CS154, Lecture 18:
CS 154 – Final Exam

Wednesday December 12, 12:15-3:15 pm
STLC 111

You’re allowed one double-sided sheet of notes
Exam is comprehensive (but will emphasize post-midterm topics)
Look for practice final and solutions
Evaluate CS 154

Your Input Really Matters
Chapter I

Finite Automata (40s-50s):
Very Simple Model (constant memory)
• Characterize what can be computed (through closure properties)
• First encounter: non-determinism (power of verified guessing)
• Argue/characterize what cannot be computed
• Optimization, learning

More modern (complexity-theoretic) perspective:
streaming algorithms, communication complexity
Chapter II

Computability Theory 30’s – 50’s

Very Powerful Models: Turing machines and beyond

(Un)decidability – what cannot be computed at all

• Foot in the door – an unrecognizable language
• Many more problems, through reductions
• Hierarchy of exceedingly harder problems

The foundations of mathematics & computation

Kolmogorov complexity (universal theory of information)
Chapter III

Complexity Theory: 60’s –
Time complexity, P vs. NP, NP-completeness
• Non-determinism comes back
• Our foot in the door – SAT, a problem that is likely hard to compute
• Many more problems through (refined) reductions
• An hierarchy of hard problems

Other Resources: space, randomness, communication, power,... Crypto, Game Theory, Computational Lens
Computing:

“evolution of an environment via repeated application of simple, local rules”

Somebody
Computational Lens
Hitchhiker's Guide to the Galaxy
Computational Game Theory

Markets computing an equilibrium.
Simple dynamics (best response)?

Bounded Rationality:
Prisoners Dilemma
Repeated Games, infinite, finite
Backward Induction
Finite Automata
Always Cooperate, Always Defect, Tit for Tat, Trigger
Limited Resources

Example 3.12: Recall Theorem 1.3, where we proved the following statement with hypotheses $H = \text{"$U$ is an infinite set, $S$ is a finite subset of $U$, and $T$ is the complement of $S$ with respect to $U$." The conclusion $C$ was "$T$ is infinite." We proceeded to prove this theorem by contradiction. We assumed "not $C$", that is, we assumed $T$ was finite.

Our proof was to derive a contradiction from $H$ and not $C$. We first showed from the assumptions that $S$ and $T$ are both finite, that $U$ also must be finite. But since $U$ is stated in the hypothesis $H$ to be infinite, and a set cannot be both finite and infinite, we have proved the logical statement "false." In logical terms, we have both a proposition $p$ (U is finite) and its negation, not $p$ (U is infinite). We then use the fact that "$p$ and not $p$" is logically equivalent to "false."
Factoring & One-Way Functions

Given two primes P and Q easy to compute N=PQ. For random such N, assume it is hard to find P and Q. Special case of One-Way Functions (the most basic cryptographic primitives).

Random Instances of SAT that are hard
Zero-Knowledge Proofs
Hardness of learning
Pseudorandom Generators
Deterministically increasing entropy
Randomness is weak
PCPs, Hardness of Approximation, Approximation-Preserving Reductions, Interactive Proofs, Zero-Knowledge, Cold Fusion, Peace in the Middle East
Coping with NP-Completeness

There are thousands of NP-complete problems

Many are solved all the time !?!

Average Case vs. Worst Case; [Beyond Worst-Case Analysis (CS264)] Heuristics vs. Algorithms – SAT Solvers

Special cases/parameters that make a problem easy – Subset Sum with small target, 2SAT, ...

Approximation Algorithms
The Vertex Cover Problem - NP-Complete

vertex cover - set of nodes C that cover all edges:
For all edges, at least one endpoint is in C
Approximating Vertex Cover

Minimization problem: find the smallest VC

A very simple (greedy) approximation algorithm A: finds a VC that is at most twice as large as the optimal (a 2-approximation).

Algorithm: Set $C=\emptyset$ and while there exist uncovered edge $e$, add both endpoints of such $e$ to $C$

Why does it work?
**MAX-SAT**

Max-SAT = given a cnf formula how many clauses can be satisfied? A maximization problem: satisfy the most clauses

Can always satisfy a constant fraction of all the clauses. Specifically:
When all clauses have at least 3 unique literals, can satisfy at least 7/8 of all clauses (how?) \( \geq 7/8 \) of clauses of clauses in optimal solution (\( \Rightarrow \) a 7/8-approximation).

Can we approximate MAX-SAT up to any constant \( < 1 \)? Can we solve Max-3SAT with \((7/8+\text{eps})\)-approximation?
Not if P\( \neq \)NP

For other problems no constant-approximation is likely - (clique n\(^{1-\text{eps}}\))
The PCP Theorem

For some constant $\alpha > 0$ and for every language $L \in \text{NP}$, there exists a polynomial-time computable function $f$ that maps every input $x$ into a 3cnf formula $f(x)$ s.t.

- If $x \in L$ then $f(x) \in \text{SAT}$
- If $x \not\in L$ then no assignment satisfies more than $(1 - \alpha)$ fraction of $f(x)$ clauses.

$\Rightarrow$ sufficiently good approximation of $\text{MAX-SAT}$ implies $P=\text{NP}$ (for tight inapproximability need better PCP theorem)
Hardness of Approximation

A rich literature giving exceedingly sophisticated approximation algorithms and exceedingly sophisticated inapproximability results

Know the best approximation factors for a wide range of problems (especially those where the algorithms are simple)

Inapproximability results via stronger PCPs and via approximation-preserving reductions

$3\text{SAT} \leq_p \text{CLIQUE}$ is (very) approximation-preserving; why?
$\text{IS} \leq_p \text{VERTEX-COVER}$ is completely not; why?
\[(x_1 \lor x_1 \lor x_2) \land (\neg x_1 \lor \neg x_2 \lor \neg x_2) \land (\neg x_1 \lor x_2 \lor x_2)\]

Approx MAX-Clique $\Rightarrow$ Approx MAX-SAT

\[|V| = 9\]

k = 3

MAX-Clique = MAX-SAT
Claim: For every graph \( G = (V,E) \), and subset \( S \subseteq V \), \( S \) is an independent set if and only if \( (V - S) \) is a vertex cover.

Therefore \( (G,k) \in IS \iff (G,|V| - k) \in \text{VERTEX-COVER} \)

Our polynomial time reduction: \( f(G,k) := (G, |V| - k) \)

Assume \( \min \text{VC is } k \ (k \ll n) \) and max IS is \( n-k \). \( c \)-approximation will give IS of size roughly \( n/c \). Giving VC of size \( n-n/c \) - No approximation guarantee for the VC.
PCPs = Probabilistically Checkable Proofs

Alternative (equivalent) statement of PCP Theorem (informal):

Every statement that has a polynomial-time verifiable proof has such a proof where the verifier only reads $O(1)$ bits of the proof such that

[perfect completeness]: if the statement is correct accept with Probability 1
[soundness]: if the statement is false reject with probability 0.99

Example of the power of randomness (probabilistically checkable)
What can we Prove?

Every problem in NP has a short and easy to verify proof

How about coNP? Can a prover P convince a verifier V that there is no k-clique?

How about PSPACE? Can P convince V that there is a winning strategy for white from a particular position?

Yes!! If we add interaction!
Interactive Proofs

PCPs add randomness to proofs, what if we also add interaction?

\[ x \in L? \]

Prover \( P \)  
Verifier \( V \)  

Interactive Proofs can be used to prove membership in powerful (PSPACE) languages. For example: \( V \) knows a winning Chess strategy: \( \text{IP}=\text{PSPACE} \)
Graph Non-Isomorphism

A graph $G$ and $H$ are isomorphic if we can rename vertices of $G$ to get $H$ (the mapping is called isomorphism).

Graph Isomorphism = $\{(G,H) | G$ and $H$ are isomorphic$\}$

Graph Non-Isomorphism = $\{(G,H) | G$ and $H$ are not isomorphic$\}$

Graph Isomorphism in NP but can we prove that $G$ and $H$ are not isomorphic?

We will see a simple interactive proof
Interactive Proof for Graph Non-Isomorphism

Prover $P$

$G_0, G_1$

Verifier $V$

Select a bit $b$ at random; Set $H$ to be a random isomorphic copy of $G_b$

Find $c$ such that $H$ and $G_c$ are isomorphic

Accept iff $c = b$

[perfect completeness]: if $G_0, G_1$ are not isomorphic $V$ accept with Probability 1

[soundness]: if $G_0, G_1$ are isomorphic $V$ accept with Probability $\frac{1}{2}$ (no matter what $P$ does)
Zero-Knowledge (Interactive) Proofs

PCPs add randomness to proofs, what if we also add interaction?

Prover $P$ --- $x \in L$? --- Verifier $V$

Zero-Knowledge Proofs – reveal no information apart of $x \in L$

ZK proofs for all of IP (PSPACE)
IP for Non-Isomorphism is ZK (for semi-honest verifier)

Prover $P$

Verifier $V$

Find $c$ such that $H$ and $G_c$ are isomorphic

Accept iff $c = b$

Select a bit $b$ at random; Set $H$ to be a random isomorphic copy of $G_b$

$G_0, G_1$

[perfect completeness]: if $G_0, G_1$ are not isomorphic $V$ accept with Probability 1

[soundness]: if $G_0, G_1$ are isomorphic $V$ accept with Probability $\frac{1}{2}$ (no matter what $P$ does)
Where is Waldo?
What’s next?

A few possibilities (more to come):

CS161 – Design and Analysis of Algorithms
CS168 - The Modern Algorithmic Toolbox
CS250 – Algebraic Error Correcting Codes
CS254 – Complexity Theory (next quarter)
CS255 - Introduction to Cryptography
CS264 - Beyond Worst Case Analysis
CS296G -Almost Linear Time Graph Algorithms
CS 352 - Pseudorandomness (next year)
CS250 - Error Correcting Codes: Theory and Applications
CS261 - Optimization and Algorithmic Paradigms
CS265 - Randomized Algorithms and Probabilistic Analysis
CS368 - Algorithmic Techniques for Big Data
Parting thoughts:

• Computation is a powerful notion, becoming increasingly central
• Theory allows us to model and analyse computation, reaching non-trivial understanding
• Much is still open – waiting for you
That's all Folks!