CPEG 421/621
- Spring 2008

Compiler Design:
The Software and Hardware Tradeoffs
# Admin. Information

<table>
<thead>
<tr>
<th>Instructor:</th>
<th>Prof. Guang R. Gao</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office:</td>
<td>312 DuPont Hall</td>
</tr>
<tr>
<td>Phone:</td>
<td>831-8218</td>
</tr>
<tr>
<td>email:</td>
<td><a href="mailto:ggao@eecis.udel.edu">ggao@eecis.udel.edu</a></td>
</tr>
<tr>
<td>Co-instructor:</td>
<td>Dr. Haiping Wu</td>
</tr>
<tr>
<td>Office:</td>
<td>306 DuPont Hall</td>
</tr>
<tr>
<td>Phone:</td>
<td>831-0332</td>
</tr>
<tr>
<td>email:</td>
<td><a href="mailto:hwu@capsl.udel.edu">hwu@capsl.udel.edu</a></td>
</tr>
<tr>
<td>TA:</td>
<td>Thomas St John</td>
</tr>
<tr>
<td>email:</td>
<td><a href="mailto:stjohn@capsl.udel.edu">stjohn@capsl.udel.edu</a></td>
</tr>
</tbody>
</table>

**webpage:** [http://www.capsl.udel.edu/courses/cpeg421/2008/](http://www.capsl.udel.edu/courses/cpeg421/2008/)
Important Dates

May 14 (Mon.) : Tentative: quiz
May 16 (Wed.) : project report due

Course work will carry the following weights towards your final grade:

Homework, Lab and class Participation: 25%
Quiz: 45%
Project: 30%
References

1. A set of papers - to be assigned

2. Books:


Other references: see course page
Other References

3. Journals

<table>
<thead>
<tr>
<th>Journal</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE</td>
<td>Computer Transactions on Computers</td>
</tr>
<tr>
<td></td>
<td>Concurrency</td>
</tr>
<tr>
<td></td>
<td>Transactions on Parallel and Distributed Systems</td>
</tr>
<tr>
<td>ACM</td>
<td>TOPLAS - Transactions on Programming Languages and Systems</td>
</tr>
<tr>
<td></td>
<td>Transaction on Computer Systems</td>
</tr>
<tr>
<td>JPDC</td>
<td>Journal of Parallel and Distributed computing</td>
</tr>
<tr>
<td>JSC</td>
<td>Journal of Supercomputing</td>
</tr>
<tr>
<td>JPP</td>
<td>International Journal of Parallel Programming</td>
</tr>
<tr>
<td>PC</td>
<td>Parallel Computing (North-Holland)</td>
</tr>
<tr>
<td>JPL</td>
<td>J. of Programming Languages</td>
</tr>
</tbody>
</table>
## Other Reference

4. **Conference Proceedings**

<table>
<thead>
<tr>
<th>Conference</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLDI</td>
<td>ACM Symposium on Programming Language Design and Implementation</td>
</tr>
<tr>
<td>POPL</td>
<td>ACM Symposium on Principles of Programming Languages</td>
</tr>
<tr>
<td>PPOPP</td>
<td>ACM Symposium on Principles and Practice of Parallel Programming</td>
</tr>
<tr>
<td>ICPP</td>
<td>International Conference on Parallel Processing</td>
</tr>
<tr>
<td>ICS</td>
<td>International Conference on Supercomputing</td>
</tr>
<tr>
<td>LCPC</td>
<td>Intern. WS. on Languages and Compilers for Parallel Computing</td>
</tr>
<tr>
<td>PACT</td>
<td>Parallel Architectures and Compilation Techniques (since 1994)</td>
</tr>
<tr>
<td>IPDPS</td>
<td>International Parallel and Distributed Processing Symposium</td>
</tr>
<tr>
<td>EUROPAR</td>
<td>European Parallel Processing Conferences</td>
</tr>
<tr>
<td>MICRO</td>
<td>ACM/IEEE Symposium on Microarchitectures</td>
</tr>
<tr>
<td>ISCA</td>
<td>ACM/IEEE International Symposium on Computer Architecture</td>
</tr>
<tr>
<td>ASPLOS</td>
<td>ACM Symposium on Architecture Support for Program Languages and Operating Systems</td>
</tr>
</tbody>
</table>
Major Topics

- Topic 1: An Overview on Compiler Design
- Topic 2: Compiler Front-End and IR
- Topic 3: Run-time Environment
- Topic 4: Middle-End and Optimizations
- Topic 5: Middle-End Optimization: Loop Nest Optimizations
- Topic 6: Back-End Optimization: Code Generation
- Topic 7: Advanced Optimization
- Topic 8: Compiler and Architecture Co-Design
Foundations for Compiler Design

1. Processor architecture design flow
2. Compiler structure and design flow
3. Code generation design flow
Why Study Compilers?

• Influences on programming language design
• Influences on computer design
• Compiling techniques are useful for software development
  – Parsing techniques are often used
  – Learn practical data structures and algorithms
  – Basis for many tools such as text formatters, structure editors, silicon compilers, design verification tools,…

Writing a compiler requires an understanding of almost all important CS subfields
Architecture Models

- Architecture features/models
  - Simultaneous multithreading
  - Vector units. SIMD
  - Instructional Level Parallelism (ILP)
    - superscalar
    - VLIW
  - Chip multiprocessing (CMP, multi-core, etc.)

What is the impact of these ideas on compilers?
Processor Architecture Design Flow Diagram

Arch./Compiler and System Software Design Toolset
- Instruction Set Architecture Design (Microarchitecture Design-I)
- Compiler Design
  - Code Optimizer
  - Code Generator
- Toolchain
  - Intel VTune™
  - IBM Performance Evaluator
- Debugger

ISA Simulator

System Level Design

System Level Simulator

System-Level Design

RTL Level Design
- RTL Level Design (Microarchitecture Design II)
- Switch Level Design
- Circuit Level Design

RTL Level Simulator

Switch Level Simulator

Circuit Level Simulator

HDL (VHDL or Verilog)

Hardware Design
- Switch Level Design
- Circuit Level design

2008/2/21
A Good Compiler Infrastructure Needed – A modern View

- Front end
  - Interprocedural Analysis and Optimization
  - Loop Nest Optimization and Parallelization
  - Global (Scalar) Optimization

- Middle-End

- Backend
  - Code Generation

Good IR
Middle-End Optimization

- Flow Analysis
  - Control flow analysis
  - Dataflow analysis
- Global scalar optimization
- Loop nest optimization
- Advanced topics:
  - Static Single Assignment form (SSA)
  - Application of SSA to scalar optimization
Backend Optimization (I)

• Instruction selection
• Instruction scheduling
• Register allocation
• Others
Backend Optimization (II)

- Loop optimization and scheduling
- Software pipelining