

Algorithms and Data Structures

Lecturers: Priv.-Doz. Dr. Peter Sanders, Dr. Kavitha Telikapalli

Exercises: Roman Dementiev

Organization

Lectures: Mo 14:15–16:00, Mi 9:15–10:00

Exercises: find slots now

Web: www.mpi-sb.mpg.de/~sanders/courses/algdat03

Contact: rooms 308,414???,322, MPII building (Building 46), times:???
 {sanders,telikapalli,dementiev}@mpi-sb.mpg.de

Mailing List: algdat-l@postino.mpi-sb.mpg.de

Written Midterm Exam: Monday before Christmas 9:00 s.t. earlier?

Written Final Exam: week after the lecture period.

Grading: 50% final 30 % midterm 20 % exercises. Must pass final and all but two of the exercises

German: you can answer the exercises in German. The exam will also have a German translation of the questions

Overview $\not\subseteq, \not\supseteq$

Graphs: Graph representation. Graph traversal: BFS, DFS, ((strongly) connected components, biconnected components, topological sorting. Shortest Paths, Minimum spanning trees maximum flows, matchings, edge coloring

Data Structures: Integer search trees, priority queues, union-find, interval maxima.

Geometry: convex hulls, line segment intersection, Delaunay triangulations, Voronoi diagrams, linear programming, smallest enclosing balls

Strings: string matching, full text indexing, string sorting

Optimization: greedy, dynamic programming, exhaustive search, black-box solvers, local search metaheuristics

Advanced Models: External memory, parallel algorithms (if time permits)

Plus selected lectures on research at MPII AG1

Literature

- [Handouts, references on the course web page](#)
- Kurt Mehlhorns Books on Algorithms and Data Structures Volume 1–3.
- Introduction to Algorithms. Cormen Leiserson and Rivest. Second Edition a bit better! Very detailed explanations.
- Computational Geometry Algorithms and its Applications. Marc de Berg, Marc van Kreveld, Marc Overmars, Ottfried Schwarzkopf

Prerequisites

- Mathematical notation: sets, relations, $\sum \forall \exists \wedge \vee \neg$
- Basic proofs: induction, invariants, sums, ...
- Some probability theory: random variables, linearity of expectations, binomial coefficients
- $\mathcal{O}(\cdot)$, $\Omega(\cdot)$, $\Theta(\cdot)$
- arrays, doubly linked lists, hash tables, binary heaps, 2-4 trees/red black trees or the like: search trees with log-time insert/delete/lookup, sorting in $\mathcal{O}(n \log n)$ time.
- Graph terminology and basic graph algorithms (Dijkstra SSSP, Prim MST)

Pseudo Code

C/C++/Java ++, --, +=, ...

Pascal like stuff for better typography and to differentiate from actual implementation

parallel assignment: e.g., $(a, b) := (b, a)$ // swap

sets: $\min \{p > k : p \text{ is prime}\}$

sequences: $\langle 1, 2, 3 \rangle$, $\langle e \in s : e < p \rangle$ implement by queues, stacks, arrays, lists, ...

Machine Model

von Neumann / RAM model

- (effectively) unlimited memory
- **uniform memory access** time
- constant time for “basic” operations on operands of size $\mathcal{O}(\log n)$ bits (n is input size) $+ - * / \ll \gg \neg \vee \wedge \& | \dots$
sometimes REAL-RAM operations on real numbers
- Analysis: count instructions
- Use $\mathcal{O}(\cdot)$ to get rid of constant factors / lower order terms
- `new`, `delete` in constant time

Advanced Models? Perhaps later

Example: Quicksort

Function `quickSort(s : Sequence of Element) : Sequence of Element`

```

if  $|s| \leq 1$  then return  $s$  // base case
pick  $p \in s$  uniformly at random // pivot key
 $a := \langle e \in s : e < p \rangle$  // (A)
 $b := \langle e \in s : e = p \rangle$  // (B)
 $c := \langle e \in s : e > p \rangle$  // (C)
return concat(quickSort(a), b, quickSort(c))

```