T-79.7001 Postgraduate Course in Theoretical Computer Science (2–10 cr) P V

Autumn 2007

Weekly Sessions and Course Personnel

Seminars: Mondays 16-20, TB353
Teacher: Prof. Ilkka Niemelä, tel. 451 3290, e-mail: Ilkka.Niemela@tkk.fi
Web: http://www.tcs.hut.fi/Studies/T-79.7001/

Topic: Propositional Proof Complexity

- Basics of Boolean functions and circuits
- Circuits as a computation model
- Proof complexity
- Complexity of proofs in major propositional proofs systems (sequent calculus, resolution, algebraic refutation systems, cutting planes, Frege systems)

Material
(Sections 1 and 5).

Passing the course

The course is organized as a research seminar and taken by
1. giving seminar talks and
2. doing homework.

- The grade of the course (0–5) is determined by the respective grades of (i) the seminar talks (70%) and (ii) homework (30%).

© 2007 TKK, Laboratory for Theoretical Computer Science
Seminar talks

A seminar talk involves:
- giving a 45 min presentation on a particular part of the course material
- no written report; slides in pdf form on the course web page
- Each seminar talk is graded by the other students w.r.t. the merits of the presentation

Grading

- The grade of a seminar talk: determined by the grades given by fellow students (given as part of their homework).
- The grade for the homework:
  - The teacher grades learning diaries (0–2 points each) (a diary can be submitted only for talks which have been attended).
  - Submitting a talk evaluation (1 point each)
  - These points form the homework points which translate into grades as follows:

<table>
<thead>
<tr>
<th>Grade</th>
<th>Lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>50%</td>
</tr>
<tr>
<td>2</td>
<td>57%</td>
</tr>
<tr>
<td>3</td>
<td>64%</td>
</tr>
<tr>
<td>4</td>
<td>71%</td>
</tr>
<tr>
<td>5</td>
<td>78%</td>
</tr>
</tbody>
</table>

Why is Proof Complexity Interesting for Computer Science

- Proof complexity is related to major open problems in CS: \( P = NP? \), \( NP = coNP? \), \ldots
- Can be used for comparing, classifying and developing automated theorem proving techniques like propositional satisfiability (SAT) checkers.
- Can be used for analysing complexity and strength of search procedures and heuristics.
- Can be used for analysing cryptographic primitives and assumptions.