Scalability Testing for a Data Intensive Molecular Dynamics Application (AWE-WQ) using Work Queue

Lydia Brothers, University of Kentucky

Abstract

Simulating the conformational dynamics of protein molecules is essential for developing probabilistic links between a protein's shape and its function. Computationally, molecular dynamics is a challenging field due to the large time scales needed to make direct observations of kinetic transitions. Due to researchers' desire to more efficiently simulate complex molecules, improvements were made to accommodate the use of approximately 10,000 cores in AWE-WQ using HTCondor for simulations in parallel. To visualize the bottlenecks of the system, a suite of graphical tools were developed for AWE-WQ, and the functionality to specify and monitor resources was added. While incrementally scaling up the application, performance metrics such as speed, efficiency, and utilization were evaluated and maintained by monitoring the distribution of task (simulation) execution times, master/worker availability, as well as network, CPU, disk, and memory usage.

Figure 1: Work Queue Architecture

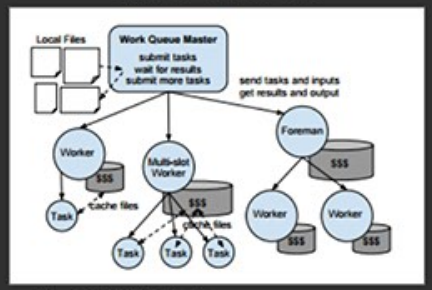
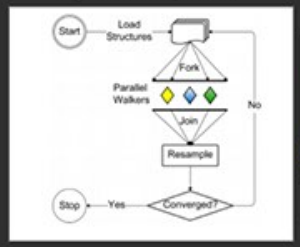


Figure 2: AWE-WQ Flowchart



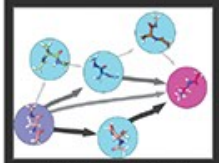
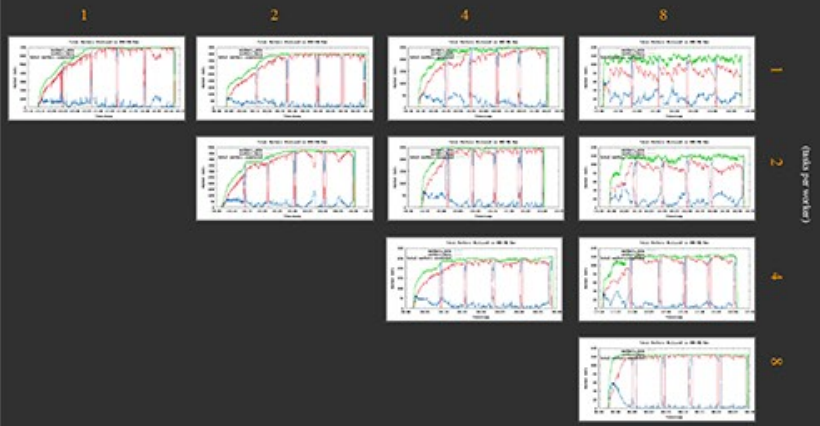


Figure 3: Directed graph of metastable protein states based on kinetic transition energy barriers

Methodology

Figure 3: Worker Availability for different task/worker size combinations in AWE-WQ when the molecule alanine dipeptide ran for 5 iterations with 10,000 tasks allocated to 1000 workers



Evaluating the effectiveness of running AWE-WQ using different task size combinations on multi-slot workers was important to this study because although larger tasks/workers complete faster, they are more sparsely available and difficult to allocate at larger scale using HTCondor.

Table 1: Run Time for 1000 Workers (Corresponding to Figure 3) in minutes

	1 core/worker	2 cores/worker	4 cores/worker	8 cores/worker
1 task/worker	58	55	52	74
2 tasks/worker		44	37	54
4 tasks/worker			38	40
8 tasks/worker				25

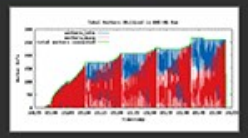
Table 2: Max GB Sent for 1000 Workers (Corresponding to Figure 3)

	1 core/worker	2 cores/worker	4 cores/worker	8 cores/worker
1 task/worker	19	11	8.5	14
2 tasks/worker		13	7.5	8.5
4 tasks/worker			7.5	4.5
8 tasks/worker				3.25

AWE-WQ was incrementally scaled using the benchmarks of 100, 1000, 4000, and 8000 cores. Modifications to AWE-WQ were made to generate these visualizations by allowing the user to specify the number of cores per worker, tasks per worker, task disk usage, and task memory usage at the top level.

Results

Figure 4: Worker Availability for "4000 one core workers" using cctools 5.4.13. As of this updated version, Work Queue masters check for returning results, submit new tasks, and then connect new workers in that order. Using HTCondor, the desired number of workers are not always allocated due to machine availability.



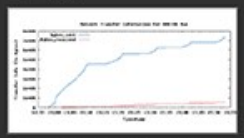


Figure 5: Network Transfer (given in bytes). The bytes sent to each worker primarily consist of cache information and binaries used to run AWE. Bytes received refer to results returned.

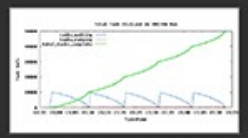


Figure 6: Tasks Utilized. This shows the total number of tasks (or simulations) that are run over 5 iterations of AWE-WQ, as well as the behavior of tasks running and waiting.

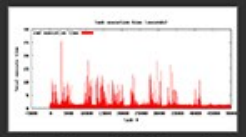


Figure 7: Sequential Task Execution Times

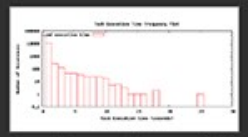


Figure 8: Frequency Histogram of Task Execution Times

Conclusions and Future Work

Thorough testing when scaling up an application is necessary to ensure its reliability and performance is maintained. Our data and visualizations indicate that the application meets performance standards using several thousand cores. Workers that frequently disconnect significantly increase network transfer due to the resending of cache and instance data, and were therefore minimized via task/worker size. More work is needed to be done to stabilize high resource runs of AWE-WQ, and deploying the application in Google Cloud's Compute Engine should clarify additional bottlenecks. Additional work with this project will include a refining of the measure of capacity (number of cores the master can consistently support), and more testing using Work Queue's fastabort. Other useful analyses include comparing cost and performance of using heterogeneous environments simultaneously (commercial cloud in conjunction with a campus shared cluster, etc).

Data Intensive Scientific Computing REU
Mentor: Dr. Douglas Thain
Lydia Brothers: lydia.brothers@gmail.com

References:
Albrecht, 2013, DOI: 10.1109/CLUSTER.2013.6702628
Badi et al., 2014, DOI: 10.1021/ci500321g