Last Class
- End Gap Amplification
- High level view of Composition

Today
- Formal Definition of AT = Assignment Tester
- Composition Theorem

1 Composition

We saw informally in the previous lecture that the composition step was recursion with a twist: the “inner” the verifier needed to perform the following “assignment testing” task: Given an assignment $a$ to the variables of a constraint $\Phi$, check if $a$ is close to a satisfying assignment of $\Phi$. The formal definition follows. We say that two strings $x, y$ are $\delta$-far from each other if they differ on at least $\delta$ fraction of coordinates.

**Definition 1.1 (Assignment Tester).** A $q$-query Assignment Tester $AT(\gamma > 0, \Sigma_0)$ is a reduction algorithm $P$ whose input is a Boolean circuit $\Phi$ over Boolean variables $X$ and $P$ outputs a system of constraints $\Psi$ over $X$ and set $Y$ of auxiliary variables such that

- Variables in $Y$ take values in $\Sigma_0$
- Each $\psi \in \Psi$ depends on at most $q$ variables in $X \cup Y$
- $\forall a : X \rightarrow \{0, 1\}$
  - If $\phi(a) = 1$, then $\exists b : Y \rightarrow \Sigma_0$ such that $a \cup b$ satisfies all $\psi \in \Psi$
  - If assignment ‘a’ is $\delta$-far from every a’ such that $\phi(a') = 1$ then $\forall b : Y \rightarrow \Sigma_0$, atleast $\gamma_0 \delta$ fraction $\psi \in \Psi$ are violated by $a \cup b$

We first observe that the above definition is stronger than a regular PCP reduction:

- $\Phi$ satisfiable $\implies \exists a \cup b$ that satisfy all constraints in $\Psi$
• $\Phi$ not satisfiable $\implies$ every assignment $a : X \to \{0, 1\}$ is 1-far from any satisfying assignment (since no satisfying assignment exists!), and hence for every $b : Y \to \Sigma_0$, $a \cup b$ violates $\Omega(1)$ fraction of constraints in $\Psi$. In particular, every assignment $a \cup b$ to the variables of $\Psi$ violates $\Omega(1)$ fraction of the constraints, like in a PCP reduction.

**Theorem 1.2 (Composition Theorem).** Assume existence of 2-query Assignment tester $P(\gamma > 0, \Sigma_0)$. Then $\exists \beta > 0$ (dependent only on $P$ and $\text{poly}(\text{size}(G))$) such that any constraint graph $(G, \mathcal{C})_\Sigma$ can transformed in time polynomial in the size of $G$ into a constraint graph $(G', \mathcal{C}')_{\Sigma_0}$ denoted by $G \circ P$ such that

- $\text{size}(G') \leq O(1)\text{size}(G)$
- $\text{gap}(G) = 0 \implies \text{gap}(G') = 0$
- $\text{gap}(G') \geq \beta \cdot \text{gap}(G)$

**Proof.** The basic idea is to apply the AT $P$ to each of the constraints $c \in \mathcal{C}$ and then define the new constraint graph $G'$ based on the output of $P$. Since the AT expects as input a constraint over Boolean variables, we need to first express the constraints of $G$ with Boolean inputs. For this, we encode the elements of $\Sigma$ as a binary string.

The trivial encoding uses $\log(|\Sigma|)$ bits. Instead we will use an error correcting code $e : \Sigma \to \{0, 1\}$ where $l = O(\log |\Sigma|)$ of relative distance $\rho = 1/4$, i.e., with the following property:

$$\forall x, y, x \neq y \implies e(x) \text{ is } \rho\text{-far from } e(y)$$

i.e. $x \neq y \implies \Delta(e(x), e(y)) \geq \rho \cdot l$

Let $[u]$ denote the block of Boolean variables supposed to represent the bits of the encoding of $u$’s label. For a constraint $c \in \mathcal{C}$ on variables $u, v$ of $G$, define the constraint $\tilde{c}$ on the $2l$ Boolean variables $[u] \cup [v]$ as follows:

$$\tilde{c}(a, b) = 1 \text{ iff } \exists \alpha, \alpha' \in \Sigma \text{ such that the following hold: }$$

$$e(\alpha) = a$$
$$e(\alpha') = b$$
$$c(\alpha, \alpha') = 1.$$

Let $\tilde{c} : \{0, 1\}^{2l} \to \{0, 1\}$ be regarded as a Boolean circuit and fed to a 2-query AT. The output is a list of constraints $\Psi_c$ which can regarded as a constraint graph over $\Sigma_0$, call it $(G_c = (V_c, E_c), \mathcal{C}_c)$ (where $[u] \cup [v] \subseteq V_c$), with the two variables in each constraint taking the place of vertices in the constraint graph. To get the new constraint graph $G'$, we will paste together such constraint graphs $G_c$ obtained by applying the 2-query AT to each of the constraints of the original constraint graph.

Formally, the new constraint graph is $(G' = (V', E'), \mathcal{C}')$ over $\Sigma_0$ where

- $V' = \cup_{c \in \mathcal{C}} V_c$
- $E' = \cup_{c \in \mathcal{C}} E_c$
• $C' = \bigcup_{c \in C} C_c$

We will assume wlog that $|E_c|$ is the same for every $c \in C$ (this can be achieved by duplicating edges if necessary).

We will now prove that the graph $G'$ obtained by the reduction above satisfies the requirements of the Composition Theorem (??). Clearly, since the size of $G'$ is at most a constant times larger than that of $G$, since each edge in $G$ is replaced by the output of the assignment tester on a constant-sized constraint, and thus by a graph of constant (depending on $|\Sigma|$) size. Also, $G'$ can be produced in time polynomial in the size of $G$.

The claim that $\text{gap}(G') = 0$ when $\text{gap}(G) = 0$ is also obvious – beginning with a satisfying assignment $\sigma : V \rightarrow \Sigma$, we can label the variables in $[u]$ for each $u \in V$ with $e(\sigma(u))$, and label the auxiliary variables introduced by the assignment testers $P$ in a manner that satisfies all the constraints (as guaranteed the property of the assignment tester when the input assignment satisfies the constraint).

It remains to prove that $\text{gap}(G') \geq \beta \cdot \text{gap}(G)$ for some $\beta > 0$ depending only on the AT.

Let $\sigma' : V' \rightarrow \Sigma_0$ be an arbitrary assignment. We want to show that $\sigma'$ violates at least a fraction $\beta \cdot \text{gap}(G)$ of the constraints in $C'$. First we extract an assignment $\sigma : V \rightarrow \Sigma$ from $\sigma'$ as follows: $\sigma(u) = \arg \min_a \Delta(\sigma'(\{u\}), e(a))$, i.e., we pick the closest codeword to the label to the block of variables (here we assume without loss of generality that $\sigma'$ assigns values in $\{0,1\}$ to variables in $[u]$ for all $u \in V$).

We know that $\sigma$ violates at a fraction $\text{gap}(G)$ of constraints in $C$. Let $c = c_e \in C$ be such a violated constraint where $e = (u, v)$. We will prove that at least a $\gamma \cdot \rho / 4$ fraction of the constraints of $G_c$ are violated by $\sigma'$. Since the edge sets $E_c$ all have the same size for various $c \in C$, it follows that $\sigma'$ violates at least a fraction $\gamma \cdot \rho / 4 \text{gap}(G)$ of constraints of $G'$. This will prove the composition theorem with the choice $\beta = \gamma \cdot \rho / 4$.

By the property of the assignment tester $P$, to prove at least $\gamma \cdot \rho / 4$ of the constraints of $G_c$ are violated by $\sigma'$, it suffices to prove the following.

Claim: $\sigma'(\{u\} \cup \{v\})$ is at least $\rho / 4$-far from any satisfying assignment to $\tilde{c}$.

Proof of Claim: Let $(\sigma''(\{u\}), \sigma''(\{v\}))$ be a satisfying assignment for $\tilde{c}$ that is closest to $\sigma'(\{u\} \cup \{v\})$. Any satisfying assignment to $\tilde{c}$ must consist of codewords of the error-correcting code $e$. Therefore, let $\sigma''(\{u\}) = e(a)$ and $\sigma''(\{v\}) = e(b)$. Moreover, $c(a, b) = 1$. Since $\sigma$ violates $c$, we have $c(\sigma(u), \sigma(v)) = 0$. It follows that either $a \neq \sigma(u)$ or $b \neq \sigma(v)$, let us say the former for definiteness. We have

$$\rho \leq \Delta(e(a), e(\sigma(u))) \leq \Delta(e(a), \sigma'(\{u\})) + \Delta(\sigma'(\{u\}), e(\sigma(u))) \leq 2\Delta(e(a), \sigma'(\{u\}))$$

where the last step follows since $e(\sigma(u))$ is the codeword closest to $\sigma'(\{u\})$. Recalling, $e(a) = \sigma''(\{u\})$, we find that at least a $\rho / 2$ fraction of the positions $\sigma'(\{u\})$ must be changed to obtain a satisfying assignment to $\tilde{c}$. It follows that $\sigma'(\{u\} \cup \{v\})$ is at least $\rho / 4$-far from any satisfying assignment to $\tilde{c}$.

This completes the proof of the claim, and hence also that of Theorem 1.2. \qed

The composition theorem needed a 2-query AT. We now show that bringing down the number of queries to 2 is easy once we have a $q$-query AT for some constant $q$. 

3
Lemma 1.3. Given a q-query Assignment Tester AT over $\Sigma_0 = \{0, 1\}, \gamma_0 > 0$, it is possible to construct a 2-query AT over alphabet $\Sigma'_0 = \{0, 1\}^q$ and $\gamma'_0 = \frac{2\gamma_0}{q}$.

Proof. Let the q-query Assignment Tester AT be on Boolean variables $X \cup Y$ (where $Y$ is the set of auxiliary variables), with set of constraints $\Psi$. Define 2-query AT as follows. The auxiliary variables are $Y \cup Z$ where $Z = \{z_{\psi} | \psi \in \Psi\}$ is a set of variables over the alphabet $\Sigma'_0 = \{0, 1\}^q$, and the set of constraints $\Psi'$ include for each $\psi \in \Psi$ a set of $q$ constraints on two variables: $(z_{\psi}, v_1), (z_{\psi}, v_2), \ldots, (z_{\psi}, v_q)$ where $v_1, v_2, \ldots, v_q$ are the variables on which $\psi$ depends (if $\psi$ depends on fewer than $q$ variables, we just repeat one of them enough times to make the number $q$). The constraint $(z_{\psi}, v_i)$ is satisfied by $(a, b)$ if $\psi$ is satisfied by $a$ and $b$ is consistent with $a$ on the value given to $v_i$.

Clearly, if all constraints in $\Psi$ can be satisified by an assignment to $X \cup Y$, then it can be extended in the obvious way to $Z$ to satisfy all the new constraints. Also, if $a : X \rightarrow \{0, 1\}$ is $\delta$-far from satisfying the input circuit $\Phi$ to the AT, then for every $b : Y \rightarrow \{0, 1\}$, at least $\gamma_0 \delta$ fraction of $\psi \in \Psi$ are violated. For each such $\psi$, for any assignment $c : Z \rightarrow \{0, 1\}^q$, at least one of the $q$ constraints that involve $z_{\psi}$ must reject. Thus, at least a fraction $\frac{2\gamma_0}{q}$ of the new constraints reject. \qed

Later on, we will give a 6-query AT over the Boolean alphabet. By the above, this also implies a 2-query AT over the alphabet $\{1, 2, \ldots, 64\}$.

2 Linearity Testing

We will now take a break from PCPs and do a self-contained interlude on “linearity testing”. Consider a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ as a table of values, the question we now consider is ”Is $f$ linear ?”. Such questions are part of a larger body of research called property testing. First, we define what we mean by a linear function.

Definition 2.1. (Linear functions) A function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is linear if $\exists S \subset \{1, 2, \ldots, n\}$ such that $f(x) = \bigoplus_{i \in S} x_i$. Or equivalently, $f$ is linear if there exists $a \in \{0, 1\}^n$ such that $f(x) = \bigoplus_{i=1}^n a_ix_i$.

Fact 2.2. The following two statements are equivalent:

- $f$ is linear
- $\forall x, y : f(x + y) = f(x) + f(y)$

For $S \subset \{1, 2, \ldots, n\}$, define $L_S : \{0, 1\}^n \rightarrow \{0, 1\}$ as $L_S(x) = \bigoplus_{i \in S} x_i$. Say $L_s(X) = \sum_{i \in S} X_i$. Given access to the truth table of a function $f$, linearity testing tries to distinguish between the following cases, using very few probes into the truth table of $f$:

- $f = L_S$ for some $S$
- $f$ is “far-off” from $L_S$ for every $S$
A randomized procedure for Linearity Testing uses $2.2$ above. Instead of testing whether $f(x + y) = f(x) + f(y)$ for every pair $x, y$, we pick one pair $(x, y)$ at random and apply the following test: Is $f(x + y) = f(x) + f(y)$? Thus we look at the value of $f$ on only 3 places. We will explore actual guarantees that this test provides in the next lecture, and go on to connect this with the proof of the PCP theorem.