Master Thesis

Algorithms
Algorithms – Who?

Faculty
Lars Arge
Gerth Stølting Brodal
Gudmund Skovbjerg Frandsen
Peter Bro Miltersen
Christian Nørgaard Storm Pedersen

Ph.d. and Master students
…

Researchers
Thomas Mailund
Henrik Blunck
Peyman Afshani
Nodari Sitchinava
Mohammad Abam
Deepak Ajwani
Kristoffer Arnsfelt Hansen
Nguyen Kim Tang
Algorithms – Where?

Algorithms (Turing 0+1)
Arge, Brodal, Frandsen,
Miltersen, Blunck, Ajwani,
Sitchinava, Abam,
Afshani, Hansen

BioInformatics
(Building 110)
Pedersen, Mailund
Algorithms – Courses

Introductory
• Programming 2 - Frandsen
• Algorithms and data structures 1+ 2 - Brodal
• Machine architecture/Operating systems - Pedersen

Advanced
• Optimization/Combinatorial search - Miltersen/Arnsfelt
• Computational geometry - Brodal
• Advanced data structures - Brodal
• I/O algorithms - Arge
• Dynamic algorithms - Frandsen
• Randomized algorithms - Frandsen
• String algorithms - Pedersen/Mailund
• Algorithms in bioinformatics - Pedersen
• Machine learning - Pedersen/Mailund
• Complexity theory - Miltersen/Hansen
• Algorithmic game playing - Miltersen
• Data compression (loseless/lossy) - Miltersen
Algorithms – Research

- I/O algorithms
- Computational geometry
- Data structures
- String algorithms
- Complexity theory
- Data compression
- Optimization
- Algebraic algorithms
- BioInformatics
- Graph algorithms
- Dynamic algorithms
- Randomized algorithms
- Algorithmic game theory

Subset of research interests
Solid lines = major interest

- Arge
- Brodal
- Frandsen
- Miltersen
- Pedersen
Algorithms – Research

• Theoretical computer science

• Tool development
  – BioInformatics, I/O algorithms

• Algorithm engineering
  – primarily in relation to thesis work

• Seminars – master students very welcome
  – BiRC, MADALGO, CAGT, …
Algorithm Research
— a typical result statement

<table>
<thead>
<tr>
<th>Problem</th>
<th>Best cache-oblivious result</th>
<th>Best cache-aware result</th>
</tr>
</thead>
<tbody>
<tr>
<td>List ranking</td>
<td>$O(Sort(V))$</td>
<td>$O(Sort(V))$</td>
</tr>
<tr>
<td>Euler Tour</td>
<td>$O(Sort(V))$</td>
<td>$O(Sort(V))$</td>
</tr>
<tr>
<td>Spanning tree/MST</td>
<td>$O(Sort(V))$</td>
<td>$O(Sort(V))$</td>
</tr>
<tr>
<td>Undirected BFS</td>
<td>$O(V + Sort(E))$</td>
<td>$O(ST(E) + Sort(E)$</td>
</tr>
<tr>
<td></td>
<td>$+ \frac{E}{B} \cdot \log V + \sqrt{VE/B}$</td>
<td>$+ \frac{E}{B} \cdot \log V + \sqrt{VE/B}$</td>
</tr>
<tr>
<td>Directed BFS &amp; DFS</td>
<td>$O((V + E/B) \cdot \log V + Sort(E))$</td>
<td>$O((V + E/B) \cdot \log V + Sort(E))$</td>
</tr>
<tr>
<td>Undirected SSSP</td>
<td>$O(V + (E/B) \cdot \log(E/B))$</td>
<td>$O(V + (E/B) \cdot \log(E/B))$</td>
</tr>
</tbody>
</table>

Table 1. I/O-bounds for some fundamental graph problems.

Types of Algorithmic Thesis

• Solve a concrete problem
  ...using algorithmic techniques
• Survey of a research area
• Implement a technical paper
  ...fill in the missing details
  ...perform experiments
• Explain all (missing) details in a technical paper
  ...how 8 pages become +100 pages
• Experimental comparison of several algorithms
• The clever idea: Describe a new algorithm
• Examples:
  www.cs.au.dk/~gerth/cv/index.html#Advising
  www.cs.au.dk/~cstorm/www/students/
Master Thesis in Algorithms

Thesis work

- Large fraction of time spend on trying to understand technical complicated constructions

- Implementations are often an ”existence proof” – most algorithm authors do not implement their algorithms (did they ever think about the missing details?)

- Hard to convince friends that it took you a year to understand an 8 page paper...
Hidden work...

Compact Oracles for Reachability and Approximate Distances in Planar Digraphs

Mikkel Thorup
AT&T Labs - Research, Shannon Laboratory
180 Park Avenue, Florham Park, NJ 07932, USA
mthorup@research.att.com

! Warning !
Nontrivial construction ahead of you

! Warning !
Need to understand another paper first

The proof is contained in [13], but somewhat hidden in other details because they need to ensure that the paths are of $O(\sqrt{n})$ length. The existence of $v$ and $w$ is what is actually proved in the proof of Lemma 2 in [13]. They find $(v, w)$ as an edge in an arbitrary triangulation of $H$. No side of the fundamental cycle of $(v, w)$ in $T$ contains more than $2/3$ of $H$. The vertices $v$ and $w$ are found in linear time in steps 1, 8, and 9 of the partitioning algorithm in §3 in [13].