INFR11102: Computational Complexity 19/11/2019

Lecture 18: Unique Satisfiability; Interactive Proofs

Lecturer: Heng Guo

1 Unique Satisfiability

The number of solutions to NP-hard problems may vary in a large range. It might be 0, 1, polynomially many, or exponentially many. One might speculate that NP-hard problems are difficult because of this varying behaviour. However, as we will see, even if we restrict the number of solutions to at most 1, the problem is still as difficult as the Sat.

Consider the following question. Suppose we know that a given formula has at most 1 satisfying assignment, is Sat getting any easier?

Name: UniSat

Input: A formula φ with at most 1 satisfying assignment.

Output: Is φ satisfiable?

A decision problem of this kind is usually called a *promise problem*, namely its input satisfies a certain promise. An algorithm $A(\cdot)$ for UniSat shall satisfy the following requirement: for an input φ,

$$A(\varphi) = \begin{cases} 0 & \text{if } \varphi \text{ is not satisfiable;} \\ 1 & \text{if } \varphi \text{ has a unique satisfying assignment;} \\ 0/1 & \text{otherwise.} \end{cases}$$

Notice that A is allowed to output whatever when φ has more than 1 satisfying assignments. What Valiant and Vazirani [VV86] showed is that UniSat is not easier than Sat, at least for randomized algorithms.

Theorem 1 (Valiant-Vazirani [VV86])**.** *If there is a (randomized) polynomial-time algorithm A for* UniSat*, then* NP = RP*.*

The main technical tool is still pairwise independent hash families. Recall its definition.

Lemma 2. *Let $S \subseteq \{0,1\}^n$ be a set with $2^m \leq |S| \leq 2^{m+1}$ for some integer $m \geq 0$, and let $\mathcal{H}_{n,m+2}$ be a pairwise independent hash family from $\{0,1\}^n$ to $\{0,1\}^{m+2}$. Then*

$$\Pr_{H \in \mathcal{H}_{n,m+2}} \left[\text{there is a unique } x \in S \text{ with } H(x) = 0^{m+2}\right] > 1/8.$$

Proof. Let $\mathbf{0} = 0^{m+2}$ and $p := 2^{-m-2}$ so that $|S| \cdot p \in \left[\frac{1}{4}, \frac{1}{2}\right]$. By the definition of pairwise independent hash families, for any $x \in S$,

$$\Pr_{H \in \mathcal{H}_{n,m+2}}[H(x) = \mathbf{0}] = p,$$

and for any $x \neq x' \in S$,

$$\Pr_{H \in \mathcal{H}_{n,m+2}}[H(x) = H(x') = \mathbf{0}] = p^2.$$

Let N be a random variable denoting $|\{x \mid H(x) = \mathbf{0}\}|$ where H is chosen uniformly at random from $\mathcal{H}_{n,m+2}$. By inclusion/exclusion,

$$\begin{aligned}\Pr[N \geq 1] &\geq \sum_{x \in S} \Pr[H(x) = \mathbf{0}] - \sum_{x \neq x' \in S} \Pr\left[H(x) = H(x') = \mathbf{0}\right] \\ &= |S| \cdot p - \binom{|S|}{2} p^2,\end{aligned}$$

and

$$\Pr[N \geq 2] \leq \sum_{x \neq x' \in S} \Pr\left[H(x) = H(x') = \mathbf{0}\right] = \binom{|S|}{2} p^2.$$

Hence,

$$\begin{aligned}\Pr[N = 1] &= \Pr[N \geq 1] - \Pr[N \geq 2] \\ &\geq |S| \cdot p - 2\binom{|S|}{2} p^2 \geq |S| \cdot p - |S|^2 \cdot p^2 > 1/8,\end{aligned}$$

using the fact that $|S| \cdot p \in \left[\frac{1}{4}, \frac{1}{2}\right]$. □

Lemma 2 is similar to the isolation lemma we showed last time. The key point is still that when the size of S is appropriate, isolation happens with high probability.

Proof of Theorem 1. The basic idea is to transform the input φ into a more restrained formula φ' so that if φ is satisfiable, then φ' has a unique satisfying assignment with high probability. To use Lemma 2, we need to know an integer m such that $2^m \leq |S| \leq 2^{m+1}$, where S is the set of satisfying assignments of φ. We get around this by choosing $m \in [n-1]$ uniformly at random. Draw a random hash function $H \in \mathcal{H}_{n,m+2}$ and let

$$\psi(x) := \varphi(x) \wedge (H(x) = \mathbf{0}).$$

Let φ' be the Boolean formula that is equivalent to ψ (with possibly auxiliary variables). We then simply output $A(\varphi')$.

To see the correctness of our algorithm, if φ is not satisfiable, then neither is φ'. Otherwise, with probability at least $1/n$, S satisfies that $2^m \leq |S| \leq 2^{m+1}$. By Lemma 2, with probability at least $1/8$, φ' has a unique satisfying assignment. Thus with probability at least $\frac{1}{8n}$ multiplied by the correct probability of A, our algorithm is correct. We may use the standard amplification method to make the correct probability arbitrarily close to 1. □

2 Private coin interactive proof systems

In the verification definition of NP, a certificate is used to verify or prove the validity of an input. This is a static proof. A game-theoretical way to look at this is that an almighty prover presents a proof, and a polynomial-time verifier can verify the correctness of the proof. Instead, we may also allow the verifier and the prover to interact with each other. It leads to the notion of interactive proof systems.

Allowing interaction does not sound like a big generalization, and indeed, if both the prover and the verifier are deterministic, then one can show that this is the same class as NP. However, what about adding some randomness?

The prover is assumed to have unlimited computational power, and thus there is no need to employ randomness on his side. The interactive proof class, IP, is defined with a probabilistic verifier. In NP, the verifier has to run within polynomial-time, and the proof can be arbitrarily difficult to compute, as long as it is correct and exists. Similarly, the verifier in IP is also bounded by polynomial-time, and the prover can be arbitrarily powerful. In fact, it is easy to see that the prover does not need power beyond PSpace, with which he can already enumerate all possible responses.

To be more precise about the "interaction", suppose on input x, V is the verifier and P is the prover. In the first round, V produces (probabilistically) some queries y_1 for P, depending on x, and P returns z_1 depending on (x, y_1). In the next round, V generates (probabilistically) another query y_2 depending on (x, y_1, z_1), and P returns z_2 depending on $P_2(x, y_1, y_2)$. The protocol continues in this fashion until V has reached a decision. We denote by $\langle V, P\rangle_k(x)$ such an outcome. The running time of V in this protocol has to be bounded by a polynomial in $|x|$, and all strings $y_1, y_2, \cdots, z_1, z_2, \cdots$ must have polynomial length as well.

Definition 1. *For any integer $k \geq 1$, a language L is in $\mathtt{IP}[k]$ if there is a probabilistic polynomial-time TM V (verifier) such that V can have a k-round interaction with a function $P : \{0,1\}^* \to \{0,1\}^*$ such that*

- *if $x \in L$, then $\exists P$, $\Pr[\langle V, P\rangle_k(x) = 1] \geq \frac{3}{4}$;*
- *if $x \notin L$, then $\forall P$, $\Pr[\langle V, P\rangle_k(x) = 1] \leq \frac{1}{4}$.*

Then $\mathtt{IP} = \bigcup_{c\in\mathbb{N}} \mathtt{IP}[n^c]$.

Once again, this class does not seem like too much of a generalization. An easy upper bound is that $\mathtt{IP} \subseteq \mathtt{PSpace}$. However, surprisingly, IP is a lot more powerful than NP.

Theorem 3 (Shamir [Sha92])**.** $\mathtt{IP} = \mathtt{PSpace}$.

By using the same amplification techniques as for BPP (repeat and take a majority vote), the constants 3/4 and 1/4 can be replaced by $1 - 2^{-n^c}$ and $2^{-n^{c'}}$ for any $c, c' > 0$. One can even do this without requiring more rounds, via a technique termed parallel repetition [Raz98]. The essential requirement here is that the gap between the two cases cannot be too small.

2.1 An interactive proof system for graph non-isomorphism

We have mentioned Graph isomorphism (GI), which is a problem with potentially intermediate complexity between P and NP.

Name: GI

Input: Two graphs G_1 and G_2.

Output: Are G_1 and G_2 isomorphic?

Recall that GI $\in$ NP. The certificate is simply a permutation of all vertices so that G_1 and G_2 are identical. On the other hand, for its complement problem, graph non-isomorphism (GNI), there does not seem to exist an easy certificate. However, there is a very simple interactive proof system for GNI.

The verifier first toss a random coin, and choose G_1 or G_2 respectively. Then, he permutes the resulting graph randomly, and then sends it to the prover. Call this random graph X. The prover is required to distinguish whether this graph is G_1 or G_2. The verifier accepts if the prover guessed it correctly.

Apparently, if G_1 and G_2 are not isomorphic, then the prover can easily distinguish them by brute force. (Recall that the prover has unlimited computational power.) On the other hand, if $G_1 \cong G_2$, then X is uniform over all graphs isomorphic to G_1 or G_2, independently from the choice at the beginning. Thus, the answer from the prover, which only depends on X, is also independent over the choice of G_1 or G_2. Hence, he can guess it correctly with probability at most $1/2$. Since the error in the protocol above is one-sided, we can repeat it many times to get a very small error probability.

Proposition 4. GNI $\in$ IP[2].

Remark (Bibliographic). Private coin system IP was introduced by Goldwasser, Micali, and Rackoff [GMR89]. Relevant chapters are [AB09, Chapter 8.1].

3 Public coin systems

In Definition 1, the prover is not allowed to see the random choice of V. Thus it is called a *private coin system*. This makes the prover's job more difficult — he needs to be prepared by whatever query the verifier comes up with. Indeed, our protocol for GNI seems to rely on this fact.[1] If, on the other hand, such a coin is public, then we have some different complexity classes, which intuitively is less powerful than IP.

Definition 2. *The class* AM[k] *is defined in the same way as* IP[k] *in Definition 1, except that the verifier's message is restricted to only the random bits, and no other random bit is allowed to be used by the verifier.*

Define AM = AM[2].

[1] However this is just an illusion! We will give a public coin system for GNI in the next lecture.

The name AM stands for "Arthur-Merlin". The prover is termed "Merlin" to reflect its infinite power, whereas the verifier, Arthur, is a mere mortal and can only perform polynomial-time computation. In addition, Arthur cannot hide coins he has tossed from Merlin.

Note the disparity between $\mathtt{AM} = \mathtt{AM}[2]$ and $\mathtt{IP} = \mathtt{IP}[poly]$. This is somewhat inconsistent but standard in the literature. In fact, it can be shown that $\mathtt{AM}[k] = \mathtt{AM}[2]$ for any constant $k \geq 2$.

An alternative definition for AM is the following.

Definition 3. *A language $L \in \mathtt{AM}$ if there exists a polynomial-time TM M such that*

$$x \in L \Rightarrow \Pr_y[\exists z, M(x, y, z) = 1] \geq \frac{3}{4};$$
$$x \notin L \Rightarrow \Pr_y[\exists z, M(x, y, z) = 1] \leq \frac{1}{4},$$

where y and z are both polynomially bounded in length.

One way to understand AM is that in a two-player game, Arthur makes a random move first, and Merlin counters with the best possible move.

Standard amplification techniques, once again, apply here. The gap can be made arbitrarily close to 1. In fact, one can show that the correct probability can be changed to 1 without changing Definition 3.

Definition 4. *A language $L \in \mathtt{AM}_1$ if there exists a polynomial-time TM M such that*

$$x \in L \Rightarrow \Pr_y[\exists z, M(x, y, z) = 1] = 1;$$
$$x \notin L \Rightarrow \Pr_y[\exists z, M(x, y, z) = 1] \leq \frac{1}{4},$$

where y and z are both polynomially bounded in length.

Theorem 5. $\mathtt{AM} = \mathtt{AM}_1$.

Proof sketch. It is clear that $\mathtt{AM}_1 \subseteq \mathtt{AM}$. We will show the other direction. Let $L \in \mathtt{AM}$. We assume the error probability in Definition 3 is 2^{-n^c} via standard amplification. The idea here is similar to the $\mathtt{BPP} \subseteq \Sigma_2^p$ proof. Let S be the set of random choices of y for an input x. Then $x \in L$ if and only if the set of certificates for x is "large". The one-sided error protocol is:

1. Arthur draws randomly a few "shifts" $s_1, \cdots, s_m$ for some m;
2. Merlin returns y and $z_1, \cdots, z_m$;
3. Arthur verifies that $M(x, y \oplus s_i, z_i) = 1$ for all i.

We have analyzed the error probability in the $\mathtt{BPP} \subseteq \Sigma_2^p$ proof and will not repeat here. □

Using Theorem 5, we have the following containments.

Theorem 6. $\mathtt{NP} \subseteq \mathtt{AM} \subseteq \Pi_2^p$ *and* $\mathtt{BPP} \subseteq \mathtt{AM}$.

Proof. $\mathtt{NP} \subseteq \mathtt{AM}$ and $\mathtt{BPP} \subseteq \mathtt{AM}$ are straightforward from Definition 3.

To see $\mathtt{AM} \subseteq \Pi_2^p$, we only need to show $\mathtt{AM}_1 \subseteq \Pi_2^p$ by Theorem 5. Rewriting Definition 4, for $L \in \mathtt{AM}_1$,

$$x \in L \Rightarrow \forall y \exists z, M(x, y, z) = 1;$$
$$x \notin L \Rightarrow \Pr_y[\exists z, M(x, y, z) = 1] \leq \frac{1}{4} \Rightarrow \exists y \forall z M(x, y, z) = 0.$$

This is indeed a Π_2^p expression. □

Remark (Bibliographic). Public coin system $\mathtt{AM}$ was introduced Babai [Bab85]. Relevant chapters are [AB09, Chapter 8.2].

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[Bab85] László Babai. Trading group theory for randomness. In *STOC*, pages 421–429. ACM, 1985.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[Raz98] Ran Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, 1998.

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

[VV86] Leslie G. Valiant and Vijay V. Vazirani. NP is as easy as detecting unique solutions. *Theor. Comput. Sci.*, 47(3):85–93, 1986.