Amplification of Hardness

Jonathan Kelner

4/28/2011

Application to Hardness Amplification

- Will almost give us what we need for worst-case to average-case hardness reduction
  - Idea:
    - Think of $f: \{0,1\}^n \rightarrow \{0,1\}$ as binary string of length $N = 2^n$
    - Encode to $f'$ in $\{0,1\}^M$ using ECC: $\{0,1\}^M \rightarrow \{0,1\}^M$, say of distance 0.2
    - Think of $f'$ as function from $\{0,1\}^{\log M} \rightarrow \{0,1\}$
    - If have corrupted $f'$ with 10% of values changed, can still recover $f'$ if can compute $f'$ correctly with prob. 0.9, can compute $f$ exactly
    - i.e., if $f$ hard in worst case, $f'$ hard in average case
  - Problem:
    - Need reduction to be efficient, convert small circuits to small circuits
    - Can’t read whole function and decode everything—way too big
    - For poly sized circuits, need to run in poly(n)=polylog(N)
    - So can only read a few values
  - Solution: Locally decodable codes

Locally Decodable Codes

- **Definition:** Local decoder for ECC $E: \{0,1\}^n \rightarrow \{0,1\}^m$ handling $\rho$ errors is alg that:
  - Given query access to $y$ that is within distance $\rho$ of codeword $E(x)$
  - Can recover $j$th bit of $x$ in $\text{polylog}(m)$ time with prob $\geq 2/3$
  - Also useful in practical settings
  - Can get local decoders for R-S, W-H, R-M, and concatenated codes
- **Claim:** Suppose have function $f \in E$ with worst-case hardness $\geq S(n)$. Then exists $g \in E$ and const. $c>0$ s.t. with $H_{\text{avg}}^{0.99}(g) \geq S(n/c)/n^c$ for sufficiently large $n$

Stronger Hardness

- For derandomization of BPP, need hard to compute with prob only slightly bigger than 1/2, so need stronger result
  - **Fundamental problem:**
    - Suppose want hard to compute with prob., say, 0.6
    - Binary ECC cannot have distance $>1/2$
    - So cannot correct if errors in more than 1/4 of bits
    - Means can’t recover if <0.75 fraction of bits are right
  - **Solution:** List decoding
    - Don’t try to uniquely decode
    - Try to get reasonably small list of possibilities
List Decoding

- **Theorem (Johnson Bound):** Let $E: \{0,1\}^n \rightarrow \{0,1\}^m$ have dist $\geq 1/2 - \epsilon$. For every $x \in \{0,1\}^m$ and $\delta \geq \epsilon^{1/2}$, exists $\leq 1/(2\delta^2)$ vectors within distance $\frac{1}{2} - \delta$
- Proof idea is similar to $1/2$ bound for unique decoding
- There are local list decoding algorithms for the codes we’ve discussed (with somewhat worse parameters)
- How do we use this?
  - Need to handle list of values instead of just one

Introduction to PCPs

- Not all NP-complete optimization problems are created equal
  - Some can be approximated much better than others
  - PCPs are our best tool for proving problems hard to approximate
  - Will combine a bunch of the tools we’ve introduced
- Example question:
  - Given 3SAT formula, NP-complete to check if satisfiable
  - Let val($f$)=max number clauses can be simultaneously satisfied
  - A $\rho$-approximation alg will output assignment satisfying $\rho$ val($f$) clauses
  - For what $\rho$ do there exist $\rho$-approx alg for 3SAT?
  - Can get $1/2$ easily, $7/8$+ with a little more work
    - $7/8$ is easy for 3CNF
  - Can we significantly do better?
  - PCP Theorem says no, unless $P=NP$ (!!!)
    - Not known whether can get $1-\epsilon$ for every $\epsilon >0$ until 1992
    - We’ll show some constant, but a little worse than $7/8$
  - Will have two different viewpoints of PCP Theorem:
    - Assertion about certain proof systems
    - Hardness of approximation theorem

Putting the pieces together

- **Claim:** Suppose have function $f \in E$ with worst-case hardness $\geq S(n)$.
  Then exists $g \in E$ and const. $c>0$ s.t. with $H_{\text{avg}}(g) \geq S(n/c)^{1/2}$ for all sufficiently large $n$
  - I.e., can’t use circuits of size $S(n/c)$ to get right with prob better than $\frac{3}{4} + 1/S(n/c)$
- **Proof sketch:**
  - Think of as $f$ as $N=2^n$ bit string, encode with ECC
  - Goal will be to use $S(n)$ alg computing $g$ on $\frac{1}{2} + 1/S(n)$ fraction of $(0,1)^n$ to compute $f$ perfectly on $(0,1)^n$
  - Use concatenation of $R \cdot M$ with $W \cdot H$, get $g$ of length $N'$
  - Think of as function from $(0,1)^n \rightarrow (0,1)$, $n'=log N'$
    - Will have parameters s.t. $n' = O(n)$
    - Take $|F|=S(n)$, $\delta$=small constant, $d_1=|F|^{1/2}$
    - Need to encode $f$ as input to $R \cdot M$ code, $\ell=2 \log N \log d$ suffices
    - Get $|F|$ elements of $f$, apply $W \cdot H$ to each $\rightarrow |F|^{\ell}$ total bits
  - List decoders take time poly$(|F|, d, \delta=S(n)^{1/2}, c=$absolute const.
  - Will handle $\frac{1}{2}$-approx of size $S(n)^{1/2}$ fraction errors with list size $|F|^{1/26} \cdot n^{O(1)}$
  - Will have parameters s.t. $n' = O(n)$
    - Set $\delta$ small enough (as fn of $\epsilon$), hardwire index of $f$ inside list
    - Need $O(\log(n))$ bits to index $f$

Proof System Viewpoint

- NP = things with poly time verifiable proofs
  - I.e., exists poly-sized certificates, poly time deterministic verifier
- Suppose too lazy to read the whole proof
- **PCP Theorem will show:** can encode a proof s.t. you can verify it by just looking at a few bits (chosen using randomness), with poly increase in length
  - How can this possibly work?
    - Formally, have verifier $V$ that runs in poly time s.t.:
      - Can query any bit by address (i.e., like RAM)
      - Will be allowed to flip coins
      - Queries will be nonadaptive, i.e., depend only on input and randomness, not on results of earlier queries
    - Let $L$ has $(r(n), q(n))$-PCP verifier $V$ if:
      - $V$ uses $\leq r(n)$ random bits
      - $V$ makes $\leq q(n)$ queries to proof
      - Proof is of length $\leq q(n)2^{O(1)}$
      - Completeness: If $x \in L$, exists proof s.t. accept with prob. 1
      - Soundness: If $x \notin L$, $V$ does not accept any proof with prob $\geq 1/2$
- PCP$(r(n), q(n))$-languages with $(O(r(n)), O(q(n))$-verifiers
- **PCP Theorem:** NP = PCP$(\log n, 1)$
  - PCP$(\log n, 1) \subset NP$ is easy
Hardness of Approximation Viewpoint

- **PCP Theorem (version 2):** \( \exists \rho < 1 \) s.t., for every \( L \in \text{NP} \), exists poly \( f \) mapping strings to 3CNF formulas s.t.:
  - \( x \in L \Rightarrow \text{val}(f(x)) = 1 \)
  - \( x \notin L \Rightarrow \text{val}(f(x)) < \rho \)
- Implies that exists \( \rho < 1 \) s.t. MAX-3SAT hard to approximate within \( \rho \) unless \( P = \text{NP} \)
  - Why?
- Will be convenient to work with more general constraint satisfaction problems (CSPs)
  - qCSP problem will allow arbitrary constraints s.t. each depends on \( \leq q \) variables (call \( q \) the “arity”)
  - For collection of constraints \( \phi \), \( \text{val}(\phi) = \text{fraction of constraints that can be simultaneously satisfied} \)
- \( \rho \)-Gap-qCSP problem: distinguish \( \text{val}=1 \) from \( \text{val}<\rho \)
- **PCP Theorem (version 3):** \( \exists q \in \mathbb{N}, \rho \in (0,1) \) s.t. \( \rho \)-Gap-qCSP is \( \text{NP-hard} \)

Dictionary Between the Two Views

- **Proof View:**
  - PCP verifier (V)
  - PCP proof (\( \pi \))
  - Length of proof
  - # queries (q)
  - # random bits (r)
  - Soundness prob. (usually \( 1/2 \))
- **Hardness View:**
  - CSP instance (\( \phi \))
  - Assignment to vars
  - # variables (n)
  - Arity of constraints (q)
  - \( \log(#\text{constraints}) \)
  - Max \( \text{val}(\phi) \) for NO instance