A High-Productivity Language for Computational Science

Josh Milthorpe

Productivity

Challenges for high-performance computing

The standard approach in high-performance computing is currently to use C++ or FORTRAN for standard sequential programming, combined with MPI for parallelism. Problems with this standard approach:

- Readability
- Correctness
- Fragmented view of computation
- Lower productivity

The DARP A High Productivity Computing Systems Program [1] aims to increase the value and reduce the cost (in terms of time to market, software development, maintenance or compute time) of high-performance computing. One theme of the program is the creation of new programming languages for emerging parallel architectures. Three languages have been created: Fortress, Chapel, and X10.

X10

A new language for high-productivity computing

X10 [2] is a managed object-oriented language, based on a small subset of Java with the addition of explicit localisation through places and dynamic, asynchronous activities for concurrency. It supports a variety of common parallel programming idioms. X10 provides a rich array sublanguage. An array has a region (the set of index points) and a distribution specifying the location (in memory) of each point in the region. Regions and distributions provide the foundation for parallel constructs such as attach which executes a statement over all the points in a raw array.

Research questions:

- What aspects of the X10 language specification, compiler and runtimes are critical for performance?
- How does the performance of X10 compare with the standard approach (FORTRAN / C++ and MPI) for bioinformatic applications?
- How can runtime optimization be used to enhance performance?
- How can language features of X10 support good performance across different architectures e.g. CPU vs. GPU?
- How do Particle Mesh Ewald and Fast Multipole methods compare for typical biochemical problems?

Supervised by Alistair Rendell

in collaboration with researchers at IBM T.J. Watson Research Center

Electrostatic interactions

The time required to exactly calculate all interactions is proportional to the square of the number of particles. However, approximate methods exist with better scaling properties. Particle Mesh Ewald methods approximate long-range potentials over a mesh of grid points in the simulation box. A 3D Fast Fourier Transform is used to solve Poisson's equation for the potential at each grid point. Particle potentials are interpolated between grid points.

Fast Multiple methods use multipole approximations for interactions between widely separated groups of particles. Although in theory these methods scale better than PME, practical implementations have not yet been competitive for biochemical problems.

Computational Biochemistry

Modeling the building blocks of life

Aim: to develop molecular dynamics codes in X10 using Particle-Mesh Ewald and Fast Multipole methods, and compare performance for typical problems, such as: simulation of the dynamic behaviour of ion channels.

Molecular dynamics is the simulation of dynamic systems of molecules and ions in terms of forces between their constituent atoms. Fundamental to this method is the force field, which defines simplified interactions between pairs of atoms in a fixed loading structure. The force field is used to simulate dynamic behaviour by integrating the forces over time. This approach is used in computational biochemistry to explore processes like protein folding and docking.

Key factors in molecular dynamics:

- Force field representation
- Time integration
- Evaluation of long-range interactions

Electrostatic interactions

The time required to exactly calculate all interactions is proportional to the square of the number of particles. However, approximate methods exist with better scaling properties. Particle Mesh Ewald methods approximate long-range potentials over a mesh of grid points in the simulation box. A 3D Fast Fourier Transform is used to solve Poisson's equation for the potential at each grid point. Particle potentials are interpolated between grid points.

Fast Multipole methods use multipole approximations for interactions between widely separated groups of particles. Although in theory these methods scale better than PME, practical implementations have not yet been competitive for biochemical problems.