

The Final Episode

Summarizing BADS

The Final Episode

Ingredients

- The contest prize
- Course Content Overview
- A glimpse into the future
 - Electives
 - Reading
 - Thesis Projects

Theory

- **Asymptotic complexity** (**big-O**, **big- Ω** , **big- Θ**): how do algorithms **scale**?
- **Worst-case** running time: what time can be guaranteed?
- **Expected** running time: what time can be expected?
- **Randomized** algorithm/data structure: (here) dramatically decreasing probability of the worst case by using randomness.

Theory [continued]

- **Heuristic** algorithm: algorithm guided by an approximation goal function (a heuristics). Typically not optimal or optimal only under some conditions.
- **Asymptotic memory use, in place** algorithms.

Toolbox: Algorithms for Graphs

- Depth-First and Breadth-First search in unweighted graphs: how many edges between to vertices?
- Single Source Shortest Paths (Dijkstra): what is the distance between two points in a network?
- A*: a heuristic algorithm for SSSP.
- Topological sorting
- Minimum Spanning Trees
- Network algorithms (leader election, distributed BFS, distributed Boruvka)

Toolbox: Algorithms for Search and Sort

- Sequential search: is an object in an array/list? ($O(n)$ worst and average)
- Binary search: is an object a sorted array? ($O(\lg(n))$ worst case)
- Sorting: quadratic and optimal ($n \lg n$). Counting based sorts.

Toolbox: Data Structures

- Simplistic: stacks, queues, ringbuffer, array (vector), extendable array
- Priority Queue: binary heap
- Dictionaries:
 - Hash-tables
 - BSTs
 - AVL trees

Skills

- Performance analysis,
- Scalability assessment.
- Implementing data structures as ADTs
- Combining algorithms into bigger programs.
- Choosing algorithms for an application.
- Limited training in design of new algorithms.

What now?

Studying Algorithms: take electives from our MSc programmes

Efficient AI Programming [F2009]

AI techniques for problem solving. Efficient algorithms for hard problems in modern IT applications such as ERP systems, decision support systems, configuration, optimization, computer games.

Advanced Algorithms [E2008/E2009]:

Greedy algorithms, divide and conquer, dynamic programming, network flow, reductions, approximation algorithms, randomized algorithms, parallel algorithms

Database Tuning [F2009]

To be able to achieve high performance, an understanding of the internal workings of RDBMSs is needed. In addition to this, the course covers indexing methods that may dramatically speed up applications. A project, comprising approximately half of the course, consists of a database development project with special emphasis on performance demands. The focus of the course is on situations where there are large amounts of data, and we consider how efficient algorithms and data structures are used to handle this in RDBMSs.

What now?

Reading onwards

- Jon Kleinberg. Eva Tardos. *Algorithm Design*.
- Anany Levitin. *Introduction to The Design and Analysis of Algorithms*.
- Cormen. Leiserson. Rivest. Stein. *Introduction to Algorithms*.

- Eric Roberts. *Thinking recursively with Java*.
- Martin Fowler. *Refactoring*.
- Adam Barr. *Find the bug. A book of incorrect programs*.

What now?

Future thesis prospects

- My general topic is
 - (model-driven) component-based development
 - especially in the context of software product lines.
- My website lists sample project proposals:
 - Software engineering (incl. testing)
 - Object-Oriented programming
 - Domain specific languages
 - Open source
 - Models of computation
 - Artificial intelligence
- A thesis project with me?