Profiling and Predictions of Large-scale QMCPack I/Os on Titan with Skel

Matthew Wezowicz[1], Michael Matheny[1], Stephen Herbein[1], Jeremy Logan[2], Jeongnim Kim[2], Jaron Krogel[2], Scott Klasky[2], and Michela Taufer[1]

[1] University of Delaware

[2] Oak Ridge National Laboratory

Motivation

- As scientific applications move to the exascale:
  - Discrepancy between I/O speed and flops increases
  - Traditional POSIX file I/O becomes inefficient
  - I/O performance becomes more system dependent
- As simulation size increases we cannot save all the simulation data
- QMCPack is an example of this type of application
  - Computation scales to peta-scale but I/O does not scale

Can we predict the I/O performance of QMCPack when simulating energies of molecules on large-scale supercomputer like Titan?

Goals and Methodology

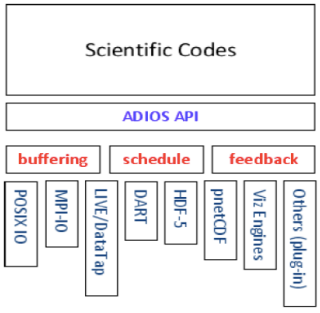
- Extend existing tool for I/O simulation, such as Skel, to integrate IO variability as shown in QMCPack and other real world applications
- Validate the simulation against a real world application
- Show that the results of such simulations can be used to steer design choices through comparisons with real data

QMCPack

- Each simulation searches for solutions to Schrodinger's equation for a relevant particle system
- Use loosely coupled algorithm
- Each process runs *n* independent trial solutions called walkers

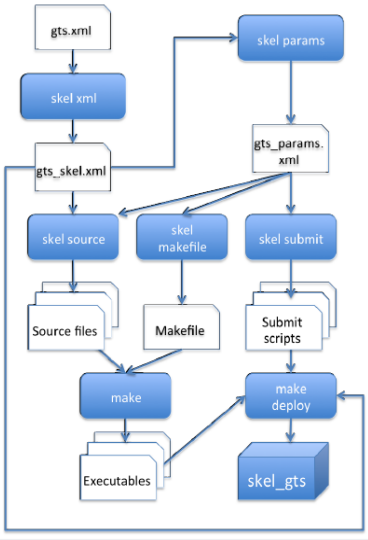
ADIOS

- Include suite of simple, easy-to-use APIs and metadata stored in external XML file
- Allow flexible selection of most effective I/O method
- Provide both high performance, reliability, and usability



Skel

- Tool for testing I/O performance of applications
- Generate and test I/O skeletal applications:
  - Take as input I/O descriptor and set of parameters
  - Generate full I/O benchmark
- Performance measurements from benchmarks correlate with performance of real applications



Skel's Limits

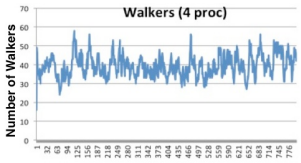
- Fixed size of data per "time step" not realistic in many real applications
- No hotspot or straggler simulation possible to reproduce

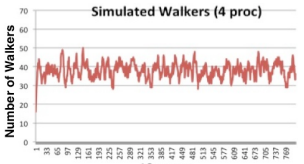
Our Skel's Extension

- Enable variable I/O per process at each Skel iteration
  - Inline definition of function:

```
< var function="static int x=5; walker_num = x++;" />
< var function_file="func.c" includes="stdlib.h" />
```

- Simulate walker numbers with spring
  - Approximate spring equation using Midpoint method
  - Dampen to prevent divergence from center
  - Randomly increment/decrement position





Test Setup

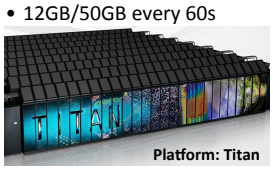
Molecular system:

Graphite 4x4x2:

- 128 carbons and 512 electrons
- 8328 x 8Bytes per walker

Writing frequency:

- 12GB/50GB every 60s



Platform: Titan

QMCPack runs:

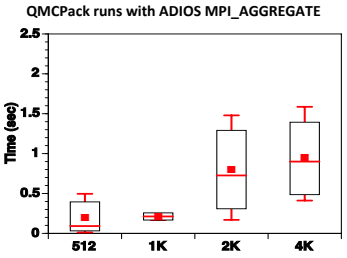
- 512, 1K, 2K, 4K nodes
- 2 MPI process per node
- ADIOS methods used:
  - POSIX, MPI_AGGREGATE

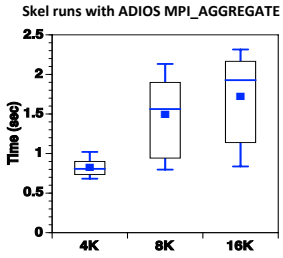
Skel runs:

- 128, 256, 512, 1K, 2K, 8K, 16K nodes
- 2 MPI processes per node
- ADIOS methods used:
  - POSIX, MPI_AGGREGATE

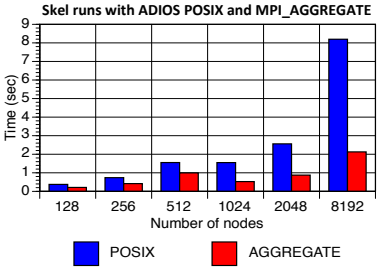
Tests and Predictions

- I/O of graphite's traces:
  - Energies of each ion and electron for each walker in each process





- Performance of different I/O methods on Titan follows expected trend of increasing divergence between POSIX and MPI_AGGREGATE
  - Due to increased strain on Lustre metadata server



ADIOS's method AGGREGATE exhibits a potential excellent scalability when the number of nodes reaches larger scales

Conclusion and Future Work

- New Skel integrates model of variable IO
  - Enable prediction I/O of QMCPack on large platforms
- Our results show potential of the ADIOS MPI_AGGREGATE method to support low-cost I/O on exascale platforms
- Future work includes more ways to specify variable I/O of applications