Syllabus
Scalable Data Science and Distributed Machine Learning, 6hp

Large clusters composed of a network of commodity computers with new hardware, including GPUs, TPUs, CPUs, and SSDs, are today's workhorses for solving various problems that scale to large datasets using parallel and distributed algorithms. This course is a theoretical and practical introduction to the subject.

Topics include analysis of distributed and parallel algorithms for sorting, optimization, numerical linear algebra, machine learning, graph analysis, streaming algorithms, and other problems that are challenging to scale on a commodity cluster. The course, offered in collaboration with Stanford University, will have a theoretical focus on the mathematical analysis of algorithms, and a practical focus on implementation in a cluster with real data.

Issued by the WASP graduate school management group 2022-08-29

Course Information

Course Type:

- **AI track: mandatory**
- **AS track: elective**
- **Joint curriculum: elective**

Course level

PhD student course

Course offered for

PhD Students in the WASP graduate school

Time: Given even years, Autumn

Teacher: Raazesh Sainudiin (Uppsala University)

Examiner: Marina Axelson-Fisk (Chalmers)

Entry requirements

The participants are assumed to have a background in mathematics corresponding to the contents of the WASP-course "Mathematics and Machine Learning", including programming experience in at least one high-level language such as Python.

Intended learning outcomes

After completing the course the student should be able to:

- describe, explain and analyze the computational complexity of common sequential, parallel and distributed algorithms including sorting, optimizing, performing linear algebraic & graph operations using appropriate abstract machine models
- describe the time, space and communication complexity of the core distributed algorithms for sorting, joining, optimizing, performing linear algebraic & graph operations in Apache Spark
- design, analyze and implement distributed algorithms for data-parallel or model-parallel optimization of ML models from linear regression to DNNs
- analyze the computational complexity and the mathematical assumptions of the distributed algorithms underpinning results of scientific publications in ML/AI
- be able to use Apache Spark core, its libraries for SQL, GraphX and ML, and other tools in its ecosystem to solve practical ML/AI problems starting from raw datasets using a cluster

Course content

The course is given in three modules.

Module 1 – Introduction to data science & analysis of parallel algorithms.

Introduction to theoretical fundamentals of parallel algorithms and runtime analysis on a parallel random access machine model will be complemented by an introduction to Apache Spark.

Theoretical topics: sequential & parallel random access machine models, work-depth models, Brent's theorem, scheduling, sum, all-prefix sum, sorting, matrix multiplication, minimum spanning trees, iterative solution of linear systems, unconstrained and constrained optimization, including gradient descent, stochastic gradient descent & hogwild,

Practical topics: introduction to the data science process, Apache Spark and Scala.

Module 2 – Introduction to distributed algorithms for data science & machine learning.

Introduction to the analysis of distributed algorithms running on a cluster of machines will be complemented by implementations in Apache Spark's ecosystem.

Theoretical topics: distributed work-depth models, communications complexity analysis, distributed summation, sorting, joining & optimisation, map-reduce model, page rank, and distributed linear algebra.

Practical topics: using distributed algorithms in core, SQL and ML libraries of Apache Spark.

Module 3 – Diving deeper

Practical pathways for students to become familiar with scalable data science processes and distributed machine learning pipelines for typical decision problems, including estimation, prediction and testing in various domains will be provided to help inspire student group

projects. The last module is an opportunity to dive deeper with a small group of 2 to 4 peers into a problem, domain and/or method of interest to the group.

Teaching and working methods

The course includes three 2-day meetings with intense interactions on-site, typically a mixture of lectures, exercises and other activities. Students are expected to go through the provided lecture materials, code notebooks, references and videos ahead of the meetings for a flipped classroom experience. The course is self-contained with a chapter on preliminaries.

Examination

Assignments hand-in: The students hand-in solutions to simple programming or theoretical exercises and it is graded according to a grading rubric at the end of the course.

Group project and presentation: The students work in groups of 2-4. They do a video presentation and a written report with code that is peer-reviewed and made publicly available at the end of the course.

The course allows one single retry for the assessment tasks at a date 6 months after the course concluded. The assessment tasks might be altered for the retry such as an oral examination instead of the group project.

Grades

Fail or Pass