Languages for programming parallel machines

- The range of programming models for parallel machines is wider than that of today’s uniprocessors. This range will widen even further.
  - Domain-Specific Languages.
    - Very High-Level (Simulink?) – Implicit parallelism
    - High-Level (Plain MATLAB, Mathematica)
  - General Purpose High-Level Languages
    - Explicit Parallelism
      - Message-passing
      - Shared-Memory
      - Vector
      - … ?
    - Implicit Parallelism – If we ever figure out how to do it.
  - Assembly Language

Two of the things we don't understand well

- Parallel programming constructs for symbolic computing.
  - Symbolic computing will continue to dominate.
  - Would the same constructs we use in OpenMP and MPI for numerical computations suffice ?
- Performance portability
  - Also difficult to achieve in the sequential domain:
    - ATLAS/FFTW
    - Manual unrolling of loops in some popular high-performance applications.
    - Assembly language programming.
  - But more difficult in the parallel domain.
  - Language features may help, but the impact of languages on effective compiling is even less well understood.

A conjecture

- Parallel programming is and will forever be harder than sequential programming. Things are much worse now and may become better in the future but parallel programming will never be as simple as sequential programming.
  - Developers are going to avoid parallel programming as long as they can.