1 Today

- We introduce Approximability and Optimization and their relationship with PCP.

- We begin Dinur’s new proof (published two days ago!) of the PCP theorem. The proof is inspired by previous papers by Dinur-Reingold. Recall the PCP theorem: \( \operatorname{NP} \subseteq \operatorname{PCP}[O(\log(n)), O(1)] \).

2 Approximability and Optimization

Consider the optimization problem of coloring a given graph with as few colors as possible. We know this problem is NP-hard (because 3-coloring a 3-colorable graph is NP-hard), but what about nearly-optimal colorings? For example, if we are given that a graph is \( k \)-colorable, is there a way to color the graph with \( k + 1 \) colors in polynomial time? We know we can always color planar graphs with \( k + 1 \) colors (because we have an algorithm for 4-coloring planar graphs) but it is open what we can do with generalized graphs. Approximability is the study of finding such near-optimal solutions.

Formally, a polynomial time algorithm \( A \) approximates a problem to within \( \alpha(\cdot) \) if for any instance \( x \), \( A(x) \) produces a solution whose Cost \( \leq \alpha(n) \text{OPT}(x) \) where \( \text{OPT}(x) \) is the cost of the optimal solution and \( \alpha(n) > 1 \). Alternatively, if we are trying to maximize a quantity, we use Profit \( \geq \text{OPT}(x)/\alpha(n) \).

For example, planar coloring has a 4/3-approximating algorithm. For general coloring, the best known result for 3-colorable graphs is that we can color a 3-colorable graph with \( n^{3/14} \) colors [Blum, Karger].

How would one prove that this is the best result, in other words that there is no polynomial time algorithm that can always color a 3-colorable graph with less than \( n^{3/4} \) colors? What would a hardness reduction look like?

Well, we could give a transformation which maps 3cnf formula \( \phi \) to graph \( G_\phi \) such that

- \( \phi \in \text{SAT} \rightarrow G_\phi \) is 3-colorable
- \( \phi \notin \text{SAT} \rightarrow G_\phi \) is not \( k \)-colorable for \( k < n^{3/14} \)

If we had such a transformation, then we’d know that either we have the best algorithm or \( \text{P} = \text{NP} \). Say we had some better algorithm \( A' \). To see if \( \phi \) is
satisfiable we’d just use the transformation to get $G_\phi$ and then color $G_\phi$ with $A'$ and see if we could do it with less than $n^{3/14}$ colors. If the algorithm worked, we’d know that $\phi$ is satisfiable. We will be using a reduction like this to show the equivalence of the PCP theorem and the optimization problem MAX-SAT.

3 MAX-SAT

We define MAX $k$-SAT-$\Sigma$ to be an optimization problem where $k$ is a positive integer and $\Sigma$ is a finite set (an alphabet). An instance of MAX $k$-SAT-$\Sigma$ consists of variables $x_1, \ldots, x_n$ taking values in $\Sigma$ and constraints $C_1, \ldots, C_m$ where $C_j$ is a constraint on up to $k$ variables. The goal is to find an assignment for $x_1, \ldots, x_n$ that maximizes the number of satisfied constraints. Notice that constraints can be defined arbitrarily (i.e., “truth-table” style) as long as they only depend on at most $k$ variables.

MAX-SAT is NP-hard for most choices of $k$ and $\Sigma$. For example, MAX 2-SAT-$\{0, 1, 2\}$ is as hard as the 3-colorable optimization problem defined above because we can let each vertex be a variable and 0, 1, and 2 be the colors and then translate the graph into a set of constraints (each edge is a not-equal constraint on two variables). Another example, MAX 2-SAT-$\{0, 1\}$ is as hard as the optimization problem MAX-CUT, in which the goal is to partition a graph into two groups so as to maximize the number of edges that are “cut” (edges that connect two vertices in separate groups). Let each vertex correspond to a vertex and each edge connecting $v_i$ and $v_j$ is represented by the constraint $x_i \oplus x_j = 1$.

Dinur proves that $\alpha$-approximating MAX $k$-SAT-$\Sigma$ is NP-hard for satisfiable instances.

4 PCP and MAX-SAT

We will show that the PCP theorem is equivalent to Dinur’s theorem. First we introduce notation:

Given $\psi$, an instance of a MAX-SAT problem, and $\sigma$, an assignment of the variables $x_1, \ldots, x_n$, we say

$$\text{UNSAT}_\sigma(\psi) = \text{fraction of constraints left unsatisfied by } \psi$$

and

$$\text{UNSAT}(\psi) = \min_\sigma \{\text{UNSAT}_\sigma(\psi)\}$$

- Claim 1: If MAX $k$-SAT-$\Sigma$ is hard to approximate within $\alpha$ then NP $\subseteq$ PCP[O(log n), O(1)]. That is, if there exists a transform $T$ which transforms a 3cnf-formula to an instance of MAX-SAT such that

$$\phi \in \text{SAT} \rightarrow \text{UNSAT}(T(\phi)) = 0$$

and

$$\phi \notin \text{SAT} \rightarrow \text{UNSAT}(\phi) \geq 1 - 1/\alpha$$

then NP $\subseteq$ PCP[O(log(n)), O(1)]
• Proof: Verifier transforms NP problem to Max SAT instance by computing 
\[ \psi = T(\phi) \] and expects as proof an assignment to the variables. Verifier 
then picks a random constraint, \( C_j \) in \( \psi \). Notice that if \( \phi \not\in \text{SAT} \) then 
there is a \( 1 - 1/\alpha \) chance that \( C_j \) will not be met. The verifier reads the 
\( k \) elements that \( C_j \) depends on and verifies that this assignment meets 
\( C_j \). Thus it requires \( k \log |\Sigma| \) queries and \( O(\log n) \) random bits. Notice 
if \( \phi \in \text{SAT} \) we accept with probability 1 and if \( \phi \not\in \text{SAT} \) then we accept 
with probability less than \( 1/\alpha \).

• Claim 2: The converse, if \( \text{NP} \subset \text{PCP}[O(\log n), O(1)] \) then \( \alpha \)-approximating 
\( \text{MAX } k-\text{SAT} - \{0,1\} \) is hard.

• Proof: Let \( X_1, \ldots, X_n \) denote the bits of the proof. Let there be \( m = 2^{O(\log n)} \) 
constraints \( C_1, \ldots, C_m \). Notice that \( m \) is the number of distinct random 
strings that could be used by the verifier. Let \( C_j \) denote the condition that leads to acceptance by the verifier on the \( j^{th} \) random 
string. Notice that \( C_j \) depends on only \( q \) variables, where \( q \) is the number 
of queries that the verifier needs. Thus we have an instance of Max 
\( q-\text{SAT} - \{0,1\} \).

5 Dinur’s Theorem

Theorem [Dinur 2005]: For all \( \epsilon > 0 \) there exists a transformation \( T' \) transforming 3cnf formulae to \( \text{MAX } 2-\text{SAT} - \Sigma \) such that 
\[ \phi \in \text{SAT} \rightarrow \text{UNSAT}(\phi) = 0 \]
and 
\[ \phi \not\in \text{SAT} \rightarrow \text{UNSAT}(T'(\phi)) \geq \epsilon \]
Notice by the above claims this is equivalent to the PCP theorem. We prove 
the theorem with a main lemma which is then proved by two sub-lemmas.

6 Main Lemma

For some constant \( \Sigma, \epsilon > 0 \), there exists a transform \( T \) transforming \( \text{MAX } 2-\text{SAT} - \Sigma \) instances to \( \text{MAX } 2-\text{SAT} - \Sigma \) preserving satisfiability such that 
\[ \text{UNSAT}(T(\phi)) \geq \min\{\epsilon, 2\text{UNSAT}(\phi)\} \]
and further \( |T(\phi)| \) is \( O(|\phi|) \).

The theorem follows from this lemma. Start by transforming \( \phi \) to \( T_0(\phi) \), 
and instance of \( \text{MAX } 2-\text{SAT} - \Sigma \) such that \( T_0(\phi) \) is satisfiable if and only if 
\( \phi \) is satisfiable. Now apply \( T \) to \( T_0(\phi) \) a logarithmic number of times so that 
\( \text{UNSAT} \) is sufficiently low and \( |T'(\phi)| \) is \( O(n \log(n)) \).

We prove this main lemma from two sub-lemmas.

7 Lemma 1: Amplification

For every \( \beta \) and \( \Sigma \) there exists a \( l \) and a transform \( T_1 \) from \( \text{MAX } 2-\text{SAT} - \Sigma \) to 
\( \text{MAX } 2-\text{SAT} - \Sigma^l \) preserving satisfiability such that 
\[ \text{UNSAT}(T_1(\phi)) \geq \beta \cdot \text{UNSAT}(\phi) \]
and further \(|T_1(\phi)| = O(|\phi|)\).

Notice that this lemma improves soundness but at the expense of a larger alphabet. We will defer the proof of this lemma until we cover derandomization techniques.

8 Lemma 2: Alphabet Reduction

There exists alphabet \(\Sigma\) and constant \(c\) such that for every finite alphabet \(\Gamma\) there is a transform \(T_2\) from \(\text{MAX } 2-\text{SAT } \Gamma\) to \(\text{MAX } 2-\text{SAT } \Sigma\) preserving satisfiability and such that

\[
\text{UNSAT}(T_2(\phi)) \geq 1/c \cdot \text{UNSAT}(\phi)
\]

Furthermore \(|T_2(\phi)| = O(|\phi|)\). Notice that this lemma decreases soundness but also decreases the alphabet size. We can combine it with lemma 1 to prove the main lemma.

9 Main Lemma from Sub-Lemmas

- Set \(\Sigma\) as in Lemma 2.
- Set \(\beta = 2c\)
- Set \(\Gamma\) to \(\Sigma'\) from Lemma 1.
- Consider the transform \(T = T_2 \circ T_1\). \(T\) is linear sized, polynomial time, and preserves satisfiability. Furthermore,

\[
\text{UNSAT}(T(\phi)) = \text{UNSAT}(T_2(T_1(\phi))) \\
\geq 1/c \cdot \text{UNSAT}(T_1(\phi)) \\
\geq \beta/c \cdot \text{UNSAT}(\phi) \\
= 2 \cdot \text{UNSAT}(\phi)
\]

10 Proof Outline for Lemma 2

We ended with a proof outline of lemma 2, which we will prove next class. We break lemma 2 down into two more lemmas:

- Lemma 2a: There exists constants \(k\) and \(c_a\) such that for every finite \(\Gamma\), there is a transform \(T_{2a}\) from \(\text{MAX } 2-\text{SAT } \Gamma\) to \(\text{MAX } k-\text{SAT } \{0,1\}\) preserving satisfiability such that

\[
\text{UNSAT}(T_{2a}(\phi)) \geq 1/c_a \cdot \text{UNSAT}(\phi)
\]
and further, \(T_{2a}\) is linear.

- Lemma 2b: For every \(k\) there is a transform \(T_{2b}\) \(\text{MAX } k-\text{SAT } \{0,1\}\) to \(\text{MAX } 2-\text{SAT } \{0,1\}^k\) such that

\[
\text{UNSAT}(T_{2b}(\phi)) \geq 1/k \cdot \text{UNSAT}(\phi)
\]
and further, \(T_{2b}\) is linear and preserves satisfiability. The proof of this lemma is analogous to reducing oracle interactive proofs to 2-prover interactive proofs.