CS 612: Software Design for High Performance Architectures

Instructor: Dr. Paul Stodghill (stodghil@cs.cornell.edu) 496 Rhodes Hall

Time: Tuesdays & Thursdays, 10:10 - 11:25 AM, Hollister 401

URL: `http://www.cs.cornell.edu/Courses/cs612/2001SP/`

Prerequisites: An undergraduate compiler course (like CS412) is helpful but not required. Programming experience is essential.

Course content: Sophisticated program analysis and transformation tools are needed in many areas such as high-performance computing, software engineering, and secure computing. For example, modern high-performance computers are complex networks of symmetric multiprocessor nodes in which each processor is pipelined or even multithreaded as in the recently-announced IBM Blue Gene machine, so sophisticated compilers are needed to deliver performance to applications. The Y2K bug highlighted the need for better automatic tools for software engineering and maintenance.

The objective of this course is to understand what is known about program analysis and automatic restructuring, study a number of problem domains to understand what problems can be solved with existing technology, and determine what technology needs to be developed. Some of this technology was developed at Cornell, and is in use in production compilers at SGI, IBM, Intel and other companies. An indirect objective of the course is to gain an appreciation for the use of algebras (such as lattice, polyhedral and relational algebras) to provide clean program abstractions that permit the design of modular software tools.

Students will implement programming projects on high-performance work-stations and the AC^3 cluster in the Theory Center. In addition, there will be three or four homework assignments, some of which may require programming. Topics covered in the course include the following.

1. High-performance architectures
   - Shared/distributed memory machines
   - Memory hierarchies and latency avoidance
   - Multithreading and latency tolerance
2. Problem domains
   - Computational science
   - Databases
   - Other problem domains of interest to course participants
3. Program analysis and transformations
   - Scalar analysis: lattice algebra, control dependence, classical and sparse data flow analysis, Roman Chariots problem, inter-procedural dataflow analysis
   - Array analysis: polyhedral algebra, Diophantine equations, array dependence analysis, distance/direction abstractions

- Pointer analysis: alias relations and their representations, computing inter-procedural alias relations
- Fractal symbolic analysis
- Transformation theories for perfectly-nested and imperfectly-nested loops, based on matrices and Farkas' lemma
- Empirical optimization: Atlas and FFTW systems