Teaching Statement

Mert Hidayetoglu (him/his)

The most important aspect of teaching for me is conveying the insights that are useful to build systems. The best way I have found to do that is to use an algebra to describe computational systems succinctly. For example, we can represent collective communication as a sparse matrix with a sparsity pattern specific to the data movement across GPUs. I show optimization as an algebraic operation, like factorization, which is generalizable to any communication pattern and system architecture. My teaching style connects the dots between special cases in applications and intuitive, generalized principles through mathematical thinking.

Giving lectures and seminars has always been a part of my studies. I was fortunate enough to contribute to my PhD advisor's teaching kit in GPU programming. I prepared lectures that not only have been taught at University of Illinois, but also globally through a MOOC which reached tens of thousands of students. Moreover, I contributed to later editions of "Programming massively parallel processors: A hands-on approach" by Kirk & Hwu, which has become the standard textbook in senior / graduate classes in GPU computing and parallel programming. I traveled every year (2016–2019) to teach at a summer school on GPU programming at the Barcelona Supercomputing Center. I updated the hands-on labs according to the relevant problems each year and gave lectures. I also led tutorials for the dissemination of the frameworks that I have developed over the years. My latest experience was at MLSys 2022 conference, where I curated a tutorial on "Sparsity in ML: Understanding sparsity in neural networks on heterogeneous systems."

I have mentored several undergraduate students in their junior and senior years in CS and CE programs with various research interests related to GPU systems. I involved mentees by carefully guiding them through the learning steps from the classroom to the frontiers of research in computing. I found it very rewarding to contribute to someone's intellectual development. I focused on creative thinking, and finding ways to communicate complex ideas through sketches on a whiteboard, pair programming, and designing example programs. In short, I tried a variety of methods to try to communicate with students in whatever way worked for them.

I would like to teach two graduate-level courses. The first covers fast parallel algorithms for solving large-scale inverse problems with low computational complexity. I would focus on developing numerical analysis and computational thinking by learning how to map physical data structures to supercomputers. The second and related course would involve programming models that make programming supercomputers easier. This course will involve i) standard APIs and language extensions for parallel programming, such as CUDA, MPI, ii) higher-level programming systems such as Legion, Charm++, TACO, and iii) domain-specific languages such as Halide and those I propose for large-scale applications in my own research.

For undergraduate programs, I can teach typical introductory classes in CS, EECS, and ECE curricula. For junior and senior classes, I would especially enjoy teaching parallel programming, vector space optimization, and GPU computing.