A Sustainable Software Innovation Institute for Computational Chemistry and Materials Modeling ((SICM)2)

Working Group on Portable Parallel Infrastructure

The (SICM)2 aims to provide leadership, resources, software, expertise, technologies, and assistance for our community as it navigates the next decade of rapid change in computer technology that impacts everything spanning from laptops in the classroom, through college clusters, to extreme scale supercomputers.
How should we write programs for and prepare students to use the computers anticipated in 2020, and how should our community reorganize to best address these challenges? Some relevant technology trends that we must consider are:

* Massive concurrency within a single socket evidenced by multi-core processors (e.g., currently 128 x86 thread units or more on single chip) with increasingly long vector units (e.g., 512 bytes on Intel MIC), and many-core light weight processors (e.g., currently over 10k+ on a chip).

* Massive numbers of sockets in the largest supercomputers implying that bleeding-edge science must coordinate over 10^9 threads possibly with applications needing for the first time to tolerate faults in the system.

* Complex memory hierarchies including limited coherence, partitioned address spaces, and software managed memories.

* Memory and communication bandwidth and capacity (not FLOP/s) being the primary constraints on performance, cost, and energy consumption.

Current programming models in computational chemistry and materials science are mostly designed for machines of the past and have not kept pace with the underlying technology, and more recent models such as CUDA are suitable only for a small subset of applications with appropriate massive, uniform parallelism. Powerful new concepts for asynchronous distributed computation, such as found in the DARPA HPCS languages (Chapel, X10, Fortress), are poorly understood by the campus chemistry community and, moreover, are only part of the required solutions.

Computational chemistry and material science have well-established histories of adopting and developing advanced software and algorithmic techniques, forced by the complexity of the theories and associated algorithms and software, as well as by our ambitions to perform definitive simulations as an equal partner with experiment in scientific discovery. These needs have forged strong connections with the computer science and applied-math research communities, and the ability to generate high-performance, massively parallel codes from high-level specifications is perhaps the crucial capability for handling the complexity of efficient and robust simulation at scale. We are in a good position to expand these abilities to a steadily increasing fraction of our science including reduced-scaling and fast algorithms and the many-body methods of materials science, to facilitate the composition of multi-physics applications, and to exploit advanced autotuning and code-generation/transformation.