



Program

Thursday, March 27

8:30

Registration

9:00

Welcome & Introduction

Mark K. Smith, Gelato Central Operations

Why Many-Core - The BIG Picture

9:15

Many-Cores in the Future

Robert Schreiber, HP

The change from single core to multi-core processors is expected to continue, taking us to many-core chips (64 processors) and beyond. Cores are more numerous, but not faster. They also may be less reliable. Chip-level parallelism raises important questions about architecture, software, algorithms, and applications. A chip-edge bandwidth crisis is looming, but new technologies may help us cope. I'll consider the directions in which the architecture may be headed, and look at the impact on parallel programming and scientific computing.

10:00

Morning Break

10:15

Taking Multi-Core and Parallelism Seriously: The Intel Perspective

Xinmin Tian, Intel

The processor architecture and micro-architecture are undergoing a vigorous shaking-up. The major chip

manufacturers have shifted their focus to “multi-core” processors with a “right turn” from GHz competition. The new focus is multiple cores with shared caches, providing increased concurrency instead of increased clock speed. As a challenge, software engineers can no longer rely on increasing clock speed to hide software bloat. Instead, they must learn to make effective use of increasing parallelism. This adaptation has never been easy. This talk consists of two parts -- the first part will focus on the parallel programming tools such as compilers, performance analysis tools and correctness checking tools that are applicable for developing mainstream parallel applications. We also share some of the challenges that developers face today in developing applications for homogeneous multi-core systems and we will discuss the situation with the advent of heterogeneous many-core systems in the next few years. The second part will cover progress on Transactional Memory technology research and development. We will discuss open transactional memory research problems, as there is a growing community of researchers and industry software and hardware vendors working on both software and hardware support for the TM approach.

11:15

Challenges for HPC Future

Richard Kaufmann, HP

This talk will describe some of the trends in the base technologies used in HPC, and how these trends will affect HPC developers and users. The two biggest developments are multi-core processors (where Moore's Law has become Moore's Cores!) and accelerators (the use of GPUs, FPGAs and other specialized ASICs for general purpose computing). After listening to the talk, perhaps the audience will be better armed to ask probing questions of their computer suppliers and application developers.

11:45

Open Discussion

This will be an open discussion led by Robert Schreiber.

12:15

Lunch (Zhengda Restaurant)

Compiling Code for Many-Core

13:45

Software Engineering for Multi-Core Systems - An Experience Report

Christoph Schaefer, University of Karlsruhe

The emergence of inexpensive parallel computers powered by multi-core chips combined with stagnating clock rates raises new challenges for software engineering. As future performance improvements will not come for free from increased clock rates, performance critical applications will need to be parallelized. However, little is known about the engineering principles for parallel general-purpose applications.

This talk presents an experience report based on four diverse case studies with multi-core software development for general-purpose applications. Our empirical findings include:

- 1) Auto-tuning is indispensable, as manually tuning thread assignment, number of pipeline stages, size of data partitions and other parameters is difficult and error prone.
- 2) Architectures that encompass several parallel components are poorly understood. Tuneable architectural patterns with parallelism at several levels need to be discovered.

Representing all case studies, I will focus on the parallelization process of a large commercial application containing multi-level parallelism and how our prototype of an auto-tuning framework is used to find the best configuration of the application's parallel sections.

14:15

May Happen in Parallel Analysis

Yao Shi, Tsinghua University

Concurrent program analysis is an urgent and useful topic for programmers targeting multi-core processors. A fundamental technique of concurrent program analysis is May-Happen-in-Parallel (MHP) analysis that determines whether any two statements may be executed in parallel. This reduces the false positive rate and makes concurrent program analysis more efficient.

However, current research on MHP is weak, which can only process at most ten KLOC with some restrictions (Object-Oriented, OpenMP model, etc.). We propose a framework in the Open64 Compiler to solve general MHP problems for C/C++ programs.

14:45

Scalable Concurrency in Many-Core Processors

Li Zhang, University of Amsterdam

The many-core technology exhibits tremendous computational capability and parallelism on a single chip. Meanwhile how to harness the power of parallelism has become a key issue in the field.

This talk will present the SVP (Self-Adaptive-Network-Entity Virtual Processor) programming model with explicit parallelism exploitation. It tackles the issue of extracting and utilizing the massive concurrency in hardware cost-effectively. Imposing the SVP model, uTC (an extension to the C language) is defined as a concurrency-oriented parallel language. An architectural solution, the micro-threaded architecture, based on the SVP model will also be introduced. It resembles the dataflow computational model and is capable of explicit context switch, register level data synchronization and dynamic concurrency management. The proposed Chip Multi-Processor as microgrid is aiming to be scalable across a large number of on-chip processing cores in terms of both power and performance.

15:15

Dynamic Optimization - An Open Discussion

Wei Chung Hsu, University of Minnesota

This will be an open discussion led by Wei Chung Hsu and focused on Dynamic Optimizations.

15:45

Afternoon Break

16:15

Communication Analysis and Optimized Mapping of Explicit Parallel Codes

Lei Shang, Institute of Computing Technology, CAS

Mapping logical computing units onto physical computing units is one of the basic problems in parallel computing, especially for hierarchy architecture or topology-sensitive systems, like SMP clusters, multi-core SMP and many-core systems. The optimized mapping is relevant to both hardware characteristics and application communication patterns. In our work, we are going to build a general framework for optimizing the mapping with synthesis of application analysis and hardware architecture. The communication analysis techniques of explicit parallel codes, adapting abstract methods and heuristic methods for graph partitioning are used in our framework.

A toolbox approach is in the works, and it will be convenient for any explicit parallel code to get the optimized mapping and improve performance cheaply.

16:45

HP Compiler Lab - The Many-Core Perspective

Shin-Ming Liu, HP

17:30

Open Discussion

This will be an open discussion led by Shin-Ming Liu.

18:00

Depart American Studies Center for Lansheng Hotel

18:30

Dinner at the China Rose Hall, 3rd Floor, Shanghai Lansheng Hotel

20:00

End of Day

Friday, March 28

8:30

Registration

Many-Core Application Development

9:00

GPU Computing Research at UIUC

Wen-Mei W. Hwu, University of Illinois at Urbana-Champaign

In the next decade, we are going to see continued performance scaling in single-chip, massively parallel compute engines. According to the semiconductor industry road map, these chips could provide up to 10,000x speedup over our current microprocessors by the end of the year 2016. Such a dramatic increase in computation power will likely enable revolutionary work in science, engineering and many other disciplines. Like any other massively parallel computer system, in order to achieve high performance, an application programmer currently has to understand the desirable parallel programming idioms, potential performance pitfalls, and proven coding strategies for the platform. However, the programming and code optimization models of GPU computing design are quite different from those of traditional CPUs. In this presentation, I will describe the vision and recent results of a collaborative effort between the University of Illinois and NVIDIA on building an infrastructure of programming tools, educational materials (www.courses.ece.uiuc.edu/ece498/al), application development experience, and architectural directions needed for application developers to fully exploit the hardware compute power of current and future GPU computing platforms.

9:45

Massively Parallel GPU Computing with NVIDIA's CUDA

David Kirk, Nvidia

In the past, graphics processors were special purpose hardwired application accelerators, suitable only for conventional rasterization-style graphics applications. Modern GPUs are now fully programmable, massively parallel floating point processors. This talk will describe NVIDIA's massively multi-threaded computing architecture and CUDA software for GPU computing. The architecture is a scalable, highly parallel architecture that delivers high throughput for data-intensive processing. Although not truly general-purpose processors, GPUs can now be used for a wide variety of compute-intensive applications beyond graphics.

10:45

Morning Break

11:00

Intel Threading Building Block (TBB)

Colt Gan, Intel

11:30

Model-Driven Development Tool for Parallel Applications

James Gan, IBM

Parallel programming is extremely difficult. Programmers must be very careful to avoid popular defects like deadlock and data race. Our tool can provide a much easier style of programming. First, it won't require explicit concurrency. Instead, the developer creates a sequential computing kernel. After that, he/she can create a model for the parallel application being developed then the model can be transformed to a parallel application. The model-driven development tool can bring the following benefits:

1. Progressive disclosure information to developer

The software engineer can easily develop the parallel program before he or she becomes an expert of parallel programming

2. Concurrent pattern can handle classical scenarios quickly

If the user case fits into one of map/reduce, master/worker, pipeline, fork/join, it can be easily done.

3. Task-oriented API

When creating special task flow, the developer only needs to specify dependencies between tasks. Tasks will be automatically scheduled to multiple cores with consideration of dependency.

12:00

Lunch (Danyuan Restaurant)

13:30

The Parallel Framework for Realizing the Power of Multi-Core Processors

Yurong Chen, Intel

This talk will discuss the methodology in analyzing the scalability bottlenecks, and demonstrate how to improve

the performance on the future chip multi-processor (CMP) systems. With the prevalence of CMP and the number of cores increasing steadily for the foreseeable future, one key issue is how to effectively manage and execute more and more threads on CMP at the same time. I will introduce the parallel framework, which uses an iterative parallel performance tuning method on the multi-core processor. Some emerging video processing applications are used to show how we can parallelize them to enable real-time performance on the multi-core processor.

I will also examine all aspects of parallel performance tuning techniques in this talk, and show how to use the analysis tools to improve the scalability performance.

14:00

Open Discussion

This will be an open discussion led by Wen-Mei W. Hwu.

New Paradigms - Thinking Parallel

14:30

Parallel Processing Models and Research at CERN

Sverre Jarp, European Organization for Nuclear Research

With its current parallel processing paradigm (High Throughput Computing) CERN has been able to embrace multicore systems since Day 1. Today, to prepare for the start-up of the Large Hadron Collider, we have a large installation of Intel-64 Woodcrest/Clovertown systems as well as an IA-64 Montecito cluster. However, the parallel processing paradigm requires additional memory per process, and leads to other complications, such as inefficient scheduling. This talk will first explain the issues with the current multi-core processing model which is unlikely to scale to many-core environments (with hundreds of cores). Next I will describe several experimental programming models, based on multi-threading, that are being tried out in order to improve the situation. I will also briefly describe the tools we have deployed, such as performance monitors and threading tools. Finally I will highlight our ongoing educational effort that we think is mandatory in order to get the programming community to "think parallel".

15:00

General Purpose Programming of Many-Core Devices and Many-Core Systems

Steven Ericsson-Zenith, Institute for Advanced Science & Engineering

This talk will highlight general purpose programming of many-core devices and many-core systems. I will discuss the process oriented programming model that is the basis of my work and also provide some historical anecdotes from my experience with Occam and the Transputer.

I will also discuss the Carnap programming language and the open source project that is implementing a compiler for that language for many-core platforms.

I will also speak in my capacity as Chief Scientist for Manycore Corporation and their architect of intellectual property for devices to support the Process Oriented Programming model.

15:30

Afternoon Break

16:00

Dynamic Helper Thread Generation

Wei Chung Hsu, University of Minnesota

A multi-core CPU (or chip-level multiprocessor, CMP) combines two or more independent cores into a single chip. Most processor vendors are offering multi-core/many-core chips today, and more such CPUs will be coming out in the near future. At present, such multi-core/many-core CPUs are mainly used to improve throughput or highly parallel application performance rather than single thread performance. Since multiple processor cores on the same chip may share the level-2 on-chip cache, one or more helper threads can be spawned and executed speculatively ahead of the main thread to prefetch data into the shared cache. This can significantly reduce the cache miss penalty of the main thread, which is often the major performance bottleneck for modern applications.

This talk presents the design and implementation of a runtime optimization system that can automatically generate and spawn helper threads to speed up single threaded applications. The performance results are measured and collected on an UltraSparc IV+ dual-core CPU system.

16:30

High Performance Data Mining

Judy Qiu, Indiana University

The ever increasing number of cores per chip will be accompanied by a pervasive data deluge whose size will probably increase even faster than CPU core count over the next few years. This suggests the importance of parallel data analysis and data mining applications with good multi-core, cluster and grid performance. This talk considers data clustering, mixture models and dimensional reduction presenting a unified framework applicable to bioinformatics, cheminformatics and demographics. Deterministic annealing is used to lessen the effect of local minima. We present performance results on 4 and 8-core systems identifying effects from cache, runtime fluctuations, synchronization and memory bandwidth.

17:00

Parallel Garbage Collection

Xiao-Feng Li, Intel

Garbage collection (GC) is one of the key components in modern programming systems, such as Java, C#, JavaScript, Ruby, etc. Its performance impacts the overall software scalability on multi-core platforms. While the major efforts in software parallelization are focused on multi-core programming, threading, and compilation, we investigate GC parallelization technology systematically. We classify the topics into the following categories: traversal of object connection graph, live object marking, object copying order, heap compaction, large object management, and concurrent collection. Each category has its own characteristics and worth separate study. In this talk, we will describe and compare the parallelization techniques in each category in a systematic approach, and also discuss their interactions with underlying platforms.

17:30

Open Discussion

This will be an open discussion led by Sverre Jarp.

18:00

End of Day

Speakers

Yurong Chen Intel

Yurong Chen is a researcher at the Intel Microprocessor Technology Lab, Beijing (Intel China Research Center). Currently he conducts research on parallel processing of emerging applications, scalable workloads, and benchmarking and performance analysis for next-generation microprocessors/platforms. He joined Intel in 2004. Before that he did two years postdoctoral research on large-scale scientific computing in the Institute of Software, Chinese Academy of Sciences. He received his Ph.D. degree from Tsinghua University in 2002. His e-mail is yurong.chen at intel.com.

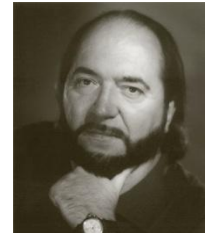


The Parallel Framework for Realizing the Power of Multi-Core Processors

Friday, March 28 • 13:30

Steven Ericsson-Zenith Institute for Advanced Science & Engineering

Steven Ericsson-Zenith is a Scholar at the Institute for Advanced Science & Engineering. He obtained his doctorate in Computer Science from the University of Pierre and Marie Curie in Paris in 1992. His dissertation discusses his work at INMOS on the Transputer microprocessor and the parallel programming language Occam as a part of David May's Computer Architecture team and his subsequent work at Yale University with David Gelernter's Linda Group.



He first introduced the Process Oriented Programming model in his work at Yale University. This model consists of parallel programming data types that allow programmers to express distributed data structures and the processes that act upon them. The model addresses well known problems in the general purpose programming of parallel computers, provides ease of programming with powerful logically shared data structures while maintaining the underlying rigor of a CSP formal basis.

The Process Orient Programming model is being implemented for many-core processors in the language Carnap (<http://carnap.info>).

General Purpose Programming of Many-Core Devices and Many-Core Systems

Friday, March 28 • 15:00

Colt Gan
Intel

Colt Gan joined Intel in 1995. Worked as senior software engineer, senior consultant and lead instructor in various Intel teams. Mainly focused on new technology enabling for China market. Colt now leads a technical consultant engineer team to support Intel software tools in APAC.



Intel Threading Building Block (TBB)
Friday, March 28 • 11:00

James Gan
IBM

Zhi Gan, a software engineer at IBM China Emerging Technology Institute. He joined IBM in 2005 after receiving a Ph.D degree in computer security from Shanghai JiaoTong University. Mr. Gan has extensive experience in SOA, Parallel Programming Models, and Eclipse. His current focus is parallel programming models for multi-core, model-driven development with patterns, and next generation tooling. Zhi can be contacted at ganzhi@cn.ibm.com



Model-Driven Development Tool for Parallel Applications
Friday, March 28 • 11:30

Wei Chung Hsu
University of Minnesota

Wei Chung Hsu received a PhD in Computer Science from the University of Wisconsin, Madison, in 1987. Currently, he is a full professor in the Department of Computer Science and Engineering at the University of Minnesota. From 1997 to 1999, he was a Runtime Optimization Architect in the California Language Lab at HP. From 1993 to 1997, he was a Technical Lead in the compiler team responsible for developing and releasing an optimizing compiler for the HP PA-8000 systems. Prior to joining HP, he was a Senior Software Engineer and an Architect at Cray Research, in Chippewa Falls, Wisconsin. His current research interests include high-performance computing systems and architectures, in particular, runtime optimization systems, optimizing compilers, and cache-related optimization techniques.



Dynamic Optimization - An Open Discussion
Thursday, March 27 • 15:15

Dynamic Helper Thread Generation
Friday, March 28 • 16:00

Wen-mei W. Hwu
University of Illinois at Urbana-Champaign

Wen-mei W. Hwu holds the Sanders-AMD Endowed Chair in the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign. His research interests are in the areas of architecture, implementation, and software for high-performance computer systems. He is the director of the IMPACT research group (www.crhc.uiuc.edu/Impact). For his contributions in research and teaching, he received the Eta Kappa Nu Holmes MacDonalld Outstanding Teaching Award, the ACM SigArch Maurice Wilkes Award, the ACM Grace Murray Hopper Award, the Tau Beta Pi Daniel C. Drucker Eminent Faculty Award, and the ISCA Most Influential Paper Award. He is a fellow of IEEE and ACM. Hwu serves on the Executive Committee of the MARCO/DARPA C2S2 (www.c2s2.org) and GSRC (www.gigascale.org) Focus Research Centers. He leads the GSRC Concurrent Systems Theme with Kurt Keutzer. He also serves on the Gelato Steering Committee. Dr. Hwu received his PhD degree in Computer Science from the University of California, Berkeley.



GPU Computing Research at UIUC
Friday, March 28 • 9:00

Sverre Jarp
European Organization for Nuclear Research



Sverre Jarp is the Chief Technology Officer at the CERN openlab, a joint collaboration with leading industrial partners in order to assess cutting-edge information technology for the Large Hadron Collider's Computing Grid from 2007 onwards.

He has been working in computing at CERN for over 30 years and has held various managerial and technical positions, promoting advanced but cost-effective computing solutions for the Laboratory. In 2001-2002, he spent a sabbatical year at HP Labs (Palo Alto, USA). Inside openlab, his main focus is currently compilers and platform optimization as well as virtualization and grid middleware.

S. Jarp holds a degree in Theoretical Physics from the Norwegian University of Science and Technology in Trondheim.

Parallel Processing Models and Research at CERN
Friday, March 28 • 14:30

David Kirk
Nvidia



David Kirk has been NVIDIA's Chief Scientist since January 1997. His contribution includes leading NVIDIA graphics technology development for today's most popular consumer entertainment platforms. In 2006, Dr. Kirk was elected to the National Academy of Engineering (NAE) for his role in bringing high-performance graphics to personal computers. Election to the NAE is among the highest professional distinctions awarded in engineering. In 2002, Dr. Kirk received the SIGGRAPH Computer Graphics Achievement Award for his role in bringing high-performance computer graphics systems to the mass market. From 1993 to 1996, Dr. Kirk was Chief Scientist, Head of Technology for Crystal Dynamics, a video game manufacturing company. From 1989 to 1991, Dr. Kirk was an engineer for the Apollo Systems Division of Hewlett-Packard Company. Dr. Kirk is the inventor of 50 patents and patent applications relating to graphics design and has published more than 50 articles on graphics technology. Dr. Kirk holds B.S. and M.S. degrees in Mechanical Engineering from the Massachusetts Institute of Technology and M.S. and Ph.D. degrees in Computer Science from the California Institute of Technology.

Massively Parallel GPU Computing with NVIDIA's CUDA
Friday, March 28 • 9:45

Xiao-Feng Li
Intel



Xiao-Feng is a VM architect in the System Software Division of the Intel Software and Solutions Group, where he leads a team developing managed runtime technologies. Prior to his current position, Xiao-Feng was a research manager in the Micro-processor Technology Labs of the Intel Corporate Technology Group. Before he joined Intel, Xiao-Feng worked at the Nokia Research Center on mobile and network technologies. Xiao-Feng is active in the open source community, frequently training and giving lectures. Xiao-Feng's current interests are in programming language design and implementations, focusing on runtime technologies and their interactions with underlying platforms.

Parallel Garbage Collection
Friday, March 28 • 17:00

Shin-Ming Liu
HP



Shin-Ming Liu is the Section Manager for the Optimization Technology Section of the Java, Compiler, and Tools Lab at HP in Cupertino, California. Liu led the development team for all levels of optimization in Java and compilers targeted for the Itanium processor. He has led the reinvention of compiler development methodology by focusing on modulization, memory footprint control, canonical internal representation, and automatic error detection. Before joining HP, he worked at MIPS/SGI in the area of compiler front end, middle end, back end, and linker. He has co-authored several technical publications and holds several US patents in the field of compiler optimization.

HP Compiler Lab - The Many-Core Perspective
Thursday, March 27 • 16:45

Judy Qiu
Indiana University



Qiu received a Masters degree in Computer Science and Engineering at Beihang University and a Masters and Ph.D in Computer and Information Science from Syracuse University. She is now a Postdoctoral researcher at the Bloomington campus of Indiana University. Her research interests include distributed and parallel computing, grid computing, collaboration systems, and computer graphics. She is involved in the SALSA many-core computing project which is developing a hybrid model of parallel computing that links grids, clusters and multi-core chips. SALSA is building a suite of parallel data mining algorithms for applications including Geographical Information Systems, cheminformatics, bioinformatics, and speech recognition. Her interests here include parallel algorithms, performance evaluation and programming models.

High Performance Data Mining
Friday, March 28 • 16:30

Christoph Schaefer
University of Karlsruhe



Christoph Schaefer received a Masters degree in Computer Science at the University of Karlsruhe, Germany. He is now an assistant researcher at the Institute of Programming Structures and Data Organization at the University of Karlsruhe. He is involved in the Multicore Software Engineering Research Group, which focuses on investigating software engineering concepts, methods, and tools for developing reliable, parallel software. Christoph's research interests include parallel patterns, tunable architectures and auto-tuning.

Software Engineering for Multi-Core Systems - An Experience Report
Thursday, March 27 • 13:45

Robert Schreiber
HP



Rob Schreiber is a Distinguished Technologist in and Assistant Director of the Exascale Computing Lab at Hewlett Packard Laboratories. Dr. Schreiber received a BA in mathematics from Cornell in 1972 and a PhD in Computer Science from Yale in 1977. He is known for research in sequential and parallel algorithms for matrix computation and compiler optimization for parallel languages. He was a professor of Computer Science at Stanford and at RPI and was chief scientist of the Saxpy Computer company. He was a co-developer of the sparse matrix extension of Matlab, and a leading designer of the High Performance Fortran programming language. He was one of the developers of the NAS parallel benchmarks. At HP, Rob helped lead the PICO Project, which developed a system for embedded processor synthesis from high-level specifications. In 2007 he was named as a Distinguished Scientist by the Association for Computing Machinery.

Many-Cores in the Future
Thursday, March 27 • 9:15

Lei Shang
Institute of Computing Technology,
CAS



Lei Shang received a Masters in Computational Mathematics from Northwestern Polytechnical University. He is an assistant researcher at the Advanced Compiling Technology Lab in the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include parallel algorithms, parallel languages and optimization techniques. He is now involved in a research group which focuses on parallel programming models for hyper parallel systems. Shang can be reached at shanglei@ict.ac.cn.

Communication Analysis and Optimized Mapping of Explicit Parallel Codes
Thursday, March 27 • 16:15

Yao Shi
Tsinghua University

Yao Shi is a PhD candidate in the Department of Computer Science and Technology, Tsinghua University. He received a Bachelor degree from Tsinghua University in 2004. He is a member of the Open64 Compiler group and his current interest is concurrent program analysis. Shi can be reached at shiyao00@mails.tsinghua.edu.cn.



May Happen in Parallel Analysis
Thursday, March 27 • 14:15

Mark K. Smith
Gelato Central Operations

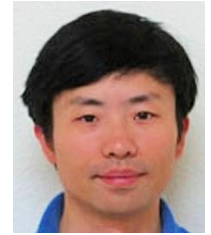
Mark K. Smith is the Managing Director of the Gelato Federation. He works with Federation members and sponsors around the world, fostering collaborative relationships among members, sponsors, and the general community to advance the Linux-Itanium platform. Mark leads a technical team at the University of Illinois and dedicates time to educating the general community about the advantages of the platform. Prior to joining Gelato, he worked in the software industry for 10 years. Mark holds a PhD in Engineering from the University of Illinois.



Welcome & Introduction
Thursday, March 27 • 9:00

Xinmin Tian
Intel

Xinmin Tian holds a PhD in Computer Science. He is a Principal Engineer and Compiler Architect at Intel. Xinmin leads parallelization, vectorization, compiler parallel debugging support and transactional memory research and development projects of Intel C++ and FORTRAN compilers for Intel IA-32, Intel 64, and Itanium multi-core architectures. He has over 30 refereed technical publications on compiler optimizations, parallel computing, and multi-threaded architectures. Xinmin Tian is a coauthor of "The Software Optimization Cookbook" (Second Edition) at Intel Press published in 2006, and a main contributor for the "Multi-Core Programming" book published by Intel Press in 2006. Xinmin Tian has 20 patents pending in the areas of compiler optimizations, parallelization, and multi-core architectures. He also served on program committees for research conferences and has been a referee for technical journals and conferences.



Taking Multi-Core and Parallelism Seriously: The Intel Perspective
Thursday, March 27 • 10:15

Li Zhang
University of Amsterdam

Li Zhang received a Masters degree in Computer Engineering at Delft University of Technology, the Netherlands. Currently, he is a PhD candidate and assistant researcher in the Informatics Institute at the University of Amsterdam. His research interests include computer architecture and implementation, memory architecture, and programming model. He is involved in the Microgrid project at computer system architecture group, which focuses on multi-core programming model, its computer architecture and memory architecture.



Scalable Concurrency in Many-Core Processors
Thursday, March 27 • 14:45