Limits of Approximation Algorithms 18 Feb, 2010 (TIFR)

Lec. 4: The PCP Theorem and inapproximability of Clique

Lecturer: Prahladh Harsha *Scribe: Girish Varma*

In the last lecture, we had introduced the PCP Theorem in terms of hardness of certain gap problems.

PCP Theorem 1. *There exists $0 < \alpha < 1$ such that gap$_\alpha$-MAX3SAT is NP-hard.*

In this lecture, we will give an alternate version of the PCP Theorem in terms of proof systems and then prove the equivalence between the two versions. In the second half of the lecture, we will prove the inapproximability of MAX-CLIQUE assuming the PCP Theorem.

The references for this lecture include Lecture 2 of the DIMACS Tutorial on Limits of Approximation [HC09] and Lectures 1 and 2 of a course on PCPs at the Univ. of Chicago [Har07]. In fact, the following lecture notes are adapted from the lecture notes scribed by Josh Grochow and Karthik Sridharan for the course on PCPs at the Univ. of Chicago [Har07] by the same instructor.

4.1 Proof Systems

A *proof system* consists of a verifier V and prover P. Given a statement x, such as "φ is satisfiable" or "G is 3-colorable," P produces a candidate proof π for the statement φ. The verifier V then reads the statement-proof pair (φ, π) and either accepts or rejects the proof π for φ. We require two properties of any proof system:

Completeness Every true statement has a proof. In other words

$$\text{if } x \text{ is true, then } \exists\pi \text{ such that } V(x,\pi) \text{ accepts.}$$

Soundness A false statement does not have a proof. In other words,

$$\text{if } x \text{ is false, then } \forall\pi,\ V(x,\pi) \text{ rejects.}$$

Variations of this general definition are (can be) used to define many complexity classes of importance, for instance:

Definition 4.1.1. *A language L is in NP if there is a deterministic polynomial time verifier V and a polynomial p such that*

1. *(Completeness)* $(\forall x \in L)(\exists\pi)(|\pi| = p(|x|)$ *and* $V(x,\pi)$ *accepts)*
2. *(Soundness)* $(\forall x \notin L)(\forall\pi(|\pi| \leq p(|x|) \Rightarrow V(x,\pi)$ *rejects)*

We can then ask the question: What happens if we allow multiple rounds of communication between P and V? It is easy to see that if V is deterministic, this is equivalent to the model in which P sends V a proof (which contains the entire transcript of the conversation). What happens if we also allow V to be randomized? Historically, this led to the development of the classes AM (Arthur-Merlin), IP (interactive proofs), which further led to the classes ZK (zero-knowledge proofs), MIP (multi-prover interactive proofs), and eventually PCPs. For a nice writeup of this history, see [Bab90] and [O'D05].

Interative proofs are defined as follows. An *interative proof system* consists of a prover P (computationally unbounded) and a verifier V (probabilistic and resource bounded) which on input x, exchange a sequence of messages with P, at the end of which V either accepts or rejects. We will denote $\langle V, P\rangle(x)$ to denote the output of the interation between P and V.

Definition 4.1.2 (IP [GMR89, BM88])**.** *A language $L \in IP$ if there is a interactive proof system (V, P) and a polynomial p such that*

1. *(Completeness)* $(\forall x \in L) \Pr[\langle V, P\rangle(x)$ *accepts* $] \geq 2/3$.
2. *(Soundness) For any prover P',* $(\forall x \notin L) \Pr[\langle V, P'\rangle(x)$ *accepts* $] \leq 1/3$.
3. *(Efficency) $\forall x$ and provers P, the total running time of V is bounded by $p(|x|)$.*

As defined above, the verifier could either reveal his random coins (public-coins-IP) or keep them hidden from the prover (private-coins-IP). Goldwasser and Sipser in a surprising result [GS86] proved that public-coins-IP=private-coins-IP.

IP naturally contains NP as well as BPP. Does IP contain more languages? In fact, Graph Non-Isomorphism (GNI) is in IP [GMW91]. A problem instance of GNI is a pair of graphs (G_1, G_2) such that $(G_1, G_2) \in GNI$ iff G_1 and G_2 are not isomorphic. The interative protocol for GNI is as follows :

1. V tosses a coin c and randomly chooses a permutation σ.
2. V constructs the graph $\sigma(G_c)$ with edge set $E' = \{(\sigma(u), \sigma(v)) : (u, v)$ is an edge in $G_c\}$ and sends it to P.
3. P is supposed to answer whether $\sigma(G_c)$ is isomorphic to G_1 or G_2.
4. V accepts iff the answer given by P is c.

Cleary if the graph are not isomorphic, a computationally unbounded prover can correctly answer in step 3 and V will accept with probability 1. If G_1 and G_2 are isomorhic, both will be isomorphic to $\sigma(G_c)$ and any prover will be able to guess the correct graph only at most half the time. If V repeats the protocol twice and accepts iff both answers of P are correct then the probability of error is $\leq 1/4$. Lund, Fortnow, Karloff and Nisan and Shamir [LFKN92, Sha92] proved that any language in PSPACE has an IP protocol thus proving that $IP = PSPACE$.

Ben-Or, Goldwasser, Kilian and Wigderson [BGKW88] then asked the question "what happens if we increase the number of provers?" This led to the definition of the class multi-prover-interactive proofs (MIP). In MIP, the provers are not allowed to interact between them and the verifier queries the provers seperately. It is an easy exercise to show that any MIP protocol can be reduced to a 2-prover protocol of the following form. V first asks a query q_1 to P_1 who gives an answer a_1 and then it asks q_2 to P_2 who gives an answer a_2. We can think of the provers P_1 and P_2 as functions $P_1 : Q_1 \to A_1$ and $P_2 : Q_2 \to A_2$ where $Q_1, Q_2 \subseteq \{0,1\}^{poly(n)}$ are the set of all queries that the verifier could ask. Since q_1 and q_2 are $poly(n)$ in length, the above functions can be thought of as tables with $2^{poly(n)}$ rows, one for each query q having the value $P_i(q)$. An $NEXP$ machine on input x can guess these two tables and simulate the protocol. Thus, it is easy to prove that $MIP \subseteq NEXP$[1]. It was then shown that the converse containment is also true: $MIP = NEXP$ [FRS94, BFL91].

The result $MIP = NEXP$ was then scaled down (from exponential to polynomial) in a sequence of works [BFLS91, FGL$^+$96, AS98, ALM$^+$98] which finally led to the celebrated PCP Theorem that $PCP = NP$. As PCPs are the topic of this course, let us define them more carefully.

4.2 Probabilistically Checkable Proof systems

We first define the notion of a restricted verifier.

Definition 4.2.1 (restricted verifier)**.** *Let $r, q, m, t : \mathbb{N} \to \mathbb{N}$ be integer valued functions. A (r, q, m, t)-restricted verifier V is a probabilistic Turing Machine (TM) with oracle access to a proof π over the alphabet Σ, which on input x of length n.*

- *tosses at most $r(n)$ coins*
- *probes at most $q(n)$ locations of*
- *a proof π of size at most $m(n)$*
- *runs in time $t(n)$*
- *and based on the proof bits it reads, it either accepts or rejects the proof.*

We will denote the verdict of the verifier V on input x and proof π and random coins R by $V^\pi[x; R]$.

We are now ready to define the PCP classes.

Definition 4.2.2 (PCP Classes)**.** *Let $r, q, m, t : \mathbb{N} \to \mathbb{N}$ be integer valued functions. We say that a language $L \in PCP_{c,s}[r, q, m, t]$ if L has a (r, q, m, t)-restricted verifier V such that*

Completeness: $\forall x \in L, \exists \pi$ *of size at most* $m(|x|), \Pr_R [V^\pi[x; R] = \mathsf{acc}] \geq c(n)$.

Soundness: $\forall x \notin L, \forall \pi$ *of size at most* $m(|x|) \Pr_R[V^\pi[x; R] = \mathsf{acc}] < s(n)$.

Similarly, we will omit mentioning the running time $t(n)$ if $t(n) = \Omega(\text{poly}(n))$ and the proof length $m(n)$ of the proof if $m(n) = \Omega(2^{r(n)+q(n)})$.

We will denote $V^\pi(x, R)$ as the output of the verifier on input x, proof π and the results of coin tosses R.

Remark 4.2.3.
- *If $c(n) = 1$, we say that the PCP verifier has perfect completeness.*

[1] $NEXP = U_{c \in \mathbb{N}} NTIME(2^{n^c})$.

- *The verifier could be either non-adaptive or adaptive (i.e., the locations probed by the verifier could depend on the earlier probes).*
- *If the verifier is non-adaptive then the size of the proof is bounded above by $q(n)2^{r(n)}$ while if the verifier is adaptive, then $m(n) \leq 2^{r(n)+q(n)}$.*
- *The number of queries is bounded by the running time (i.e., $q(n) \leq t(n)$), which could be as large as polynomial in the length of the input.*
- $PCP_{c,s}[r, q] \subseteq NTIME(2^{r(n)+q(n)})$*: The non-deterministic verifier guesses the proof of length 2^{r+q} and runs the PCP verifier for all possible random coins and accepts if the accepting probability is at least $c(n)$.*
- *It follows from definition that* $NP = PCP_{1,0}[0, \text{poly}(n)], BPP = PCP_{\frac{2}{3},\frac{1}{3}}[\text{poly}(n), 0], P = PCP_{1,0}[0, 0]$.

4.2.1 The PCP Theorem in terms of proof systems

The PCP Theorem states that all languages in NP have probabilistically checkable proofs. That is, $\forall x \in L$, there exists a verifier that checks the veracity of a proof π by tossing at most logarithmically many coins and querying the proof π in a constant number of locations. More formally, we have the following.

PCP Theorem 2 ([AS98, ALM$^+$98]). *There exists a constant Q, such that $\forall L \in NP$, $\exists$ a constant c such that, $L \in PCP_{1,1/2}[c \log n, Q]$.*

That is

$$\exists q > 0 : NP = \bigcup_{c>0} PCP_{1,\frac{1}{2}}[c \log n,\ q]$$

4.3 Equivalence of the two versions of the PCP Theorem

In this part of the lecture, we will show that the two version of the PCP theorem, , one – the hardness of approximation viewpoint and the proof checking viewpoint are, in fact, equivalent. It will be convenient for us to state PCP Theorem 1 in terms of gap_α-MAX3SAT* , defined as follows:

Definition 4.3.1 (gap_α-MAX3SAT*).

$$\begin{aligned} YES &= \{\varphi \mid \varphi \in 3SAT\} \\ NO &= \{\varphi \mid \textit{all assignments satisfy } < \alpha m \textit{ clauses}\} \end{aligned}$$

(where, as above, φ is a 3CNF formula with m clauses). gap_α-MAX3SAT* is identical to gap_α-MAX3SAT but for the fact that the second argument k of the instance $\langle \varphi, k \rangle$ of gap_α-MAX3SAT is forced to be m, the number of clauses.

PCP Theorem 3. *There exists $0 < \alpha < 1$ such that gap_α-MAX3SAT* is $NP - hard$.*

Clearly, PCP Theorem 3 implies PCP Theorem 1. We now show that PCP Theorem 3 and PCP Theorem 2 are equivalent.

Lemma 4.3.2. *PCP Theorem 3 $\Leftrightarrow$ PCP Theorem 2.*

Proof. (PCP Theorem 3 $\Rightarrow$ PCP Theorem 2) Assuming that gap_α-MAX3SAT* is $NP-hard$. Now $\forall L \in NP$ we construct a restricted verifier. The verifier uses the reduction from L to gap_α-MAX3SAT* to convert the input x to a formula $\varphi(x)$ that has the property that if $x \in L$, $\varphi(x)$ is satisfiable and for $x \notin L$, at most α fraction of the clauses of $\varphi(x)$ can be satisfied. Now it randomly picks a clause, makes 3 queries to the proof which is an assignment to the variables of $\varphi(x)$ to find the values of variables within the choosen clause. It accepts if and only if the clause evaluates to true. So if $x \in L$, there is a proof (the satisfiable assignment) such that the verifier accepts with probability 1. If $x \notin L$, for any assignment, at most α fraction of clauses are satified, and hence verifier will accept with probability at most α. It follows that $L \in PCP_{1,\alpha}[\log n, 3]$.

(PCP Theorem 2 $\Rightarrow$ PCP Theorem 3) Given the PCP Theorem 2, we wish to Karp-reduce any $L \in PCP_{1,1/2}[c \log n, Q]$ to gap_α-MAX3SAT* . The basic idea is to encode the verifier's possible actions by a Boolean formula. For each random string R, the verifier's action is a Q-ary Boolean function h_R (where Q is the number of bits of the proof probed by the verifier). A candidate for the MAX3SAT formula is the following

$$\Psi = \bigwedge_{\text{coins } R} h_R.$$

However, each local constraint of Ψ is (1) a constraint on Q variables instead of 3 and (2) the constraint is not a simple disjunction of literals but an arbitrary predicate of the the Q variables. We can get around this problem using the following fact, which we state without proof.

Fact 4.3.3. *For every q, there exists $\ell(q), k(q)$ such that any q-ary Boolean function h can be encoded by a 3CNF formula φ_h with $k(q)$ clauses over $q+\ell(q)$ variables $x_1, \ldots, x_q, z_1, \ldots, z_{\ell(q)}$ such that*

$$\begin{aligned} h(x) = 1 &\quad\Rightarrow\quad \exists z, \varphi_h(x,z) = 1 \\ h(x) = 0 &\quad\Rightarrow\quad \forall z, \varphi_h(x,z) = 0 \end{aligned}$$

The variables $z_1, \ldots, z_{l(q)}$ are called extension variables.

This fact, essentially follows from the Cook-Levin Theorem, that shows the NP-completeness of 3SAT.

Thus, given a verifier V, we construct the formula

$$\Phi = \bigwedge_{\text{coins } R} \varphi_{h_R}.$$

Let $M = 2^R k(q)$ be the number of clauses in Φ. If $x \in L$, there exists some proof π such that V accepts for all random coins R or equivalently $h_R(\pi) = 1$ for all R. Hence, we can set the value of all the extension variables such that $\varphi_{h_R}(\pi, z) = 1$ for all R. Hence, Φ is satisfiable. On the other hand, if $x \notin L$, then less than half of the choices of R cause the verifier to accept or equivalently for all π, for at least half the number of random coins R, $h_R(\pi) = 0$. It then follows from Fact 4.3.3 that for all assignments π, z, at least half of the $\varphi(h_R)$ are not satisfied. For each $\varphi(h_R)$ not satisfied, at least one clause of it must be satisfied, so at most $k(q) - 1$ clauses of it can be satisfied. Thus the total number of clauses of Φ satisfies is less than $\frac{M}{2} + \frac{M}{2}(1 - \frac{1}{k}) = M(1 - \frac{1}{2k})$. Thus, any $L \in PCP_{1,1/2}[c \log n, Q]$ Karp-reduces to gap_α-MAX3SAT* for $\alpha = 1 - 1/2k$. □

4.4 Hardness of Approximating Clique

We will now assume the PCP Theorem and prove the NP-hardness of approximating the MAX-CLIQUE problem. The corresponding descision problem is called gap_α-CLIQUE.

Definition 4.4.1 (gap_α-CLIQUE)**.** *The instance of gap_α-CLIQUE (for each $0 < \alpha \leq 1$)are of the form $\langle G, k\rangle$, where G is a graph and k a positive integer. The YES and NO instances of gap_α-CLIQUE are define*

$$\begin{aligned} YES &= \{\langle G, k\rangle | CLIQUE(G) \geq k\} \\ NO &= \{\langle G, k\rangle | CLIQUE(G) < \alpha k\} \end{aligned}$$

where $CLIQUE(G)$ denotes the size of the largest clique in G.

We will prove the following reduction due to Feige et al. [FGL+96], which will prove the NP-hardness of gap_α-CLIQUE for some $0 < \alpha < 1$, which in turns proves the NP-hardness of approximating MAX-CLIQUE to a factor better than $1/\alpha$.

Lemma 4.4.2. *If $L \in PCP_{c,s}[r, q]$ then there exists a deterministic reduction running in time $\mathrm{poly}(2^{r+q})$ reducing L to $gap_{s/c}$-CLIQUE.*

Proof. Consider a PCP verifier Ver for L that shows $L \in PCP_{c,s}[r, q]$. We use this verifier to reduce any instance x of L to $\langle G, k\rangle$ of $gap_{s/c}$-CLIQUE. The basic idea is to encode the actions of the PCP verifier Ver by the graph G such that if $x \in L$ then G has a clique of size at least k and if $x \notin L$ then G does not have any clique of size greater than $(s/c)k$ for some k.

What are the actions of the PCP Verifier Ver? On input the x, it tosses random coins R (uniformly at random of 2^r possibilities). Based on these random coins R, the verifier decides to probe the proof at q locations $(i_1(R), \ldots, i_q(R))$. It then probes the proof π at these locations to obtain $(\pi_{i_1(R)}, \pi_{i_2(R)}, \ldots, \pi_{i_q(R)})$. We call this sequence of q bits, the "view" of the verifier. Note that there are exactly 2^q possible views for each random coin (depending on the proof). Some of these views are accepting (i.e., they cause the verifier to accept) while others are rejecting. For a given proof π and random coins R, we denote the corresponding view of the proof by $Q(R, \pi) = (\pi_{i_1(R)}, \pi_{i_2(R)}, \ldots, \pi_{i_q(R)})$.[2]

[2]The above description assumes the verifier is non-adaptive. We could do a similar argument if the verifier were adaptive. The views in this case as in the non-adaptive case are a sequence of q bits in the proof. But the positions in the proof these views correspond to, will be different. For instance, in this case $Q(R, \pi)$ is defined as $Q(R, \pi) = (\pi_{i_1(R)}, \pi_{i_2(R, \pi_{i_1(R)})}, \pi_{i_3(R, \pi_{i_1(R)})}, \pi_{i_2(R, \pi_{i_1(R)})}, \ldots,)$. However, for simplicity we will assume the verifier is non-adaptive.

We are now ready to give the description of the graph $G = (V, E)$. The graph G will have $|V| = 2^{r+q}$ vertices, distributed over 2^r layers, each layer consisting of 2^q vertices. The 2^r layers correspond to the 2^r different random coins, while the 2^q vertices within each layer correspond to the possible 2^q views. Thus,

$$V(G) = \{(R, \text{view}) | R \in \{0,1\}^r, \text{view} \in \{0,1\}^q\}.$$

Two vertices (R, view) and (R', view') are connected by an edge if both the views are accepting and furthermore they do not contradict each other. In other words, there exists a proof π such that (i) view $= Q(R, \pi)$, (ii) view$' = Q(R', \pi)$ and (iii) $\mathsf{Ver}^\pi[H; R] = \mathsf{Ver}^\pi[H; R'] = \mathsf{acc}$. This completes the description of the graph $G = (V, E)$.

We now discuss the size of the largest clique in the two cases, depending on whether $x \in L$ or $x \notin L$.

Completeness: If $x \in L$, then there exists a proof π such that $\Pr_R[\mathsf{Ver}^\pi[x; R] = \mathsf{acc}] \geq c$. Consider the following set of vertices.

$$C_\pi = \{(R, Q(R, \pi)) | \mathsf{Ver}^R[H; R] = \mathsf{acc}\}.$$

Clearly the vertices given by C_π form a clique since they correspond the accepting views from the same proof π. We have, $|C_\pi| \geq c\ 2^r$. Thus, in this case, we have CLIQUE$(G) \geq c2^r$.

Soundness: if $x \notin L$, then for all proofs π, $\Pr_R[\mathsf{Ver}^R[H; R] = \mathsf{acc}] < s$. In this case, we will show that CLIQUE$(G) < s2^r$. Suppose otherwise, then there exists a set of C vertices in G of size $s2^r$ that form a clique. Since edge exist only between non-contradicting accepting views, there exists a proof π such that for all $(R, \text{view}) \in C$, we have view $= Q(R, \pi)$ and $\mathsf{Ver}^\pi[x; R] = \mathsf{acc}$. But then, we have $\Pr_R[\mathsf{Ver}^\pi[x; R] = \mathsf{acc}] \geq s$, contradicting that $x \notin L$.

Hence, the reduction $x \mapsto \langle G, c2^r \rangle$ is a reduction from L to $gap_{s/c}$-CLIQUE. Furthermore, this reduction can be performed in time at most linear in the size of the graph G (i.e., 2^{r+q}). This completes the proof of the lemma □

Thus, starting from the PCP Theorem that $NP = \bigcup_{c>0} PCP_{1,\frac{1}{2}}[c \log n,\ q]$ for some constant q, we get the following inapproximability result for MAX-CLIQUE.

Corollary 4.4.3. *$gap_{0.5}$-CLIQUE is NP-hard.*

4.4.1 Improving the inapproximability factor

We observe that the inapproximability factor in the above reduction only depends on the ratio s/c. However, by a simple sequential repetition of the verifier improves this factor to any constant $\alpha > 0$ as shown in the following proposition.

Proposition 4.4.4. *For all $k > 0$, $PCP_{c,s}[r, q] \subseteq PCP_{c^k, s^k}[kr, kq]$*

Proof. Sequentially repeat the actions of the verifier of "$PCP_{c,s}[r, q]$" k-times and accept only if all the k views are accepting. □

Combining this proposition with any constant k with the reduction, we have the improved hardness result.

Corollary 4.4.5. *$\forall 0 < \alpha < 1$, gap$_\alpha$-CLIQUE is NP-hard.*

We can improve the hardness factor further by choosing k to be super-constant. However, then the number of random coins tossed by the verifier $kr = O(k \log n)$ becomes super logarithmic and hence the running time of the reduction 2^{kr+kq} becomes super-polynomial. The problem here is that we are using kr random coins to repeat the verifier k times. We can instead use techniques from derandomization to recycle random coins. In fact, it is known that $r + O(k)$ (as opposed to kr) random coins suffice to repeat a randomized protocol k times achieving the same exponential improvement in error.

Lemma 4.4.6. *For all k, $PCP_{1,s}[r,q] \subseteq PCP_{1,2s^k}[r + O(k), kq]$.*

Combining this with the hardness result, we get

Corollary 4.4.7. *There exists a $\delta > 0$, gap$_{n^{-\delta}}$-CLIQUE in NP-hard. In other words, approximating MAX-CLIQUE to a factor better than n^δ is NP-hard.*

It is known that we can "recycle queries" and improve the inapproximability factor to $n^{1-\varepsilon}$ for any ε (under randomized reductions by Håstad [Hås99] which was later extended to work even under deterministic reductions by Zuckerman [Zuc07]). We will not cover these results in this course. Note this is almost optimal, since outputting a single vertex gives a n-approximation algorithm for MAX-CLIQUE.

References

[ALM$^+$98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. *Proof verification and the hardness of approximation problems.* J. ACM, 45(3):501–555, May 1998. (Preliminary Version in *33rd FOCS*, 1992). eccc:TR98-008, doi:10.1145/278298.278306.

[AS98] Sanjeev Arora and Shmuel Safra. *Probabilistic checking of proofs: A new characterization of NP.* J. ACM, 45(1):70–122, January 1998. (Preliminary Version in *33rd FOCS*, 1992). doi:10.1145/273865.273901.

[Bab90] László Babai. *E-mail and the unexpected power of interaction.* In *Proc. 5th IEEE Conference on Structure in Complexity Theory*, pages 30–44. 1990. doi:10.1109/SCT.1990.113952.

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. *Non-deterministic exponential time has two-prover interactive protocols.* Comput. Complexity, 1(1):3–40, 1991. (Preliminary Version in *31st FOCS*, 1990). doi:10.1007/BF01200056.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. *Checking computations in polylogarithmic time.* In *Proc. 23rd ACM Symp. on Theory of Computing (STOC)*, pages 21–31. 1991. doi:10.1145/103418.103428.

[BGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. *Multi-prover interactive proofs: How to remove intractability assumptions.* In *Proc. 20th ACM Symp. on Theory of Computing (STOC)*, pages 113–131. 1988. doi:10.1145/62212.62223.

[BM88] László Babai and Shlomo Moran. *Arthur-merlin games: A randomized proof system, and a hierarchy of complexity classes.* J. Computer and System Sciences, 36(2):254–276, April 1988. doi:10.1016/0022-0000(88)90028-1.

[FGL+96] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. *Interactive proofs and the hardness of approximating cliques.* J. ACM, 43(2):268–292, March 1996. (Preliminary version in *32nd FOCS*, 1991). doi:10.1145/226643.226652.

[FRS94] Lance Fortnow, John Rompel, and Michael Sipser. *On the power of multiprover interactive protocols.* Theoretical Comp. Science, 134(2):545–557, 21 November 1994. (Preliminary Version in *3rd IEEE Symp. on Structural Complexity*, 1988). doi:10.1016/0304-3975(94)90251-8.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. *The knowledge complexity of interactive proof systems.* SIAM J. Computing, 18(1):186–208, February 1989. (Preliminary Version in *17th STOC*, 1985). doi:10.1137/0218012.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. *Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems.* J. ACM, 38(3):691–729, July 1991. (Preliminary Version in *27th FOCS*, 1986). doi:10.1145/116825.116852.

[GS86] Shafi Goldwasser and Michael Sipser. *Private coins versus public coins in interactive proof systems.* In *Proc.* 18*th ACM Symp. on Theory of Computing (STOC)*, pages 59–68. 1986. doi:10.1145/12130.12137.

[Har07] Prahladh Harsha. *CMSC 39600: PCPs, codes and inapproximability*, 2007. A course on PCPs at the University of Chicago (Autumn 2007).

[Hås99] Johan Håstad. *Clique is hard to approximate within $n^{1-\varepsilon}$.* Acta Mathematica, 182(1):105–142, 1999. (Preliminary Version in *28th STOC*, 1996 and *37th FOCS*, 1997). doi:10.1007/BF02392825.

[HC09] Prahladh Harsha and Moses Charikar. *Limits of approximation algorithms: PCPs and unique games*, 2009. (DIMACS Tutorial, July 20-21, 2009). arXiv:1002.3864.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. *Algebraic methods for interactive proof systems.* J. ACM, 39(4):859–868, October 1992. (Preliminary Version in *31st FOCS*, 1990). doi:10.1145/146585.146605.

[O'D05] Ryan O'Donnell. *A history of the PCP theorem*, 2005.

[Sha92] Adi Shamir. *IP = PSPACE.* J. ACM, 39(4):869–877, October 1992. doi:10.1145/146585.146609.

[Zuc07] David Zuckerman. *Linear degree extractors and the inapproximability of max clique and chromatic number.* Theory of Computing, 3(1):103–128, 2007. (Preliminary Version in *38th STOC*, 2006). doi:10.4086/toc.2007.v003a006.