A Theorist's Toolkit **(CMU 18-859T, Fall 2013)**

Lecture 25: Sketch of the PCP Theorem

December 4, 2013

Lecturer: Ryan O'Donnell *Scribe: Ben Cousins*

1 Overview

The complexity class NP is the set of all languages $\mathcal{L}$ such that if $x \in \mathcal{L}$, then there exists a polynomial length certificate that $x \in \mathcal{L}$ which can be verified in polynomial time by a deterministic Turing machine (and if $x \notin \mathcal{L}$, no such certificate exists). Suppose we relaxed the requirement that the verifier must be deterministic, and instead the verifier could use randomness to "approximately" check the validity of such a certificate. The notion of a *probabilistically checkable proof* (PCP) arises from this idea, where it can be shown that for any $\mathcal{L} \in NP$, a polynomial length certificate that $x \in \mathcal{L}$ can be verified or disproven, with probability of failure say at most 1/2, by only looking at a constant number (independent of $\mathcal{L}$) of bits in the certificate. More details are given in Section 2, where the existence of such a verifier is implied by the so-called PCP theorem; this theorem was first proven by [2, 3], but here we will cover a proof given by Dinur that is simpler [4]. Some of the more interesting applications of the PCP theorem involve showing hardness of approximability, e.g. it is NP-Hard to approximate Max-3SAT within a 7/8 factor, but we do not discuss such applications here.

2 PCP

We give a statement of the PCP Theorem; for a more complete description, please see [4, 1].

Theorem 2.1. *The complexity class* NP *is equivalent to the class of all languages $\mathcal{L}$ for which, given some polynomial length input x, there exists a polynomial time computable proof system for $\mathcal{L}$ and a verifier such that:*

- *If $x \in \mathcal{L}$, then there is a polynomial length certificate y which proves $x \in \mathcal{L}$, and given y, the verifier always concludes $x \in \mathcal{L}$.*
- *If $x \notin \mathcal{L}$, then for all polynomial length certificates y, the verifier can determine with probability $\geq 1/2$ that $x \notin \mathcal{L}$ by only examining a constant number of random locations in y.*

It is helpful to consider the above description in relation to the standard definition of the class NP, where the verifier V will instead look at all of the bits in y to conclude with certainty whether y implies that $x \in \mathcal{L}$. So, we go from a looking at all of the bits in y to only looking at some constant number, while introducing a probability that we are "fooled" into thinking that $x \in \mathcal{L}$. Theorem 2.1 will be proven through another theorem, for which we give a proof sketch in Section 3. The following definition is helpful for denoting approximation factors for algorithms.

Definition 2.2. We say an algorithm produces an (α, β)-approximation to a problem if for all possible inputs with $\mathsf{OPT} \geq \beta$, the algorithm outputs a valid solution to the problem instance with objective value $\geq \alpha$.

Theorem 2.3. *For all languages $\mathcal{L} \in \mathsf{NP}$, there is a polynomial time deterministic reduction from $\mathcal{L}$ to an instance of 3SAT and an absolute constant $\varepsilon_0 > 0$ independent of $\mathcal{L}$ such that $(1 - \varepsilon_0, 1)$-approximating Max-3SAT on this instance is NP-Hard.*

This theorem may at first seem counterintuitive. For instance, define the language $\mathcal{L} \in \mathsf{NP}$ as the language of all 3SAT instances that have a satisfying assignment. We could have some instance $x \notin \mathcal{L}$ of 3SAT with m clauses and n literals $y = \{y_1, \ldots, y_n\}$ for which there is not a satisfying assignment, but there does exist some assignment $y \in \{0,1\}^n$ such that all clauses but one are satisfied. If, for a proof that $x \in \mathcal{L}$, we simply ask for the values $\{y_1, \ldots, y_n\}$, then even using randomness, we need to examine $O(n)$ locations before we can say that $x \notin \mathcal{L}$ with probability $\geq 1/2$. Alternatively, we could ask for a list of the assignments to each clause, but still we must examine $O(m)$ random locations to conclude $x \notin \mathcal{L}$ with probability $\geq 1/2$.

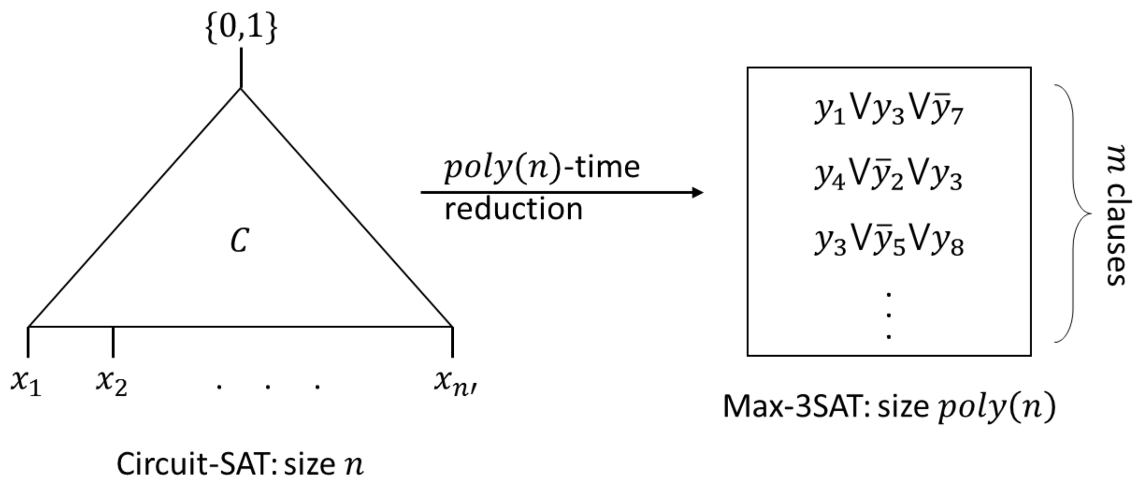

Figure 1: A visualization of a polynomial time reduction from Circuit-SAT to 3SAT.

So how do we get to a constant number of locations for this instance? This is the crucial idea of the theorem, which implies that there is a polynomial time reduction that transforms this 3SAT instance into another instance of 3SAT that has some absolute constant ε_0 clauses that cannot be satisfied. A term that is used to describe this transformation is *gap amplification*, where we increase the unsatisfiability of a problem. More generally, the outline for the proof is: given some language $\mathcal{L} \in \mathsf{NP}$, apply a standard polynomial time reduction to $\mathcal{L}$ to get the NP-Complete Circuit-SAT. We then transform this Circuit-SAT problem into Max-3SAT such that (in the notation of Figure 1):

- If $x \in \mathcal{L}$, then there exists a y satisfying all clauses.
- If $x \notin \mathcal{L}$, then every assignment y does not satisfy at least ε_0 fraction of the clauses, for some absolute constant $\varepsilon_0 > 0$.

The fact that $\varepsilon_0 = 0$ suffices, or even something like $\varepsilon_0 = 1/m$ or $\varepsilon_0 = 1/\mathsf{poly}(n)$, in the above reduction should be apparent since 3SAT is NP-Complete, but the (perhaps surprising) result of the PCP Theorem is that ε_0 can in fact be taken to be a fixed positive constant.

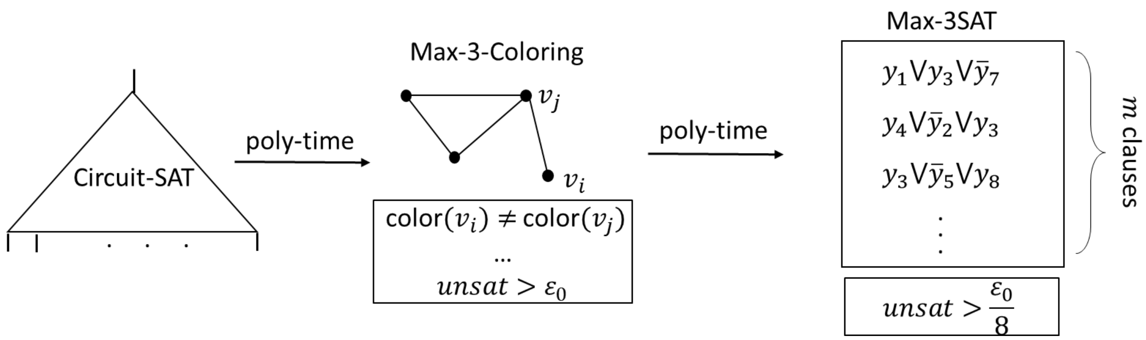

Figure 2: In Section 3, we will actually sketch a proof of Theorem 2.3 for Max-3-Coloring, i.e. there will be some $\varepsilon_0 > 0$ fraction of clauses unsatisfied for the 3-Coloring instance. This result will imply Theorem 2.3 because we can then encode the graph into 3SAT constraints, where the unsatisfiability will be some $\varepsilon_0' > 0$.

Remark 2.4. While Theorem 2.3 was stated for Max-3SAT, it is equivalent to state it for any NP-Hard constraint satisfaction problem (CSP) with constant arity and constant domain size. The absolute constant ε_0 will be different depending on which CSP the theorem is stated for. We will actually prove Theorem 2.3 for Max-3-Coloring, which then implies it for Max-3SAT (Figure 2).

3 Sketch of Dinur's Proof

Definition 3.1. Let G be an instance of some arity-2 CSP. Then define $\mathsf{unsat}(G) :=$ fraction of constraints violated by the best assignment to G.

Theorem 3.2. *There exist constants $\varepsilon_0 > 0, K < \infty$ and a polynomial time reduction R mapping $G \to G'$, where G, G' are arity-2 CSP's with domain size 3, such that*

- $\mathsf{size}(G') \leq K \cdot \mathsf{size}(G)$.

- $\mathsf{unsat}(G') \geq 2 \cdot \mathsf{unsat}(G)$ *if* $\mathsf{unsat}(G) \leq \varepsilon_0$.
- $\mathsf{unsat}(G') = 0$ *if* $\mathsf{unsat}(G) = 0$.

Figure 3 illustrates how we can apply Theorem 3.2 to prove Theorem 2.3. We have our instance of Circuit-SAT, which we reduce in polynomial time to 3-Coloring. Then applying the reduction R from Theorem 3.2 $\log N$ times so that $\mathsf{unsat}(H) > \varepsilon_0$ will imply Theorem 2.3, which then implies the PCP Theorem.

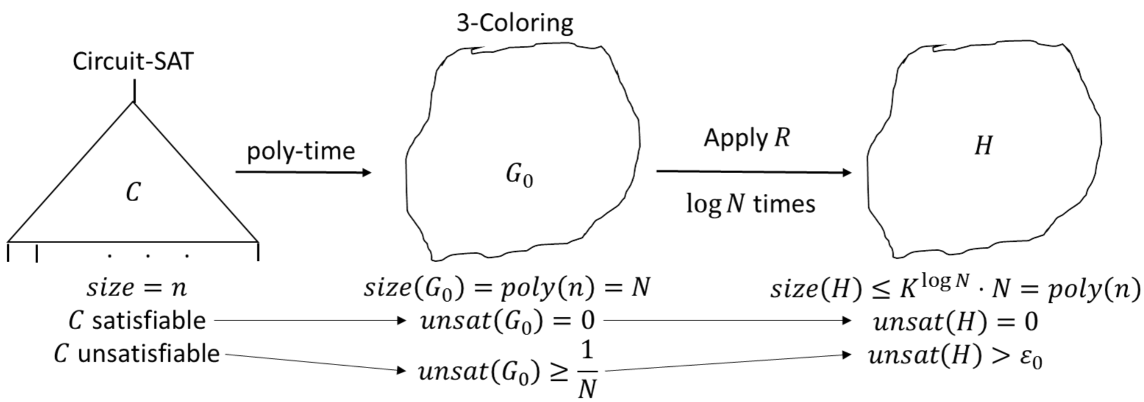

Figure 3: An illustration of how Theorem 3.2 will imply the PCP Theorem.

3.1 Outline of steps

The proof of Theorem 3.2 has 4 main steps, and we outline the effect of each step here. An additional side effect, which is omitted in the below outline, is that for each step $1-4$ is that $\mathsf{unsat}(G) = 0 \Rightarrow \mathsf{unsat}(G') = 0$. That is, if an instance is satisfiable, it remains satisfiable after applying the reduction.

1. **Degree Reduction** (easier)
   - **Main effect**: underlying graph now has degree 4.
   - **Side effects**:
     - $\mathsf{size}(G') \leq K \cdot \mathsf{size}(G)$.
     - $|\Omega'| = 3$, where Ω' is the CSP domain for G'.
     - $\mathsf{unsat}(G') \geq \mathsf{unsat}(G)/C_1$, for some constant $C_1 > 0$.
2. **Expanderizing** (easier)
   - **Main effect**: new underlying graph is a good expander.
   - **Side effects**: same as step 1.

3. **Powering** (innovation, key step)

- **Main effect**: $\mathsf{unsat}(G') \geq t/C_3 \cdot \mathsf{unsat}(G)$, where t is some very large constant (so we have $\mathsf{unsat}(G') \geq \mathsf{unsat}(G)$).
- **Side effects**:
  - $\mathsf{size}(G') \leq 2^{O(t)} \cdot \mathsf{size}(G)$. Note that t was chosen as a constant.
  - $|\Omega'| \approx 3^{8^t}$, so some very large, but constant size, domain.

4. **Mini-PCP** (innovative, but already known)

- **Main effects**:
  - $|\Omega'| = 3$ (the target domain size of Theorem 3.2).
  - $\mathsf{unsat}(G') \geq \mathsf{unsat}(G)/C_4$.
  - $\mathsf{size}(G') \leq 2^{2^{2^{O(t)}}} \cdot \mathsf{size}(G)$ (...still a constant!).

3.2 Proof sketch for each step

Here we will give a sketch of **Step 1-3** in the previous section. For more details in the proof of each step, we refer the reader to lecture notes from a course by Guruswami and O'Donnell [5].

Step 1: Degree Reduction

As shown in Figure 4, the idea of this step is to transform the constraint graph G so that it is 4-regular, with the side effects mentioned in Section 3.1. Note that 3SAT is still NP-Complete when each variable appears in a constant number of constraints.

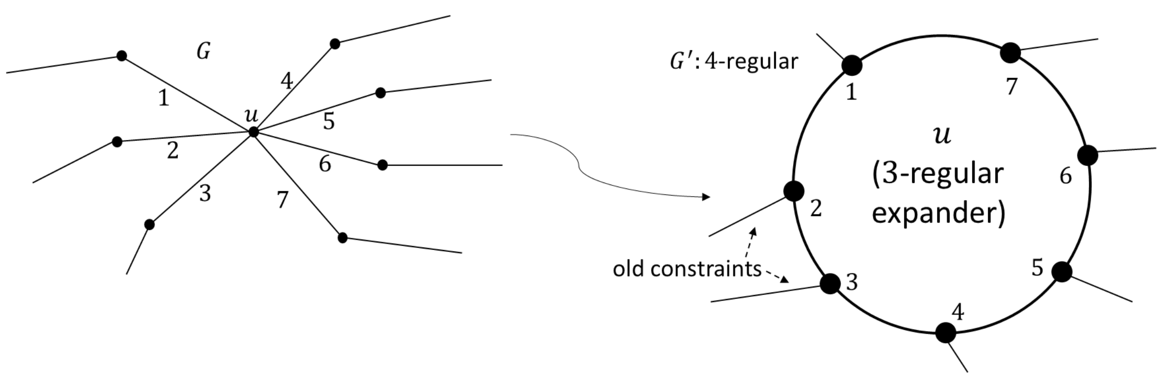

Figure 4: An illustration of the transformation of G to a new graph G' that is 4-regular. We take each node in $u \in G$, along with its neighbors in G, and transform it into a 3-regular expander graph in G'.

We now give a sketch of why $\mathsf{unsat}(G') \geq \mathsf{unsat}(G)/C_1$, for some constant $C_1 > 0$. Let A' be the best assignment in G'. By defintion, we know that A' violated $\geq \mathsf{unsat}(G') =: \varepsilon$

fraction of the contraints in G'. We now construct the assignment A for G in the following way:

- Assign A for G by letting $A(u) =$ plurality vote of $\{A'(\text{cloud for } u)\}$, where the cloud for u was the 3-regular expander in Figure 4.
- We know that A violates $\geq \mathsf{unsat}(G)$ fraction of constraints in G.

For each constraint (u, v) in G (i.e. an "inter-cloud" constraint in G'), either A' violates it in G', or u' or v' has a minority label in its cloud. This then implies that $\Omega(\mathsf{unsat}(G))$ intercloud edges in G' are violated, or that $\Omega(\mathsf{unsat}(G))$ intracloud edges in G' are violated. The intracloud edges conclusion follows from each cloud being an expander.

Step 2: Expanderizing

We want the underlying graph that resulted from **Step 1** to have good expansion. We add this for "free" by adding constraints to G' that are always satisfied, but improve the expansion, and only a constant overhead of constraints will be added to the graph. Specifically, here we add some 3-regular expander, and use the fact that combining two regular graphs gives a good expander, when only one of the graphs has good expansion. We allow for repeat edges when combining the two graphs, so the new graph G' will be a multigraph. After adding the edges, the resulting graph will be 7-regular.

Step 3: Powering(t)

The input to this step is a graph G with arity-2 and domain size $|\Omega| = 3$ that is 7-regular and a good expander. We will output a graph G' that satisfies the following properties:

- Same variable/vertex set as in G.
- Introduce edges/constraints for each lazy path of length $\approx t$ in G, i.e. a random walk with length $\mathsf{Geom}(1/t)$.
- The degree of each vertex in G is $\approx 8^t$.
- The size of the new domain is: $|\Omega'| = |\Omega|^{1+8+8^2+\ldots+8^t} \approx 3^{8^t}$.

Consider the case of $t = 3$. Note we are keeping the variable/vertex set the same in G', but introduce different contraints. We will think of a labeling in G' as assigning the colors (R, G, B) to all nodes in G that are $\leq t$ distance away from the vertex. For a node $u \in G$ and a node $v \in G$ that is within distance t of u, we denote the label (R, G, B) that u assigns to v as u's "opinion" of v (Figure 5). A constraint in G' will be to "test everything"; this step is where the large blowup in degree and domain size comes from. For an example, we pick two vertices from Figure 5: a and f. Then, for any $a - f$ path in G, we constrain all of a's opinions match all of f's opinions for vertices along that $a - f$ path. Specifically, for any

$a - f$ path P in G and two vertices $u, v \in P$ such that (u, v) is a constraint in G, then we add a constraint in G' so that a's opinion of u and f's opinion of v satisfy the (u, v) constraint in G.

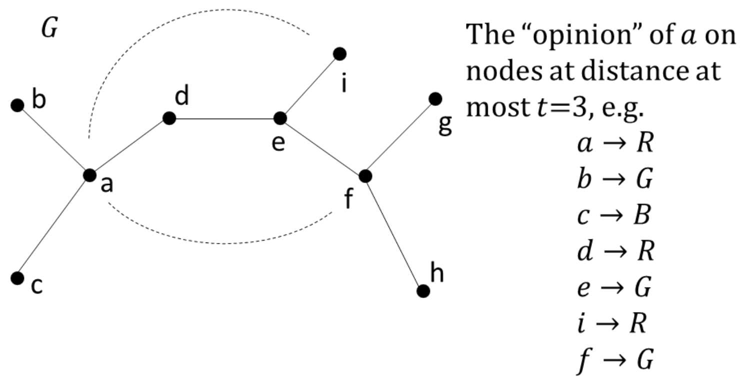

Figure 5: In the above graph G, the nodes a, b, c, d, e, i, and f are all within distance $t = 3$ of a, and we consider some "opinion" assigned to each of these nodes by a. We then turn pairwise opinions into constraints in G', using the constraints in the original graph G.

Even though it may seem we are adding much too many constraints, it will only give a constant size increase in size (remember t is a constant and G was 7-regular). The domain size will also increase by a constant factor. Recalling the effects mentioned in Section 3.1 for this step:

- **Problem Size**: $\mathsf{size}(G') \leq 2^{O(t)} \cdot \mathsf{size}(G)$ will follow from the above procedure.
- **Domain Size**: $|\Omega'| \leq 3^{8^t} \cdot |\Omega|$ also follows from the above.
- **Satisfiability**: $\mathsf{unsat}(G') \geq t/C_3 \cdot \mathsf{unsat}(G)$ still needs to be shown.

We now give an overview of the proof of unsatisfiability. Let A' be the best assignment for G'. Write $A'(w)_v \in \Omega = \{R, G, B\}$ for w's opinion of v under A'. Then, define an assignment A for G from a plurality vote with the opinions in A':

$$A(v) = \text{plurality}\{A'(w)_v | w \text{ is distance at most } \approx t \text{ from } v\}.$$

We have that A violates $\geq \varepsilon := \min\{\mathsf{unsat}(G), \varepsilon_0 := 1/t\}$ fraction of constraints in G, which just follows from the definition of $\mathsf{unsat}(G)$. Let F be the constraints in G that the assignment A violates.

Goal: Show that A' violates an $\Omega(t) \cdot \varepsilon$ fraction of the path constraints in G'.

Let $a \to f$ be a random path/constraint in G', and suppose this path passes through some $(u, v) \in F$. There is $\geq 1/9$ chance that $A'(a)_u = A(u)$ and $A'(f)_v = A(v)$, since A was chosen by the plurailty vote of opinions and $a \to f$ was a random path. If this happens, then A' violates the $a \to f$ constaint in G'.

It remains to argue that whatever F is, there is at least an $\Omega(t) \cdot \varepsilon$ chance that a random-length $\approx t$ path in G hits F. This will be true because G is an expander.

References

[1] Sanjeev Arora and Boaz Barak. *Computational complexity*. Cambridge University Press, Cambridge, 2009.

[2] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[3] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP. *J. ACM*, 45(1):70–122, 1998.

[4] Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):Art. 12, 44, 2007.

[5] Venkatesan Guruswami and Ryan O'Donnell. The PCP theorem and hardness of approximation. 2005. http://courses.cs.washington.edu/courses/cse533/05au/.