6.841: Advanced Complexity Theory Fall 2012

Lecture 19 — November 15, 2012

Prof. Dana Moshkovitz *Scribe: Badih Ghazi*

1 Overview

In the last lecture, we saw the graph powering method, which allows us to increase the soundness gap but has the drawback of increasing the alphabet size from $|\Sigma|$ to $|\Sigma|^{d^t}$ (where $|\Sigma|$ is a constant). In this lecture, we describe how to reduce the alphabet size (from $|\Sigma|^{d^t}$ back to $|\Sigma|$), while causing only a small decrease in the gap (and a constant factor increase in the constraint graph size). Therefore, overall, the soundness gap has been amplified. This step is called "composition" in the PCP literature.

2 Alphabet Size Reduction

2.1 Notation

The constraint graph after powering but before composition is denoted by $G = (V, E, \Sigma^{d^t}, \Phi)$ (where d and t are constants). Φ is the set of edge constraints in G, i.e. $\Phi = \{\varphi_e \mid e \in E\}$ where for every $e \in E$, φ_e is the constraint associated with edge e in G. Let $\sigma : V \to \Sigma^{d^t}$ denote an assignment of alphabet values to the vertices of G (which is provided by the prover and may not be a valid assignment). The graph after the composition step is denoted by $G' = (V', E', \Sigma, \Phi')$.

2.2 Overview of the transformation

The main idea of the transformation taking G as input and producing G' as output is the following. For every vertex $v \in V$, G' contains a collection $\{(v, i)\}_i$ of vertices corresponding to v. Each assignment of values in Σ^{d^t} to v will induce an assignment of values in Σ to $\{(v, i)\}_i$.
For every edge $e = (u, v) \in E$, G' contains a collection $\{(e, i)\}_i$ of vertices (that will be assigned values in Σ) corresponding to e. G' also contains edges between $\{(u, i)\}_i$, $\{(v, i)\}_i$ and $\{(e, i)\}_i$. Those edges, along with the corresponding constraints in Φ', will be specified at the end of Section 2.4. The particular form of those assignments will be specified after introducing "long codes", in Section 2.3.

The edges and constraints between $\{(u, i)\}_i$, $\{(v, i)\}_i$ and $\{(e, i)\}_i$ will satisfy the following two properties:

1. Assume that $\sigma(u), \sigma(v) \in \Sigma^{d^t}$ satisfy edge e and that $\sigma(u)$ and $\sigma(v)$ induce some assignments of values in Σ to $\{(u, i)\}_i$ and $\{(v, i)\}_i$ respectively. Then, there exist assignments of values in Σ to $\{(e, i)\}_i$ that satisfy the corresponding constraints in G'.

2. Let $\delta, \delta' > 0$ be given constants and let $e = (u, v) \in E$. For any assignment of values in Σ to $\{(u, i)\}_i$, $\{(v, i)\}_i$ and $\{(e, i)\}_i$ that satisfies a $(1 - \delta)$ fraction of the corresponing constraints in G', there exist unique assignments $\sigma(u), \sigma(v) \in \Sigma^{d^t}$ satisfying the edge e and s.t. a $(1 - \delta')$ fraction of the $\{(u, i)\}_i$'s have their assignments induced by $\sigma(u)$ and a $(1 - \delta')$ fraction of the $\{(v, i)\}_i$'s have their assignments induced by $\sigma(v)$.

Lemma 1. *There exists a constraint graph $G' = (V', E', \Sigma, \Phi')$ with $|V'| + |E'| \leq c(|V| + |E|)$ (where c is a constant that depends on $|\Sigma|$, d and t) and constants $\delta, \delta' > 0$ that satisfy properties 1 and 2 above for every edge $e \in E$.*

Lemma 1 implies composition, by noting that the new alphabet is Σ and the completeness and soundness parameters are essentially maintained.

Remarks

1. The subgraph of G' (corresponding to a particular edge e of G) is by itself a PCP! That PCP (which we call the *inner* PCP) is for deciding whether a particular assignment to $\{(u, i)\}_i$, $\{(v, i)\}_i$ corresponds to a satisfying assignment for the edge e in G. The latter is a stronger notion than the usual PCP (deciding whether *there exists* a satisfying assignment). Nonetheless, many PCP constructions also yield this stronger notion (called "assignment testing" or "PCP of proximity").
2. $|\Sigma^{d^t}|$ is a constant. In particular, the constant c in the statement of Lemma 1 will be exponential in $|\Sigma^{d^t}|$.
3. It is enough to prove the existence of a 3-hypergraph[1] satisfying the above properties with an alphabet of $\{\pm 1\}$. Given such a 3-hypergraph, we can construct a graph satisfying the above properties by introducing a new vertex per hyperedge and assigning values in $\{\pm 1\}^3$ to these new vertices (see pset 5 regarding transforming a 3-query PCP to a 2-query PCP).

2.3 Long codes

First, we define the concept of a coordinate function (also known as a "dictator").

Definition 2. *Let $W \in \mathbb{N}$. For every $w \in \{1, .., W\}$, the coordinate function $f_w : \{\pm 1\}^W \to \{\pm 1\}$ is defined by $f(x_1, .., x_W) = x_w$ for all $(x_1, .., x_W) \in \{\pm 1\}^W$.*

Next, we define "long codes" which were introduced by Bellare, Goldreich and Sudan in [BGS95].

Definition 3. *Let $W \in \mathbb{N}$. A long code is a coding scheme where the message $w \in [W]$ is encoded by the table of the coordinate function $f_w : \{\pm 1\}^W \to \{\pm 1\}$.*

The fact that the size of a codeword (represented by 2^W bits) is doubly exponential in the size of the message (represented by $\log W$ bits) gives rise to the name "long code". Note that a long code encoder can be equivalently described as assigning to each $w \in [W]$ the concatenation of the values of all functions $f : W \to \{\pm 1\}$ on input w.

[1] A 3-hypergraph is a generalization of a graph where an edge connects either 2 or 3 vertices.

Convention
Note that each codeword in a long code is inherently "balanced" in the sense that it consists of 2^{W-1} 1's and 2^{W-1} -1's. This is due to the fact that every coordinate function f_w has the property that $f_w(-v) = -f_w(v)$ for all $v \in \{\pm 1\}^W$. Thus, in a codeword, a function and its negation can be represented by 1 bit instead of 2 bits, thereby reducing the length of a codeword by a factor of 2. We represent the supposed long code codewords this way, ensuring they are balanced. This is called "folding" in the PCP literature.

We now give an useful property of long codes.

Lemma 4. *Let $W \in \mathbb{N}$. Define $LC_1 : \{\pm 1\}^{\log W} \to \{\pm 1\}^{2^W}$ to be the long code encoder with message length $\log W$ and block length 2^W. Also, define $LC_2 : \{\pm 1\}^{\log W} \times \{\pm 1\}^{\log W} \to \{\pm 1\}^{2^{W^2}}$ to be the long code encoder with message length $2 \log W$ and block length 2^{W^2}. Then, $LC_2(u, v)$ "contains" $LC_1(u)$ and $LC_1(v)$ for all $u, v \in \{\pm 1\}^{\log W}$. More precisely, there exist two one-to-one functions $h : [2^W] \to [2^{W^2}]$ and $r : [2^W] \to [2^{W^2}]$ s.t. for all $u, v \in \{\pm 1\}^{\log W}$ and all $i \in [2^W]$, $\big(LC_1(u)\big)_i = \big(LC_2(u, v)\big)_{h(i)}$ and $\big(LC_1(v)\big)_i = \big(LC_2(u, v)\big)_{r(i)}$.*

Noise stability
We say that a function is noise stable if flipping at random a small fraction of the input bits typically causes no change to the output. For general Boolean functions, it can be seen that the constant functions are the most noise stable. However, if we consider the set of all "balanced" Boolean functions, it turns out that dictators are the most noise stable. This property will be useful in designing constraints that identify long code codewords.

Assignment form
We will use a long code with parameter $W = |\Sigma^{d^t}|$ (where $\Sigma = \{\pm 1\}$). For every $v \in V$, the assignments $\{(v, i)\}_i$ are supposed to be the bits of the codeword $L_1(\sigma(v))$. For every $e = (u, v) \in E$, the assignments to $\{(e, i)\}_i$ are supposed to be the bits of the codeword $L_2(\sigma(u), \sigma(v))$, where $\sigma(u)$, $\sigma(v)$ are satisfying assignments for e. Moreover, for every $e = (u, v) \in E$, G' we have a set of hyperedges each connecting 3 vertices in $\{(u, i)\}_i \cup \{(v, i)\}_i \cup \{(e, i)\}_i$, and Φ' contains a constraint associated with each such hyperedge.

2.4 Hastad's test

Notation
Let $\sigma : V \to \Sigma^{d^t}$ denote the assignment of alphabet values to the vertices of the initial constraint graph G and φ_e be the constraint corresponding to the edge $e = (u, v)$ in G. Also, let A be the assignment in the new constraint graph G'.

We will now prove Hastad's test (introdcuced in [Hås01]), show that it has perfect completeness and then complete the construction of the graph G'.

Description of Hastad's test
The NP verifier V_H picks $f : \Sigma^{d^t} \times \Sigma^{d^t} \to \{\pm 1\}$ and $g : \Sigma^{d^t} \to \{\pm 1\}$ uniformly and independently at random. It then sets $g' : \Sigma^{d^t} \times \Sigma^{d^t} \to \{\pm 1\}$ as follows: With probability $1/2$, $g'(\sigma_1, \sigma_2) = g(\sigma_1)$ for all $\sigma_1, \sigma_2 \in \Sigma^{d^t}$ and with probability $1/2$, $g'(\sigma_1, \sigma_2) = g(\sigma_2)$ for all $\sigma_1, \sigma_2 \in \Sigma^{d^t}$. Then, V_H chooses

$\mu : \Sigma^{d^t} \times \Sigma^{d^t} \to \{\pm 1\}$ as follows: For all $\sigma_1, \sigma_2 \in \Sigma^{d^t}$, we distinguish 2 cases. If $f(\sigma_1, \sigma_2) = 1$, then $\mu(\sigma_1, \sigma_2) = -1$. If $f(\sigma_1, \sigma_2) = -1$, then:

$$\mu(\sigma_1, \sigma_2) = \begin{cases} 1 & \text{with probability } 1 - \tau. \\ -1 & \text{with probability } \tau. \end{cases}$$

(where $\tau > 0$ is some parameter). We view φ_e as a function mapping $\Sigma^{d^t} \times \Sigma^{d^t}$ to $\{\pm 1\}$ where the constraint associated with edge e in G is satisfied if and only if φ_e has value -1. Let $f \wedge \varphi_e$ denote the ± 1 version of the usual $\wedge$, i.e. for all $x \in \Sigma^{d^t} \times \Sigma^{d^t}$, $(f \wedge \varphi_e)(x) = -1$ if and only if $f(x) = -1$ and $\varphi_e(x) = -1$. Moreover:

1. $A(f \wedge \varphi_e)$ denotes the $\{\pm 1\}$ value in the assignments $\{(e, i)\}_i$ corresponding to the function $f \wedge \varphi_e$.
2. If $g'(\sigma_1, \sigma_2)$ is set to $g(\sigma_1)$ for all $\sigma_1, \sigma_2 \in \Sigma^{d^t}$, then $A(g)$ is the $\{\pm 1\}$ value in the assignments $\{(u, i)\}_i$ corresponding to the function g. On the other hand, if $g'(\sigma_1, \sigma_2)$ is set to $g(\sigma_2)$ for all $\sigma_1, \sigma_2 \in \Sigma^{d^t}$, then $A(g)$ is the $\{\pm 1\}$ value in the assignments $\{(v, i)\}_i$ corresponding to the function g.
3. $A(g' \cdot \mu)$ (where $g' \cdot \mu$ is the coordinatewise product of g' and μ) denotes the $\{\pm 1\}$ value in the assignments $\{(e, i)\}_i$ corresponding to the function $g' \cdot \mu$.

Then, V_H rejects if $A(f \wedge \varphi_e) = A(g) = A(g' \cdot \mu) = 1$ and accepts otherwise.

Completeness
We now show that Hastad's test has perfect completeness.

Lemma 5. *If $\sigma(u)$ and $\sigma(v)$ satisfy the constraint on edge e in G (i.e. if $\varphi_e(\sigma(u), \sigma(v)) = -1$) and if A is the long code assignment described at the end of Section 2.3, then V_H accepts.*

Proof. We proceed by contradiction. Assume that $\varphi_e(\sigma(u), \sigma(v)) = -1$ and $A(f \wedge \varphi_e) = A(g) = A(g' \cdot \mu) = 1$. Since A is the long code assignment, it follows that $(f \wedge \varphi_e)(\sigma(u), \sigma(v)) = 1$ which implies that $f(\sigma(u), \sigma(v)) = 1$ (since $\varphi_e(\sigma(u), \sigma(v)) = -1$). By the definition of μ, we then have that $\mu(\sigma(u), \sigma(v)) = -1$. Moreover, $A(g) = 1$ implies that $g'(\sigma(u), \sigma(v)) = 1$ thereby contradicting the fact that $A(g' \cdot \mu) = 1$. □

Soundness
We will prove the soundness of Hastad's test in the next lecture.

Construction of G'
We now complete the construction of G' (started in Section 2.2) by specifying the edges and associated constraints between $\{(u, i)\}_i$, $\{(v, i)\}_i$ and $\{(e, i)\}_i$ where $e = (u, v)$ is an edge of G.
For every possible sequence of coin tosses that can be obtained by V_H, we add a 3-hyperedge and an associated constraint to the graph G'. More precisely, for every possible choice of the functions f, g, g' and μ that can be obtained in a run of Hastad's test, we add a hyperedge connecting $(e, f \wedge \varphi_e)$, $(e, g' \cdot \mu)$ and either (u, g) or (v, g) depending on the outcome of the coin toss for g'. The constraint corresponding to this hyperedge is the negation of the constraint $A(f \wedge \varphi_e) = A(g) = A(g' \cdot \mu) = 1$.

For more details, please see [Din07] and [AB09].

In the next lecture, we will prove the soundness of Hastad's test, which will complete the proof of Lemma 1.

References

[AB09] S. Arora and B. Barak. *Computational complexity: a modern approach*, volume 1. Cambridge University Press Cambridge, UK, 2009.

[BGS95] M. Bellare, O. Goldreich, and M. Sudan. Free bits, pcps and non-approximability-towards tight results. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 422–431. IEEE, 1995.

[Din07] I. Dinur. The pcp theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.

[Hås01] J. Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.