CS 194 Checkpoint

Nathan Beckmann

Expected Progress

- Run ODE on unmodified MINT (1 week)
- Simulator very close to completion (2 months)

Accomplishments

- I worked on getting ODE running in the simulator (SESC)
  - Convert ODE to use SESC synchronization API's
  - Finished majority of calls; handed off thread pool model to Tom to convert because he was familiar with code
- Familiarity with SESC
  - Application API's for threading and synchronization
  - How API calls are handled internally

Accomplishments (contd.)

- Paul worked on revising SESC to fit our architecture (Parallax)
  - Heterogeneous cores worked out-of-the-box
  - Timing is contained in a separate library that has been separated from SESC
  - During down-time, I tested thread affinity on cores

Surprises

- Originally thought getting ODE running on simulator would be trivial
  - SESC API's are *very* small (~10 functions) and do not provide equivalent functionality to pthread
- Also thought getting simulator would take most of the work
  - Expected to use multiple instances of SESC to model heterogeneity, but it already worked
  - Then we thought that all our problems were solved, but realized that mesh latency wasn't sufficiently modeled

Future Steps

- Get ODE running in simulator
  - Tom had code for several weeks; was working on another part of the project. Now I have it again and I will convert the thread pool model.
  - Everything else is complete: Makefile, configuration files, etc.
- Connect libnet to SESC to model mesh latency
  - Latency is already modeled in SESC for cache misses
  - Incorporate libnet to retrieve actual mesh latency