Week 10 - Parallel Computing

Hongzheng Chen

Mar 30, 2020

1 Basis of Computer Architecture

About the history and developments of computer, there are many books and papers you can refer to. *The Landscape of Parallel Computing Research: A View from Berkeley* written by the scholars in UC Berkeley in 2006 may be a good literature for you to know why we need parallel computing and what are the challenges of computer architectures in the 21st century.

Other resources about nowadays computer architecture may include the books written by John Hennessy and David Patterson, the 2018 Turing Award winners:

- *Computer Architecture: A Quantitative Approach* (Chinese version)
- *Computer Organization and Design: The Hardware/Software Interface* (Chinese version)

The second book is also the textbook of our *Computer Organization Principle* course, and you can read it ahead of the schedule. Moreover, *Computer Systems: A Programmer's Perspective (CS:APP)* is still highly recommended for you to get through, which covers the basic concepts from applications to computer architecture.

2 Different Hardware

Basically, we focus on CPU, GPU and FPGA. CPU is the hardware you most familiar with — easy-to-program, flexible, and general to different tasks. GPU, also known as graphical cards, is traditionally used for CG tasks such as rendering your PC games. Later, people found that GPU's vectorized computation ability can be extended to more tasks. Then the concept of General Purpose GPU (GPGPU) is proposed, and GPU is leveraged to accelerate deep learning, scientific computing, etc. GPU is much harder to program than CPU, which uses CUDA (Compute Unified Device Architecture) and SPMD (Single Program Multiple Data) model. But the emergence of deep learning frameworks like Tensorflow and PyTorch lowers the programming barrier and enables programmers quickly deploy their deep learning models onto GPU, which is also an important reason for the boom of deep learning.

Regarding to Field-Programmable Gate Array (FPGA), abbreviated as FPGA, it is the one you most unfamiliar with. It is quite different with CPU and GPU that FPGA is non-von-Neumann architecture, and the execution is not based on instructions. FPGA directly synthesizes[1] programs to circuits. When the data flows through FPGA, the execution is done.

About CPU's architecture, you can find it in any computer architecture courses or books, as

[1] You can basically think synthesis is a form of compilation, but they are not the same. See this article for the differences.

illustrated in Section 1. About GPU's, you can also find that in parallel computing courses. I will give some in Section 3. For FPGA, there are several resources you can refer to:

- FPGA 结构、编译与应用: My blog about overview of FPGA
- 《FPGA 原理和结构》: This book is written by a Japanese and is very up-to-date (published in 2019). It is filled with details and must be an excellent book for beginners.
- *Cook FPGA*: A Github repository contains all you need to know about FPGA from its architecture to programming guide.

Moreover, as commonsense, you should know the biggest enterprises in the world producing these hardware. For example, most of the desktop/server CPUs are x86 architecture and made by Intel or AMD[2] For edge devices (smartphones), ARM (RISC architecture) takes most of the market share. For GPU[3], Nvidia has much greater advantages than AMD and takes control of most of the GPGPU field. For FPGA, Xilinx and Intel[4] are the two leaders, where more than half of the market are under the control of Xilinx.

3 Parallel Computing

This seminar covers most of the programming models, and you can find them in the following courses:

- CMU CS15-418: *Parallel Computer Architecture and Programming*
- UCB CS267: *Applications of Parallel Computers*

The former course may be more CS-principle-related covering the topics I mention including ILP, SIMD, CUDA, OpenMP, MPI, etc. The latter course is more application-oriented and covers common scientific computing topics including matrix multiplication, graph processing, fast fourier transform (FFT), computational biology, etc.

Otherwise, you can refer to my notes/blogs on parallel computing:

- 并行计算
- 并行编程-C/C++ 多线程
- 并行编程-OpenMP
- 并行编程-Cilk
- 并行编程-AVX 指令集

[2]The 2019 data shows Intel takes 80% of the market of desktop CPUs while AMD takes 20%; and the data for server CPUs is 93% to 7%.

[3]Here I mean isolated graphical cards. Otherwise, most of Intel CPU now has initial integrated GPU, and Intel must be the greatest winner.

[4]At the very beginning, Intel did not make FPGAs. But at 2015, Intel bought the FPGA design company Altera and became the second leader in FPGA market, see this news.

- 并行编程-MPI
- 并行编程-MapReduce
- 并行编程-Spark

But these blogs are relatively concise, and you can find more materials in the references after the blogs.