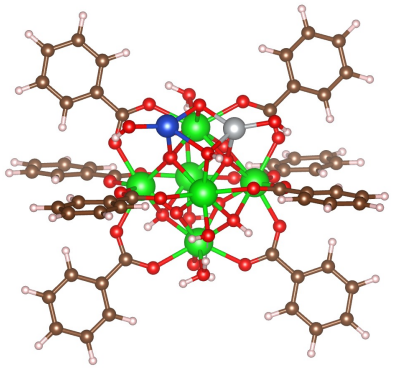


# PETASCALE TO EXASCALE CHALLENGE

## How large problem can we solve?



TiO2 polymorphs
216 atoms with 1536 electrons, 10 meV/f.u.
YL et al. New J. Phys. 18 113049 (2016)



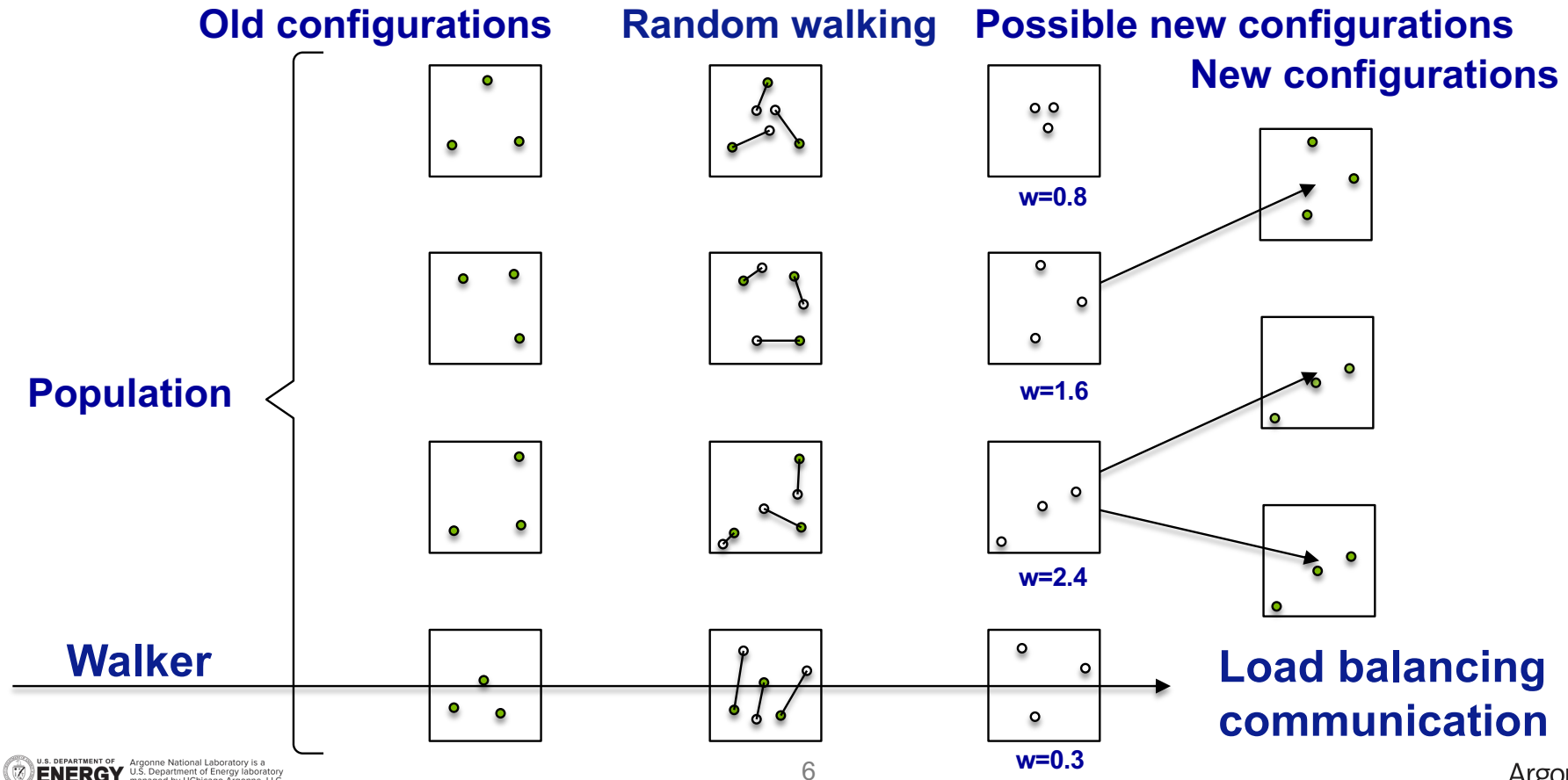
Metal organic framework
153 atoms with 594 electrons, 10 meV total energy.
A Benali, YL, et al. J. Phys. Chem. C, 122, 16683 (2018)

### What is next?

1. Solve faster and more petascale problems
2. Solve much larger problems

1k atoms
10k electrons

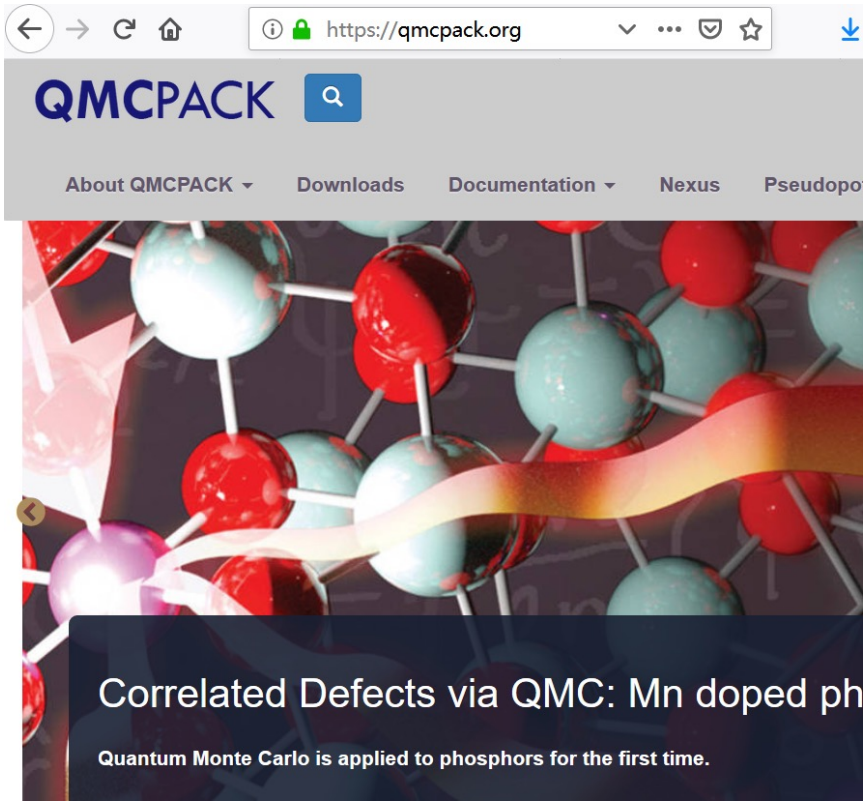
# DIFFUSION MONTE CARLO SCHEMATICS



# A HIGH-PERFORMANCE DESIGN FOR HIERARCHICAL PARALLELISM

# QMCPACK

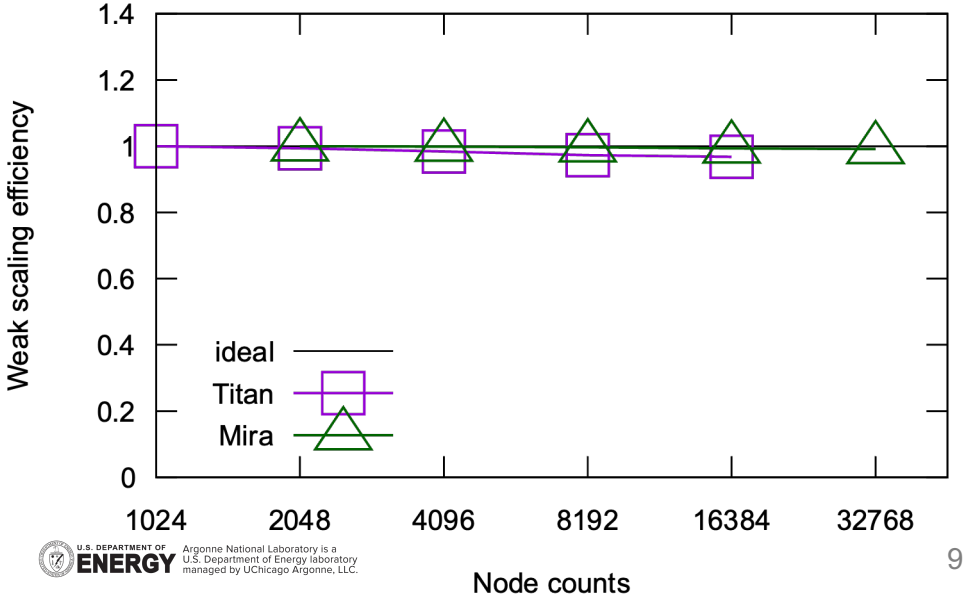
- QMCPACK, is a modern high-performance open-source Quantum Monte Carlo (QMC) simulation code for electronic structure calculations of molecular, quasi-2D and solid-state systems.
- The code is C/C++ and adopts MPI+X (OpenMP+offload/CUDA/HIP/SYCL)
- Monte Carlo: massive Markov chains (walkers) evolving in parallel. 1st level concurrency. Good for MPI and coarse level threads.
- Quantum: The computation in each walker can be heavy when solving many body systems (electrons). 2nd level concurrency. Good for fine level threads and SIMD.

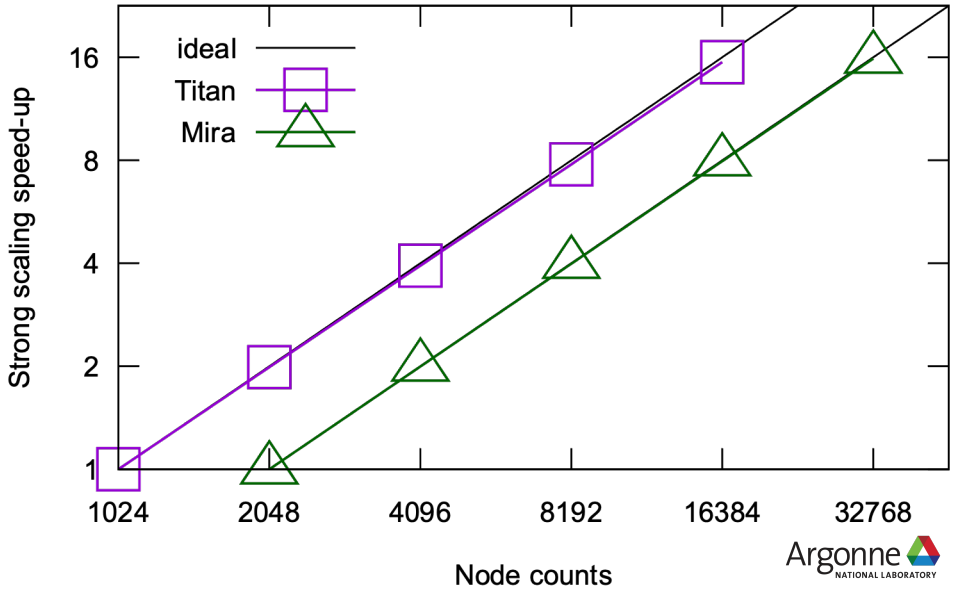


# WALKER BASED PARALLELISM

## Works extreme well on petascale supercomputers

- Weak scaling efficiency 99% on 2/3 Mira and 95% on almost full Titan.
- Weak scaling, fix work per node. Strong scaling, fix the total number of samples.
- Equilibration excluded.



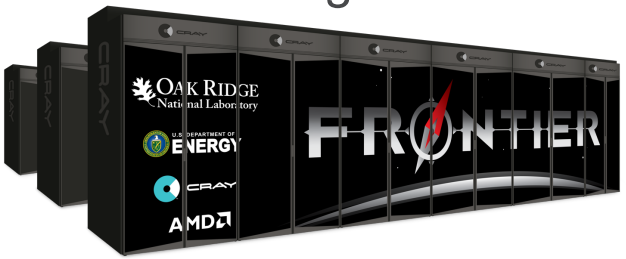
# PRE AND EXASCALE SYSTEMS IN 2023

## Need to make the code work on all machines

NERSC Perlmutter


Argonne Polaris


Oak Ridge Frontier



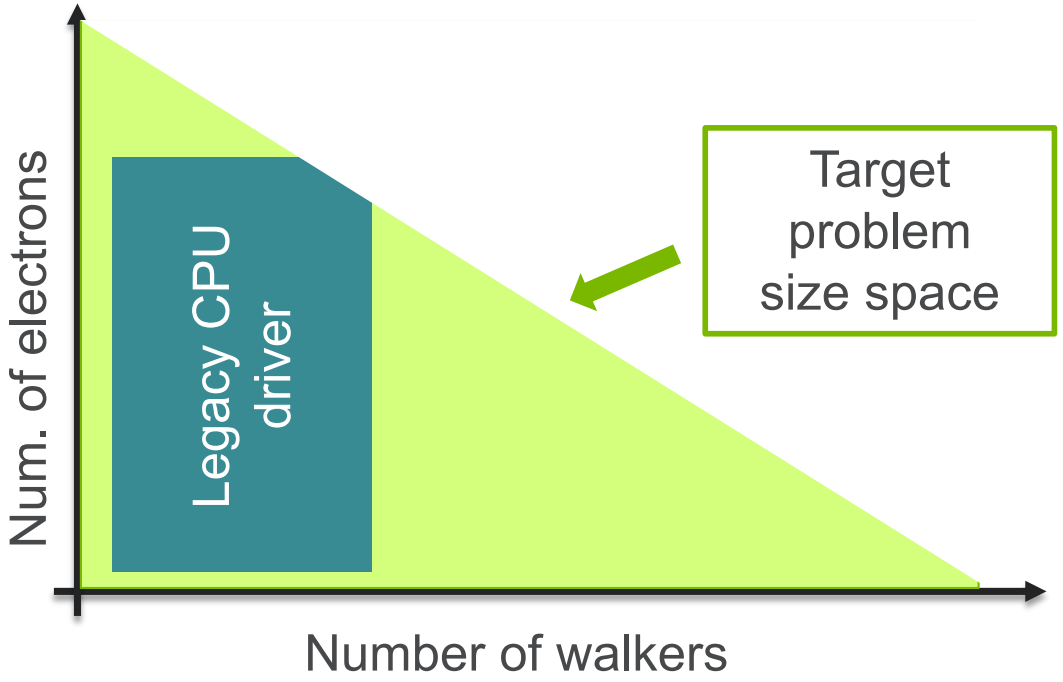

Argonne Aurora


Make QMCPACK run well everywhere
including CPU-only systems
On-node performance is the focus

# LEGACY CPU IMPLEMENTATION

**Algorithm 2** Pseudocode for the multi-threaded implementation.

```
1: for MC generation = 1 ··· M do                    seq.
2:     #pragma omp parallel for
3:     for walker = 1 ··· N_w do                     para.
4:         for particle k = 1 ··· N do               seq.
5:             ...
6:         end for{particle}
7:     end for{walker}
8: end for{MC generation}
```

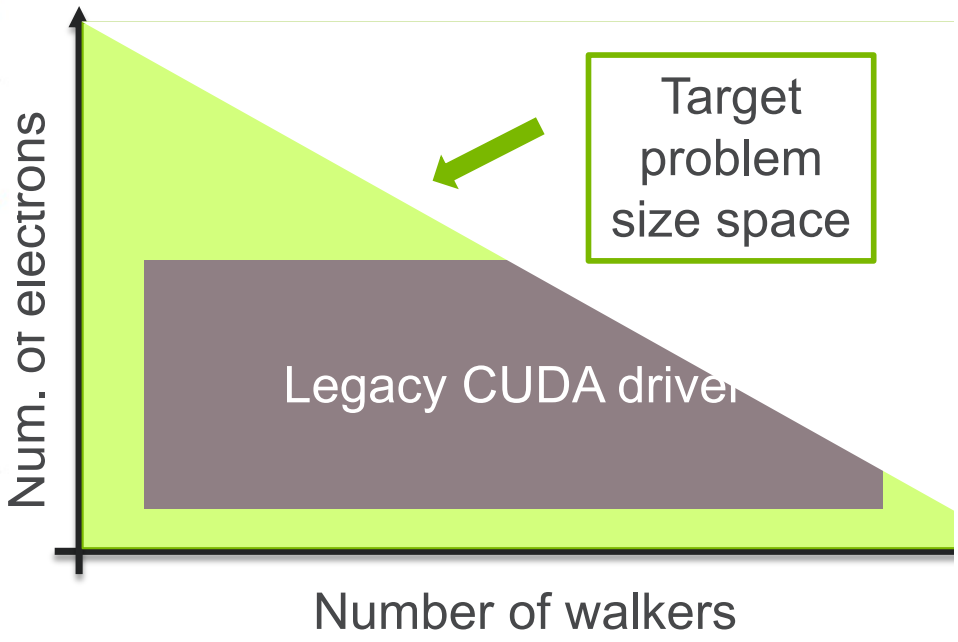


# CUDA-BASED GPU IMPLEMENTATION

**Algorithm 3** Pseudocode for the CUDA-based implementation.

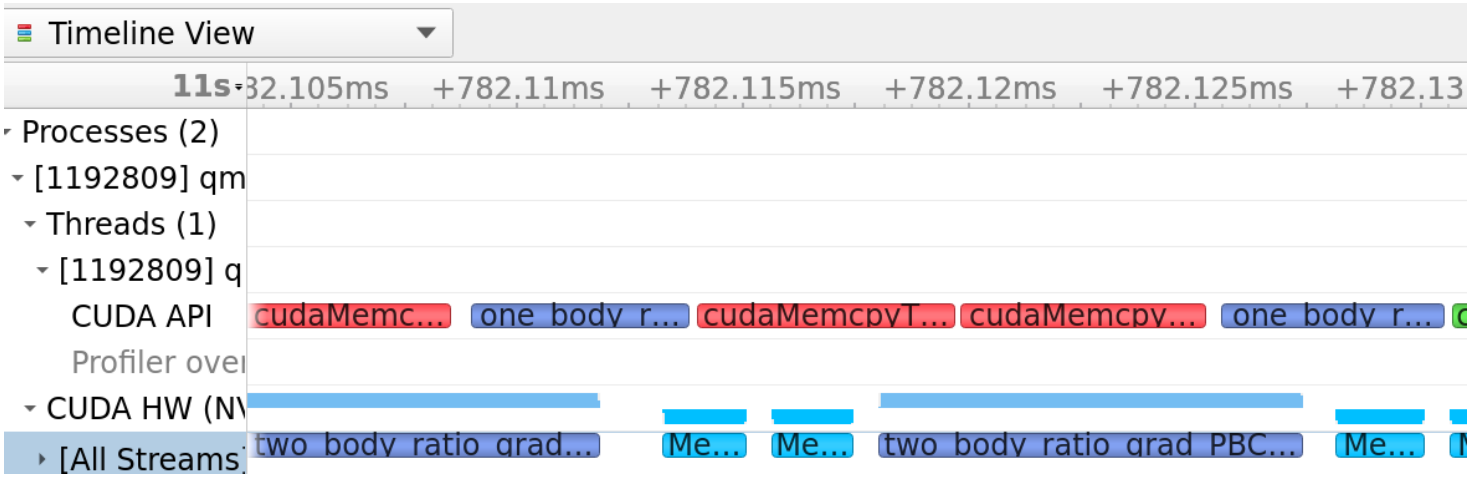
1: **for** MC generation $= 1 \cdots M$ **do** seq.
2: **for** particle $k = 1 \cdots N$ **do** seq.
3: Algorithm 1. Line 5,6,7,8,9 over all the $N_w$ walkers batched
4: **end for**{particle}
5: **local energy** $E_L = \hat{H}\Psi_T(\mathbf{R})/\Psi_T(\mathbf{R})$ over $N_w$
6: reweight and branch walkers based on $E_L - E_T$
7: update $E_T$ and load balance via MPI.
8: **end for**{MC generation}

Diverge APIs



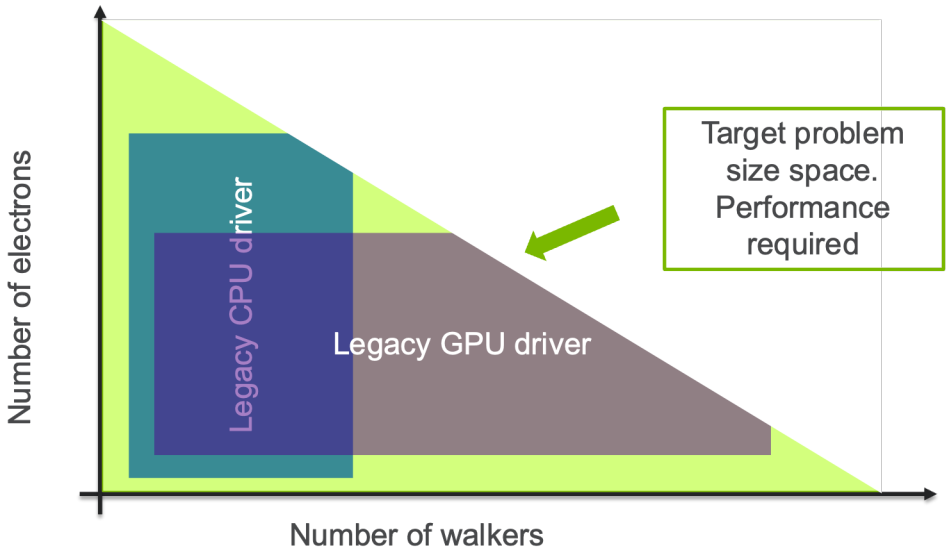
# TRACING ON NVDIA GPUS

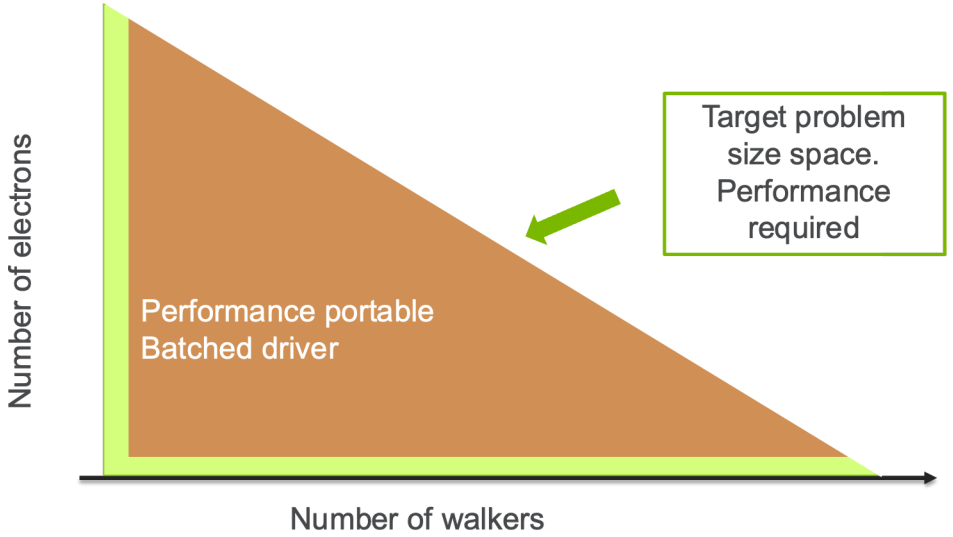
## Single thread + GPU idle gaps



# UNIFY BOTH IMPLEMENTATIONS

## By design

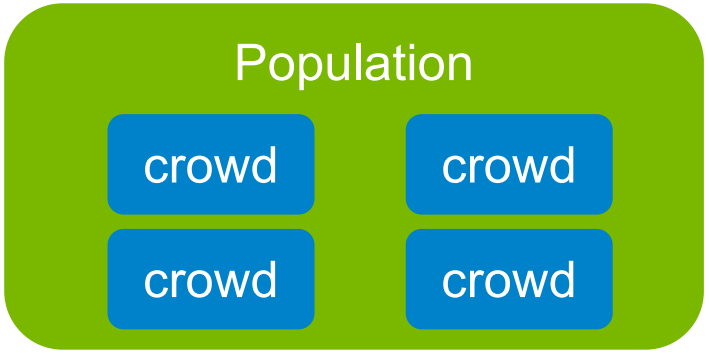


# NEW DESIGN WITH CROWDS

**Algorithm 4** Pseudocode for the batched DMC driver.

1: **for** MC generation $= 1 \cdots M$ **do** — seq.
2: #pragma omp parallel for
3: **for** crowd $= 1 \cdots C$ **do** — para.
4: **for** particle $k = 1 \cdots N$ **do** — seq.
5: Algorithm 1. Line 5,6,7,8,9 over all walkers with in this crowd — batched
6: **end for**{particle}
7: **local energy** $E_L = \hat{H}\Psi_T(\mathbf{R})/\Psi_T(\mathbf{R})$ over this crowd
8: reweight and branch walkers based on $E_L - E_T$
9: update $E_T$ and load balance via MPI.
10: **end for**{crowd} CG
11: **end for**{MC generation}



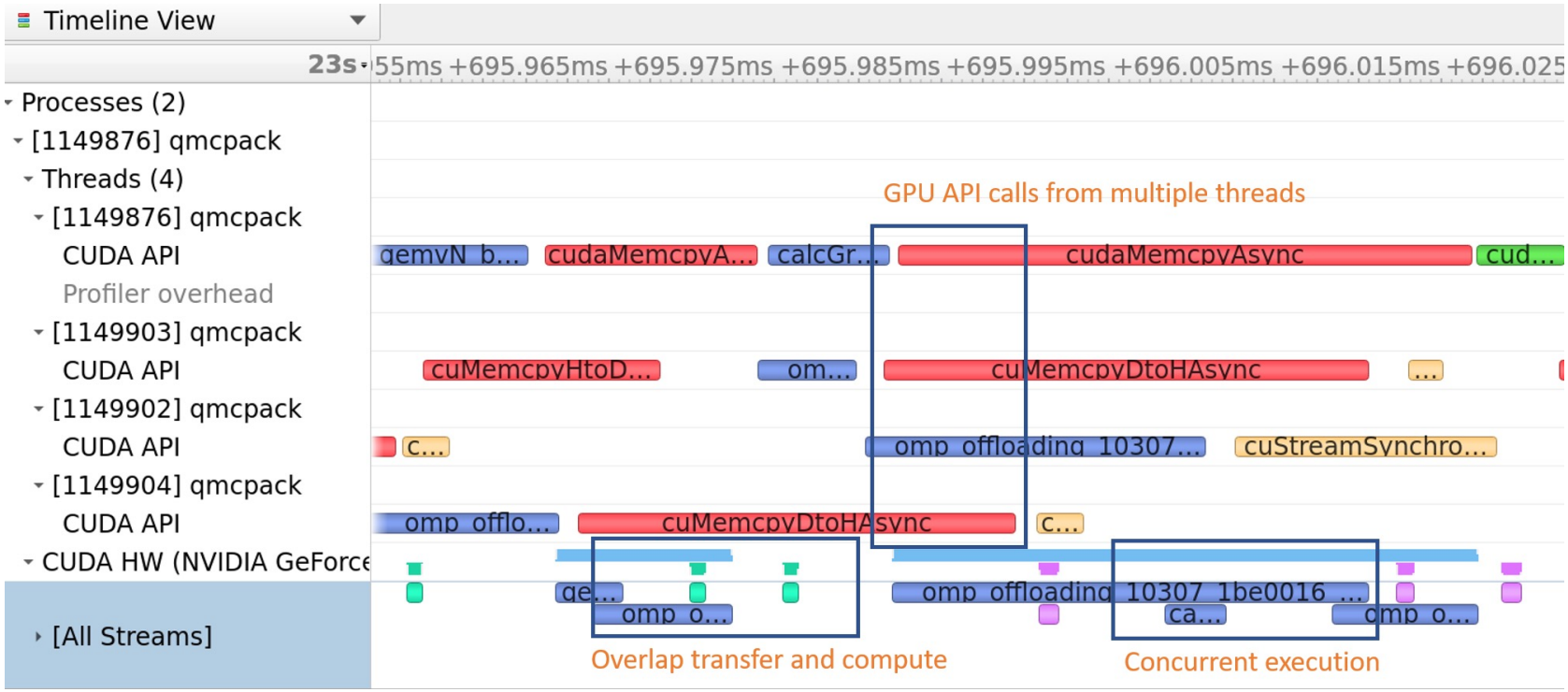

- lock-step walkers within a crowd
- Independent crowds
- Decay to legacy implementations

# OPENMP OFFLOAD GPU IMPLEMENTATION

## A bit more software technology to handle GPUs

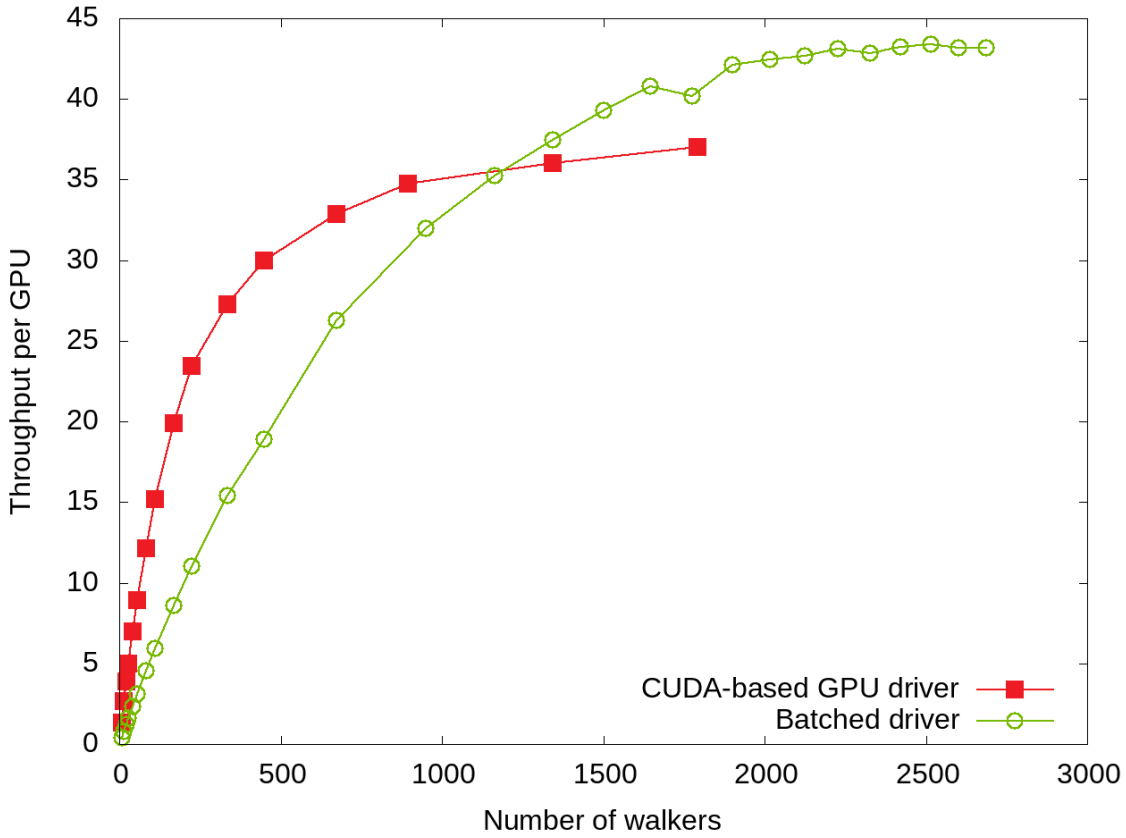
- Use portable OpenMP target feature
  - Portable on NVIDIA, AMD, Intel GPUs. Fallback on CPU as well.
  - Multiple compilers. GNU, Clang, AOMP, NVHPC, OneAPI
- Multiple crowds (CPU threads) to launch kernels to GPUs
  - Maximize GPU utilization. Overlapping compute and transfer by OpenMP.
- Specialized in CUDA/HIP/SYCL to call NVIDIA/AMD/INTEL accelerated libraries.
  - cuBLAS/cuSolver, hipBLAS/rocSolver, MKL

# CROWDS ON OPENMP THREADS



# ONE VS A FEW CROWDS

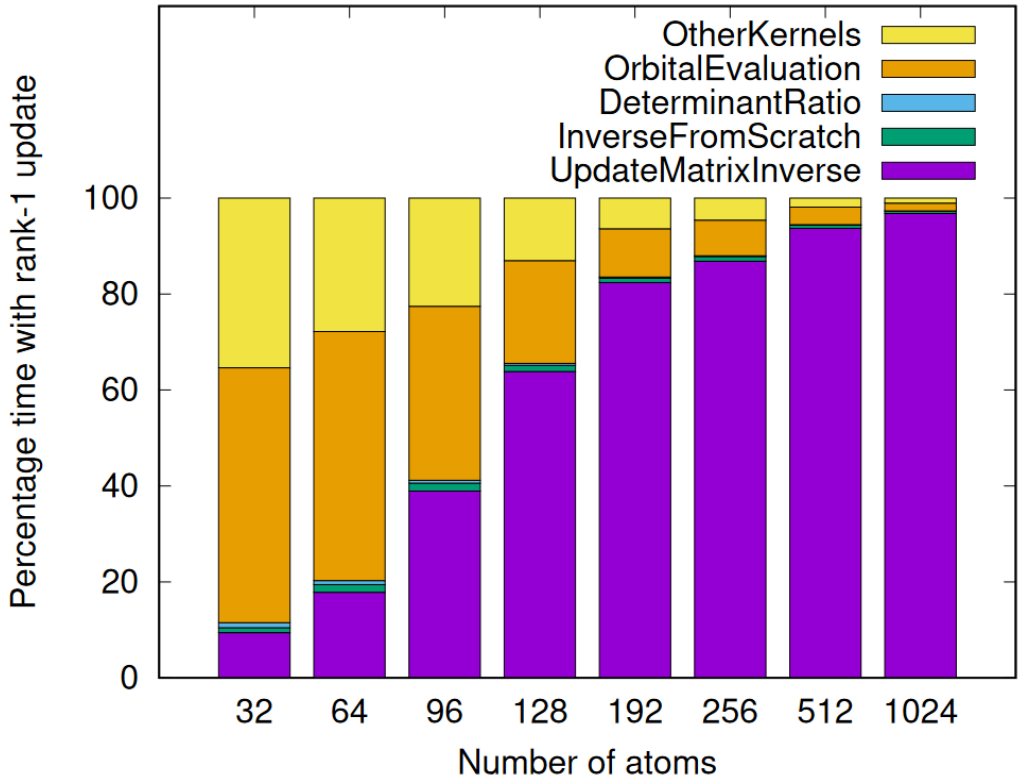
- 7 cores per GPU on OLCF Summit
- Fixed 7 crowds
- Small walker count, performance drops
- Large walker count, performance improves.



# COMPUTATIONAL COST

## Determinant update dominates

- Overall cost in a Monte Carlo step $\alpha N_e^3+\beta N_e^2$
- In NiO solid benchmark, at medium to large atom counts, determinant update with rank-1 update takes almost all the time.
- Reason: rank-1 update uses $N_e^2$ FLOP and read/write $N_e^2$ numbers to memory. Unfriendly to hardware.
- Actual timing can be worse than cubic scaling due to out-of-cache.



# DELAYED UPDATE

## Using Sherman-Morrison-Woodbury formula

- Based on Fahy's variant not relying on U.
- Do not actually invert B. $B^{-1}_{n+1}$ can be constructed based on $B^{-1}_{n}$.
- Compute $v^{new}$ without the up-to-date inverse matrix.
- Computational cost $2N_e^3$ per Monte Carlo step.
- Matrix-matrix multiplication is the most efficient calculation on compute hardware.

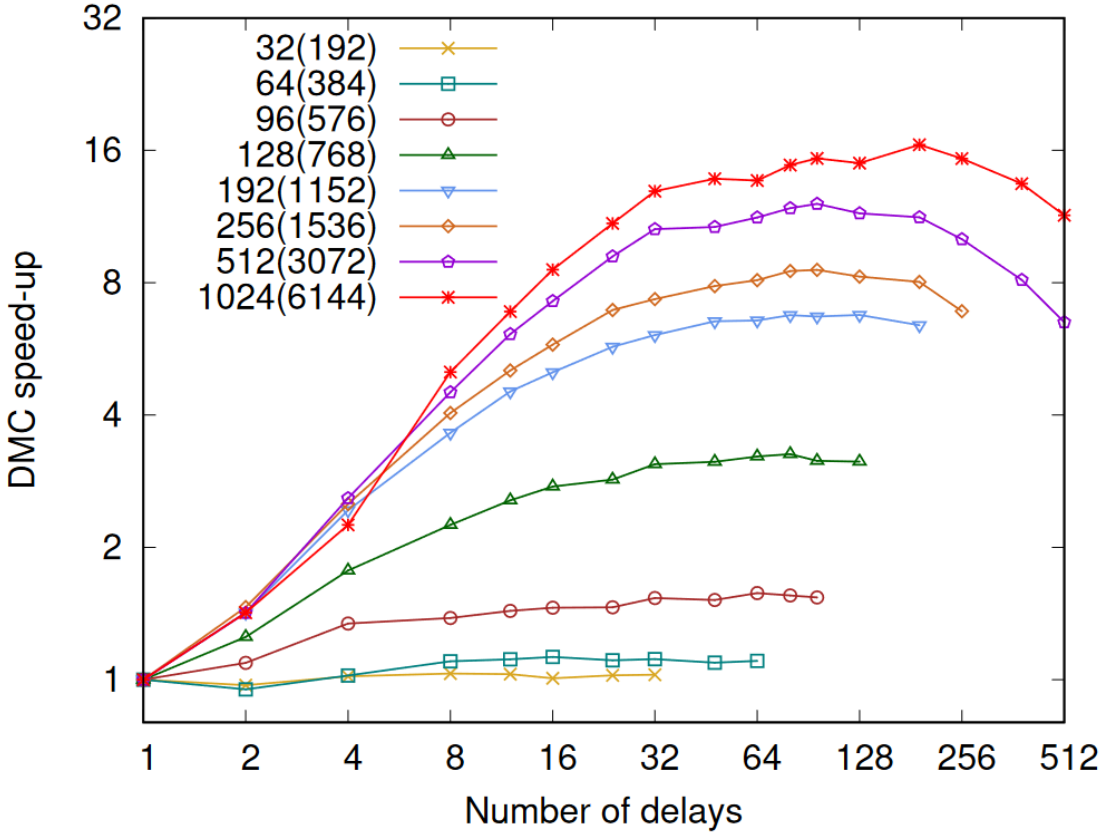
$$\begin{aligned}\tilde{\mathbf{A}}^{\text{new}} &= \{[\mathbf{A} + \mathbf{W}^T(\mathbf{U}^{\text{new}} - \mathbf{U})]^T\}^{-1} \\ &= \tilde{\mathbf{A}} - [\tilde{\mathbf{A}}(\mathbf{U}^{\text{new}})^T - \mathbf{W}^T]\mathbf{B}^{-1}\mathbf{V} \\ \mathbf{B} &= \mathbf{V}(\mathbf{U}^{\text{new}})^T\end{aligned}$$

$$\begin{aligned}\mathbf{v}^{\text{new}} &= \left(\mathbf{w}^T\tilde{\mathbf{A}}^{\text{new}}\right)^T \\ &= \mathbf{v} - \mathbf{V}^T(\mathbf{B}^T)^{-1}\mathbf{U}^{\text{new}}\mathbf{v}\end{aligned}$$

# SPEED UP WITH DELAYED UPDATE

## Up-to 16X

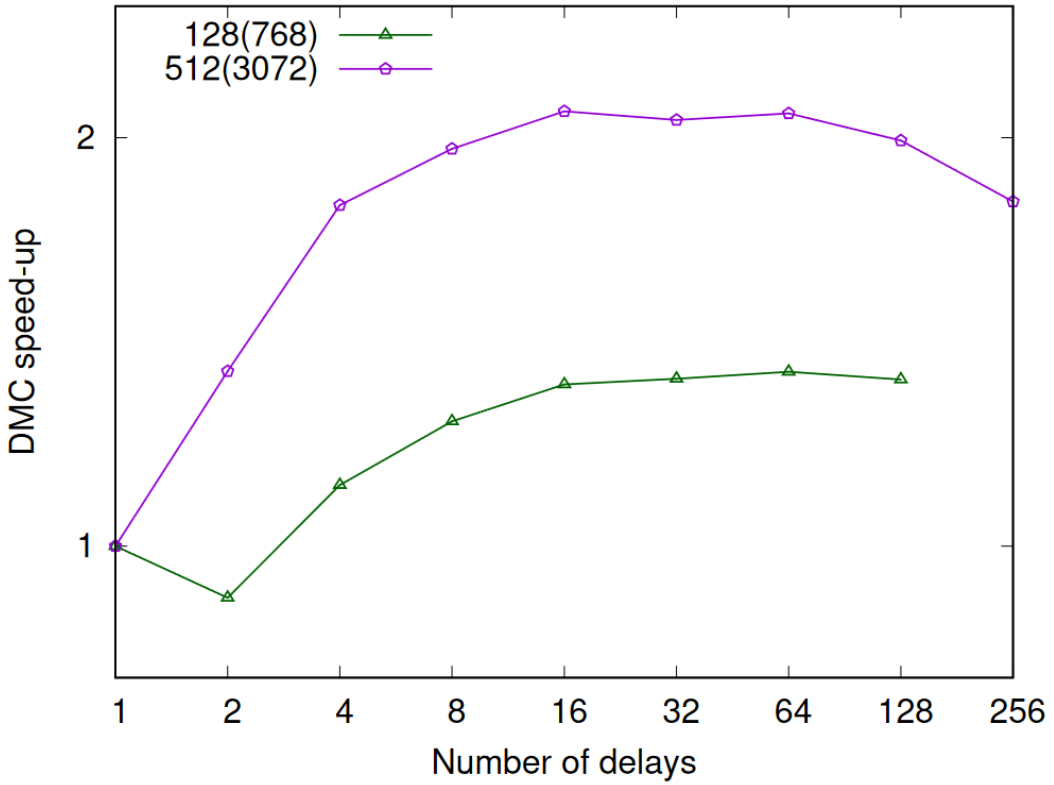
- Gain on all problem sizes.
- 16X speed-up with 1024 atom.



# GPU

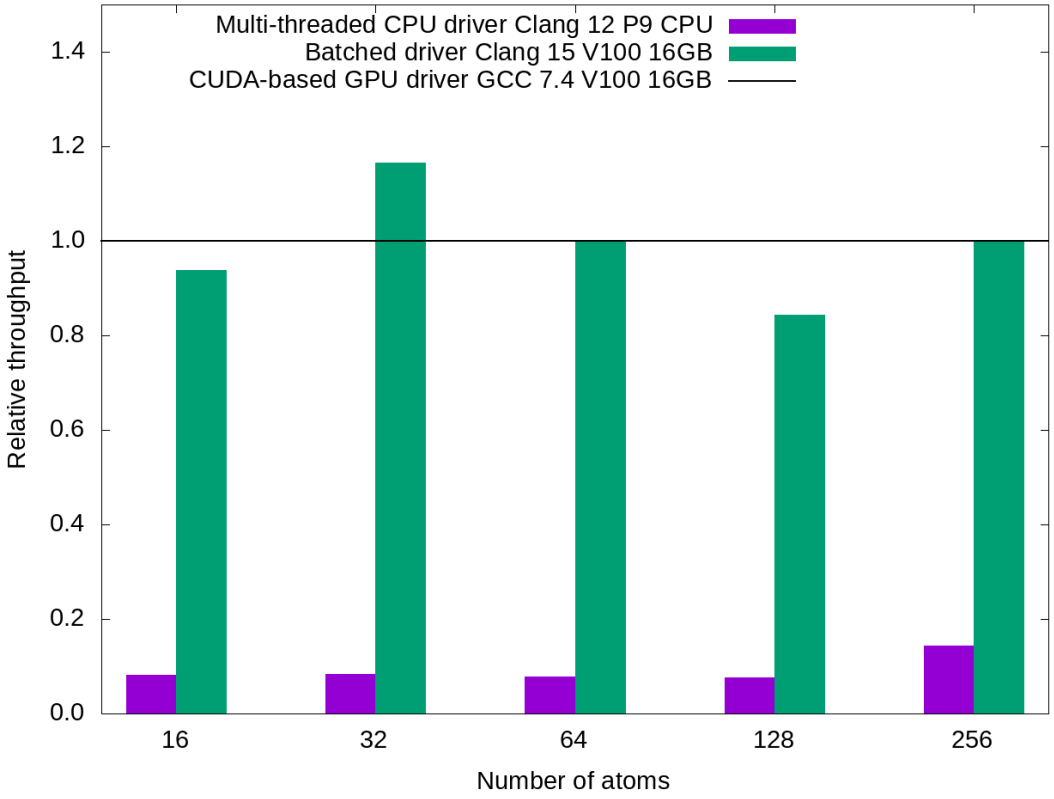
## Always gain

- At low electron counts, like a128, walkers are batched and run in lock-step. We add fake accept with unmodified orbitals to ensure all the walkers carry the same delay count and keep consistent sizes of matrices.
- At high electron counts, like a512, walkers are fully independent like the CPU case but data movement is more complicated.



# NEW IMPLEMENTATION PERFORMANCE

- The new implementation rivals the original CUDA-based GPU implementation.
- Production quality on NVDIA GPU systems like Summit and Polaris.
- Still problematic on AMD GPUs due to ROCr bugs.
- Intel software works well with this design



Aurora

# Computing Quantum Mechanical Properties faster.


intel

ISC High Performance
The HPC Event.

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

# PORTING CHALLENGES

- Multiple threads sharing a GPU.
- Matrix solver stability and performance
- Programming model interoperability. OpenMP offload + CUDA/HIP/SYCL
- Correctness and performance of compiler and runtime libraries.
- Continuous science production.

# CONCLUSION

- The new design of hierarchical parallelism maximized performance on hardware
- The new algorithm enable much more efficient calculation following hardware trend.
- The performance portable version of QMCPACK is ready for Exascale machines.

# GPU PORTING GUIDE

- OpenMP offload optimization guide: beyond kernels -Lessons learned in QMCPACK. SC23 booth talk on youtube
- Lessons Learned in Designing Performance Portable QMCPACK Using OpenMP Offload to GPUs. Jun. 2023 OpenMP Users Monthly Telecon
- Path to Exascale Material Simulation on Aurora Supercomputer. IXPUG 2023 annual conference