Programming Languages

Keshav Pingali
Cornell University
We work with roughly 20 research associates and graduate students.
PL Research Themes

• Language design
• Reasoning about program behavior
• Checking your reasoning about program behavior
• Exploiting knowledge of program behavior
  – Find bugs
  – Optimize performance
PL Research Projects

• **Language design**
  – Polyglot: extensible Java front-end (Myers)
  – Jx: type-safe nested inheritance (Myers)

• **Reasoning about program behavior**
  – Kleene algebra with tests (Kozen)
  – Data structure analysis (Rugina)

• **Checking your reasoning about program behavior**
  – NuPRL proof assistant (Constable)

• **Exploiting knowledge of program behavior**
  – Find bugs
    • Safety properties of program resources (Rugina)
    • Certifying compiler for device drivers (Kozen)
  – Optimize performance
    • Self-optimizing systems (Pingali)
    • Mobile programs in computational grids (Pingali, Rugina)
    • Static memory management (Rugina)
Mobile Computational Grid Programs

- Programs run for many hours on large machines with hundreds of processors.
- Mobile programs can adapt to changing resource availability by migrating to new sites on grid.
- New site may have different number and type of processors.
- Goal: programs → mobile programs in a semi-automatic way.
Programs → Mobile Programs

• **Solution: program transformation**
  - Insert code for saving/restoring application state (done by C³ compiler)
  - Use type information to reconstruct state at remote site
  - Applications become “self-checkpointing” and “self-restarting”
  - Application-level checkpointing (ALC)
    (cf. system-level checkpointing)
  - Efficient
    • 5% overhead or less for sequential codes
    • About 10% for MPI codes (homogenous platforms)
Ongoing Work

- **Program analysis to reduce the amount of saved state**
  - Joint work with Radu Rugina
- **Heterogenous platforms**
  - Different architectures
  - Different number of processors
- **Self-optimization**
- **Other applications:**
  - Speculative computation
  - Backward differentiation