E0 249: Approximation Algorithms April 4, 2022

Week 14 Lecture 2 - PCP

Instructor: Anand Louis *Scribes: Anish Hebbar*

In this lecture, we'll be looking at Probabilistically Checkable Proofs and some applications to things such as Hardness of approximation.

1 Probabilistically Checkable Proofs (PCP)

1.1 3-SAT

In order to motivate the idea behind Probabilistically Checkable Proofs, we first take a look at the 3-SAT problem. In 3-SAT, we are given a boolean formula consisting of m clauses, each of length exactly 3, on a total of n variables, and we wish to satisfy all the clauses simultaneously. For example, consider the instance

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (x_2 \vee x_4 \vee x_5) \wedge \neg(x_1 \vee \neg x_2 \vee \neg x_4)$$

We classify an instance as YES if there exists an assignment satisfying it (evaluates to TRUE), else we call it a NO instance. Note that if a SAT instance is a YES instance, the binary string corresponding to the satisfying assignment suffices as a "proof" of this fact.

Given a polynomial sized proof, a **Verifier** (Turing Machine) can verify in polynomial time that the given instance is a YES instance (Simply by evaluating each clause). One such "proof" to the above instance , which is indeed a YES instance is 11101, namely $x_1 = 1, x_2 = 1, x_3 = 1, x_4 =, x_5 = 1$. Note that the "proof" (satisfying assignment) does not have to be unique, other assignments such as 10011 also work. We now ask the following question:

Does the verifier need to read the entire proof, or can the verifier (Turing machine) make a "good" decision after only reading $O(1)$ bits of the proof?

We will make precise what "good" means soon. We will make use of a randomised scheme to verify the proof.

- Say the verifier can make use of r random bits, and read q locations in the proof, in order to give a decision that is correct with "good" probability.
- The prover then writes down a "proof" X in a predecided format (in this case, a binary string corresponding to the assignment).
- The Verifier then used the r *independent* random bits to generate q random locations $l_1, l_2, \cdots l_q$ of the proof to query. These locations can be thought of as some deterministic function of these r random bits.
- The Verifier then computes some $0-1$ function $g(X_{l_1}, \cdots X_{l_q})$, and accepts the proof if the function evaluates to 1, and rejects it otherwise (Evaluates to 0). Basically,

  $$g : \{0,1\}^q \to \{0,1\}$$

  is some evaluation function that will depend on the exact problem we are working with.

- If we have a YES instance, we require that there *exists* a proof that the verifier accepts with probability **at least** c (Correctness parameter). If we have a NO instance, we require that *for all* "proofs", the verifier accepts it with probability **at most** s (Soundness parameter).

Now, we denote $PCP_{c,s}(r,q)$ to be the class of languages which have a probabilistically checkable proof with parameters c, s, r, q satisfying the above. Intuitively, we want $q = O(1)$ (queries at most constantly many locations), and $r = O(\log n)$ (We want the proof to be polynomial in the input size n. Each location can be identified using $\log n$ many bits, and we want to query only $q = O(1)$ many locations). Moreover, we want $c - s = \Omega(1)$ ideally, namely that it is large.

Let us now come up with a trial toy scheme for the 3-SAT problem.

- Our prover gives a satisfying assignment (binary string) to the 3-SAT instance as a proof. The verifier then uses random bits to sample a clause, uniformly at random from the 3-SAT instance. If the binary string given satisfies this random clause, we ACCEPT, else we REJECT.
- Note that since there are m clauses, we need $r = O(\log m)$ many bits to choose one at random. Each clause is length 3, so we need to query only $q = 3$ locations.
- If we had a YES instance, then the prover would have given a satisfying assignment that satisfies each clause, hence we would always ACCEPT

  $$c = \mathbb{P}(\text{Verifier ACCEPTS}) = 1$$

  If we had a NO instance, then at most $m-1$ clauses in the instance can be satisfied by any assignment, so the probability that we make an error is at most

  $$s = \mathbb{P}(\text{Verifier ACCEPTS}) \leq \frac{m-1}{m}$$

- However, in this case $c - s$ could be as small as $\frac{1}{m} \neq \Omega(1)$. Thus, we need a better strategy.

We now prove that

$$PCP_{c,s}(O(\log n), O(1)) \subseteq NP$$

Consider any language $L \in PCP_{c,s}(O(\log n), O(1))$ Namely, given any string $x \in L$, we will show that there exists a polynomial time proof P_x that x belongs to L.

- First, enumerate all 2^r ($r = O(\log nn)$) possible values of the random bits, and correspondingly compute $g(X_{l_1}, \cdots, X_{l_q})$ for each case whether at least c fraction of these evaluate to 1 (Verifier accepts), or whether $\leq s$ fraction make the verifier accept.
- The runtime is $poly(n)$ as $r = O(\log n)$, so there that $O(n^{O(1)})$ many possible values of the random bit, each of which is checked in polynomial time. Hence, we have exhibited a proof in polynomial time that $x \in L \implies L \in NP$.

The PCP theorem states something stronger:

Theorem 1 (PCP Theorem [1, 2])**.** *$PCP_{1,s}(O(\log n), O(1)) = NP$ for some absolute constant $s < 1$.*

Dinur [3] significantly simplified the proof of the above theorem, using ideas like expander graphs, and her proof is the one usually covered in graduate courses on complexity theory today.
We will now look at the idea of serial repetition, in order to improve our soundness guarantee s. We follow the below scheme.

Take $PCP_{1,s}(r,q)$, and repeat the protocol k many times. If the verifier answered Yes in all cases, output YES (ACCEPT). Otherwise, output NO (REJECT). We now need $k \cdot r$ random bits, and query $k \cdot q$ many locations. As long as k is constant, we still query at most constantly many locations.

- If we have a YES instance, then the verifier will answer Yes in all cases and we output YES, showing $c = 1$ for this new protocol.
- If we have a NO instance, then the only way the verifier outputs YES is if the answer is YES in all cases, which happens with probability at most s^k, as these are k independent trials. So the new soundness parameter for this protocol is $s' = s^k$.

Thus, we have shown that $\forall k \in \mathbb{N}$, $PCP_{1,s}(r,q) \subseteq PCP_{1,s^k}(kr, kq)$.

1.2 Hardness of Approximation of Max-g

We will now study a link between PCP and hardness of approximation. Let $g : \{0,1\}^q \to \{0,1\}$ be a fixed function.

Max-g: Given a set of n variables and a set of m constraints

$$\{g(X_{l_1^{(i)}}, \cdots X_{l_q^{(i)}}) : i \in [m]\}$$

compute an assignment to the variables that maximised the fraction of constraints satisfied.

We will show the following theorem.

Theorem 2. *If $3SAT \in PCP_{c,s}(O(\log n), O(1))$, with g as the test function, then it is NP-hard to obtain a polytime approximation algorithm with approximation factor better than s/c for $3SAT$.*

Proof. Let X be the proof that is provided by the prover. The verifier tosses r random coins, $R \sim \{0,1\}^r$ random coins, giving q locations $l_1^{(R)}, \cdots l_q^{(R)}$ to query. Now, consider the set of tests

$$\{g(X_{l_1^{(R)}}, \cdots X_{l_q^{(R)}}) : R\}$$

performed by the verifier. The probability that the verifier accepts the proof is simply the fraction of tests satisfied by the proof. Hence, we can view the prover's task as finding an assignment to the "variables" X such that the fraction of satisfied constraints in $\{g(X_{l_1^{(R)}}, \cdots X_{l_q^{(R)}}) : R\}$ is maximised.

- Say the 3SAT instance was a YES instance. Then, the verifier accepts the proof with probability at least c, as we assumed $3SAT \in PCP_{c,s}(O(\log n), O(1))$. Hence, there exists an assignment to X that satisfies at least c fraction of the constraints in $\{g(X_{l_1^{(R)}}, \cdots X_{l_q^{(R)}}) : R\}$.
- Say the 3SAT instance was a NO instance. Then, the verifier accepts the proof with probability at most s. Hence, *any* assignment to X satisfies at most s fraction of the constraints in $\{g(X_{l_1^{(R)}}, \cdots X_{l_q^{(R)}}) : R\}$.

Thus, by the PCP theorem, for $\{g(X_{l_1^{(R)}}, \cdots X_{l_q^{(R)}}) : R\}$, it is NP-hard to determine if there is any assignment to X which satisfies at least c fraction of the constraints, or whether all such assignments satisfy at most s fraction of the constraints. This then implies that for Max-g, it is NP-hard to obtain a polytime approximation algorithm with approximation factor better than s/c, as if there were such an algorithm with

approximation factor $s/c + \varepsilon$, as then if there existed an assignment satisfying at least c fraction of the constraints, we can use this algorithm to obtain an assignment satisfying at least $(s/c + \varepsilon) \cdot c = s + c \cdot \varepsilon > s$ fraction of the constraints. So, we would be able to differentiate between the 2 cases in polytime, which is impossible unless $P = NP$.

□

Given the above close connection, we now look at the Gap version of this problem.

Gapg$_{\mathbf{c,s}}$: Given an instance of Max-g promised to be one of the two cases

1. YES: $\exists$ an assignment to the variables satisfying $\geq c$ fraction of constraints.
2. NO: $\forall$ assignments to the variables $\leq s$ fraction of constraints are satisfied.

We need to determine the case that the instance falls in.

MAX - E3LIN: Given a system of linear equations over $\mathbb{F}_2$, with variables $x_1, \cdots x_n$ and equations of the form

$$x_i + x_j + x_k = c$$

where $1 \leq i, j, k \leq n$, $c \in \{0, 1\}$, the goal of the problem is to find an assignment to these variables that satisfies the maximum fraction of equations.
Following are some results related to E3SAT and E3LIN due to Hastad [4].

- For every $\varepsilon > 0$, and every $L \in NP$, there is a PCP with $q = 3$, $c \geq 1 - \varepsilon$ and $s \leq \frac{1}{2} + \varepsilon$. Moreover, the verifier chooses indices $(l_1, l_2, l_3) \sim [m]^3$ and $b \sim \{0, 1\}$ according to some distribution, and checks whether $X_{l_1} + X_{l_2} + X_{l_3} = b \ (mod\ 2)$.
- Thus, we see that for any $\varepsilon > 0$, GapE3LIN$_{1-\varepsilon, \frac{1}{2}+\varepsilon}$ is NP-hard. Using the s/c approximation hardness result, we see we cannot get better than a $\frac{1}{2}$ approximation for this problem.
- Note that if a solution to the system of equations exists, we can recast the above into matrix form and solve it in polynomial time via gaussian elimination. If a solution does not exist, we observe that for any given equation $x_i + x_j + x_k = c$, if we *randomly* assigned values to the x_i, the sum $x_i + x_j + x_k$ is even with probability half, and odd with probability half. Thus, if there are m equations, the expected number of satisfied equations, by linearity of expectation is

  $$\mathbb{E}\left[\text{No. of equations satisfied}\right] \geq \frac{1}{2} \cdot m \geq \frac{OPT}{2}$$

  showing that a random assignment gives a half approximation.
- A reduction from Max-E3LIN to MaxE3SAT shows that for any $\varepsilon > 0$, GapE3SAT$_{1-\varepsilon, \frac{7}{8}+\varepsilon}$ is NP-hard. Hastad [4] also proved GapE3SAT$_{1, \frac{7}{8}+\varepsilon}$ is NP-hard.

1.3 Hardness for Max-Clique

We will use a reduction from PCP to a max-clique problem to get a hardness result. For $PCP_{1,s}(r, q)$, we construct the FGLSS graph [5] (named after Feige, Goldwasser, Lovasz, Safra, Szegedy) as follows.

- Vertices: We make $R = 2^r$ rows, each containing $\leq 2^q$ vertices. The total number of vertices is thus $N \leq 2^{r+q}$. The vertices in each row correspond to views on the queried bits that make the verifier accept the proof.

- Edges: We add edges between any 2 vertices that do not conflict, i.e., represent *consistent* partial assignments.
- Note that each row is an independent set, as each vertex on the row differs at atleast one position in the corresponding assignment. If two rows query different sets of indices, then they form a complete bipartite graph as there aren't any edges among vertices in the same row, and any 2 vertices in different rows have an edge (query different indices)
- Now, say this graph has a clique of size M. These vertices much be in M different rows. Then, the partial assignment of any two vertices in the clique are consistent with each other (as they are all joined by edges). The probability that the verifier accepts the proof is then $\geq \frac{M}{2^r}$, as if we land in any of the M rows defined by the clique, we can choose the assignment corresponding to the clique, and then extend the assignment arbitrarily on the remaining vertices.
- Thus, assuming $3SAT \in PCP_{1,s}(r,q)$. we get that there is no $1/s$ approximation for max-clique.
- By serial repetition, we get that there is no $\frac{1}{s^k}$ approximation for max-clique assuming $3SAT \in PCP_{1,s^k}(kr,kq)$

Theorem 3. *$\forall \varepsilon > 0$, there is no $1/2^{\log^{1-\varepsilon} N}$ approximation for max-clique assuming $NP \not\subseteq \bigcup_{t>0} DTIME(2^{\log^t n})$ (Polynomial time)*

Proof. In the serial repetition argument, choose $k = \log^t n$, then

$$N \approx 2^{k(r+q)} = 2^{O(\log^{t+1} n)}$$

The gap between correctness and soundness is

$$\frac{1}{s^k} = \frac{1}{2^{O(\log^{t+1} n)}} = 2^{O(\log^{\frac{t}{t+1}} N)}$$

and for any given $\varepsilon > 0$, we can choose t large enough that $\frac{t}{t+1} > 1 - \varepsilon$ as desired.

□

References

[1] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. J. ACM, 45(1):70–122, 1998.

[2] 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, 1998.

[3] Irit Dinur. The PCP theorem by gap amplification. J. ACM, 54(3):12–es, 2007.

[4] Johan Hastad. Some optimal inapproximability results. J. ACM, 48(4):798–859, July 2001.

[5] Feige, Uriel, et al. "Interactive Proofs and the Hardness of Approximating Cliques." Journal of the ACM, vol. 43, no. 2, Mar. 1996, pp. 268–92.