Statement:

After I left Cornell in the spring of 2012, I lived in Shanghai for less than a month and enjoyed Shanghai Library. While I was exploring more on philosophy using the library resources, I got a phone call from a friend of mine, who worked in a study abroad service agency in Zhengzhou. He wanted me to work in the agency and I accepted. I left the job after two months since I found it is not that technology-involved and I could not learn much from it.

In the following years, I lived in several cities in China (Beijing, Tongling, Shanghai) and continued to learn using available resources (books, MOOCs, websites). At the time, I thought I could land a job in IT industry after acquiring sufficient knowledge and skills and I spent quite an amount of time learning computer science.

I benefited a lot from learning computer organization by reading Stallings [11], who is also one of my favorites computer science textbook author. Meanwhile, I learned how to program in C by reading Kernighan and Ritchie [5]. I became interested in how computer works on a fundamental level and spent a lot time with Mano and Kime [7], which covers the fundamentals of digital circuits. It was such a pleasant experience to see how combinatorial and sequential logic works. I wanted to know more circuit theory and explored more on it. The more I learned, The more I wanted to dig deeper into the fundamental level. I believe I cannot understand how computers work without knowing the underlying physics (esp. electromagnetism). However, physics is a hard subject to explore on my own and I encountered numerous difficulties without instructors.

I have been a fan of Bach since long and was deeply moved by his sonatas and partitas for solo violin (BWV 1001-1006). This motivated me to learn the violin. In the end of 2013, I took a course in violin. I played the guitar and thought I could transfer the knowledge of guitar to violin playing. Even though I have an instructor this time, violin is the hardest musical instrument I have encountered. Despite my unsuccessful attempt with violin, I really enjoyed the learning experience.

Over the years, I successfully finished 21 MOOCs from various course providers, many of which were accomplished with distinction. Two of them are officially verified by Coursera. One was *Logic: Language and Information 2* [9], the other was *Functional Programming Principles in Scala* [8]. The list of those 21 courses and correspondent statements of accomplishment can be found on *CourseFinder* [12].

I want to mention two MOOCs specifically here. The first is *Hardware/Software Interface* [1] and the other is *Calculus: Single Variable* [4], being the most rewarding MOOCs for computer science and mathematics respectively.

Hardware/Software Interface provided some challenging virtual labs through the Fedora (a Linux distro) virtual machine. The textbook for this course was the classic Bryant and O'Hallaron [2]. Understanding how high-level languages are translated into the basic instructions of assembly language and how software interacts with hardware through those virtual labs was a really wonderful experience.

Calculus: Single Variable, which did not use any textbook, offered a perspective that was very different from most other calculus courses. It was definitely not an oversimplified calculus course

that only teaches you the mechanics of calculus. It was neither an analysis-like calculus course that constructs the real number from an axiomatic approach. It offered the right amount of justification of the subject as a second course in calculus and tried to unify the entire course through Taylor series.

One day in 2014, it was quite crowded as usual in the Shanghai Library. Fortunately, I managed to find a seat. The guy next to me was coding with a fancy IDE on his MacBook Pro. At the moment, I thought he was just another programmer. I was curious about what he was doing and looked at his screen to find out. To my surprise, his code was written in Scala, a then rare programming language in China. I initiated the talk with him by saying that I just completed a related MOOC [8] by the creator of Scala, Martin Odersky and really liked some ideas of the functional programming paradigm (FP). Possibly due to the rarity of FP in China as opposed to object-oriented programming paradigm (OOP), he also replied with surprise that he agreed but was more interested in making real-world application than just theory. He told me that already made an office automation web app with the language for his company and kept improving it. We then talked about some other computer-related stuff. Some time later, he revealed that he owned the company mentioned earlier and asked me whether I was interested in having an internship in his company. I did not accept right away since I wanted to focus on learning new things at the time, but we exchanged our contact info.

However, some months later, I wanted to examine what I had learned and the best way to do it was through internship. I contacted the aforementioned man to see if I was still able to intern at his company and he accepted. I worked there as a programmer for three months and gained some real project experience. This story was covered by *Global Times* [6].

What I benefited the most from this internship is not any specific skills related to IT but the self-discovery that I have an aptitude for academics, specifically mathematics. I am more interested in how things work rather than how to make things only for the immediate profits. If I were to learn how to drive, I would care about not only how to turn the wheel but also what goes under the hood.

Last summer, I taught myself LaTeX and became a frequent user. I almost abandoned Microsoft Word completely since then and used LaTeX whenever I found suitable. In fact, this file was typeset in LuaLaTeX and source code can be found on *GitHub* [13]. I am able to typeset formulae like

$$\begin{aligned}\frac{d}{dx}\int_{t=\sin x}^{\tan x} e^{-t^2}\,dt &= \frac{d}{dx}\left(\int_{t=0}^{\tan x} e^{-t^2}\,dt - \int_{t=0}^{\sin x} e^{-t^2}\,dt\right)\\ &= \frac{d}{dx}\int_{t=0}^{\tan x} e^{-t^2}\,dt - \frac{d}{dx}\int_{t=0}^{\sin x} e^{-t^2}\,dt\\ &= e^{-\tan^2 x}\cdot\sec^2 x - e^{-\sin^2 x}\cdot\cos x.\end{aligned}$$

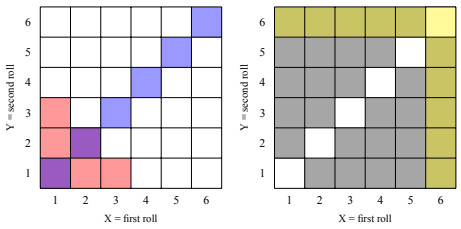

Figure 1: On the left, the blue region shows the doubles and the red region indicates the outcomes whose sum is 4 or less; on the right, the yellow region indicates outcomes with at least one 6 and the gray region shows the outcomes where two rolls differ.

With the PGF/Ti*k*Z package, I could make graphics like the figure on the right. Many standard computer tools (like LaTeX and GNU toolchain) come as free and open source software (FOSS) and I came to know the underlying philosophy [3] of it by this way. That is one of numerous benefits by studying computer science.

That said, I did not find in computer science the same theoretical depth I found in mathematics and physics. Don't get me wrong. I did not mean all computer science and just most computer science curricula are not for theoretical computer scientists.

I did not know that it is permissible and free to audit at a Chinese university until last fall. Hence I audited two courses at Fudan University, being *mathematical analysis I* and *advanced linear algebra I* respectively.

Intended Major and Career Plan:

I intend to double major in mathematics and physics. I still love computer science and there is a lot more to learn in the field. But with the double major, I do not know if I have enough time and effort for it. In addition, there are general requirements to meet.

If possible, I want to skip some lower division courses in mathematics and computer science like MAT 122, MAT 221, CSC 140, CSC 151. Due to my strong interest in mathematics, I will definitely have a wonderful time with mathematics. Since I do not have much background in physics, I will probably have a very hard time with it. But I like challenges; hard courses just turn me on and easy courses are time-wasters.

I want to stay in the academia and become an academic (if not a mathematician). This means I will seek master and doctoral degrees after graduation from Cornell.

Term I wish to return:

I wish to return on the block 1 of the 2016-2017 academic year.

Housing Preference:

I do not have any particular housing preference, except that I want to make the habit of going to bed early.

References

[1] Gaetano Borriello and Luis Ceze, *Hardware/Software Interface*, from the University of Washington, Coursera, URL: https://www.coursera.org/course/hwswinterface (visited on 03/20/2016).

[2] Randal E. Bryant and David R. O'Hallaron, *Computer Systems: A Programmer's Perspective*, 2nd ed., Prentice Hall, Feb. 4, 2010, 1080 pp., ISBN: 978-0-13-610804-7.

[3] Free Software Foundation, *What is free software?*, URL: http://www.gnu.org/philosophy/free-sw.en.html (visited on 03/22/2016).

[4] Robert W. Ghrist, *Calculus: Single Variable*, from the University of Pennsylvania, Coursera.

[5] Brian W. Kernighan and Dennis Ritchie, *The C Programming Language*, 2nd ed., Prentice Hall, 274 pp., ISBN: 0-13-110362-8.

[6] Lin Li, "Back to school online", in: *Global Times* (July 17, 2014), URL: http://www.globaltimes.cn/content/871111.shtml (visited on 03/17/2016).

[7] M. Morris R. Mano and Charles R. Kime, *Logic and Computer Design Fundamentals*, 4th ed., Prentice Hall, June 17, 2007, 696 pp., ISBN: 978-0-13-198926-9.

[8] Martin Odersky, *Functional Programming Principles in Scala*, from École Polytechnique Fédérale de Lausanne, Coursera, June 29, 2014, URL: https://www.coursera.org/account/accomplishments/records/jeAXpfyLDdj7TBYK (visited on 03/17/2016).

[9] Greg Restall and Jen Davoren, *Logic: Language and Information 2*, from the University of Melbourne, Coursera, June 17, 2014, URL: https://www.coursera.org/account/accomplishments/records/AqU3pfW4qRTd8FzE (visited on 03/17/2016).

[10] Walter Rudin, *Principles of Mathematical Analysis*, 3rd ed., McGraw-Hill Education, Jan. 1, 1976, 325 pp., ISBN: 0-07-054235-X.

[11] William Stallings, *Computer Organization and Architecture: Designing for Performance*, 8th ed., Prentice Hall, Apr. 3, 2009, 792 pp., ISBN: 978-0-13-607373-4.

[12] Lei Zhao, *Lei Zhao's Completed Courses*, Sept. 29, 2015, URL: https://coursefinder.io/u/leizhao (visited on 03/17/2016).

[13] Lei Zhao, *Readmission Letter on GitHub*, URL: https://github.com/LeeiFrankJaw/CollegeApplication (visited on 03/21/2016).

[14] Vladimir A. Zorich, *Mathematical Analysis I*, trans. by Roger Cooke, 1st ed., Universitext, Springer-Verlag, Jan. 22, 2004, 574 pp., ISBN: 3-540-40386-8.