Advanced Topics in Complexity Theory
Introduction and Lecture Overview

Daniel Borchmann
Theoretical Computer Science

March 23, 2016

cc
Organization
Course Tutors

Daniel Borchmann
Lectures and Exercises
Organization

Lectures
Monday, 5. DS, in APB E005

Exercise Sessions
Tuesday, 3. DS, in APB E005

Web Page
https://ddll.inf.tu-dresden.de/web/Advanced_Topics_in_Complexity_Theory_(SS2016)

Lecture Notes
There will probably be no lecture notes.
Goals and Prerequisites

Goals
Discuss some of the topics of complexity theory usually not taught in introductory courses, namely

- Approximation complexity
- Interactive Proof Systems
- Counting complexity

Prerequisites
Although this is an “advanced lecture”, it is meant to be as self-contained as possible. However, some familiarity with the basic notions of complexity theory is necessary:

- Basic complexity classes like P, NP, PSpace, ExpTime, ...
- Reductions and Completeness
Reading List

- Michael R. Garey and David S. Johnson: *Computers and Intractability*; Bell Telephone Laboratories, Inc. 1979
- Erich Grädel: *Complexity Theory*; Lecture Notes, Winter Term 2009/10
The Story of a Remarkable Theorem
A Well-Known Characterization of NP

Recall

NP is the class of problems that have “short proofs”: A language $L$ is in NP if and only if there exists a deterministic polynomial-time Turing machine $M$ and a polynomial $p$ such that

$$L = \{ x \in \Sigma^* \mid \exists y : |y| \leq p(|x|) \land (x, y) \in L(M) \}.$$ 

This characterization can also be seen “interactively”:

- a powerful Prover devises for input $x$ a proof $y$;
- a polynomial-time Verifier checks whether $y$ is indeed a proof for $x$.

Then $x \in L$ if and only if the Prover can find a proof that is accepted by the Verifier.
A Natural Followup Question ...

One could now ask ...

What is the most general notion of an “efficient proof”? 

**Definition 1**

In an *interactive proof system*, a randomized polynomial-time verifier with private coin tosses interacts with an unrestricted prover by sending messages back and forth in polynomially many rounds, such that:

- correct statements should have proofs accepted with probability 1 ("completeness")
- incorrect statements should be accepted, regardless of the proof, with probability at most 1/2 ("soundness")

Denote with IP the class of languages with interactive proofs that run for at most polynomially many rounds.
Graph Non-Isomorphism

Question
Is IP more powerful than NP?

Observation
If one leaves out the randomization, it is easy to see that IP = NP.

Anyway, there are indications for an affirmative answer ...

Definition 2 (Graph Non-Isomorphism, GNI)
The GNI-problem gets two graphs as input and answers true if and only if these graphs are not isomorphic.

- It is known that GNI ∈ coNP, but not whether GNI ∈ NP;
- It is easy to see that GNI ∈ IP
Graph Non-Isomorphism

Example 3 (A Protocol for GNI)

Let $G_1, G_2$ be the input graphs.

- **Verifier:** randomly guess $i \in \{0, 1\}$ and send randomly permuted graph $H = \pi(G_i)$ to Prover
- **Prover:** Return $i$ such that $H = G_i$ to Verifier
- **Verifier:** accept if the answer of the Prover corresponds to the original choice of $i$

Analysis

- **Completeness:** if $G_1 \not\equiv G_2$, then given $H$, the Prover can find $i \in \{0, 1\}$ such that $H \simeq G_i$ with certainty. Thus the input is accepted with probability 1.
- **Soundness:** if $G_1 \equiv G_2$, then given $H$, the Prover can only guess the original value of $i$. Thus the input is rejected with probability 1/2.
The Subjective Relationship between NP and IP

- IP can be seen as a “reasonable extension” of the notion of an efficient proof as used in NP
- There exists an oracle with respect to which coNP $\not\subseteq$ IP; this makes showing coNP $\subseteq$ IP rather difficult ...
- Exception: IP is a kind of randomized version of NP, but not much more than this

Thus, the following result was a big surprise when it came in 1990:

**Theorem 4 (Lund, Fortnow, Karloff, Nisan)**

There is an interactive protocol for computing the permanent of a binary matrix. Consequently, $PH \subseteq P^{#P} \subseteq IP$. 
The Permanent

Definition 5 (Permanent)
Let $A = (a_{ij}) \in \{0, 1\}^{n \times n}$ be a binary matrix. Then the permanent of $A$ is

$$\text{perm}(A) := \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i\pi(i)}$$

Remarks
- Note the similarity to the definition of the determinant of $A$.
- If $A$ is the adjacency matrix of an undirected bipartite graph $G$, then $\text{perm}(A)$ is the number of perfect matchings in $G$.
- Computing the permanent of a binary matrix is $\#P$-complete.
Counting Complexity

Caveat
Sometimes known that one answer exists is not enough, but one is merely interested in how many solutions exist.

Definition 6 (#P)
The class #P consists of all function problems of the form: given a non-deterministic polynomial time Turing machine \( M \), compute for input \( x \) the number of accepting paths of \( M \) on input \( x \).

Counting versions of simple problems can easily be #P-complete:

- 2SAT \( \in \) P, but #2SAT is #P-complete
- Finding perfect matchings in bipartite graphs is easy, but counting them is #P-complete
Two Central Results in Counting Complexity

Theorem 7 (Valiant, 1979)

*Computing the permanent is \#P-complete.*

Theorem 8 (Toda, 1989)

\[ \text{PH} \subseteq \text{P}^{\#P}. \]
A Central Result

As it now seems, IP is not so “small” anymore ...

**Theorem 9 (Shamir, 1990)**

\[ \text{IP} = \text{PSpace} \]
Well, actually ...

We wanted a slight generalization of the notion of “provable”, but we ended up with something supposedly much bigger.

**Question**

Can we “scale down”? Can we somehow restrict the verifier to get back to NP?
Definition 10
Denote with $\text{PCP}(r(n), q(n))$ the class of languages with an interactive proof systems of completeness 1 and soundness $1/2$, in which the verifier
- uses $\mathcal{O}(r(n))$ bits of randomness and
- is allowed to only access $\mathcal{O}(q(n))$ bits of the proof.

PCP stands for *probabilistic checkable proofs*.

Theorem 11 (Feige, Goldwasser, Lovász, Safra, Szegedy, 1991)
$\text{NP} \subseteq \text{PCP} (\log n \cdot \log \log n, \log n \cdot \log \log \log n)$. 
Further Improvements and the Final PCP-Theorem

**Question**

Why $\log n \cdot \log \log n$?

**Theorem 12 (Arora, Safra, 1992)**

$\text{NP} \subseteq \text{PCP}(\log n, \log n)$. *In fact, NP $\subseteq$ PCP($\log n$, $(\log n)^{0.5+\epsilon}$).*

**Question**

Can it work with even less query bits?

**Theorem 13 (The PCP-Theorem; Arora, Lund, Motwani, Sudan, Szegedy, 1992)**

$\text{NP} \subseteq \text{PCP}(\log n, 1)$.

NYTimes: “New shortcut found for long math proofs”
Consequences for Hardness of Approximation

The PCP-Theorem has consequences on *hardness of approximation*:

**Approach**

Since solving NP-complete problems cannot be done in polynomial time (for all we know), one often settles for *approximate solutions*.

**Caveat:** Sometimes this does not work.
But the PCP-Theorem makes this approach even less helpful:

- There is some $\gamma < 1$ such that computing $\gamma$-approximations for MinVertexCover is NP-hard.
- For every $\rho < 1$, computing a $\rho$-approximation of MaxIndependentSet is NP-hard.
- If there exists a *polynomial-time approximation scheme* for Max3SAT, then $P = NP$. 
Lecture Overview
Topics of This Lecture

Planned Topics

▶ Approximation Complexity
  ▶ Approximation Algorithms
  ▶ Approximation and Complexity
  ▶ Non-approximability and the PCP-Theorem

▶ Interactive Proof Systems
  ▶ Private and Public Coin Protocols
  ▶ IP = PSpace

▶ Counting Complexity
  ▶ Counting Problems and the class #P
  ▶ #P-completeness, Valiant’s Theorem
  ▶ Toda’s Theorem

Note

While this is an “advanced” lecture, it will be as self-contained as possible.