

Expander Graphs and Their Applications (X)

Yijia Chen
Shanghai Jiaotong University

Review of the Previous Lecture

Probabilistic Verifier

Definition

A verifier V is a *probabilistic polynomial time algorithm* with access to an input $x \in \Sigma^*$ and a string $\tau \in \Sigma^*$ of (internal) *random binary bits*. Furthermore, V has access to a proof $\pi \in \Sigma^*$.

V will either accept or reject the input x , depending on (x, τ, π) .

We require that V is non-adaptive, i.e., it first reads the input x and the random bits τ , and then decides which positions in the proof π it wants to query. That is, the positions V queries do not depend on the answers that V got from *previous queries*.

The result of V 's computation on x , τ and π is denoted by $V(x, \tau, \pi)$. As usual 1 means *accepting*, 0 for *rejecting*.

Probabilistic Verifier (cont'd)

1. V is **polynomial time** in $|x|$.
2. The query positions depend on and only on x and τ (non-adaptive), so we might view the query positions as a function

$$p_V : \Sigma^* \times \Sigma^* \rightarrow \mathbb{N}^*,$$

such that $p_V(x, \tau)$ is computable in time polynomial in $|x|$. We may assume p_V is given *explicitly* with the verifier V .

3. The run of $V(x, \tau, \pi)$ that does not involve π might also depend on the random string τ .
4. We can divide the computation of $V(x, \tau, \pi)$ into 3 parts:
 - (i) Compute from x and τ a sequence of positions $\bar{p} = p_1 p_2 \dots p_\ell = p_V(x, \tau)$.
 - (ii) Read the letters on the positions \bar{p} of π to form a string

$$\pi \upharpoonright \bar{p} := \pi(p_1)\pi(p_2)\dots\pi(p_\ell),$$

where $\underline{\pi(p)}$ denotes the letter in the p -th position of π .

- (iii) Compute an answer from x , τ , and $\pi \upharpoonright \bar{p}$.

$(r(n), q(n))$ -Restricted Verifier

Definition

Let $r, q : \mathbb{N} \rightarrow \mathbb{N}$ be two **monotone** functions.

An $(r(n), q(n))$ -restricted verifier is a verifier that for inputs of length n uses at most $O(r(n))$ random bits and queries at most $O(q(n))$ bits from the proof.

1. If V is $(r(n), q(n))$ -restricted, then for any input $x \in \Sigma^*$, Then $|\tau| = O(r(n))$ and $|p_V(x, \tau)| = O(q(n))$.
2. There is no restriction on the length of the proof π .

PCP Classes

Definition

Let $r, q : \mathbb{N} \rightarrow \mathbb{N}$ be two monotone functions.

The class $\text{PCP}(r(n), q(n))$ consists of all languages Q where there exists an $(r(n), q(n))$ -restricted verifier V such that for all x

$$x \in Q \iff \exists \pi \Pr_{\tau}[V(x, \tau, \pi) = 1] = 1,$$

$$x \notin Q \iff \forall \pi \Pr_{\tau}[V(x, \tau, \pi) = 1] < \frac{1}{2}.$$

Here in $\Pr_{\tau}[\dots]$ the probability is taken over all random strings τ of length $O(r(n))$.

Remark. $1/2$ in the above definition can be replaced by any $0 < \varepsilon < 1$.

Definition

Let $R, Q \subseteq \mathbb{N} \rightarrow \mathbb{N}$ be two classes of monotone functions.

$$\underline{\text{PCP}}(R, Q) := \bigcup_{r \in R, q \in Q} \text{PCP}(r(n), q(n)).$$

Let poly denote the class of *all polynomials over* \mathbb{N} , i.e. $\mathbb{N}[x]$.

Theorem

$$\text{PCP}(\text{poly}, 0) = \text{coRP},$$

$$\text{PCP}(0, \text{poly}) = \text{NP}.$$

MAX- d SAT

Let $d \in \mathbb{N}$.

MAX- d SAT

Input: An $\alpha \in d\text{CNF}$.

Solution: An assignment \mathcal{V} for α .

Cost: the number of clauses in α that \mathcal{V} satisfies.

Goal: max.

Definition

For every $d\text{CNF}$ -formula α and an assignment \mathcal{V} for α ,

$\text{sat}(\alpha, \mathcal{V})$:= the number of clauses in α that \mathcal{V} satisfies

and

$\text{optsat}(\alpha)$:= $\max_{\mathcal{V}} \text{sat}(\alpha, \mathcal{V})$.

Hardness of Approximating MAX- d SAT

Definition

Let \mathbb{A} be an algorithm that on every d CNF-formula α always output an assignment for α which we denote by \mathcal{V}_α . The performance ratio of \mathbb{A} is

$$\max_{\alpha \in d\text{CNF}} \frac{\text{optsat}(\alpha)}{\text{sat}(\alpha, \mathcal{V}_\alpha)}.$$

Theorem

*For some constants $d \in \mathbb{N}$ and $r > 1$, there is **no** polynomial time approximation algorithm for MAX- d SAT with performance ratio r , unless $P = NP$.*

Proof.

Let $Q \subseteq \Sigma^*$ be an NP-complete language. By the PCP Theorem, there is a $(\log n, 1)$ -restricted verifier V accepting Q : for $x \in \Sigma^*$, V uses $c \cdot \log |x|$ random bits and makes d oracle queries for two constants $c, d \in \mathbb{N}$.

$$(R1) \quad x \in Q \iff \exists \pi \Pr_{\tau \in \{0,1\}^{c \cdot \log |x|}} [V(x, \tau, \pi) = 1] = 1,$$

$$(R2) \quad x \notin Q \iff \forall \pi \Pr_{\tau \in \{0,1\}^{c \cdot \log |x|}} [V(x, \tau, \pi) = 1] < \frac{1}{2}.$$

Proof. (cont'd)

We define the algorithm $\mathbb{A}(x, \tau, a_1, a_2, \dots, a_d)$:

1. $\bar{p} = p_1 \dots p_d \leftarrow p_V(x, \tau)$.
2. Simulate $V(x, \pi)$ for an “imaginary” π by replacing each π_{p_i} by a_i for $1 \leq i \leq d$.

where $\tau \in \{0, 1\}^{c \cdot \log |x|}$, and $a_1, a_2, \dots, a_d \in \{0, 1\}$.

Clearly \mathbb{A} is polynomial time in $|x|$.

Proof. (cont'd)

We reduce Q to MAX- d SAT.

Given an $x \in \Sigma^*$, we will construct a d CNF formula α_x satisfying

$x \in Q \Rightarrow \alpha_x$ is satisfiable,

$x \notin Q \Rightarrow$ for any assignment \mathcal{V} ,
at least 2^{-d-1} -fraction of clauses are not satisfied.

First let

$$P_x := \{p \mid \text{for some } \tau \in \{0, 1\}^{c \cdot \log |x|} \text{ } p \text{ appears in the sequence } p_V(x, \tau)\},$$

i.e. P_x is the set of positions that V makes queries.

Obviously P_x is computable in time polynomial in $|x|$. In addition

$$|P_x| \leq d \cdot |x|^c.$$

For each $p \in P_x$, we introduce a propositional variable X_p . Then, every proof π can be viewed as an assignment \mathcal{V}_π with

$$\mathcal{V}_\pi(X_p) := \pi(p).$$

Proof. (cont'd)

For every $\tau \in \{0, 1\}^{c \cdot \log |x|}$, we define a d CNF formula $\alpha_{x, \tau}$: For every $a_1, \dots, a_d \in \{0, 1\}$, and $\bar{p} = p_1 \dots p_d = p_V(x, \tau)$, if $\mathbb{A}(x, \tau, a_1, \dots, a_d)$ *rejects*, then $\alpha_{x, \tau}$ contains the clause

$$T_1 \vee T_2 \vee \dots \vee T_d$$

where

$$T_i := \begin{cases} X_{p_i} & \text{if } a_i = 0, \\ \neg X_{p_i} & \text{if } a_i = 1. \end{cases}$$

$\alpha_{x, \tau}$ has *at most 2^d clauses*. More importantly: Let π be a proof with corresponding assignment \mathcal{V}_π

- (F1) if $V(x, \tau, \pi)$ accepts, then \mathcal{V}_π satisfies $\alpha_{x, \tau}$,
- (F2) if $V(x, \tau, \pi)$ rejects,
then \mathcal{V}_π makes at least *one* of the clauses in $\alpha_{x, \tau}$ *false*.

Proof. (cont'd)

Let

$$\alpha_x := \bigwedge_{\tau \in \{0,1\}^{c \cdot \log |x|}} \alpha_{x,\tau}$$

α_x has *at most* $|x|^c \cdot 2^d$ clauses. We need to show:

$x \in Q \Rightarrow \alpha_x$ is satisfiable,

$x \notin Q \Rightarrow$ for any assignment \mathcal{V} ,

at least 2^{-d-1} -fraction of clauses are not satisfied.

Proof. (cont'd)

Let $x \in Q$. By

$$(R1) \quad x \in Q \iff \exists \pi \Pr_{\tau \in \{0,1\}^{c \cdot \log |x|}} [V(x, \tau, \pi) = 1] = 1,$$

we have a proof π such that $V(x, \tau, \pi)$ always accepts for any $\tau \in \{0, 1\}^{c \cdot \log |x|}$.

Thus \mathcal{V}_π satisfies all $\alpha_{x,\tau}$ by

$$(F1) \quad \text{if } V(x, \tau, \pi) \text{ accepts, then } \mathcal{V}_\pi \text{ satisfies } \alpha_{x,\tau},$$

and hence α_x , i.e.,

$$x \in Q \Rightarrow \alpha_x \text{ is satisfiable.}$$

Proof. (cont'd)

Let $x \notin Q$. For any assignment \mathcal{V} of α_x , it is easy to see we can view it as some \mathcal{V}_π for an appropriate proof π .

The by

$$(R2) \quad x \notin Q \iff \forall \pi \Pr_{\tau \in \{0,1\}^{c \cdot \log |x|}} [V(x, \tau, \pi) = 1] < \frac{1}{2}.$$

we have for *more than half* of all possible $\tau \in \{0,1\}^{c \cdot \log |x|}$, $V(x, \tau, \pi)$ rejects. For any τ with $V(x, \tau, \pi)$ rejecting, then

$$(F2) \quad \text{if } V(x, \tau, \pi) \text{ rejects,} \\ \text{then } \mathcal{V}_\pi \text{ makes at least } \textit{one} \text{ of the clauses in } \alpha_{x,\tau} \textit{ false.}$$

implies that *at least one clause* in $\alpha_{x,\tau}$ is false under $\mathcal{V}_\pi = \mathcal{V}$.

Recall $\alpha_{x,\tau}$ contains at most 2^d many clauses. Thus in total we have *more than*

$$2^{-1} \cdot 2^{-d} = 2^{-d-1}$$

fraction of clauses in α_x are not satisfied under \mathcal{V} .

Proof. (cont'd)

Let

$$r := \frac{1}{1 - 2^{-d-1}}.$$

We claim MAX- d SAT *cannot* have a polynomial time approximation algorithm with **performance ratio** r .

Assume \mathbb{B} is such an algorithm. We would use it to decide Q :

Let $x \in \Sigma^*$. We compute the d CNF-formula α_x as described above. Assume α_x has **m clauses**.

If $x \in Q$, then α_x is satisfiable. Thus \mathbb{B} would find an assignment that satisfies at least

$$\frac{m}{r} = m \cdot (1 - 2^{-d-1})$$

clauses of α_x .

Proof. (cont'd)

If $x \notin Q$, then

$$m \cdot (1 - 2^{-d-1})$$

many clauses of α_x can be satisfied by any assignment. In particular the assignment that \mathbb{B} outputs on α_x cannot satisfy $m \cdot (1 - 2^{-d-1})$ many clauses.

In summary:

$x \in Q \Rightarrow \mathbb{B}(\alpha_x)$ satisfies $\geq m \cdot (1 - 2^{-d-1})$ clauses;

$x \notin Q \Rightarrow \mathbb{B}(\alpha_x)$ satisfies $< m \cdot (1 - 2^{-d-1})$ clauses.

□