Graphical Processing Units (GPUs) are very attractive targets for compute-intensive applications today because they are widely available in computing systems ranging from laptops to supercomputers and offer around an order of magnitude higher peak performance than conventional multicore processors. However, achieving high performance for applications on GPUs is generally non-trivial and several potential performance bottlenecks must be avoided. This course is targeted at those interested in developing high-performance GPU algorithms. The basics of GPU architectural features and GPU programming will first be reviewed before delving into details on performance bottlenecks and approaches to avoiding or mitigating them. A number of GPU algorithms from data analytics and machine learning will be used to illustrate techniques for performance optimization on GPUs.

A) Review of GPU Basics

1. Introduction to GPU and CUDA
    a. Hardware view
    b. Software view
    c. CPU vs GPU

2. Performance monitoring and debugging
    a. Nvprof, nsight
    b. Cuda gdb
    c. Cuda mem check

B) GPU Optimization

3. Minimizing warp divergence
    a. Warps revisited
    b. Warp divergence
        i. Illustration using just one step of prefix sum
        ii. Reduce warp divergence/ alleviate warp divergence overhead
            1. Predicated instructions
            2. Reduce # instructions inside branch dependent code
    c. occupancy
4. Maximizing memory coalescing
    a. Explanation with array add (threads are remapped)
    b. vectorization Parallel prefix sum (step 1)
    c. SOA VS AOS
5. Optimizing shared memory usage
    a. Matrix Multiplication
    b. Bank conflict: Matrix transpose
    c. (Nvprof)

6. Warp-level primitives
    a. Shuffle
    b. Reduce
    c. Vote

C) GPU Algorithms

7. Algorithms
    a. Parallel prefix sum
    b. laplace3d
    c. Kmeans
    d. Convolutions
8. Sparse Algorithms:
    a. Representation
    b. Load balancing : spmv
    c. Tiling : LDA
    d. Tiling + fusion : ccd++
    e. Graph algorithms

D) Other Topics

9. Other high level optimizations
    a. Streams
    b. Async copy
    c. Pinned memory
    d. A small intro to unified memory
    e. Intrinsics
    f. Dynamic parallelism

10. Multi GPU programming
    a. Intro to mpi
    b. Cuda aware mpi
    c. Histogram (dense and sparse)