My Background

I am a third year graduate student in the Department of Mechanical Engineering. My research is focussed on the biomechanics of trabecular bone, which is the primary load-bearing biological structure in the human spine as well as at the end of long bones such as the femur. It has a highly porous structure and becomes structurally weak with age-related disease such as osteoporosis. My research involves the study of failure mechanisms in trabecular bone using finite element models obtained from 3D micro-computed tomography images of cadaver bone. Through this course, I would like to have a basic understanding of the techniques used in writing a parallel code to better appreciate the power of parallel computing in solving computationally intensive problems.

A parallel computing application: Bone Biomechanics

Finite element method has been widely used for solving large complex problems in the field of mechanical, civil and aeronautical engineering. The method involves solving a partial differential equation (PDE) describing a physical phenomena over a domain of interest, by discretizing the domain and converting the PDE into a system of algebraic equations with finite number of unknowns. The system of equations can be written of the form $[K]\{u\} = \{f\}$, where $[K]$ is the stiffness matrix of size $n \times n$, $\{f\}$ is the force vector and $\{u\}$ is the vector of $n$ unknowns. Often, in real life problems, the number of unknowns are large.

In the field of bone mechanics, finite element method has been used to study the mechanics of trabecular bone. A discretized domain of a trabecular bone structure can contain over a billion unknowns. Such a problem is computationally intensive and thus finds a good use of parallel computing to solve the set of equations $[K]\{u\} = \{f\}$.

In a recent study, a parallel finite element framework *Olympus* was used to perform a non-linear finite element analysis of a trabecular bone structure from the human vertebral body [1]. This study was performed on the supercomputer ASCI White IBM SP at Lawrence Livermore National Laboratory. Olympus is composed of a parallelizing finite element code Athena, which uses a parallel graph partitioner (ParMetis) to construct a sub-problem on each processor. Each sub-problem is solved using the serial finite element code FEAP and is mapped to the global Olympus operator by a parallel finite element object pFEAP. The problem is solved using the parallel algebraic multigrid package Prometheus, which is built on numerical kernels PETSc. Explicit message passing (MPI) is used for performance and portability of the algorithm. Clusters of symmetric multi-processors (SMPs) were the target platforms for this project. Faster communication within an SMP is implicitly exploited by first having Athena partition the problem onto the SMP compute nodes, and then recursively calling Athena to construct the processor subdomain problem on each node.

Scalability studies were performed by solving the same problem with six different finite element meshes with number of unknowns varying from 7.4 million in the coarse mesh to 537 million in the fine mesh. The ASCI White IBM SP was ranked fourth on the top 500 list at the time these simulations were run (ISC2003). The machine reached a theoretical peak flop rate of 24 Gflop/sec per node, each node had sixteen 375MHz Power 3 processors with 16GB memory per node. The number of compute nodes were selected to place $\sim 131$K unknowns per processor. It was shown that
the linear solver was scaling with the number of iterations for the first linear solve in each Newton loop a constant. But the number of Newton iterations required to solve the nonlinear problem kept increasing as the mesh was refined. The average flop rate per processor was scaling reasonably well, with a total parallel flop rate efficiency of about 90%. The figure below [2] shows the compressive stress distribution from a mid-sagittal cutaway of a vertebra, solved using *Olympus*.

![Compressive Stress Distribution](image)

**References**
