Program Optimization for Multi-core Architectures - Web course

COURSE OUTLINE

This is an advanced undergraduate/postgraduate course on compiler optimization for multi-core architectures. The course will cover the following broad topics.

- What are multi-core architectures
- Issues involved in writing code for the multi-core architectures
- How to develop software for these architectures
- What are the program optimization techniques
- How to build some of these techniques in compilers
- OpenMP and parallel programming libraries

Contents:

Introduction to parallel computers: Instruction Level Parallelism (ILP) vs. Thread Level Parallelism (TLP); performance issues: brief introduction to cache hierarchy and communication latency.

Shared memory multiprocessors: general architecture and the problem of cache coherence; synchronization primitives: atomic primitives; locks: TTS, tickets, array; barriers: central and tree; performance implications in shared memory programs.

Chip multiprocessors: why CMP (Moore's law, wire delay) ; shared L2 vs. tiled CMP; core complexity; power/performance; snoopy coherence: invalidate vs. update, MSI, MESI, MOESI, MOSI;memory consistency models: SC; chip multiprocessor case studies: Intel Montecito and dual core Pentium 4, IBM power4, Sun Niagara.

Introduction to program optimization: overview of parallelism, shared memory programming; introduction to OpenMP; data flow analysis, pointer analysis, alias analysis, data dependence analysis, solving data dependence equations (integer linear programming problem); loop optimizations; memory hierarchy issues in code optimization.

Operating system issues for multiprocessing: need for pre-emptive OS, scheduling techniques: usual OS scheduling techniques, threads, distributed scheduler, multiprocessor scheduling , gang scheduling; communication between processes, message boxes, shared memory; sharing issues and synchronization, sharing memory and other structures.

Sharing I/O devices, distributed semaphores, monitors spin locks, implementation techniques for multi-cores; case studies from applications: digital signal processing, image processing, speech processing.

COURSE DETAIL

S.No	Topics
1	Introduction to multi-core architectures

NP-TEL

NPTEL

http://nptel.iitm.ac.in

Computer Science and Engineering

Pre-requisites:

Undergraduate computer organization and architecture, undergraduate compilers, undergraduate operating systems.

Coordinators:

Dr. Mainak Chaudhuri
Department of Computer Science and EngineeringIIT Kanpur

Prof. Rajat Moona
Department of Computer Science and EngineeringIIT Kanpur

Prof. Sanjeev K Aggarwal
Department of Computer Science and EngineeringIIT Kanpur

2	Virtual memory and caches
3	Introduction to parallel programming
4	Cache coherence and memory consistency models
5	Hardware support for synchronization
6	Case studies of chip-multiprocessor
7	Introduction to program optimization
8	Control-flow analysis
9	Data-flow analysis
10	Compilers for high-performance architectures
11	Data dependence analysis
12	Loop optimizations
13	CPU scheduling
14	OS support for synchronization
15	Multi processor scheduling
16	Security issues
17	A tutorial on OpenMP
18	A tutorial on Intel threading tools

References:

1. J. L. Hennessy and D. A. Patterson. Computer Architecture: A Quantitative Approach. Morgan Kaufmann publishers.

2. D. E. Culler, J. P. Singh, with A. Gupta. Parallel Computer Architecture: A Hardware/Software Approach. Morgan Kaufmann publishers.

3. Steven S. Muchnick. Advanced Compiler Design and Implementation.

Morgan Kaufmann publishers.

4. Wolfe. Optimizing Supercompilers for Supercomputers. Addison-Wesley publishers.
5. Allen and Kennedy. Optimizing Compilers for Modern Architectures. Morgan Kaufmann publishers.
6. A. S. Tanenbaum. Distributed Operating Systems. Prentice Hall.
7. Coulouris, Dollimore, and Kindberg. Distributed Systems Concept and Design. Addison- Wesley publishers.
8. Silberschatz, Galvin, and Gagne. Operating Systems Principles. Addison-Wesley publishers.