China Course Restructuring Plan of Integrating CS2013/TCPP Topics into Undergraduate Curricula

Hai Jin[1] Feng Lu Zhenzhen Song Zirui Wang
School of Computer Science and Technology, Huazhong University of Science and Technology
{hjin, lufeng}@mail.hust.edu.cn

Introduction

In respect of high performance computing field, Parallel and Distributed Computing (PDC) technologies with CUDA/MPI/OMP on shared memory/computers/clusters/GPU and multi-core architectures now take increasingly important role in the fast development of computing curricula. Hence, it is no longer sufficient for modern computer programmers to solely acquire the traditionally sequential programming skills.

Since 2010, NSF/IEEE-TCPP Curriculum Initiative on PDC suitably proposed the core topics of PDC for CS and CE undergraduate curricula. In February 2012, ACM and IEEE-Computer Society proposed the Computer Science Curricula 2013 (CS2013) Strawman Draft. To coordinate the increasing importance of PDC, CS2013 dedicates a Knowledge Area (KA) to this area. This KA includes materials on programming models, programming pragmatics, algorithms, performance, computer architecture, and distributed systems.

	Core-Tier1 hours	**Core-Tier2 hours**	**Includes electives**
PD/Parallelism Fundamentals	2		N
PD/Parallel Decomposition	1	3	N
PD/Communication and Coordination	1	3	Y
PD/Parallel Algorithms, Analysis, and Programming		3	Y
PD/Parallel Architecture	1	1	Y
PD/Parallel Performance			Y
PD/Distributed Systems			Y
PD/Formal Models and Semantics			Y

Based on the reports of CS2013, CS2008 and CC2001, the following table shows the comparisons among the number of core hours of Tier1 and Tier2 in respects of Operating System and PD KAs. From this table, it is worth to note that the tier structure of OS KA is explicitly similar to the PD KA. In addition, CS2013 instructs that readers can also be guided by the NSF/TCPP Curriculum Initiative on PDC.

Knowledge Area	**CS2013**		**CS2008**	**CC2001**
	Tier1	**Tier2**	**Core**	**Core**
OS-Operating Systems	4	11	18	18
PD-Parallel and Distributed Computing	5	10	--	--

In China, both the quantities and practical experiences of faculties for teaching PDC courses are insufficient. Hence, it is challenging to restructure undergraduate courses to meet the requirements listed in PDC development and CS2013/TCPP Curriculum Initiative on PDC.

Background

In Huazhong University of Science & Technology (HUST), the School of Computer Science & Technology offers degrees of bachelor, master and doctor in the majors of Computer Science and Technology, Information Security and Software Engineering. Both the majors of Computer Science and Technology and Information Security are awarded as the “National Distinguished

[1]School Dean

Major" in China. The school has over 1,860 declared undergraduate majors and about 1,000 master students.

The HUST faculties with PDC background have taught the following courses to graduate students for more than 10 years:

- 210.503: Parallel Processing, Electives, for master, professional and PhD students, since 2002
- 210.805: Grid and Cloud Computing, Electives, for professional and PhD students, since 2005
- 210.527: Distributed System and Middleware, Electives, for master, professional and PhD students, since 2008
- 210.550: Multi-core Operating System Engineering, Electives, for master, professional and PhD students, since 2008
- 210.520: Parallel Programming, Electives, for master, professional and PhD students, since 2008
- 0820041: Cluster and High Performance Computing, Electives, for Undergraduate students, since 2004

During the past decade, we have accumulated sufficient capabilities in teaching undergraduate and graduate students. In our opinion, the courses relevant to the topics in PDC for undergraduate curricula should be different from the courses for graduate students. In addition, we believe that there are rich connections between the topics and the existing CS courses, such as multi-core and computer architecture, programming constructs and programming, parallel algorithms and data structure and algorithms, etc. Thus, we can integrate the topics in CS2013/TCPP Curriculum into the traditional CS curricula. This will be a good way to disseminate parallelism throughout the CS curriculum.

We are currently funded by the Project of "Parallel Programming Principle and Practice" from the Ministry of Education (MOE) of the People's Republic of China to design a syllabus for senior undergraduate students on parallel and distributed programming. Meanwhile, we serve as the China's first pilot of restructured courses to integrate topics of CS2013/TCPP Curriculum Initiative on PDC requirements into undergraduate curriculum.

Target Course 1: Parallel Programming Principle and Practice

(1) Description

This course aims to provide an introduction to parallel computing, including parallel computer architectures, analytical modeling of parallel programs, principles of parallel algorithm design. In addition, materials on TBB, OpenMP, CUDA, OpenCL, MPI, MapReduce will be contained.

Through the training of this course, the students will be good at using some of popular existing parallel programming tools, and solving several related open research questions.

Architecture Topics	**Bloom #**	**Learning Outcome**
Superscalar	K	Multiple instructions on different data
Pipelines	K	Single vs. Multicycle, multiple instructions, OoO execution
Multicore	C	OpenMP project
Heterogeneous	C	Hybrid OpenMP&MPI
SMP	C	OpenMP project

MIMD	K	Multicore, cluster, etc.
SIMD/Vector	K	Same operations on different data
Streams	K	GPU
Shared vs. distributed memory	K	UMA, NUMA architectures; message passing
Performance metrics	A	How to define, measure different benchmarks
Multi-Threading	K	e.g. Intel's hyper-threading
Message Passing	K	Shared memory and no shared memory

Programming Topics	**Bloom #**	**Learning Outcome**
SIMD	K	Understand common vector operations
Distributed memory	C	Different ways of message passing
Shared memory	A	Protecting shared data and obtaining speed up
Hybrid	K	CPU, GPU, etc
Task/thread spawning	A	producer/consumer synchronize
Data parallel	A	Be able to write a correct data-parallel program and get speedup, should do an exercise
Parallel loop	C	openMP, intel's TBB
Compiler directives/pragmas	C	parallel loop, concurrent section
Deadlocks	C	
Tasks and threads	K	Know the relationship between number of tasks/threads/processes
Performance metrics	C	speedup, efficiency, work, cost, Amdahl's law

Algorithms Topics	**Bloom #**	**Learning Outcome**
Speedup	C	solve same problem faster or larger problem in same time
PRAM	K	An exemplar of simplest parallel model
Task graphs	C	A concrete, algorithmic abstraction
Synchronization	K	Aware of methods of controlling race condition
Sorting	C	Parallel merge sort
Asynchrony	K	As exhibited on a distributed platform
Selection	K	Min/max, can be accomplished by sorting

(2) Evaluation plan

The learning outcomes in the major topic of architecture will be evaluated in written assignments. For the programming topic, one or two programming projects using Intel's Parallel Studio or AMD's OpenCL Studio and associated tools will be provided to the students. Students will test their programs in both single core and multi-core environments. The topics of algorithms will be discussed as part of lectures and may be evaluated in written assignments.

In addition, students are encouraged to participate in the competition about parallel programming such as AMD's Accelerate Contest and nVIDIA's CUDA Programming Contest .The competition results are taken into consideration towards the final score of this course.

(3) Pilot in Fall 2012: Hai Jin

Target Course 2: Parallel Data Structure and Algorithm

(1) Description

This course aims to introduce students to the fundamentals of parallel algorithm and data structure, and the traditional serial algorithm transformed into the scalability of parallel algorithms.

The data structure includes the basic data structures: heap, hash table, red-black tree, the map etc. It also introduced the advanced data structure in the latest parallel programming model, such as Map-Reduce, CUDA multi-thread.

Parallel algorithms include the traditional algorithm (dynamic programming, graph theory

algorithms, greedy algorithms) and its parallel extensions.

Architecture Topics	Bloom #	Learning Outcome
MIMD	K	Identify MIMD instances in practice;
SMT	K	Distinguish SMT from multicore, e.g. CUDA,OpenCL
NUMA(Shared Memory)	N	UMA, NUMA architectures;
message passing	N	message passing (no shared memory).
Performance metrics	A	How to define, measure different benchmarks.

Programming Topics	Bloom #	Learning Outcome
Synchronization	A, C	write shared memory programs, producer- consumer, and know how to speed-up
SPMD notations	C,K	See different examples, know the existence of MPI, CUDA, and some others (such as OpenCL, Global Arrays, BSP library)
Concurrency defects	C, K	Including the notions of deadlock (detection, prevention), race conditions (definition), determinacy/indeterminacy in parallel programs
Data	C	Understand impact of data distribution, layout and locality (what it means, general issues) on performance; know false sharing and its impact on performance (e.g., in a cyclic mapping in a parallel loop); notion that transfer of data has fixed cost plus bit rate (irrespective of transfer from memory or interprocessor)

Algorithms Topics	Bloom #	Learning Outcome
Divide & conquer	**C**	Introduce tree and expose parallelism
Recursion	**C**	unfold yielding tree structures have subtrees that can be computed independently, in parallel
Scan	**N**	Cover a sampler of Blelloch's examples
map-reduce	**N**	Tree structure implicit in scalar product
Graph algorithms	**N**	Graph theory and algorithms
“Out-of-core” algorithms	**N**	
Matrix product	**N**	Canon algorithm

(2) Evaluation plan

The learning outcomes in the major topic of architecture will be evaluated in written assignments and programming test.

For the programming topic, students will be given an C/C++ environment to implements a parallel formulation of a problem (e.g., MergeSort, DFS, graph algorithms such as Dijkstra’s single-source, Floyd’s all-pairs shortest path algorithms) in a programming test which will test their acknowledge of complex data structure and various algorithmic techniques. The algorithms topics and architecture topics will be related to their written assignments and final exam.

(3) Pilot in Spring 2013: Hai Jin

Budget Requirements

Budget Requirements	Cash
Teachers’ training budget	$1000
Teaching materials and equipments budget	$200
Conference attending budget	$1000
Total	**$2200**