QMCPACK USERS WORKSHOP 2023

Argonne NATIONAL LABORATORY

REAL WORLD CPU AND GPU PERFORMANCE OF THE BATCH CODE

YE LUO
Computational Scientist
Argonne National Laboratory

Dec 11th 2023, Argonne National Laboratory, Lemont, IL

GPU RELATED CMAKE OPTIONS

-DXXX=ON/OFF, TRUE/FALSE, 1/0

- ENABLE_OFFLOAD=ON for OpenMP offload on NVIDIA, AMD and Intel GPUs
- For vendor accelerated libraries and optimizations (pick one)
  - ENABLE_CUDA=ON for CUDA acceleration on NVIDIA GPUs
  - ENABLE_CUDA=ON QMC_CUDA2HIP=ON for HIP acceleration on AMD GPUs
  - ENABLE_SYCL=ON for SYCL acceleration on Intel GPUs
- Use both OpenMP and vendor options in production for optimal performance.
- Use separate option for development
- (Optional) QMC_GPU_ARCHS="sm_80;sm_70" applies to both OpenMP and vendor.

CTEST

- When GPU is enabled, one test a time. No concurrent testing.
- Set MPI launcher options at CMake.
  - MPIEXEC_EXECUTABLE Specify the mpi wrapper, e.g. srun, aprun, mpirun, etc.
  - MPIEXEC_NUMPROC_FLAG Specify the number of mpi processes flag, e.g. "-n", "-np", etc.
  - MPIEXEC_PREFLAGS Flags to pass to MPIEXEC_EXECUTABLE directly before the executable to run. Pay attention to affinity.
    mpirun -np 12 -ppn 12 -d 8 --cpu-bind $CPU_BIND_VERBOSE qmcpack input.xml
  - ctest -j 16 to maximize potential CPU utilization.

INPUT TAG

- ‘gpu’ tag. Accepted values: yes/no/cuda/sycl/omptarget/cpu
- Coarse selection gpu=yes/no. In a GPU build, pick the most performant implementation usually cuda/sycl, can be omptarget depending on each feature.
- Precise selection gpu=cuda/sycl/omptarget/cpu

```
<sposet_collection type="einspline" ... gpu="yes">
  <sposet name="spo-ud" size="8">
     <occupation mode="ground" spindataset="0"/>
  </sposet>
</sposet_collection>
```

CPU-GPU NODE ARCHITECTURES

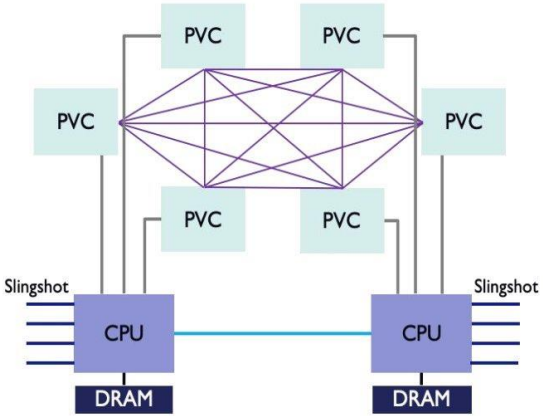
Aurora is complicated

0. One rank per NUMA
1. Put 1 MPI rank per PVC tile(GCD on AMD). Use 12 MPI ranks.
2. 52 CPU cores. Use 48 cores divisible by 6 per socket.
3. Don’t bothered by indivisible network links

Argonne NATIONAL LABORATORY

Aurora Compute Node

- 2 Intel Xeon (Sapphire Rapids) processors
- 6 X^e Architecture based GPUs (Ponte Vecchio)
  - All to all connection
  - Low latency and high bandwidth
- 8 Slingshot Fabric endpoints
- Unified Memory Architecture across CPUs and GPUs



Overview of the Argonne Aurora Exascale System
2:30pm - 3:30pm, Feb 5
Legends Ballroom

AFFINITY!!!

CPU CORE AFFINITY

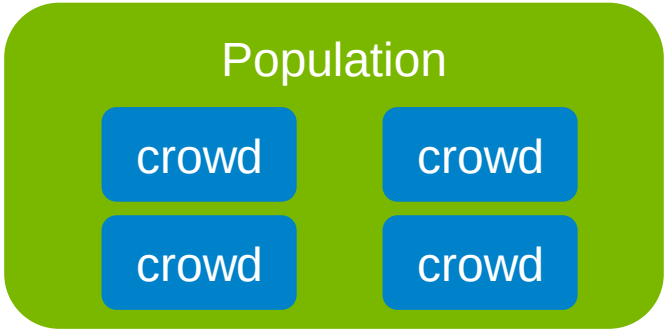
- When not using GPUs, choose 1 MPI process per CPU socket. bind the process to the socket and set OMP_NUM_THREADS=52 on a Aurora node case.
- Do not ignore it when using GPUs. Settle before GPU affinity. Filter out 48 cores of each socket and set OMP_NUM_THREADS=8.

GPU AFFINITY

- If the MPI launcher (mpiexec/mpirun) supports, pick 1 GPU per MPI rank and pick the closest one.
- If it doesn't, expose all the GPUs. QMCPACK picks a GPU for each MPI process in round-robin fashion within a node.

CONCEPT OF CROWDS

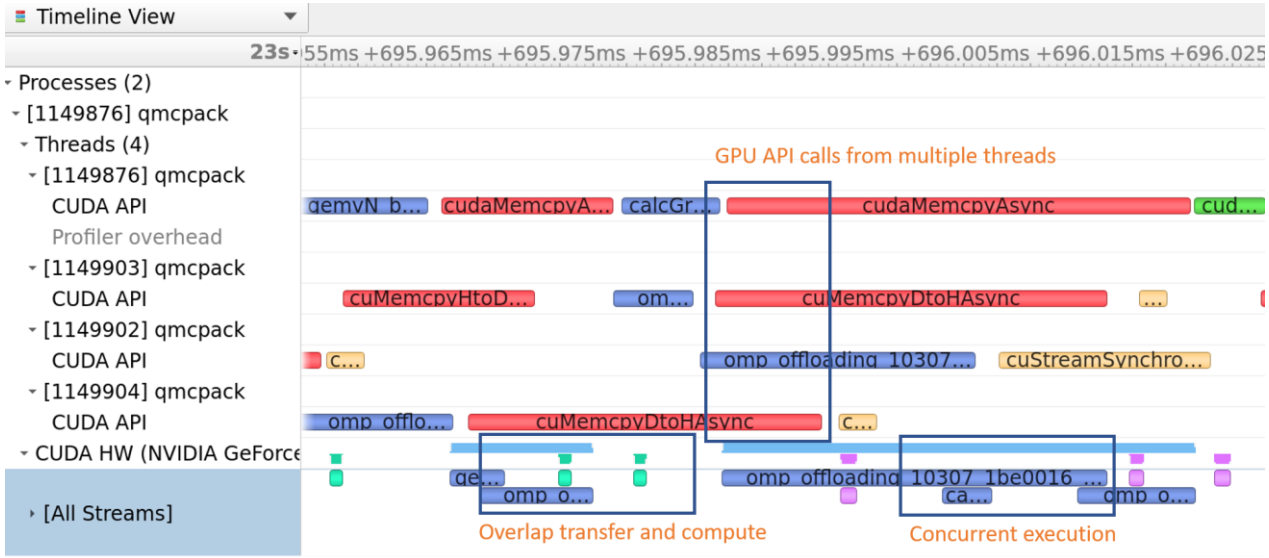
- The population within each MPI process(task) is divided into crowds. It is meant to maximize code performance.
- Crowds are mapped to CPU threads (<=16 preferred, default to #threads)
  - Crowd size 1, legacy CPU driver behavior
  - Number of crowds is 1, legacy CUDA driver behavior
- [Debug] crowd_serialize_walkers=yes driver input to calculate one walker at time.



- lock-step walkers within a crowd
- Independent crowds (8 on Aurora)
- Decay to legacy implementations

WHY ARE CROWDS NEEDED

- The assigned GPU is time shared by crowds and potentially concurrent execution.
- Overlap data transfer and computation.

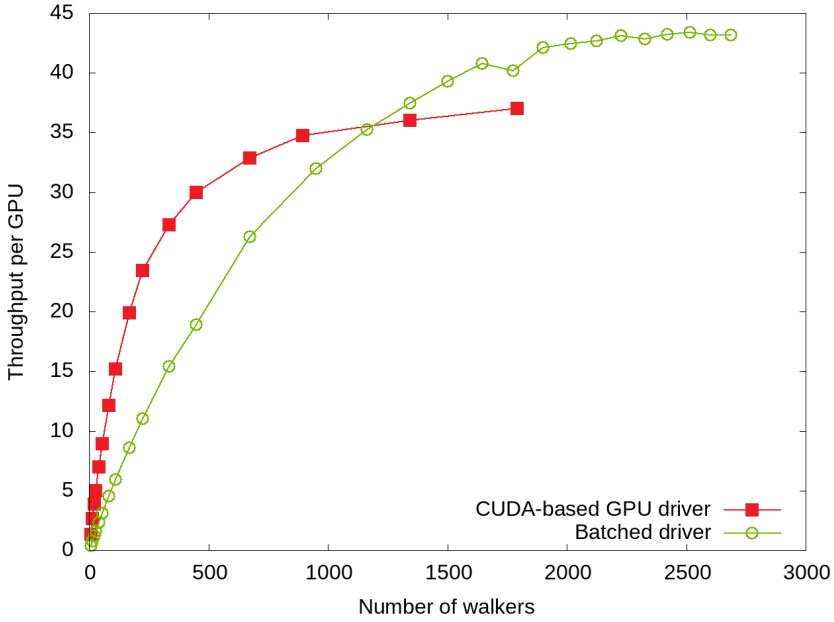


QMC THROUGHPUT

- Throughput definition
  - Workload / time. Number of samples / time in QMC.
  - Walkers steps per second.
  - `population size x steps x blocks / driver time`
  - Double node counts and double population size. How does throughput change?
  - Double steps or blocks. How does throughput change?

WALKERS PER RANK

- More GPU resident walkers, more performance
- At lower walker counts, fewer crowds are beneficial



CHOOSING THE NUMBER OF WALKERS

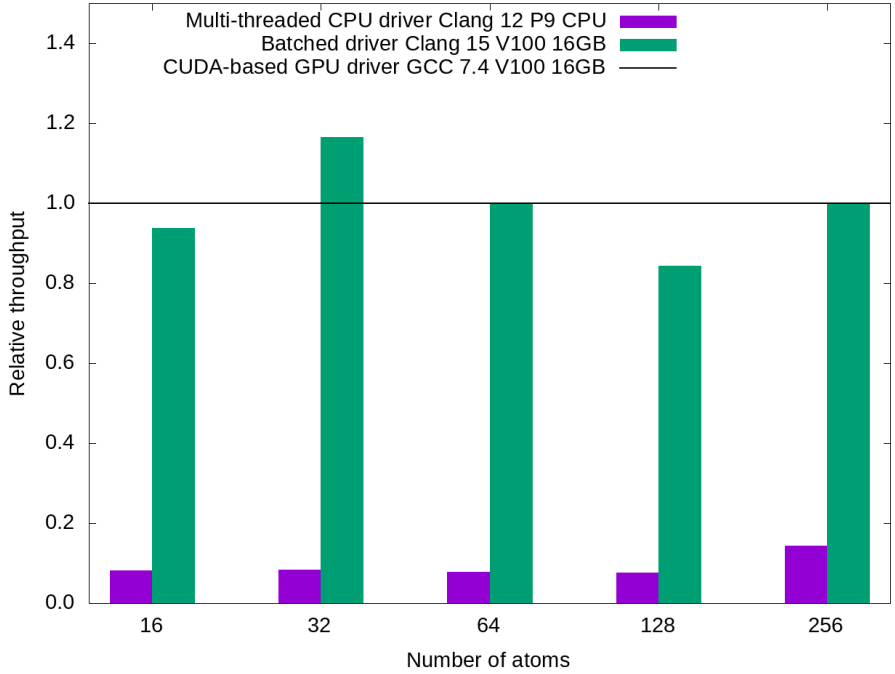
- walkers_per_rank vs total_walkers
  - “walkers” no longer accepted.
  - walkers_per_rank = MPI ranks x total_walkers must be satisfied.
  - Set one and the other is derived.
  - walkers_per_rank is set the number of crowds if neither provided. Recommended in CPU only runs.
- Walkers on each rank is directly related to resources
  - CPU memory per MPI process
  - GPU on-board memory
  - Optimal value depends on the selected features

MEMORY CONSIDERATION

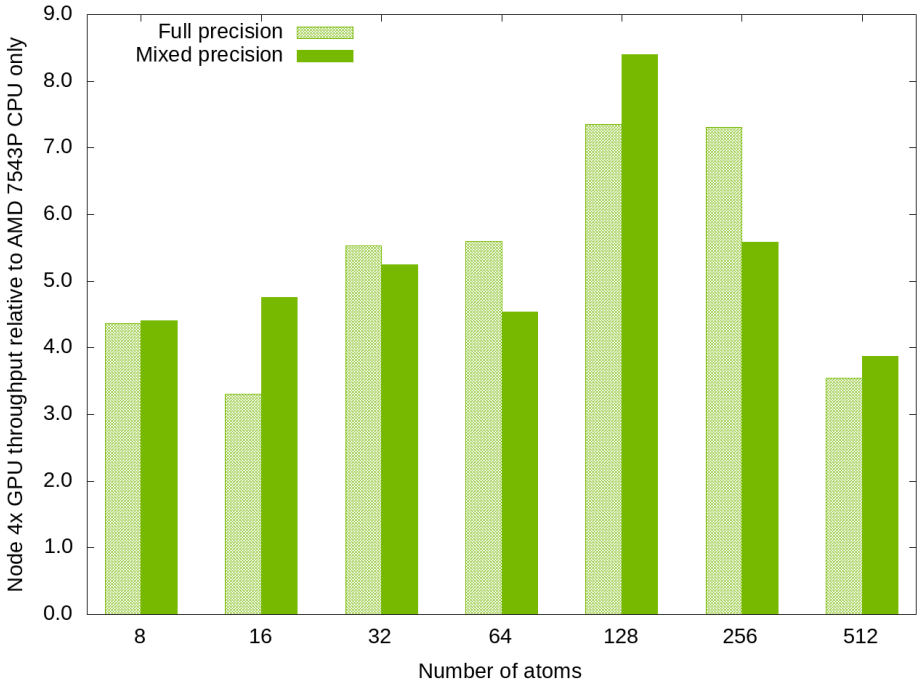
In the case of B-spline orbitals

- The B-spline table is GPU resident and shared among all the walkers within the MPI process.
- Determinant related memory scales linearly to the number of walkers
- Matrix inversion is on GPU by default. matrix_inverter="host" to run on CPU and save the scratch space on GPU.
- <particleset gpu="no"> leave distance tables on CPU and thus saves GPU memory.

GPU ACCELERATION ON SUMMIT

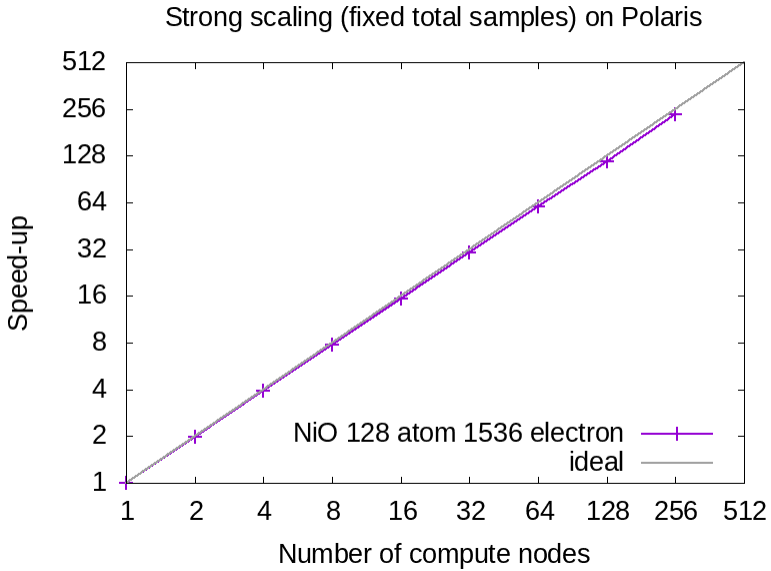


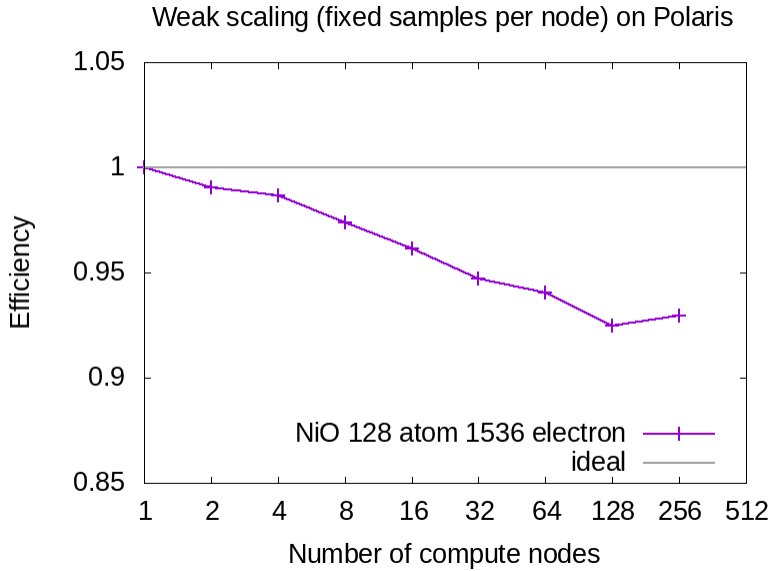
GPU ACCELERATION ON POLARIS



SCALING PLOTS

Expected Boring





REMINDER

- The GPU implementation may not always accelerate due to implementation feature constraints. 10 electron 1M-determinant system may be accelerated but 10 electron B-spline single-determinant system may be decelerated regardless of walker counts.
- Take advantage of runtime controls to mix and match GPU usage and maximize the code performance.
- QMCPACK command-line option --enable-timers=fine.