Advanced Topics on Proofs
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?) \[\geq \frac{7}{8}\] 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 $x \notin L$ then no assignment satisfies more than $(1 - \alpha)$ fraction of $f(x)$ clauses.

$\Rightarrow$ sufficiently good approximation of MAX-SAT implies P=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

3SAT $\leq_p$ CLIQUE is (very) approximation-preserving; why?
IS $\leq_p$ 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

\(|V| = 9\) \quad \text{and} \quad \(k = 3\)

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 $(G,k) \in IS \iff (G,|V| - k) \in VERTEX-COVER$

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

Assume min 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 $P$ \hspace{5cm} 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 $= \{ (G, H) \mid G$ and $H$ are isomorphic $\}$
Graph Non-Isomorphism $= \{ (G, H) \mid G$ 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 $P$

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

Verifier $V$

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

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

$x \in L$?

Verifier $V$

Accept/Reject

Prover $P$

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

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

Prover $P$

Verifier $V$

$G_0, G_1$

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)
Where is Waldo?
Parting thoughts:
Computational perspective of proofs bare beautiful fruits