Supplying Cycles for High Performance Computing with a Focus on Chemistry: Community Needs to be Considered

David A. Dixon, Department of Chemistry, The University of Alabama, Tuscaloosa, AL 35487-0336
email: dadixon@ua.edu

There are a number of needs in the chemistry community driving the access to high performance computing. These include reliable computational methods involving accurate electronic structure methods at the correlated molecular orbital (MO) theory level for as large a system as possible (scales minimally with N^7 for large basis sets), density functional theory (DFT) for medium-accuracy results as well as for *ab initio* molecular dynamics (MD) (Car-Parinello) (scales as N^3-N^4 depending on the functional for a single molecule, no dynamics), solid state calculations with DFT or correlated MO theory, semi-empirical molecular orbital theroy scaling as N^2-N^3 depending on the matrix diagonalization approach for electronic structure of 5 to 10K atoms including MD potenitallys, and MD with classical force fields or highly parameterized atom-atom potentials from electronic structure methods (1 to 10+ million atoms for as long a time as possible, 10s to 100s of nanosec. In addition, there is now the need for large scale computations for interpreting experiment and experimental data analysis and for large scale searching of databases with automatic addition to data bases.

A key issue in any decision about computational resources is the balance between the desktop, a research group cluster, a University or state center, and a large national facility. This involves not only the compute cycles and their cost, but also the staff to run the computers and to make sure that the proper codes can run efficiently. In my experience, we do little computing at the desktop level simply because it is difficult to have enough resources on the desktop to do anything but the smallest computations as the jobs take a long time to complete, running jobs in the background impacts desktop performance, and it is difficult to keep all of the desktops up o the same level for any application code. This is especially true when training undergraduates in computational chemistry. It is much simpler to manage the computational resources, ease of job submission, and the data resulting from the calculations if one has a central computational facility where the code versions and resources are known ahead of time and a central data storage systems, the latter to meet NSF and other government requirements for federally funded research. Thus the desktop is best used as a workstation. This is especially true for managing data, especially with undergraduate who do not stay in a research group many times for more than one or two semesters. For example, we developed a fully redundant data storage system in our laboratory with 8 processors and 12+ TBytes of storage. This system has redundant disks to deal with disk failure and is further backed up to a separate system weekly. The raw computational data files are maintained in ASCII format in a flat file format. Intermediate data files such as Gaussian checkpoint files are kept as long as they are needed in machine form and converted to ASCII format as needed. A robust metadata database for the massive amounts of computational chemistry raw data has been designed and implemented to manage the data. It performs data synchronization and simultaneously extracts the metadata. This system goes out daily and checks all sites where a user has an account and returns the raw data files to the data system and then generates a metadatabase automatically.

Local group and university machines are predominantly used for capacity computing as many smaller jobs can be run. Many universities do not provide centralized omputing facilities even for mandatory storage of research dat. An issue with university and sate systems (and even national systems) system is not that they may not be useful for large scale chemical computing due to queuing policies, which tend to favor smaller, short jobs as opposed to the large long jobs needed for geometry optimizations, high level correlated MO calculations, etc. It is useful to be able to have input into the queuing system to make sure that jobs are scheduled fairly and that the machines are

available to run large jobs. As an example, we have input into the policies of our State system but not for the university system which makes the state system a more valuable resource.

There is also a need for capability computing, which will be most likely provided by a national user facility. As an example, recent studies of the catalytic reactions of alcohols on transition metal oxide clusters, showed that coupled cluster CCSD(T) calculations were needed just to get the initial Lewis acid-base interactions correct for an initial system with the correct number of reactants (not measured) in order to explain the experimental observations. DFT failed in this study and access to very large systems with large memory per node and a significant number of nodes for up to 48 hours was required to do the calculations. Without these calculations the potential energy surfaces would have been unreliable and we could not explain the experimental results correctly.

On the basis of the combined experience of many computational chemists, we have substantial knowledge on the scaling of many of these algorithms, how they perform computationally, and the computer requirements for a simulation. For example, most electronic structure code algorithms are built on a cache-blocked architecture and therefore, cache latency, bandwidth, and size are important to performance and scalability. Also key to performance is the ability to interleave computation, communication, and I/O operations (e.g. use of asynchronous I/O, use of non-blocking communication operations). Because many of the calculations involve large matrix operations, communication bandwidth and latency is very important. In addition, large local memories and access to large amounts of fast storage for temporary storage of intermediate results both benefit the calculations. A key issue is our ability to deal with distributed data structures on a fully distributed memory system. Such requirements of course need to be provided to any system acquisition team.

Thus, there are a number of needs for any type of computing infrastructure. A generic computing infrastructure may not be all that useful for many scientific domains, for example using commercial cloud resources. Users and code developers will have to have input into various policies, from what science will be produced to how the results can be generated in terms of the code. One will have to have input into the type of computer that one will need to have access to. This is a critical issue as a commercial system such as available from Google cloud which can provide excess cycles to users may not be appropriate for much scientific computing. It may be excellent for work such as Google does, for example, in terms of large database searches and data analytics but may not have the local memory and disk scratch space needed for electronic structure calculations. In addition, it may not be possible to have large, long jobs that can be run over days. Furthermore, it will be very interesting to see how the software is managed and how commercial community codes such as Gaussian will be managed and paid for. In addition, one will have to be concerned about how the codes and versions are maintained. Will the appropriate compilers be available?

The model that currently exists at the DOE and NSF supported facilities is the development of ‘Greenbooks’ by the community which provide the best estimate of what a specific research community needs in terms of computational resources for both hardware and software. There is a growing realization that not all scientific communities need the same type of computing resource in terms of memory, latency, processor count, disk for scratch space, etc. Any plan for supplying compute cycles to the simulation community must account for this. In addition, it is critical not to add compute cycle costs to grants without revising the overhead structure for these costs. Furthermore, it should be noted that many groups take advantage of unused cycles on long weekend, etc. to get large amounts of work done.