Computação de Alto Desempenho

2018/2019

Knowledge Requirements

Beginner (up 10 values in 20)

- What is a parallel computer, parallel computing and parallel programming.
- What is high-performance computing.
- Understand the difference between the different types of architectures (Flynn's Taxonomy)
- Understand the basic architectural differences between a CPU and a GPU
- Know what is GPGPU.
- Understand how a GPU connects to the rest of the system.
- Know what a cluster is.
- Know what a bottleneck and a hotspot are.
- Understand the work-span model
- Know what are embarrassing parallel computations
- Know what speedup and efficiency are
- Understand the CUDA programming model:
    - Separate host and device addressing spaces
    - Memory management
    - Grid definition
    - Kernel execution
- Implement simple CUDA kernels
- Know and understand the steps of parallel algorithm design
- Understand the Map, Reduce and Scan patterns
- Implement CUDA kernels that perform a single Map operation

- Understand what Distributed Parallel Programming is (Parallel Programming on Distributed Memory Architectures)
- Understand the Message Passing programming model and its differences against shared memory programming
- Understand the SPMD execution and programming models
- Know what MPI is
- Know what MapReduce is
- Understand the map and reduce stages of the MapReduce programming model
- Know what is a Spark RDD
- Understand what is a Spark action and Spark transformation
- Implement simple Spark programs
- Understand data and task decomposition techniques

Intermediate (up to 16 values in 20)

- Understand why parallel computing is crucial for harnessing the full potential of today's architectures.
- Understand the difference between the Big Data and Big Compute (HPC) fields and their convergence in the Big Analytics field.
- Understand ILP, Vector processing, shared memory
- Understand memory consistency and why it is important

- Understand base components of a GPU's architectures (SM, computation cores, Memory hierarchy)
- Understand Multi-GPU connectivity
- Understand advantages/disadvantages of distributed memory architectures versus shared memory architectures versus hybrid shared memory + accelerated architectures.
- Understand how performance may be measured.
- Understand why parallel applications may not be scalable
- Recognize an embarrassing parallel computation
- Understand what linear and super-linear speedup is
- Understand the difference between speedup and efficiency
- Understand asymptotic complexity
- Be able to calculate the complexity of simple algorithms
- Understand the BSP machine model
- Parallelize a sequential computation into one or more CUDA kernels
- Implement CUDA kernels that make use of shared memory
- Understand CUDA atomic operations
- Understand the CUDA execution model
    - Understand the mapping of CUDA constructs into the GPU hardware
    - Understand the concept of warp
- Understand why divergence in kernels is bad
- Understand CUDA's (the GPU's) synchronization mechanism
- Use CUDA streams to overlap computation with communication
- Understand the Unified Virtual Addressing (UVA) feature
- Understand the challenges of each step of the parallel algorithm design
- Implement CUDA kernels that perform more than one Map operation
- Implement CUDA kernels that apply a Reduction or Scan operation
- Understand the CSR graph storage model.
- Understand the 4 base operations offered by the Gunrock programming model, and how these may be used to implement graph algorithms on the GPU.
- Recognize an unbalanced GPU computation.

- Understand what one-sided communication is
- Know which stages of the distributed parallel execution of a program are taken care by MapReduce (e.g. scheduling)
- Know what the shuffling process in MapReduce is
- Understand what a combiner in MapReduce is and why they are needed
- Understand how limitations of MapReduce regarding iterative computations
- Understand the use of a distributed file system in MapReduce
- Understand the Spark execution model
- Understand how data is partitioned in Spark and how such partitioning may be controlled at program-level
- Understand the impact of data placement in iterative Spark computations that join data sets
- Understand Spark RDD dependencies
- Understand and components of a Spark execution (task, stage, DAG, RDD)
- Understand Spark Datasets
- Understand Task decomposition and interaction graphs
- Understand task to processor mapping
- Understand and recognize Output and Input data decomposition
- Understand exploratory and speculative decomposition techniques
- Understand the impact of task granularity

- Understand and recognize task interactions
- Understand static mapping
- Understand centralized and distributed dynamic mapping

Advanced (up to 20 values in 20)

- Understand cache coherence issues in shared memory architectures, especially in NUMA.
- Understand what differs from sequential to parallel performance.
- Relate the BSP machine model to the CUDA execution model
- Implement CUDA kernels that efficiently access global and/or shared memory
- Understand GPU warp scheduling and how to hide latency of arithmetic and memory operations.
- Avoid divergence in kernel execution
- Understand the tradeoffs of using explicitly management of the GPU's memory versus UVA
- Understand which steps of the parallel algorithm design
- Understand when it is possible to merge Map operations is a single kernel, and when it is not.
- Efficiently implement a CUDA kernel that applies a Reduction or Scan operation
- Understand how Gunrock's filter operations may be implemented on the GPU.
- Implement graph algorithms in CUDA.
- Alter an unbalanced CUDA kernel to become balanced

- Understand how asynchronous message passing communication can be used to overlap communication and computation
- Understand the differences between MPI and MapReduce, and the trade-offs of using one of the systems
- Understand where computation to data affinity in MapReduce
- Know what Resource Managers, like Yarn, are and what is their purpose
- Contrast the MPI, MapReduce and Spark execution models
- Know which components of a Spark execution (task, stage, DAG, RDD) are generated from an application
- Understand Spark performance from component execution
- Understand and recognize Intermediate data decomposition
- Understand and recognize hybrid decompositions
- Understand the Work Stealing load balancing algorithm and its design decisions