Complexity Classes VIII

Stronger Approximation Bounds

Eric Rachlin
Review of PCP

- $\text{PCP}_{c, s}[q(n), r(n)]$ is the class of languages that can be recognized with by some Turing machine (which we call a “verifier”) with soundness $s$ (or less) and completeness $c$ (or more) using $O(r(n))$ random bits and $O(q(n))$ queries to a proof.
  - Completeness $c$ means that there exists a proof such that strings in $L$ are accepted with probability $c$.
  - Soundness $s$ means that for all proofs the TM accepts strings not in $L$ with probability $s$. 
PCP Verifiers

Input string, x \rightarrow \text{querier} \rightarrow \text{Proof string, y}

\text{Output} \leftarrow \text{Finite Control Unit, M} \leftarrow \text{Random Bits, r}

\text{Work tape}
The PCP Theorem

- PCP Theorem: $\text{NP} = \text{PCP}_{1, \frac{1}{2}} [1, \log(n)]$.
  - which implies $\text{NP} = \text{PCP}_{1, \frac{1}{n}} [\log(n), \log(n)]$.
- By representing the behavior of a verifier as an instance of 3SAT, we saw that a PTAS for 3SAT implied $\text{P} = \text{NP}$.
- By representing the behavior of a verifier as an instance of clique, we were able to show a constant factor approximation for CLIQUE implied $\text{P} = \text{NP}$.
- In both cases, our results were functions of $q$, the number of queries required to verify some NP-complete language.
IP

- IP is the class of languages “recognized” by a polynomial rounds of randomized interaction:
  - $q_1 = V(x, r_1)$
  - $a_1 = P(x, q_1)$
  - ... 
  - $a_{\text{poly}(|x|)} = P(x, q_1, a_1, \ldots, q_{k-1}, a_{k-1}, q_k)$
  - $y = V(x, q_1, a_1, \ldots, q_{k-1}, a_{k-1}, q_k, r_k)$

- $V$ is PTIME, $P$ can safely be restricted to PSPACE.
  - Since $P$ takes a random input, $r$, languages are recognized with completeness and soundness parameters.

- We saw last time that IP = PSPACE.
IP Verifier

Prover, P

Input string, x

Output, y

Finite Control Unit, V

Random Bits, r

Work tape
MIP

- MIP is defined like IP, except with multiple provers. The provers can be pitted against each other.
  - It turns out MIP = NEXPTIME.
  - Just two provers and one round of interaction suffice.
- In either IP or MIP, multiple repetitions of the proof protocol can drive error rates exponentially low.
- With MIP, we have an additional option. Ask many questions at once.
  - This is known as parallel repetition.

*See end of lecture 7 for citations.*
MIP Verifier

Input string, $x$

Output, $y$

Finite Control Unit, $V$

Work tape

Prover, $P_1$

Random Bits, $r$

Prover, $P_2$
PCP and MIP

● MIP is incredibly powerful, but what if we keep questions short (O(log(n)))?
● The PCP theorem implies NP-Complete problems can be reduced to gap instances of 3SAT.
● We have short a two prover one round protocol:
  ● Ask P₁ the assignment to the three variables in a clauses
  ● Ask P₂ the assignment to one of the variables
  ● Check for disagreement.
● Multiple sequential repetitions increase soundness.
  ● What about asking multiple questions at once? Yes!
  ● This results from the parallel repetition theorem (Raz, 1998)
Parallel Repetition

- The parallel repetition theorem says that
  - Through $t$ parallel repetitions, we can reduce the soundness from $s$ to $(1 - s^a)^{bt}$, where $a, b = o(1)$
- A PCP can be considered a one round MIP.
  - Different parts of the proof represent each prover.
- This implies the result we proved:
  - $\text{PCP}_{1, \frac{1}{2}}[1, \log(n)] = \text{PCP}_{1, \frac{1}{n}}[\log(n), \log(n)]$.
- Since $\text{MIP} = \text{NEXPTIME}$, we also have:
  - $\text{NEXP} = \text{PCP}_{1, \exp(-n)}[\text{poly}(n), \text{poly}(n)]$
MIP with PCP Verifier

Input string, $x$

Finite Control Unit, $M$

Work tape

Output

Proof 1

Proof 2

querier

Random Bits, $r$
A 2 query PCP (sort of)

- Now we have a two query PCP, in a larger alphabet.
- We know that NP-complete problems can be reduced to 3SAT via a gap introducing reduction.
- Using our 2 prover model
  - Ask the first prover the assignment to all variables in some set of k clauses.
  - Ask the second the assignment to some set of k variables, at least one from each clause.
  - The parallel repetitions theorem states that regardless of the original gap, there is some k such that our soundness is close to 0.
Reducing the Alphabet

- In the PCP theorem, we were able to reduce the alphabet from \{1,\ldots,m\} to \{0,1\} using Walsh-Hadamard Codes.
  - If \(x\) is the binary representation of a variable ranging 1 to \(m\), \(WH(x)\) lists all \(m\) different sums of \(x\)'s bits.
  - \(WH(x)\) and \(WH(y)\) are \(1/2\)-close if \(x \neq y\).
  - If \(WH'(x)\) is \(\varepsilon\)-close to \(WH(x)\), we can use two queries to determine any entry in \(WH(x)\) with probability at least \(2\varepsilon\).
    - RECALL: \(WH(x)_s = WH(x)_{s'} + WH(x)_{s'+s}\).
    - If \(WH'(x)\) is not \(\varepsilon\)-close to any codeword \(WH(x)\), it turns out we can detect this with probability at least \(\min(\varepsilon, 1/2)\).
      - \(\text{Prob}[WH'(x)_s = WH'(x)_{s'} + WH'(x)_{s'+s}] \leq \max(1 - \varepsilon, 1/2)\)
- Now we use an even longer code...
The Long Code

- If \( x \) is a \( b = \log(m) \) bit string, \( \text{LONG}(x) \) has \( 2^m \) entries, one for each function of \( x \).
- A function of \( b \) bits is represented by a \( 2^b = m \) row truth table. There are \( 2^m \) possible functions.
- Again, \( \text{LONG}(x) \) and \( \text{LONG}(y) \) are \( 1/2 \)-close if \( x \neq y \).
- If \( \text{LONG}'(x) \) \( \epsilon \)-close to \( \text{LONG}(x) \), we can use two queries to find any \( f(x) \) with probability at least \( 2\epsilon \).
  - \( \text{LONG}'(x)_{f(x)} = \text{LONG}'(x)_{g(x)} + \text{LONG}'(x)_{f(x)} + g(x) \)
- Again, if \( \text{LONG}'(x) \) is not close to a codeword, we can detect this with probability close to \( 1/2 \).
  - TEST: \( \text{LONG}'(x)_{f(x)} = \text{LONG}'(x)_{g(x)} + \text{LONG}'(x)_{f(x)} + g(x) \)
PCP using the Long Code

- Our 2 query PCP verifier asked two questions:
  - What are the values in k clauses?
  - What are the values of k variables?
- It then checks if the two answers, $a_1$ and $a_2$, are consistent:
  - For some function $h$, check $h(a_1) = a_2$.
- If we pick a second random boolean function, $f$, we can perform the check with probability 1/2.
  - For some random $f$, let $g(x) = f(h(x))$, check $g(a_1) = f(a_2)$.
  - The soundness of this test can’t be guaranteed.
- This type of test uses the long codes of $a_1$ and $a_2$. 
A Three Query PCP

- Assume the proof consists of $\text{LONG}(a_1)$ and $\text{LONG}(a_2)$ for every possible pair of queries.
- We check $g(a_1) = f(a_2)$ using a random function, $g'(x)$
  - $\text{LONG}(a_2)_{f(x)} = \text{LONG}(a_1)_{g'(x)} + \text{LONG}(a_1)_{g(x)} + g'(x)$
- This test would have perfect completeness, but it must be modified slightly.
  - When checking $\text{LONG}(a_2)_{f(x)}$, we sometimes check $f$’s compliment.
  - The function indexed in the last term must have a small amount of “random noise” added to it.
- It leads to a 3 query PCP with completeness $1 - \epsilon$ and soundness $1/2 + \epsilon$.
- For each randomly chosen pair of questions, the test is linear.
Hardness of 3SAT

- The protocol shows $\text{NP} = \text{PCP}_{1-\varepsilon, \frac{1}{2}+\varepsilon} [3, \log(n)]$.
  - Language in NP can be recognized by a verifier $M$ that makes 3 queries to a proof $y$ given some random input $r$.
  - In other words, $M_{r,x}(y)$ is a function of 3 bits of $y$, plus it is linear.

- The DNF of each $M_{r,x}(y)$ involves 4 clauses of 3 variables each.

- If a string is not in the language, close to 1 out of 8 clauses is unsatisfied.

- Unless $P = \text{NP}$, the known 7/8-approximation of 3SAT is the best possible.
Completeness and Soundness

- Proving completeness is easy. If the answers are correctly encoded, we will accept w.h.p.
- Proving soundness is more difficult. To show it is difficult to cheat, Fourier Analysis is used.
- We prove the simpler soundness result we used in Step 3 of the PCP theorem.
  - If $WH'(x)$ is not $\epsilon$-close to any codeword $WH(x)$,
    $\text{Prob}[WH'(x)_s = WH'(x)_{s'} + WH'(x)_{s'' + s}] \leq \max(1 - \epsilon, 1/2)$
Soundness of WH’(x)

- For a binary string \( x \) in \( \{0,1\}^b \), \( WH(x) \) lists all \( 2^b = m \) sums of \( x \)'s bits.
  - Each sum is indexed by one of \( m \) binary \( b \)-tuples.
- For the purpose of our analysis, we take binary to mean \( \{-1, 1\} \) (that is, 0 is replaced with -1).
  - Now \( WH(x) \) is all \( m \) products of \( x \)'s bits.
  - Also, let \( \text{dot}(x, y) = E[x_i y_i] \) (meaning the standard dot product by \( m \))
- The Fourier basis of \( \{0,1\}^m \) is \( f_\alpha \) for each subset \( \alpha \) of \( \{1,\ldots,b\} \), where \( f_{\alpha,x} = \prod_{i \in \alpha} x_i \).
  - Each \( f_\alpha \) corresponds to a linear functions of \( \{0,1\}^b \).
- Notice that the basis is orthonormal, so every \( WH(x) \) can be represented as the sum of basis elements, \( \sum WH_\alpha(x) \).
- Also \( \text{dot}(WH(x), WH(y)) = \sum WH_\alpha(x)WH_\alpha(y) \).
More Soundness

- Using our new alphabet of \{-1, 1\}, we wish to show that:
  - If \( \text{Prob}[WH'(x)_s \cdot WH'(x)'_s] \geq 1/2 + \varepsilon \), there is some basis element \( f_\alpha \) such that \( \text{dot}(WH'(x), f_\alpha) \geq 2\varepsilon \).
  - We call \( \text{dot}(WH'(x), f_\alpha) \) fourier coefficient \( c_\alpha \) of \( WH'(x) \).
  - \( s' \) is chosen at random and \( s' \cdot s \) is pairwise multiplication.

- We can prove \( \mathbb{E}[WH'(x)_s \cdot WH'(x)'_s \cdot WH'(x)'_s \cdot s] \geq 2\varepsilon \)
  - Write each codeword as sum of Fourier coefficients.
  - Simplify using \( f_{\alpha, s'} \cdot s = f_{\alpha, s} \cdot f_{\alpha, s} \).
  - Use linearity of expectation.
  - Simplify with orthonormality.
Conclusion

- Using the PCP theorem and parallel repetition, we can construct a high soundness, high completeness PCP.
- Using the Long Code, we can ask long questions using only a few queries.
- Combining these, Hastad (2001) was able to show that $\text{NP} = \text{PCP}_{1-\varepsilon, 1/2+\varepsilon}[3, \log(n)]$.
- This implied that a PTIME 7/8-approximation for 3SAT is as good possible unless $P = \text{NP}$.
- The approach applies to other problems as well.