Lecture 4: Introduction to Hardness of Approximation

In this lecture, we introduce some of the tools that will enable us to prove strong statements about hardness of approximation in the next lecture.

1 PCPP builders and hardness of approximation

During the last lecture, we introduced the concept of probabilistically checkable proofs of proximity (PCPP). To recall:

Definition 1.1 *A property $\mathcal{P}$ of m-bit strings has PCPPs of length $l(m)$ if there exists a non-adaptive $O(1)$-query tester T with access to a string $w \in \{-1,1\}^m$ and a proof $\pi \in \{-1,1\}^{l(m)}$ such that:*

1. *If $w \in \mathcal{P}$, then $\exists \pi$ such that $\mathbf{Pr}[T(w,\pi)$ accepts$] = 1$, and*
2. *If w is ϵ-far from $\mathcal{P}$, then $\forall \pi$ $\mathbf{Pr}[T(w,\pi)$ accepts$] \leq 1 - \Omega(\epsilon)$.*

After introducing PCPPs, we saw that every property $\mathcal{P}$ of m-bit strings has PCPPs of length 2^{2^m}. While this may not look so bad given that there are 2^{2^m} possible properties, we will see in this section how we can generate shorter PCPPs when we restrict our attention to properties that are decided by polynomial-size circuits. Furthermore, unlike the existential proof that was presented in the last lecture, our proof of the strengthened result provides an explicit construction for the desired PCPPs. This proof will rely on *PCPP builders*.

Definition 1.2 *A poly(m)-size circuit C over m bits has a* length-$l(m)$ PCPP builder *if there is a poly($l(m)$)-time algorithm that given C outputs a tester T_C for a length-$l(m)$ PCPP for the property C. Here we are identifying a circuit C with the property of being a string satisfying C.*

Before examining some results on PCPP builders, we also introduce the notion of weighted constraint satisfaction problems.

Definition 1.3 *A* weighted constraint satisfaction problem *(CSP) over variables $v_1, \ldots, v_N$ is a table containing predicates $\phi_1, \phi_2, \ldots$ that act on those variables and nonnegative weights $p_1, p_2, \ldots$ associated with each predicate, such that $\sum_i p_i = 1$.*

The following table shows an example of a possible CSP:

Lecture 4: Introduction to Hardness of Approximation

weight	constraints
p_1	$\phi_1(v_1, v_6, v_{10})$
p_2	$\phi_2(v_7, v_{10}, v_{20})$
p_3	$\phi_3(v_4, v_{19})$
$\vdots$	$\vdots$

An important algorithmic problem related to CSPs is the following: given an instance of a CSP, find the assignment of values to the variables $v_1, \ldots, v_N$ that maximizes the sum of the weights whose corresponding predicates are satisfied.

We can also construct a CSP to correspond to any nonadaptive tester T_C. In this case, we let the variables represent the possible characters in the string(s) to be tested. The constraints in the CSP are the possible predicates that the tester may check on its input, with each weight representing the probability that the tester runs the associated predicate.

Theorem 1.4 *There is a PCPP builder of length ...*

(a) 2^{2^m} *(see Theorem 2.3 in Lecture 4; the proof is constructive)*

(b) $2^{\mathrm{poly}(m)}$ *(based on BLR tests)*

(c) $\mathrm{poly}(m)$ *(The PCP Theorem$^+$)*

(d) $m \cdot \mathrm{polylog}(m)$ *(Dinur, Ben-Sasson & Sudan; requires size(C)* $= m \cdot \mathrm{polylog}(m)$*)*

The major result (c) is obtained using the results from (a) (or (b)). The basic idea is that the construction for (c) is recursive, and the base case of the recursion involves building PCPPs for properties of constant-length strings. Thus we can use (a) or (b), since we don't mind blowing up one constant to another. The proof of (a) constitutes perhaps a quarter to a third of Dinur's proof of the PCP theorem.

Fact 1.5 *The following is on the homework: If there exists a length-*$l(m)$ *PCPP builder, then there exists a length-*$\mathrm{poly}(l(m))$ *PCPP builder where all tests are OR*$_3$ *predicates (with negations); i.e., look like* $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ *or* $\overline{x}_{i_1} \vee x_{i_2} \vee x_{i_3}$ *or* $\overline{x}_{i_1} \vee \overline{x}_{i_2} \vee \overline{x}_{i_3}$ *etc.*

The PCP Theorem has important implications for hardness of approximation results. To present these results precisely, we first define the notion of *value* of a CSP.

Definition 1.6 *Given a CSP* $\mathcal{C}$*, we define* $val(\mathcal{C})$ *to be the maximum total weight that can be satisfied by any assignment.*

The notion of value can be used to rephrase standard hardness results.

Observation 1.7 *"3-SAT is NP-hard" is equivalent to "Given* $\mathcal{C}$ *where all constraints are OR*$_3$*, it is NP-hard to distinguish* $val(\mathcal{C}) = 1$ *and* $val(\mathcal{C}) < 1$*."*

Furthermore, the notion of value of CSPs also extends naturally to hardness of approximation results.

Lecture 4: Introduction to Hardness of Approximation

Proposition 1.8 *The PCP Theorem$^+$ implies: "Given $\mathcal{C}$ where all the constraints are OR_3, it is NP-hard to distinguish $val(\mathcal{C}) = 1$ and $val(\mathcal{C}) < 1 - \epsilon$ for some explicit $\epsilon > 0$." In particular, there is no PTAS for Max-3SAT.*

Proof: The PCP Theorem$^+$ implies that there is a $\mathrm{poly}(m)$-time reduction which, given a circuit C of polynomial size outputs a tester T_C over variables $w_1, \ldots, w_m$ and $\pi_1, \ldots, \pi_{\mathrm{poly}(m)}$, where T_C uses OR_3 predicates such that:

1. If w satisfies C then there exists π such that $\mathbf{Pr}[T_C(w, \pi)$ accepts$] = 1$, and
2. For every w ϵ-far from satisfying C, $\forall \pi$ $\mathbf{Pr}[T_C(w, \pi)$ accepts$] \leq 1 - \Omega(\epsilon)$.

In particular, if C is satisfiable, the theorem implies that $\exists w, \pi$ such that $\mathbf{Pr}[T_C(w, \pi)$ accepts$] = 1$. On the other hand, if C is not satisfiable then in fact $\forall w$, w is 1-far from satisfying C; hence $\forall w, \forall \pi$, $\mathbf{Pr}[T_C(w, \pi)$ accepts$] \leq 1 - \Omega(1)$.

This shows that

- C is satisfiable $\Rightarrow val(T_C) = 1$
- C is not satisfiable $\Rightarrow val(T_C) \leq 1 - \Omega(1)$.

But Circuit-Satisfiability is NP-hard, so this completes the proof of Proposition 1.8. □

2 Testing averages

Proposition 1.8 shows that it is NP-hard to distinguish between 3CNFs that are satisfiable and 3CNFs that are at most $(1 - \epsilon)$-satisfiable for some ϵ, but this ϵ may be very small. We now would like to get stronger results that improve those bounds. To reach that goal, we will to give stronger testing algorithms.

Definition 2.1 (Testing averages) *Given a function tester T for functions $\{-1,1\}^n \to \{-1,1\}$, "applying T to a collection $\{f_1, \ldots, f_d\}$" means: run T, but whenever it queries x, it first picks $i \in [d]$ at random and uses $f_i(x)$.*

Definition 2.2 *The* average *of $\{f_1, \ldots, f_d\}$ is the function $h : \{-1,1\}^n \to [-1,1]$, where $h(x) = \mathbf{E}_i[f_i(x)]$.*

Using the linearity of the Fourier transform, it's easy to verify that $\hat{h}(S) = \mathbf{E}_i[\hat{f}_i(S)]$.

Proposition 2.3 *Given a Fourier-coefficient formula for $\mathbf{Pr}[T(f)$ accepts$]$, the same formula holds for $\mathbf{Pr}[T(\{f_1, \ldots, f_d\})$ accepts$]$ when you use h's coefficients instead.*

For example, consider the Håstad$_\delta$ test:

Lecture 4: Introduction to Hardness of Approximation

Example 2.4 $\mathbf{Pr}[\textit{Håstad}_\delta(f)\ \textit{accepts}] = \frac{1}{2} + \frac{1}{2}\sum_S (1-2\delta)^{|S|}\hat{f}(S)^3$

$$\Rightarrow \mathbf{Pr}[\textit{Håstad}_\delta(\{f_1,\ldots,f_d\})\ \textit{accepts}] = \frac{1}{2} + \frac{1}{2}\sum_S (1-2\delta)^{|S|}\hat{h}(S)^3.$$

The proof of Proposition 2.3 uses a derivation very similar to the one we used to prove Håstad's test:

Proof: (Sketch.)

$$\begin{aligned}
\mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{j},\boldsymbol{k},\boldsymbol{l}}[\tfrac{1}{2} + \tfrac{1}{2}f_{\boldsymbol{j}}(\boldsymbol{x})f_{\boldsymbol{k}}(\boldsymbol{y})f_{\boldsymbol{l}}(\boldsymbol{z})] &= \mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}[\mathop{\mathbf{E}}_{\boldsymbol{j},\boldsymbol{k},\boldsymbol{l}}[\cdots]] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}[\tfrac{1}{2} + \tfrac{1}{2}\mathop{\mathbf{E}}_{\boldsymbol{j},\boldsymbol{k},\boldsymbol{l}}[f_{\boldsymbol{j}}(\boldsymbol{x})f_{\boldsymbol{k}}(\boldsymbol{y})f_{\boldsymbol{l}}(\boldsymbol{z})]] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}[\tfrac{1}{2} + \tfrac{1}{2}\mathop{\mathbf{E}}_{\boldsymbol{j}}[f_{\boldsymbol{j}}(\boldsymbol{x})]\mathop{\mathbf{E}}_{\boldsymbol{k}}[f_{\boldsymbol{k}}(\boldsymbol{y})]\mathop{\mathbf{E}}_{l}[f_{\boldsymbol{l}}(\boldsymbol{z})]] \\
&= \mathop{\mathbf{E}}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}[\tfrac{1}{2} + \tfrac{1}{2}h(\boldsymbol{x})h(\boldsymbol{y})h(\boldsymbol{z})],
\end{aligned}$$

where we used linearity of expectation and the independence of $\boldsymbol{j},\boldsymbol{k},\boldsymbol{l}$. The rest of the proof follows the proof of Håstad's test identically. □

Proposition 2.3 actually holds for all tests because the acceptance predicate, being a function on $\{-1,1\}^q$, can be expressed as a multilinear formula (the Fourier expansion!) and this multilinearity is all that we used in the proof sketch above.

Definition 2.5 *Given $f : \{-1,1\}^n \to \{-1,1\}$, define $f^\dagger$ by $f^\dagger(x) = -f(-x)$, and $f^{odd} : \{-1,1\}^n \to [-1,1]$ by $f^{odd} = (f + f^\dagger)/2$.*

Fact 2.6 *As we saw in Question 5 of the first homework, the Fourier expansion of f^{odd} is*

$$f^{odd} = \sum_{S:|S|\ odd} \hat{f}(S)\chi_S.$$

Allowing for a bad pun, we define the Håst-Odd$_\delta$ test on boolean functions f:

Definition 2.7 *The Håst-Odd$_\delta$ test is defined to be to be*

$$\textit{Håst-Odd}_\delta(f) := \textit{Håstad}_\delta(\{f, f^\dagger\}).$$

Remark 2.8 *Note that the Håst-Odd$_\delta$ can be applied to f using only an oracle for f; this oracle can be used to "simulate" queries to $f^\dagger$.*

Applying Proposition 2.3, we get a formula for evaluating the probability that the Håst-Odd$_\delta$ test accepts a given function.

Corollary 2.9 $\mathbf{Pr}[\textit{Håst-Odd}_\delta(f)\ \textit{accepts}] = \frac{1}{2} + \frac{1}{2}\sum_{|S|\ odd}(1-2\delta)^{|S|}\hat{f}(S)^3$.

One interesting property of the Håst-Odd$_\delta$ test is that it provides a third way to fix the dictator test introduced in Lecture 2, since this test accepts constant functions only with probability $1/2$.

As an aside: the Håst-Odd$_\delta$ test is called "folding" the Håstad$_\delta$ test in the literature, but we will not use that terminology in these lectures.

Lecture 4: Introduction to Hardness of Approximation

3 Influence of variables

In the first lecture, we alluded to the influence of coordinates on functions. Let us now define the term formally.

Definition 3.1 *For $f : \{-1,1\}^n \to \mathbb{R}$, $i \in [n]$, the* influence of i on f *is*

$$\mathrm{Inf}_i(f) = \sum_{\substack{S \subseteq [n] \\ s.t.\ i \in S}} \hat{f}(S)^2.$$

Notation 3.2 *From now on we will abbreviate that summation as "$\sum_{S \ni i}$".*

In Lecture 1 we gave an alternative definition for influences in reference to boolean-valued functions $f : \{-1,1\}^n \to \{-1,1\}$, namely: $\mathrm{Inf}_i(f) = \mathbf{Pr}[f(\boldsymbol{x}) \neq f(\boldsymbol{x}^{(i)})]$. As we show in the next proposition, the definition is equivalent in this special case:

Proposition 3.3 *For any boolean-valued $f : \{-1,1\}^n \to \{-1,1\}$,*

$$\mathrm{Inf}_i(f) = \mathop{\mathbf{Pr}}_{\boldsymbol{x}}[f(\boldsymbol{x}) \neq f(\boldsymbol{x}^{(i)})].$$

Proof: Briefly,

$$\begin{aligned}
\mathbf{Pr}[f(\boldsymbol{x}) \neq f(\boldsymbol{x}^{(i)})] &= \mathbf{E}[\tfrac{1}{2} - \tfrac{1}{2} f(\boldsymbol{x}) f(\boldsymbol{x}^{(i)})] \\
&= \tfrac{1}{2} - \tfrac{1}{2}\,\mathbf{E}[f(\boldsymbol{x}) f(\boldsymbol{x}^{(i)})] \\
&= \tfrac{1}{2} - \tfrac{1}{2} \sum_{S,T} \hat{f}(S)\hat{f}(T)\,\mathbf{E}[\boldsymbol{x}_S (\boldsymbol{x}^{(i)})_T] \\
&= \tfrac{1}{2} - \tfrac{1}{2} \sum_{S,T} \hat{f}(S)\hat{f}(T)\,\mathbf{E}[\boldsymbol{x}_{S \triangle T} \cdot (-1)^{\mathbf{1}_{i \in T}}]
\end{aligned}$$

Now $(-1)^{\mathbf{1}_{i \in T}}$ can be pulled out of the sum, and $\mathbf{E}[x_{S \triangle T}] = 1$ if $S = T$, 0 otherwise. Hence:

$$\mathbf{Pr}[f(\boldsymbol{x}) \neq f(\boldsymbol{x}^{(i)})] = \tfrac{1}{2} - \tfrac{1}{2} \sum_{S} \hat{f}(S)^2 \cdot (-1)^{\mathbf{1}_{i \in S}}$$

By Parseval's Theorem, we can replace the first $\frac{1}{2}$ above with $\frac{1}{2} \sum_S \hat{f}(S)^2$; hence:

$$\begin{aligned}
\mathbf{Pr}[f(\boldsymbol{x}) \neq f(\boldsymbol{x}^{(i)})] &= \tfrac{1}{2} \left(\sum_S \hat{f}(S)^2 - \sum_S \hat{f}(S)^2 \cdot (-1)^{\mathbf{1}_{i \in S}} \right) \\
&= \sum_{S \ni i} \hat{f}(S)^2.
\end{aligned}$$

□

For some functions, such as the parity function, all variables have high influence. However, when this is the case we might prefer to say that none of the variables are influential. Toward this goal, we define the concept of *attenuated influence*:

Lecture 4: Introduction to Hardness of Approximation

Definition 3.4 *The* ρ-attenuated influence *of* i *on* f*, for* $0 \leq \rho \leq 1$*, is*

$$\mathrm{Inf}_i^{(\rho)}(f) = \sum_{S \ni i} \rho^{|S|-1} \hat{f}(S)^2.$$

One reason for the slightly strange -1 in the exponent of ρ is that we want it to still be the case that the ith coordinate has ρ-attenuated influence 1 on the ith dictator function, for any ρ. (See Fact 3.5(c) below.) We will also see a somewhat natural combinatorial definition for attenuated influence on the homework that agrees with the above Fourier definition. Also note that we will usually apply this definition with values of ρ close to 1, in which case we will write $\rho = 1 - \delta$.

Fact 3.5 *The following fundamental facts about the attenuated influence of variables can be checked easily:*

(a) $\mathrm{Inf}_i(f) = \mathrm{Inf}_i^{(1)}(f)$.

(b) *In general,* $\mathrm{Inf}_i^{(\rho)}(f)$ *is an increasing function with* ρ.

(c) $\mathrm{Inf}_i^{(\rho)}(\chi_i) = 1$.

(d) $\mathrm{Inf}_i^{(\rho)}(\chi_{[n]}) = \rho^{n-1}$.

One important property of the definition of attenuated influence is that any function has only a constant number of coordinates with large attenuated influence, as we show in the following proposition.

Proposition 3.6 *Let* $f : \{-1,1\}^n \to [-1,1]$ *and let* $\mathcal{I}_{\epsilon,\delta} = \{i : \mathrm{Inf}_i^{(1-\delta)}(f) \geq \epsilon\}$. *Then* $\sum_{i=1}^n \mathrm{Inf}_i^{(1-\delta)}(f) \leq \frac{1}{\delta}$*, and hence* $|\mathcal{I}_{\epsilon,\delta}| \leq \frac{1}{\epsilon\delta}$.

Proof: For the proof of this proposition, we require the following claim.

Claim 3.7 *For any values of* $|S| \in [0,n]$ *and* $0 < \delta < 1$*,*

$$|S|(1-\delta)^{|S|-1} \leq \frac{1}{\delta}.$$

Let us begin by proving the proposition, assuming that Claim 3.7 holds.

$$\begin{aligned}
\sum_{i=1}^n \mathrm{Inf}_i^{(\rho)}(f) &= \sum_{i=1}^n \sum_{S \ni i} (1-\delta)^{|S|-1} \hat{f}(S)^2 \\
&= \sum_{S \ni i} |S|(1-\delta)^{|S|-1} \hat{f}(S)^2 \qquad \text{(each } S \text{ is counted } |S| \text{ times)} \\
&\leq \sum_S (\frac{1}{\delta}) \cdot \hat{f}(S)^2 \qquad \text{(by Claim 3.7)} \\
&= \frac{1}{\delta} \cdot \mathbf{E}[f(x)^2] \leq \frac{1}{\delta}.
\end{aligned}$$

To complete the proof, we now need to prove Claim 3.7. A nice proof of this claim was provided in class by Daniel Golovin: Since $(1-\delta) < 1$, then $(1-\delta)^{|S|-1} \leq (1-\delta)^{i-1}$ for every $i \leq |S|$. So

$$|S|(1-\delta)^{|S|-1} \leq \sum_{i=0}^{|S|-1}(1-\delta)^i \leq \sum_{i=0}^{\infty}(1-\delta)^i = \frac{1}{\delta}.$$

This completes the proof of the claim and of the proposition. □

4 Quasirandomness

With the notion of influence of variables, we are now ready to formally define another term that was presented in the first lecture: quasirandomness.

Definition 4.1 *Given $f : \{-1,1\}^n \to [-1,1]$, we say that f is* (ϵ,δ)-quasirandom *if for all $i \in [n]$,*

$$\mathrm{Inf}_i^{(1-\delta)}(f) < \epsilon.$$

In other words, f is (ϵ,δ)-quasirandom if $|\mathcal{I}_{\epsilon,\delta}| = \emptyset$.

One reason we call such functions "quasirandom" is that any such function is close to being uncorrelated with any function on a small number of bits; this is reminiscent of the related notion for graphs. See Homework #2 for details.

Example 4.2 *Many functions we have seen so far satisfy the definition of quasirandomness.*

Function	*Quasirandom?*
Dictator	*no*
Majority	*yes*
Parity	*yes*
Random	*yes*
Constant	*yes*

Observation 4.3 *The definition of quasirandomness becomes stricter as $\epsilon \to 0$ and $\delta \to 0$.*

The reason we introduce the definition of quasirandom functions is to make strong claims about the Håst-Odd$_\delta$ test: specifically, that it rejects quasirandom functions with high probability.

Theorem 4.4 *If $h : \{-1,1\}^n \to [-1,1]$ is (ϵ^2,δ)-quasirandom,*

$$\mathbf{Pr}[\textit{Håst-Odd}_\delta(h) \textit{ accepts}] \leq \tfrac{1}{2} + \tfrac{1}{2}\epsilon.$$

We will see the proof of this theorem in the next lecture[1]. During that lecture, we will also show how the results we have derived above can be used to establish even stronger results about hardness of approximation.