Lecture 9

More applications of pairwise independence

- Interactive proofs.
  Public coins vs. private coins

Derandomization via method of conditional expectations
Another setting in which k-wise independence is useful:

**Interactive Proofs**

\[ \text{NP} = \text{all decision problems for which "Yes" answers can be verified in polytime by a deterministic TM ("verifier")} \]

**IP:**

- Generalization of NP
- Short proofs \( \Rightarrow \) short interactive proofs
  
  “Conversations that convince”
**IP Model**

- "All-powerful" prover $P$: unbounded time but recursive
- e.g. can't solve halting problem
- Input
- $R$
- $W$
- Conversation tapes
- $R$
- $W$
- Private workspace

- poly-time verifier $V$
- Private workspace
- Random bits $\$`

**Def. [Goldwasser Micali Rackoff]**

An Interactive Proof System (IPS) for language $L$ is protocol $st.$

- if $x \in L$ & both $V, P$ follow protocol then
  
  $\Pr_{V \text{'s coins}} [V \text{ accepts } x] \geq 2/3$

- if $x \notin L$ & $V$ follows protocol then (no matter what $P$ does)
  
  $\Pr_{V \text{'s coins}} [V \text{ rejects } x] \geq 2/3$
**Thm [Goldwasser Sipser]**

\[ \text{IP}_{\text{private coins}} = \text{IP}_{\text{public coins}} \]  

GS's Answer: NO!

...anything that has protocol with private coins also has (possibly different) protocol with public coins.

Today we will see a building block for theorem:

Informally:

- Given set \( S \) s.t. \( S \in \text{IP} \) ← interesting even if \( S \in \text{P} \)
- Protocol in which \( P \) can convince \( V \) that size of set \( S \) is "big"

Let \( S_P = \exists x \mid x \text{ satisfies formula } \phi \)  
(note \( S_P \in \text{P} \))

Claim \( \exists \) protocol s.t. on input \( \emptyset \)

- if \( |S_P| > k \) + if \( V \), \( P \) follow protocol
  then \( \Pr[V \text{ accepts}] \geq 2/3 \)

- if \( |S_P| < \frac{k}{\Delta} \) + if \( V \) follows protocol
  then \( \Pr[V \text{ accepts}] < 1/3 \)

(can be replaced by any \( L \in \text{IP} \))
Can use protocol to show that # random strings which cause algorithm $A$ to accept on input $x = 2/3$.

**First idea**: Random Sampling

Repeat $? \times$ times:

- $V$ picks random assignment $x$
- Evaluates $\Psi(x)$

Output $\frac{\text{# satisfying x's}}{\text{total # repetitions}}$

How many repetitions?

$\Omega\left(\frac{\# \text{ total assignments}}{\# \text{ satisfying assignments}}\right)$

$\Omega(2^n)$
All assignments

Problem: what if $\mathcal{S}$ is small?

\[ \text{SAR assignments to } \emptyset \]

\[ \text{Fix: Universal hashing} \]

Recall:

Family of funs $\mathcal{H} = \{ h_1, h_2, \ldots \}$

for $h_x : [N] \to [M]$ is "pairwise independent" if

when $h \in \mathcal{H}$

1. $\forall x \in [N], \; h(x) \in_u [M]$

2. $\forall x_1 \neq x_2 \in [N], \; (h_1(x_1), h_2(x_2)) \in_u [M]^2$

equivalently:

$\forall x_1 \neq x_2 \in [N]
\forall y_1, y_2 \in [M]$

$\Pr_{h \in \mathcal{H}} \left[ h(x_1) = y_1 \land h(x_2) = y_2 \right] = \frac{1}{M^2}$
How does it help?

Need:
1. \( |h(S_p)| \approx |S_p| \)
2. \( h \) computable in poly time

- idea
  - clearly \( |h(S_p)| \leq |S_p| \)
  - hopefully \( |h(S_p)| \) is not too much smaller than \( |S_p| \)
    (we will show that whp \( |h(S_p)| > \frac{|S_p|}{\Delta} \))
  - \( \Rightarrow \) if \( l \) s.t. \( 2^l \) is roughly \( |h(S_p)| \)
    then most of \( 1..2^l \) gets mapped to by \( h(S_p) \)

(uses that \( H \) is p.i.)
A comment about p.i. hash fcn.

Typical use:

- Map set $S$ into smaller "space"
- Good for storage, reducing size of "name" of elements...
- Need property of "few collisions"
  since collisions cause problems, so need to minimize (e.g., in hash tables, collisions $\Rightarrow$ chaining length)
  - Here "few collisions" $\Rightarrow |h(S)|$ is not too much smaller than $|S|$

Why is that good?

- Pick any pt in range, say $0^e$
- If $h(s)$ big, it will probably hit $0^e$ uses that $h(x)$ is unit dist
Protocol: for distinguishing set of size $K$ from set of size $K/\Delta$

Given $H$ (p.i. fetus mapping $\{0,1\}^n \rightarrow \{0,1\}^k$)

1. $V$ picks $h \in_R H$
2. $V \rightarrow P$: $h$
3. $P \rightarrow V$: $x \in S_F$ s.t. $h(x) = 0^\ell$
4. $V$ accepts iff $x \in S_F$

Idea: hope: $h(S_F)$ fills "random" portion of range, so can distinguish $|h(S_F)|$ large or small.

Case 1: $|S_F| > K$:

- hopefully $|h(S_F)| \approx K$ so $0^\ell$ is "hit" with reasonable ($\geq \frac{1}{2}$?) probability.
- Then all-powerful $P$ can find preimage in $S_F$

Case 2: $|S_F| \leq \frac{K}{\Delta}$:

- $|h(S_F)| \leq \frac{K}{\Delta}$ so less likely $0^\ell$ hit.
- if not hit, $P$ can't find preimage.
- If $P$ sends $V$ a fake preimage, $V$ will detect.
Lemma \[ H \] is p.i., \( U \subseteq \mathcal{P} \), \( \alpha = \frac{|U|}{2^k} \), then \( \alpha - \frac{\alpha^2}{2} \leq \Pr_h [O^l \in h(U)] \leq \alpha \)

Proof:

RHS:
\[
\forall x \Pr_h [O^l = h(x)] = 2^{-l} \quad \text{since } \, H \text{ is p.i.}
\]

So \( \Pr_h [O^l \in h(U)] \leq \sum_{x \in U} \Pr_h [O^l = h(x)] = \frac{|U|}{2^k} = \alpha \)

LHS:\( \Pr [UA_1] \geq \sum \Pr [A_i] - \sum \Pr [A_i \cap A_j] \)

\[ \Pr_h [O^l \in h(U)] \geq \sum \Pr_h [O^l = h(x)] - \sum \Pr_h [O^l = h(x) \cap h(y)] \]

\[
= \frac{|U|}{2^k} - \binom{|U|}{2} \frac{1}{2^{2k}} \geq \frac{|U|}{2^k} - \frac{|U|^2}{2^k} \cdot \frac{1}{2^{2k}}
\]

\[ \geq \alpha - \frac{\alpha^2}{2} \]

\[ \square \]
Finishing up:

Pick \( l \) s.t. \( 2^{l-1} \leq k \leq 2^l \)

let \( a = \frac{|S_\varnothing|}{2^l} \)

If \( |S_\varnothing| > k \) then \( a \geq \frac{1}{2} \)

so \( \Pr[\text{0}^l \in h(S_\varnothing)] \geq a - \frac{a^2}{2} \geq \frac{3}{8} \)

if \( |S_\varnothing| \leq k \Delta \) then \( a \leq \frac{k}{2^l} \leq \frac{1}{\Delta} \)

so \( \Pr[\text{0}^l \in h(S_\varnothing)] \leq a \leq \frac{1}{\Delta} \)

\( \text{e.g. picking } \Delta = 4 \) gives \( \leq \frac{1}{4} \)

If repeat \( O(\log \frac{1}{\beta}) \) times,

Chernoff \( \Rightarrow \) with prob \( 1-\beta \)

if \( |S_\varnothing| \geq k \) then \( P \) is successful \( \geq \frac{3}{8} - o(1) \) of repetitions

if \( |S_\varnothing| \leq \frac{k}{\Delta} \) then \( P \) is successful \( \leq \frac{1}{4} + o(1) \) of repetitions
Comments

- Can improve so $\Delta = 1 - \varepsilon$ (how?)

- Can use same idea to prove
  
  $1P_{\text{private coins}} = 1P_{\text{public coins}}$

  argue that 1.6 protocol can be used to show size of accept region probability mass is large.

  (need that $V$ can verify a conversation/random coin flips transcripts falls into accept region).
Derandomization via the method of conditional expectations

Idea: view coin tosses of algorithm as path down tree of depth $m \leq \# \text{coin tosses}$

\[ \text{good} = \text{correct} / \text{reach witness} / \text{good approximation} / \text{pass} \ldots \]

Good randomized algorithm $\Rightarrow$ most leaves good

Idea: find a good path to leaf "bit-by-bit"
more formally:

Fix randomized algorithm $A$

\[ \text{input } x \]
\[ m = \# \text{ random bits used by } A \text{ on } x \]

For $1 \leq i \leq m \Rightarrow r_1^x \cdots r_x^x \in \Sigma_0,1^X$

\[ \text{let } p(r_1^x \cdots r_i^x) = \text{ fraction of continuations } \]
\[ \text{that end in "good" leaf} \]

\[ e(r_1^x \cdots r_i^x) = \text{ average cut value if set } \]
\[ \text{first } i \text{ nodes to } r_1^x \cdots r_i^x \]

\[ p(r_1^x \cdots r_i^x) = \frac{1}{2} \cdot p(r_1^x \cdots r_i^x, 0) \]
\[ + \frac{1}{2} \cdot p(r_1^x \cdots r_i^x, 1) \]

by averaging, if setting of $r_{i+1}^x$ to 0 or 1

st. $p(r_1^x \cdots r_{i+1}^x) \geq p(r_1^x \cdots r_i^x)$

Can we figure out which one?
if \( p(r_1 \ldots r_{x+1}) = p(r_i \ldots r_x) \quad \forall i \)

then \( p(r_1 \ldots r_m) \geq p(r_i \ldots r_{m-1}) \geq \ldots \geq p(r_i) = \text{fraction of good paths} \geq \frac{2}{3} \)

this is a leaf so value is 1 or 0

but if \( \geq \frac{2}{3} \)

it must be 1

main issue:

how do we choose best \( r_i \) at setting at each step?

Example Max cut (second way to de-randomize)

recall algorithm:

flip \( n \) coins \( r_1 \ldots r_n \)

put node \( i \) in \( S \) if \( r_i = 0 \) \& \( T \) if \( r_i = 1 \)

Output \( S, T \)
Recall from lecture 7:

**Analysis:**

Let \( 1_{u,v} = \begin{cases} 1 & \text{if } r_u \neq r_v \\ 0 & \text{o.w.} \end{cases} \)

\[
\text{Cut size} = \sum_{(u,v) \in E} 1_{u,v}
\]

\[
E[\text{cut size}] = E\left[ \sum_{(u,v) \in E} 1_{u,v} \right] = \sum_{(u,v) \in E} E[1_{u,v}] = \sum_{(u,v) \in E} \Pr[1_{u,v} = 1]
\]

\[
= \sum_{(u,v) \in E} \Pr[(r_u = 1 \land r_v = 0) \lor (r_u = 0 \land r_v = 1)]
\]

\[
= |E| \cdot \left( \frac{1}{4} + \frac{1}{4} \right) = \frac{|E|}{2}
\]

So expect \( \frac{1}{2} \) the edges to cross cut!

Note: \( E[\text{cut size}] = \frac{|E|}{2} \implies \exists \text{ cut of size } \frac{|E|}{2} \)
derandomization:

e(r_1...r_{\lambda}) = E_{R_1...R_\lambda} \left[ \left| \text{cut } (S, T) \right| \text{ given } r_1...r_{\lambda} \text{ choices made} \right]

e (no choices fixed yet) = \frac{|E|}{2} \quad \text{(previous lecture)}

how do we calculate \( e(r_1...r_{\lambda}) \)?

Let

\( S_{\lambda+1} = \{ \text{nodes } j \mid j \leq i+1, \; r_j=0 \} \supset S+T \)
\( T_{\lambda+1} = \{ \text{nodes } j \mid j \leq i+1, \; r_j=1 \} \supset \text{so far} \)
\( V_{i+1} = \{ \text{nodes } j \mid j > i+2 \text{ or } j \leq n \} \supset \text{undecided} \)

so:

\text{fact } e(r_1...r_{\lambda}) = \left( \text{# edges between } S_{\lambda+1} + T_{\lambda+1} \right)
+ \frac{1}{2} \left( \text{# edges touching } V_{i+1} \right)

(follows from same reasoning as last lecture)

\text{Note: } \text{don't need to calculate } e(r_1...r_{\lambda})
\text{ just need to figure out which is bigger } e(r_1...r_{\lambda0}) \text{ or } e(r_1...r_{\lambda1})
how do we do this?

# edges between $S_{ini} + T_{ini}$ same for both

$U_i$ is same for both

$U_{ini}$ differs only on edges to node $iti$

......

to maximize this, place node $iti$ to maximize cut size:

compare
# edges between $V_{ini} + S_{i}$
VS. " " " " " " $T_i$

$\Rightarrow$ Can deterministically pick which choice gives bigger # edges touching $U_{ini}$

$\Rightarrow$ if do this for each $i$, get solution which is $\geq$ expected value in deterministic way

yields:

Greedy Algorithm

1) $S \leftarrow \emptyset$, $T \leftarrow \emptyset$

2) For $i=0 \ldots n-1$

   place $V_i$ in $S$ if $\#$ edges between $V_i + T \geq " " " "$ S

   else place $V_i$ in $T$