Advanced Algorithmics (4AP)

Jaak Vilo
2008 fall

Goals

• To learn the main concepts and techniques of the algorithm design and analysis

• To be able to compare and analyze algorithms and data structures

• To learn to learn, use, solve, read, write, and present
Algorithms

• Al-Khwārizmī
  • http://en.wikipedia.org/wiki/Muhammad.ibn.M%C5%ABs%C4%81.al-Khw%C4%81rizm%C4%81

2 6 7 1

\[ 2 \times (6+7) + 1 = 27 \]
\[ 2 \times 6 \times (7+1) = 96 \]

Q: How to maximize the value of any expression?

13 x 11

\[
\begin{array}{c}
1 & 1 & 0 & 1 \\
\times & 1 & 0 & 1 \\
\end{array}
\]

\[ 1 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 \\
+ & 1 & 1 & 0 & 1 \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 (binary 143)
\]

\(O(n^2)\)
But Al Khwarizmi knew another way to multiply, a method which is used today in some European countries. To multiply two decimal numbers \( x \) and \( y \), write them next to each other, as in the example below. Then repeat the following: divide the first number by 2, rounding down the result (that is, dropping the .5 if the number was odd), and double the second number. Keep going till the first number gets down to 1. Then strike out all the rows in which the first number is even, and add up whatever remains in the second column.

\[
\begin{array}{c}
11 & 13 \\
5 & 26 \\
2 & 52 \text{ (strike out)} \\
1 & 104 \\
\end{array}
\]

143 \text{ (answer)}

\[
\begin{array}{c}
1101 \\
\times 1011 \\
\hline
1101 \text{ (1101 times 1)} \\
1101 \text{ (1101 times 1, shifted once)} \\
0000 \text{ (1101 times 0, shifted twice)} \\
+ 1101 \text{ (1101 times 1, shifted thrice)} \\
\hline
1000111 \text{ (binary 143)}
\end{array}
\]

is the same as:

\[
\begin{array}{c}
11 & 13 \\
5 & 26 \\
2 & 52 \text{ (strike out)} \\
1 & 104 \\
\hline
143 \text{ (answer)}
\end{array}
\]
## Analysis of algorithms

- *The theoretical study of computer-program performance and resource usage.*
- *What’s more important than performance?*

| Modularity | User-friendliness |
| Correctness | Programmer time |
| Maintainability | Simplicity |
| Functionality | Extensibility |
| Robustness | Reliability |

## Why study algorithms and performance?

- Algorithms help us to understand *scalability*.
- Performance often draws the line between what is feasible and what is impossible.
- Algorithmic mathematics provides a *language for talking about program behavior*.
- Performance is the *currency of computing*.
- The lessons of program performance generalize to other computing resources.
- Speed is fun!
Course structure

• Lectures: 2x / week
  – Some will be omitted (total ~24 lectures)
• Weekly homework (obligatory min 50% assignments)
  – Starting from the 2nd week
• Practical project
  – a practical implementation and comparison of efficiency of some algorithms
• Reading textbooks and possibly papers
• Writing an essay
• Presentations by students (TBD)
• Exam

Schedule:

• Lectures: (Prof. Jaak Vilo)
  – Wednesday 14:15-16, Liivi 2-405
  – Thursday 12:15-14, Liivi 2-111

• Practical:
  – Group 1: Wed. 16:15, Liivi 2-315 (Meelis Kull)
  – Group 2: Thu. 14:15, Liivi 2-315 (Liina Kamm)
Contacts:

• Jaak Vilo – prof. of bioinformatics
  – vilo @ ut.ee
• Liina Kamm, PhD student
  – kamm @ ut.ee
• Meelis Kull, PhD student
  – meelisk @ ut.ee
• ati.algorithmics @ lists.ut.ee

Your expected workload

24 Lectures 48h
12 Practicals 24h
Self study 12\*4 = 48h
Term paper 8h
Project work 20h
Exam 4h + 8h = 12h

----------------------------------------
Total 160h (4AP)
Grade

- Homework 30 (80%=max) + bonus
- Term paper 10 *writing skills*
- Project work 10 *results*
- Exam 50

-------------------------------------

- Total 100p

- 91+ = A, 81+ = B, ...

---

Homework

- Essential part of the course
- Obligatory to perform minimum 50% tasks
- Presentations orally during the practicals

- In rare cases of not attending:
  - Solutions must be in before the class by email both to the TA (Liina or Meelis) and prof. Vilo
  - State clearly which tasks have been completed.
Term paper

• Will be an essay based on some article

• To be decided during the course

• Reading and writing skills

• A format of the extended abstract (2 pages, with a short abstract, contents, citations, etc)

Project

• A practical algorithm development and comparison task

• Presentation of results during practicals (or special sessions) at the end ~10 min.
Exam

• Will be based on questions similar to the homework assignments

• Knowledge of the basic principles of algorithms

• Creative use of the algorithms

Short CV (Jaak Vilo)

• Tallinna Reaalkool (2. Keskkool) 1984
• TÜ Rakendusmatemaatika/arvutiteadus 1991
• University of Helsinki: 1989/90; 1991-1999
• European Bioinformatics Institute 1999-2002
• PhD 2002, Univ. of Helsinki. /Pattern discovery/
• EGeen 2002 => Quretec 2006- ...
• U. Tartu: docent, sen. researcher,
• Professor (from dec 2007)
Research

- Bioinformatics
- Data mining, Machine Learning, Visualization, ...
- Practical algorithms
- Data management and analysis

• [BIIT](http://biit.cs.ut.ee/)

Questionnaire

- To assess the basic starting point and expectations before the course start

- Please fill in the form to the best of your ability *as is* during the next 30 minutes.
Contents

• Algorithms and Data structures – basics
• Abstract data types (+ STL, LEDA, ...)
• Analysis of algorithm complexity, O(), recurrences
• Turing machines, NP, NP-complete, P-Space, ...
• Graphs (TSP, shortest path, ...)
• Computational geometry: Some key algorithms
• Basic designs: Dynamic programming, divide and conquer, randomised algorithms, ...
• Heuristics (approximations):
  — Greedy Algorithms (packing, set cover, weighted set cover, ...) local search, tabu search
  — Simulated annealing (e.g. TSP)
  — Genetic algorithms (?), ...
  — Monte Carlo, Las Vegas
• Very tough (NP) problems ("Monkey problem": unsolvable problems?)
• Online algorithms vs offline
• Deterministic or non-deterministic
• Randomized algorithms
• Exact or approximate:
• Parallel algorithms
• Data Mining?
• Clustering: hierarchical clustering, k-means, k-medoids
• ...

Programming languages:

• Pseudocode
  — primarily this is about pseudocode
• Pointers, arrays, memory handling
  — Low- and high level abstractions
• C/C++
• Java, Python, perl, ...
  — Explicit data structures and algorithms
• Whichever choice: you must be able to explain it
• Clarity and simplicity is a key
“The Textbook”

- **Introduction to Algorithms**, Second Edition
  Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein
  
  - [http://mitpress.mit.edu/algorithms/](http://mitpress.mit.edu/algorithms/)
  - [http://books.google.com/books?id=NLngYyWFf_YC](http://books.google.com/books?id=NLngYyWFf_YC)

- **Algorithmics: The Spirit of Computing** (3rd Edition)
  - David Harel, Yishai Feldman
  - Addison Wesley; 3 edition (June 11, 2004)
  
  - [http://books.google.com/books?id=txxLovFWkCUC](http://books.google.com/books?id=txxLovFWkCUC)
  - [http://www.wisdom.weizmann.ac.il/~dharel/algorithmics.html](http://www.wisdom.weizmann.ac.il/~dharel/algorithmics.html)
• **Foundations of Computer Science: C Edition**
  • **Alfred V. Aho, Jeffrey D. Ullman**
  • **W. H. Freeman (October 15, 1994)**

• **The Art of Computer Programming (TAOCP)**
  • **Donald E. Knuth.**
• **The Algorithm Design Manual**
  – *Steven S. Skiena*

• **Algorithms [ILLUSTRATED]**
  – *Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani*

---

**Mathematics**

• **Concrete Mathematics: A Foundation for Computer Science (2nd Edition) (Hardcover)**
  – *Ronald L. Graham, Donald E. Knuth, Oren Patashnik*

  - Hardcover: 672 pages
  - Addison-Wesley Professional;
  - 2 edition (March 10, 1994)
More books

• Parallel algorithms
  — An Introduction to Parallel Algorithms Joseph Jaja

• Heuristics
  — How to Solve it: Modern Heuristics. By Zbigniew Michalewicz, David B. Fogel
  — Edition: 2, Illustrated Published by Springer, 2004

• Randomized algorithms

• Complexity theory
  This book is a classic, developing the theory, then cataloguing many NP-Complete problems.

Wikipedia:

• http://en.wikipedia.org/wiki/Algorithm
• http://en.wikipedia.org/wiki/List_of_algorithm_general_topics
• http://en.wikipedia.org/wiki/List_of_algorithms
• http://en.wikipedia.org/wiki/List_of_data_structures
• http://en.wikipedia.org/wiki/Computational_complexity_theory
• http://en.wikipedia.org/wiki/List_of_NP-complete_problems
• http://en.wikipedia.org/wiki/Randomized_algorithm
• http://en.wikipedia.org/wiki/Parallel_algorithm
• ...