Enabling Fine-grained Gathering of Scientific Data in QMCPack Simulations on Titan

Stephen Herbein
University of Delaware
sherbein@udel.edu

ABSTRACT

Traditional petascale applications, such as QMCPack, can scale their computation to completely utilize modern supercomputers like Titan, but cannot scale their I/O. Because of this lack of scalability, scientists cannot save data at the granularity level they need to enable scientific discovery.

In this work, we tackle the problem of increasing the granularity of data collected from QMCPack simulations without increasing I/O overhead or compromising the simulation's scalability. Our solution relies on the redesign of the QMCPack algorithms to gather fine-grained information and the integration of the ADIOS API to help select effective I/O methods without major code changes.

Results presented in the poster outline how we can increase the quality of the scientific knowledge extracted from Titian simulations of QMCPack while keeping the I/O overhead below 10%.

1. MOTIVATION AND GOALS

As supercomputers and their applications begin to move closer to the exascale, scientists have to sacrifice saving important scientific data from their simulations at fine granularity level to maintain the simulation's scalability. The Quantum Monte Carlo code, QMCPack, has already reached this point at the petascale. QMCPack's existing I/O methods cannot save fine-grained data at scale. The current solution is to compute and save only energy averages of the entire molecular system, rather than each energy values for each particle.

Our main goal is to increase the data granularity saved from QMCPack without increasing I/O overhead or compromising scalability. This will facilitate scientific discovery at a faster pace.

2. BACKGROUND

QMCPack is a Quantum Monte Carlo simulation that finds solutions to Schrodinger's equation. The algorithm is loosely coupled where each process is a set of independent walkers. The code exploits both CPUs and GPUs. While in theory, QMCPack seems the perfect exascale application, it is hindered by its I/O performance.

ADIOS is an I/O framework that includes a suite of simple APIs and metadata stored in an external XML file. The metadata contains not only a representation of the data generated but also how the data should be written out to disk. The simple API allows for minimal changes to the existing code base while the metadata enables I/O method switching without recompiling the code.

3. METHODOLOGY

The contributions of this poster are twofold. First, we extend QMCPack's I/O from a single set of statistics per simulation using MPI_gather to a collection of traces per process using ADIOS. Our work resulted in two versions: STATS and TRACES. Second, we compare and contrast QMCPack's I/O performance with ADIOS versus the same code using HDF5.

3.1 STATS

This version of QMCPack creates an average across the system for each characteristic and saves those singular values to disk. This allows for almost no I/O overhead and high scalability. In the version with ADIOS, we simply replace HDF5 calls with ADIOS calls.

3.2 TRACES

This version of QMCPack includes a significantly expanded set of energy values to save per MC step. This version required a redesigned of algorithms to allow for collection of data at a per-walker granularity. These two changes significantly increase the amount of data saved per MC step at the cost of higher I/O overhead. Table 1 compares and contrasts the data sizes for both STATS and TRACES.

Table 1. Data output sizes for both QMCPack versions

Number of cores	STATS - HDF5	STATS - ADIOS	TRACES - HDF5	TRACES - ADIOS
2048	56KB	11KB	63GB	62GB
4096	56KB	11KB	126GB	124GB
8192	56KB	11KB	249GB	250GB
16384	56KB	11KB	502GB	502GB

4. RESULTS

4.1 Test and Platform Setup

We compared and contrasted I/O performance of QCMPack for an existing version of the code using HDF5 versus our modified version using ADIOS. Our experiments were conducted on Titan at ORNL using a Lustre file system, Spider. Each simulation had two MPI processes per node with eight OpenMP threads per MPI process. We ran on 2,048 cores up to 131,072 cores.

We consider three molecular systems different in size, i.e., graphite 3x3x1 with 36 carbons and 144 electrons, graphite 4x4x1 with 64 carbons and 256 electrons, and graphite 4x4x2 with 128 carbons and 512 electrons.

4.2 Benchmark Results

The STATS versions of QMCPack with ADIOS and HDF5 have similar performance to the code without I/O, because of the small amount of data stored per iteration. This comparison does not add any interesting I/O aspects and thus is not further discussed here.

The version of QMCPack with TRACES writes significantly more data. Graphite 4x4x2 writes between 6GB and 400GB every 60s; Graphite 4x4x1 writes between 3GB and 200GB every 12s; and Graphite 3x3x1 writes between 1.5GB and 100GB every 4s depending on the number of cores used on Titan. For this code we observed two major trends for the three graphite systems. For large systems such as graphite 4x4x2, ADIOS and HDF5 have similar performance. This trend is outlined in Figure 1. For smaller systems such as graphite 3x3x1, smaller, more frequent writes result in inefficiency at scale for both ADIOS and HDF5.

5. CONCLUSION AND FUTURE WORK

In this poster, we extended the QMCPack code with its high computing scalability, but very poor I/O performance to use the ADIOS API with its different I/O methods for the sake of collecting higher-in-granularity simulation data. We compared and contrasted the I/O performance of QMCPack with ADIOS versus a version using HDF5 and showed that the I/O overhead is less than 10% for both codes when using less than 32,768 cores. Using more than 32,768 cores results in more I/O overhead and greater variability in the I/O performance.

Future work includes integrating staging and in-transit analysis, modeling scientific queries on QMCPack data using skeleton I/O applications and determining the optimal data representations for reads and writes.

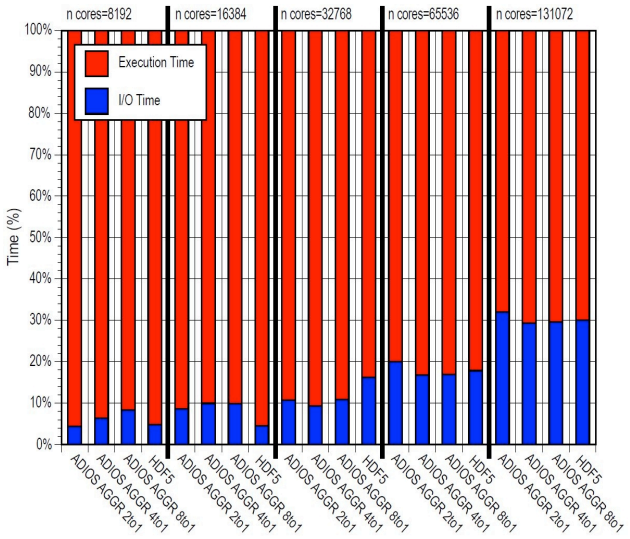

Figure 1. I/O overhead for the 4x4x2 graphite system

6. ACKNOWLEDGMENTS

Thanks to ORNL, SULI, and OLCF for the resources needed to perform this research. Many thanks to Dr. Jeongnim Kim, Dr. Jaron Krogel, and Dr. Scott Klasky for their help in integrating and testing QMCPack with ADIOS.

7. REFERENCES

[1] Kim et al., Hybrid algorithms in quantum Monte Carlo. *Journal of Physics: Conference Series,* 402:1, 012008, 2012.

[2] Lofstead et al., ADIOS: Adaptable, Metadata Rich I/O Methods for Portable High Performance I/O. *PDSW* 2008