

Algorithms II

2025

Michael Kapralov

EPFL

Welcome!

Welcome!

**As Master students, you know that
algorithms are central for**

Welcome!

**As Master students, you know that
algorithms are central for**

- **computer science**

Welcome!

As Master students, you know that algorithms are central for

- ~~computer science~~ **ALL SCIENCES**
- **getting a job**
- **having fun**

Algorithmic Universe is Huge



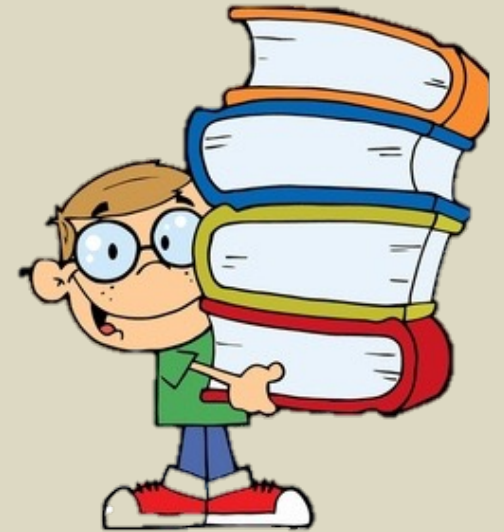
Powerful algorithmic techniques

Interesting applications/models

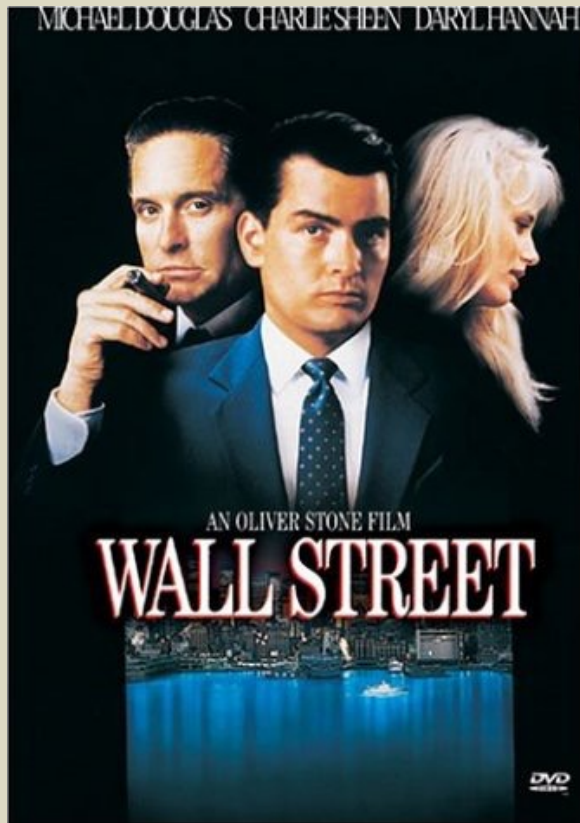
Theoretical analysis

To enjoy this course (prerequisites)

- You have taken and enjoyed basic algorithms
 - Basic datastructures (lists, queues, stacks, binary search trees, heaps...)
 - Basic algorithms (divide-and-conquer, dynamic programming, greedy, shortest path, spanning trees, flows)
 - Asymptotic analysis
- You have good basic knowledge of (discrete) math and probabilities
- I'll assume that you are now comfortable doing basic proofs



LIST OF (POTENTIAL) TOPICS



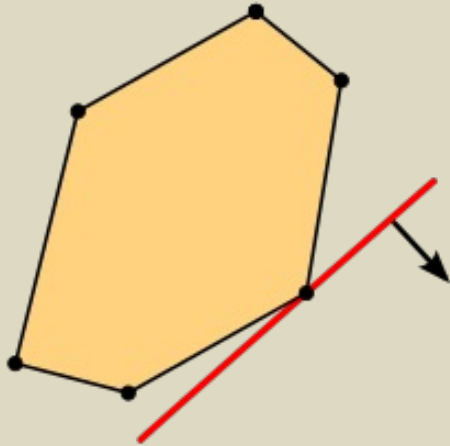
“Greed is good. Greed is right. Greed works. Greed clarifies, cuts through and captures the essence of the evolutionary spirit.”

- Gordon Gecko

When does basic greedy work? The answer is matroids

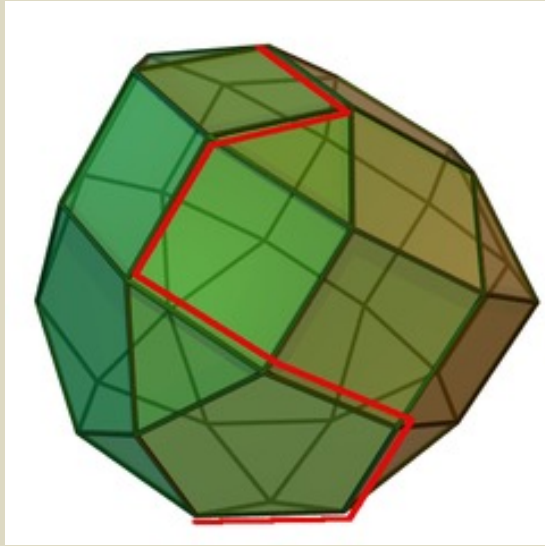
Also matroid intersection

Linear Programming

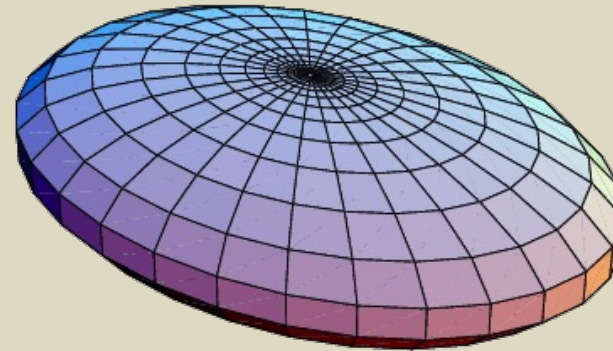


- Important big hammer
- Leonid Kantorovich - “founder of linear programming” Nobel prize in economics’ 1975
- Solution structure, duality
- Their use in the design of algorithms
- How to solve them?

Optimization



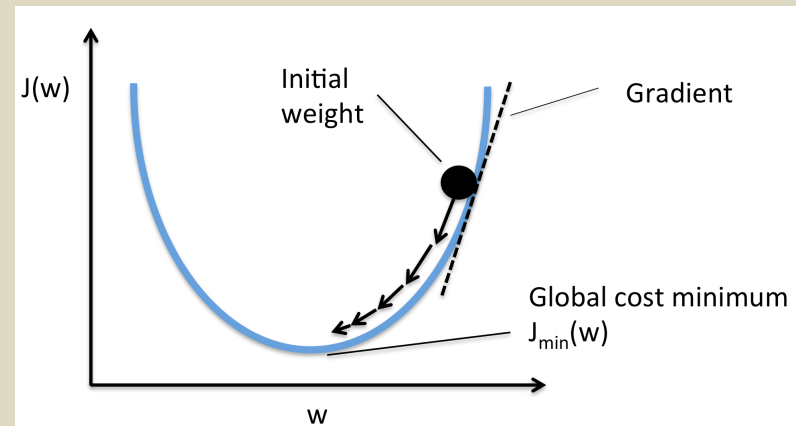
Simplex method



Ellipsoid method



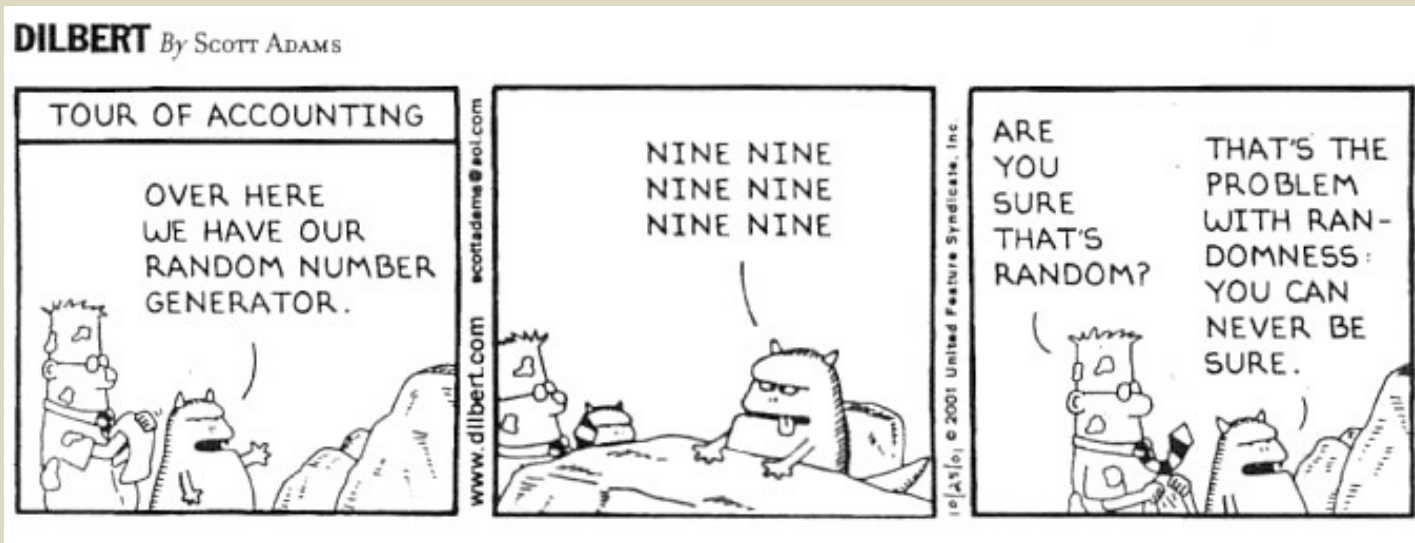
Multiplicative Weight Update



Gradient Descent

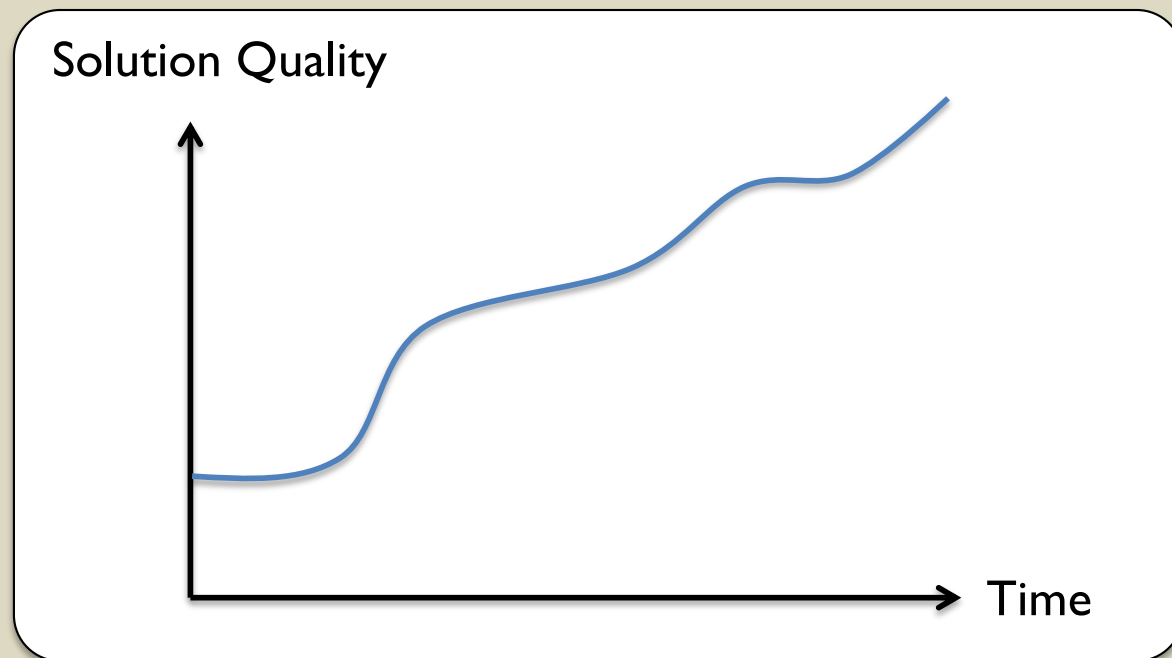
Randomized Algorithms

How to use  to design better algorithms



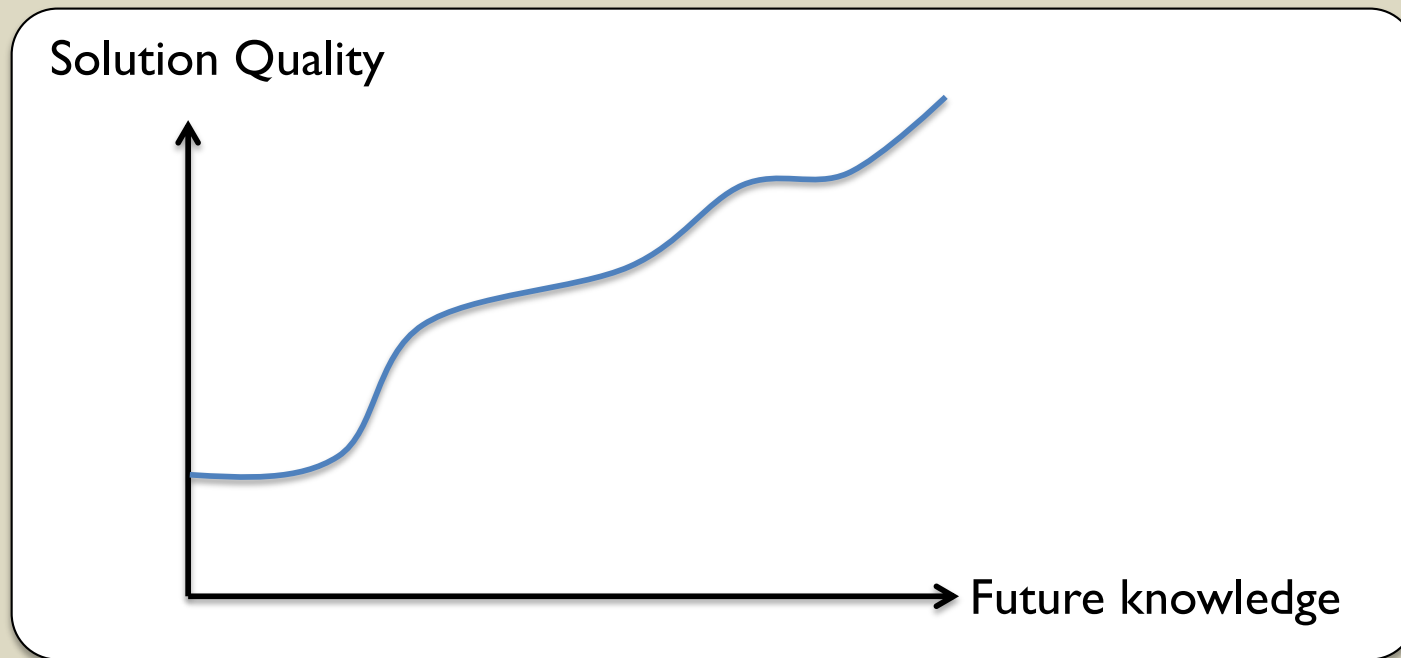
Approximation Algorithms

The study of tradeoff between solution quality and time



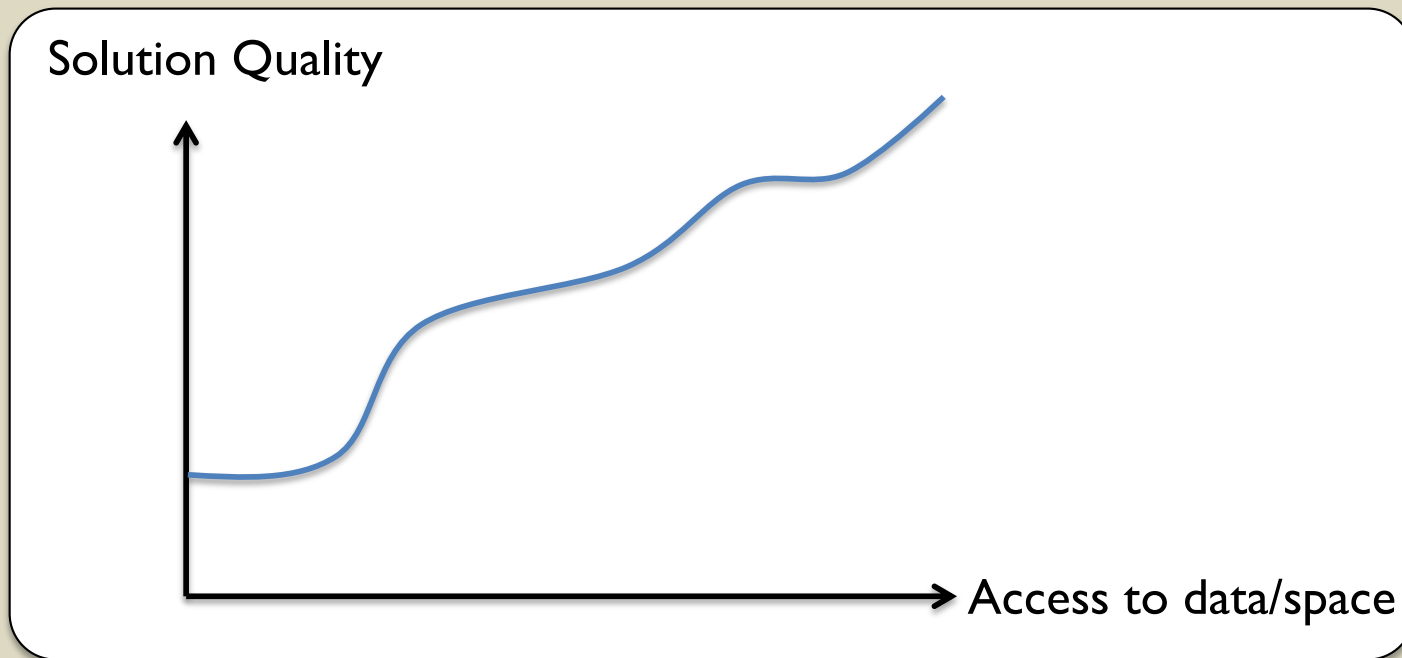
Online Algorithms

The study of tradeoff between solution quality and knowledge about future



Streaming Algorithms

The study of tradeoff between solution quality and massive data (so limited space and access)



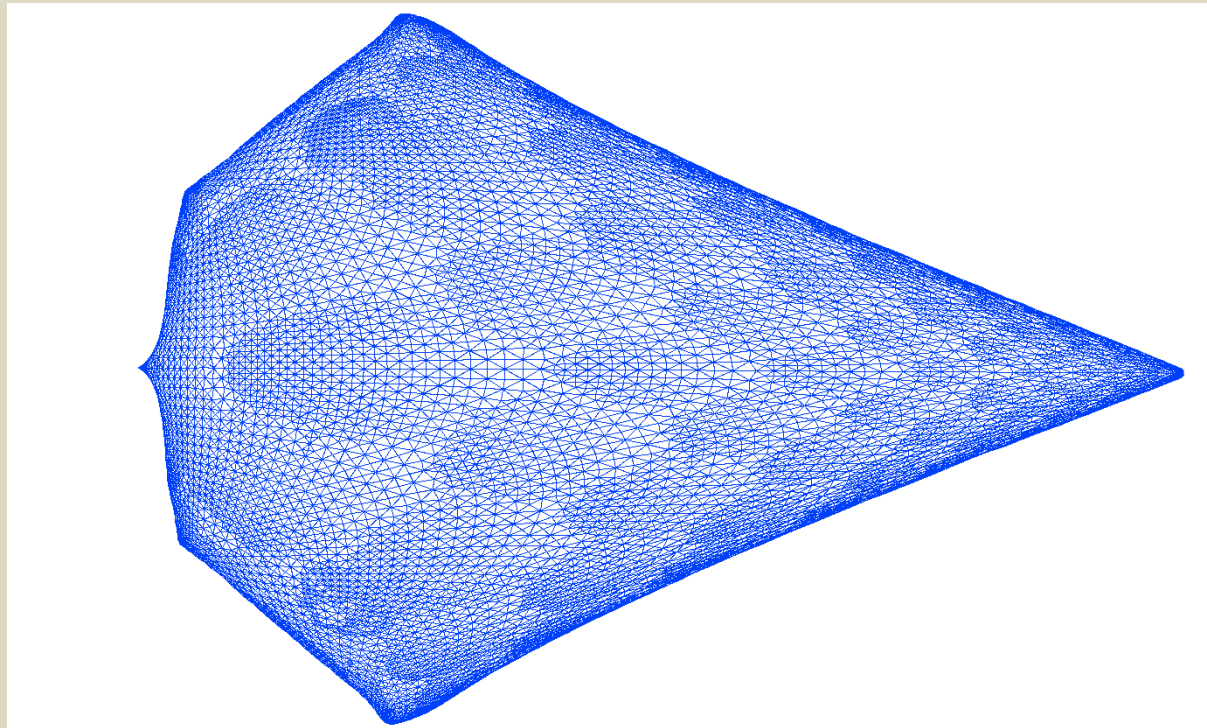
A better illustration...



How much information
must flow in a system to
perform a given task?

Spectral Graph Theory

- Study properties of graph using the eigenvalues/eigenvectors of adjacency matrix



BASIC INFORMATION ABOUT COURSE

Who is he (the course responsible)?



- Michael Kapralov
 - michael.kapralov@epfl.ch
 - <http://theory.epfl.ch/kapralov>
- Faculty of computer science
 - Research on algorithms
- Office: INJ 113
- Happy to get feedback and to answer questions!

Dream team of TAs



Alberts



James



Jan



Polina



Davide



Ekaterina



Weronika

The Course Essentials

Course homepage: <http://theory.epfl.ch/courses/AdvAlg/>

Topics in Theoretical Computer Science

Credits: 4
Lecturer: Ola Svensson
Office Hours: Wednesdays 15-18 in INJ 112
Schedule: Mondays 9:00-13:00 in CM 011

Short description

The students gain an in-depth knowledge of several current and emerging areas of theoretical computer science. The course familiarizes them with advanced algorithmic techniques, and develops an understanding of fundamental questions that underlie some of the key problems of modern computer science.

This year's course will concentrate on exciting results in computational complexity. Computational complexity is the part of computer science that studies the computational resources (time, memory, communication...) needed to solve problems that we care about. It also looks at the tradeoffs and relationships between different types of computation (e.g. whether randomness helps, exact vs approximate solutions, average vs worst case parameters...). After starting with basic concepts, concrete topics include:

- Boolean circuits and nonuniform computation
- Role of randomness in computation
- Interactive proofs and zero-knowledge proofs
- Probabilistically checkable proofs and their characterization of the complexity class NP (PCP Theorem)
- Communication complexity

Recommended books: Sanjeev Arora and Boaz Barak: *Computational Complexity: A Modern Approach*, Cambridge University Press, 2009. A draft of the book is available for free [here](#).

For previous editions of the course (covering different topics), see 2011 and 2015.

Assignments

There are currently no assignments.

Schedule and references

- Lecture 1: Introduction, Diagonalization, Time hierarchy theorem, Chapter 0-3. Another good reference is [here](#).
- Lecture 2: Why diagonalization above worst P vs NP, Space complexity, Chapter 3-4.

Grading

Links, material, information, topics, ...

MOODLE

Course: Advanced Algorithms

Navigation: Dashboard, Site pages, Current course, Participants, General, 20 February - 28 February, 27 February - 5 March, 6 March - 12 March, 13 March - 19 March, 20 March - 26 March, 27 March - 2 April, 1 April - 9 April, 10 April - 16 April, 17 April - 23 April, 24 April - 30 April, 1 May - 7 May, 8 May - 14 May

General

Recommended Reading: *Introduction to Algorithms* by Cormen et al.

Goals:

- To learn the main techniques for analyzing and for designing algorithms, while also building a repository of basic algorithmic solutions to problems in many domains.

Prerequisites:

Basic data structures (arrays, sets, stacks, queues, trees) and algorithms (binary search, sorting, including mergesort, graph connectivity, graph search), basic discrete mathematics (proof methods, induction, enumeration and counting, basic graph theory), data abstraction.

Topics:

Algorithm analysis techniques: worst-case, average-case, randomized, competitive, approximation, topics: basic algorithm design techniques: greedy, dynamic programming, divide-and-conquer, convex optimization, spectral techniques, admissible function estimation, and diagonalization. For details, see the weekly schedule on the main page.

Times and Place:

Lectures on Tuesdays 8-10 (INJ 219) and Thursdays 13-15 (INJ 219), exercise sessions on Fridays 8-11 (INJ 219). Topics to be covered from

Course Contacts: Professor Ola Svensson

Search Forums

Upcoming Events: Free Lecture Sunday 27 February, 08:00 - 09:00

Lecture notes, detailed schedule, forum, all dynamic information, videos of last year's lectures

In short, the place to be

Grading: 30% homeworks (will be 2 assignments can be made in a group of up to 3 students), 30% midterm exam, 40% final exam

Comparison to Bachelor Algorithms

- Not only algorithms for solving problems in 20 minutes (more advanced techniques for more complex problems)
- Assume more mathematical maturity

Taking as granted that it would be like most courses, where one can spend a week working on the material during the exam session. **Emphasizing that this is not the case during the presentation of the course seems important to me.**

--- Comment on a previous instalment of the course