Syllabus
Scalable Data Science and Distributed Machine Learning, 6hp

Approved by the WASP graduate school management 20260910

Large clusters composed of a network of commodity computers with new hardware, including GPUs, TPUs, CPUs, and SSDs, are today's workhorses for solving problems that scale to large datasets using parallel and distributed algorithms. This course provides 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 tandem with Stanford University, combines theoretical instruction in asymptotic algorithmic analysis with intensive hands-on implementation that runs free and locally in Docker on the student's own laptop (and scales to a distributed cluster where resources are available), using agent-based tools to assist with software development and data analysis on real-world datasets.

Main field of study
AS, Software, AI/MLX, AI/math

Course level
PhD student course
AI track: mandatory
AS track: elective
Joint curriculum: elective

Course offered for
PhD Students in the WASP graduate school
Time: Given even years, Autumn
Teacher: Raazesh Sainudiin
Examiner: Benny Avelin (UU)

Entry requirements
The participants are assumed to have a background in mathematics corresponding to the contents of the WASP-course “Introduction to 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 and 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
- 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
- understand how to design and operate modular feature, training, and inference pipelines for batch, real-time, and LLM-based systems using feature stores as the central data layer to build production ML systems end-to-end.
- understand how to engineer features, manage data across training and inference, implement observability and monitoring, and safely deploy and update ML systems at scale using practical MLOps and data engineering approaches.

Course content

The course is given in three modules. In addition to lectures by the organizer there will be invited guest speakers from industry.

Module 1* — *Practical: Introduction to Data Science & Machine Learning Systems* || *Theoretical: Analysis of Sequential, Parallel and Distributed Algorithms

This module introduces foundational concepts in data science and machine learning systems alongside theoretical fundamentals of sequential, parallel and distributed computation.

Practical topics: introduction to the data science process; setting up the course toolchain (git, Docker, and a proof-checker container — every lab runs free and locally in Docker on the student's own laptop); a short machine-checked-proof exercise; feature store concepts; building modular ML pipelines with batch prediction systems; feature engineering and transformations; embedding and vectorization techniques; a first hands-on encounter with running a small language model locally. The practical track runs on open components (Apache Spark and open feature-store concepts); no commercial platform is required or singled out.

Theoretical topics: sequential, parallel and distributed random access machine models (RAM, PRAM, DPRAM/MPC); work-depth models and Brent's theorem; reductions, sum and all-prefix sum; sorting; scheduling and load balancing; sampling and streaming sketches; the map-reduce model.

Module 2* — *Practical: Building Machine Learning Systems* || *Theoretical: Analysis of Sequential, Parallel and Distributed Algorithms

This module covers designing and operating production ML systems at scale, complemented by theoretical analysis of distributed computation.

Practical topics: building batch and real-time ML systems using feature stores, implementing feature engineering pipelines, managing training and inference data, model versioning and experiment tracking, observability and monitoring, deploying models with MLOps principles, building LLM agents using agent-to-agent (A2A) protocol, model context protocol (MCP), and retrieval-augmented generation (RAG), and applying LLMOps principles to manage large language models for building ML systems.

Theoretical topics: distributed matrix multiplication and communication-avoiding

algorithms; randomised linear algebra (dimensionality reduction and SVD); distributed solvers and least squares; optimization at scale — gradient descent, stochastic gradient descent, mini-batching and Hogwild!; distributed graph algorithms including PageRank; communications-complexity analysis; and the modern synthesis — large language models and distributed deep learning, assembled from the course's own components.

***Module 3* — Diving Deeper || Group Project Work**

Practical pathways for students to become familiar with end-to-end ML systems and production machine learning pipelines for typical decision problems, including estimation, prediction and testing in various domains, will be provided to help inspire student group projects. This 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 — in past instances, students analysed data from their own research groups or other publicly available datasets, and the same flexibility applies in 2026. Each group keeps an append-only audit log of its project work — every load-bearing step committed and replayable from the group's git repository — so results are reproducible and each member's contributions are transparent.

Teaching and working methods

The course includes three 2-day mandatory meetings with intense onsite interaction, typically a mixture of lectures, exercises and other activities. Students are expected to go through the provided course materials ahead of the meetings for a flipped classroom experience. The course is self-contained: the course book's closing Preliminaries chapters cover any background on demand.

Examination

- **Assignments hand-in:** students complete weekly comprehension quizzes and submit machine-generated receipts from the hands-on labs (run free and locally in Docker on the student's own laptop) by the specified deadlines, with at least 70% of the points in each assignment; the assignments are autograded on submission, with multiple tries permitted.
- **Group project and presentation:** students work in groups of 2-4, produce a video presentation and a written report with code, peer-reviewed and generally made publicly available at the end of the course after peer-review akin to academic conference dissemination standards. Graded out of 10 points: method / results quality (5), written report (2), video presentation (1), and a working replay.sh audit chain (2) that is verified against the group's public git repository.

Retry: one single retry is allowed for assessment tasks, six months after course conclusion. Assessment tasks may be altered for the retry, for example an oral examination instead of a group project.

Grades

Fail or Pass

Information for the intranet

Course literature

The course materials (including type-scripted mathematical lecture notes, recommended books, open MOOCs, etc.) will be made available from the course's KTH Canvas page. The required reference book for the practical aspects of the course is freely available from https://www.hopsworks.ai/lp/full-book-oreilly-building-machine-learning-systems-with-a-feature-store.