Randomized Computation

Shuangjun Zhang

Fudan University

zhangsj17@fudan.edu.cn

June 16, 2019

Overview

Polynomial Identity Testing

Let A(x), B(x) be polynomials(over R) of degrees n

- Question: Is $A(x) = B(x)$?
- Naive Idea: Converting the two polynomials to their canonical forms($\Sigma_{i=0}^{n} a_i x^i$),two polynomials are equivalent iff all the coeficients in their canonical forms are equal.
- Another Idea: Reduce problem to checking zero's of a polynomial.

$$A(x) = B(x) \Rightarrow A(x) - B(x) = 0$$

  - If $A(x) = B(x)$, $\forall r \in R, A(r) - B(r) = 0$.
  - If $A(x) \neq B(x)$, $A(x) - B(x) = 0$ has at most n roots.

Polynomial Identity Testing

Algorithm PIT:
Input: A(x), B(x), with maximum degree is n

- Choose $r \in \{1, 2, ..., 100n\}$ at random;
- Computer A(r) and B(r);
- If $A(r) \neq B(r)$ Return 0;
- else Return 1;

Polynomial Identity Testing

Analysis:

- If $\exists$ r, s.t. A(r) $\neq$ B(r), then A(x) $\neq$ B(x),that is

$$Pr[A(x) \neq B(x) | PIT() = 0] = 1$$

- If $A(x) = B(x)$,then $\forall$ r, $A(r) = B(r)$,that is

$$Pr[PIT() = 1 | A(x) = B(x)] = 1$$

- If $A(x) \neq B(x)$,then with probability at most $1/100$, $A(r) = B(r)$, that is

$$Pr[PIT() = 1 | A(x) \neq B(x)] \leq 1/100$$

Probabilistic Turing Machines(PTM)

- So far, considered the following types of machines for input x:
  - TM: only one computation that either accepts or rejects
  - NTM: there are many sequences of computation, and x accepted if there exists a sequence that accepts
  - ATM: many sequences of computations, and x accepted based on the type of nodes ($\exists$ or $\forall$) along its path
  - OTM:a TM with access to an oracle
- Another kind of machine: Probabilistic Turing Machine
  - Very similar to a TM , except at each step, M could toss a coin to get a random bit, and decide which transition rule to follow based on the outcome of the coin toss
- A PTM takes input x and a random bit string r
- $Pr[M(x, r) = z]$: probability M outputs z

Failure Probability

A major aspect of randomized algorithms or probabilistic machines is that they may fail to produce the desired output with some specified failure probability.

- One-sided error
  - The machine may err only in one direction i.e. either on 'yes' instances or on 'no' instances.
- Two-sided error
  - The machine may err in both directions i.e. it may result a 'yes' instance as a 'no' instance and vice versa.
- Zero-sided error
  - The algorithm never provides a wrong answer.
  - But sometimes it returns 'don't know'

Randomized Complexity Classes

- RP (Randomized Polynomial time)
  - One-sided error
- coRP (complement of RP)
  - One-sided error
- BPP (Bounded-error Probabilistic Polynomial time)
  - Two-sided error
- ZPP (Zero-error Probabilistic Polynomial time)
  - Zero error

RP and coRP

Definition (RP)

The complexity class RP is the class of all languages L for which there exists a polynomial PTM M such that

$$x \in L \Rightarrow Pr[M(x) = 1] \geq 1/2$$

$$x \notin L \Rightarrow Pr[M(x) = 0] = 1$$

Definition (coRP)

The complexity class coRP is the class of all languages L for which there exists a polynomial PTM M such that

$$x \in L \Rightarrow Pr[M(x) = 1] = 1$$

$$x \notin L \Rightarrow Pr[M(x) = 0] \geq 1/2$$

RP is in NP

Theorem

$$RP \subseteq NP$$

Proof.

- Let L be a language in RP . Let M be a polynomial probabilistic Turing Machine that decides L.
- If x $\in$ L, then there exists a sequence of coin tosses r such that M accepts x with r as the random string.
- So we can consider r as a certificate instead of a random string. r can be verified in polynomial time by the same machine.
- If x $\notin$ L, then $Pr[M(x) = 1] = 0$. So there is no certificate r s.t. $M(x, r) = 1$.
- So, L $\in$ NP

$\square$

Error Reduction for RP

- The constant $1/2$ in the definition of RP can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:
- Clearly, if $x \notin L$, all t runs will return a 0
- If $x \in L$, if any of the t runs returns a 1, we return the correct answer.
- If $x \in L$, if all t runs returns 0, we make mistake
- Pr[algorithm makes a mistake t times] $\leq \frac{1}{2^t}$
- Thus, we can make the error exponentially small by polynomial number of repetitions.

Bounded-error Probabilistic Polynomial time

Definition (BPP)

The complexity class BPP is the class of all languages L for which there exists a polynomial PTM M such that

$$x \in L \Rightarrow Pr[M(x) = 1] \geq 2/3$$

$$x \notin L \Rightarrow Pr[M(x) = 1] \leq 1/3$$

M answers correctly with probability 2/3 on any input x regardless if $x \in L$ or $x \notin L$

$P \subseteq BPP \subseteq EXP$

$BPP = coBPP$

Error Reduction for BPP

- $2/3$ is arbitrary and can be improved as follows:
  - Repeat the algorithm t times, say it returns X_i at the i-th run
  - Take majority answer, i.e., if $\geq$ t/2 times M returns 1, return 1, otherwise return 0
- The Chernoff Bound
  - Suppose $X_1, ..., X_t$ are t independent random variables with values in $\{0, 1\}$ and for every i, $Pr[X_i = 1] = p$. Then

$$Pr[\frac{1}{t}\sum_{i=1}^{t} X_i - p > \epsilon] < e^{-\epsilon^2 \frac{t}{2p(1-p)}}$$

$$Pr[\frac{1}{t}\sum_{i=1}^{t} X_i - p < -\epsilon] < e^{-\epsilon^2 \frac{t}{2p(1-p)}}$$

Error Reduction for BPP

- x $\in$ L, $Pr[X_i = 1] = p \geq 2/3$, M outputs correctly if $\sum_{i=1}^{t} X_i \geq t/2$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < t/2$
- Pr[Algorithm outputs the wrong answer on x]
  $= Pr[\sum_{i=1}^{t} X_i < t/2]$
  $= Pr[\frac{1}{t}\sum_{i=1}^{t} X_i < 1/2]$
  $= Pr[\frac{1}{t}\sum_{i=1}^{t} X_i - p < 1/2 - p]$
  $< e^{-(\frac{1}{2}-p)^2 \frac{t}{2p(1-p)}}$
  $\leq e^{-2t(\frac{1}{2}-p)^2}$
  $\leq e^{-2t(\frac{1}{2}-\frac{2}{3})^2}$
  $= e^{-ct}$, where c is some constant
  $= \frac{1}{2^n}$ if we set $t = \frac{n \ln 2}{c} = O(n)$
- So by running the algorithm O(n) times, reduce probability exponentially.

Another Definition of BPP

Definition (BPP)

The complexity class BPP is the class of all languages L for which there exists a polynomial PTM M such that

$$x \in L \Rightarrow Pr[M(x) = 1] \geq 1 - \frac{1}{2^{|x|}}$$

$$x \notin L \Rightarrow Pr[M(x) = 1] \leq \frac{1}{2^{|x|}}$$

M answers correctly with probability $1 - \frac{1}{2^{|x|}}$ on any input x regardless if x $\in$ L or x $\notin$ L

RP $\cup$ coRP $\subseteq$ BPP.

We have showed that $RP \subseteq NP$, But We don't know if $BPP \subseteq NP$

Zero-error Probabilistic Polynomial time

Definition (ZPP)

The complexity class ZPP is the class of all languages L for which there exists a polynomial PTM M such that

$$x \in L \Rightarrow M(x) = 1 \ or \ M(x) = \ 'Don't \ know'$$

$$x \notin L \Rightarrow M(x) = 0 \ or \ M(x) = \ 'Don't \ know'$$

$$\forall x, Pr[M(x) = \ 'Don't \ know'] \leq 1/2$$

Whenever M answers with a 0 or a 1, it answers correctly. If M is not sure, it'll output a 'Don't know'. On any input x, it outputs 'Don't know' with probability at most 1/2.

Another Definition of ZPP

Algorithm M':

Input:x

while(1):

- Invoke M to determine $x \in L$ or not;
- If $b = M(x)$, ouput b and halt;
- else continue;

If the running time of M is $T(|x|)$, the expecting running time of M is $2T(|x|)$.

Definition (ZPP)

The complexity class ZPP is the class of all languages L for which there exists a expecting polynomial PTM M such that

$$x \in L \Rightarrow M(x) = 1$$

$$x \notin L \Rightarrow M(x) = 0$$

A Theorem on ZPP

Theorem

$ZPP = RP \cap coRP$

- This theorem is surprising, since the corresponding question for nondeterministic(i.e. $\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP}$) is open.
- We have to prove both ways:

$$ZPP \subseteq RP \cap coRP$$

$$RP \cap coRP \subseteq ZPP$$

First direction

$$ZPP \subseteq RP \cap coRP$$

- We prove ZPP $\subseteq$ coRP, the other proof is similar
- Let L $\in$ ZPP. Then $\exists$ PTM M that, $\forall$ x, either correctly decides x $\in$ L or outputs 'Don't know'
- Let M' be the Turing Machine that on input x, returns 1 if M(x) ='Don't know' and otherwise returns M(x).
  - x $\in$ L: M(x) = 1 or M(x) ='Don't know', so M'(x) = 1.
  - x $\notin$ L: M(x) = 0, which is correct, or M(x) = 'Don't know' where M'(x) returns incorrectly with probability $\leq 1/2$
- So L $\in$ coRP, and ZPP $\subseteq$ coRP

Other direction

$$RP \cap coRP \subseteq ZPP$$

- L $\in$ RP $\cap$ coRP. Then there exist two TM's s.t.:
  - $M_1(x) = 1$ if x $\in$ L. Incorrect for x $\notin$ L with prob. $\leq 1/2$
  - $M_2(x) = 0$ if x $\notin$ L. Incorrect for x $\in$ L with prob. $\leq 1/2$
- Construct a PTM M' which works as follows on M'(x):
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin$ L
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in$ L
  - Else return 'Don't know'
- Claim: If M'(x) returns a 0 or a 1, it is correct.
- Claim: M'(x) returns 'Don't know' with prob. $\leq 1/2$
  - Assume x $\in$ L.
  - Pr[M' return 'Don't know'] $=$ Pr$[M_1(x) = 1 \wedge M_2(x) = 0]$
    $=$ Pr$[M_2(x) = 0] \leq 1/2$
  - The case for x $\notin$ L similar
- So, L $\in$ ZPP

BPP is in P/poly

Theorem

$BPP \subseteq P/poly$

Proof:

- L $\in$ BPP. Define L(x), L(x) $=$ 1 if x $\in$ L, else L(x) $=$ 0
- $\exists$ A PTM M on input x $\in \{0,1\}^n$, use m random bits s.t. $\forall$ x $\in \{0,1\}^n$, Pr[M(x, r) $\neq$ L(x)] $\leq 2^{-n-1}$
- Say r $\in \{0,1\}^m$ is bad for an input x $\in \{0,1\}^n$ if M(x, r) $\neq$ L(x), otherwise call r good for x
- $\forall$ x, at most $\frac{2^m}{2^{n+1}}$ string r are bad for x
- Adding over all x $\in \{0,1\}^n$ there are at most $2^n \frac{2^m}{2^{n+1}} = 2^m/2$ string r are bad
- So, $\exists$ $r_0 \in \{0,1\}^m$ good for every x $\in \{0,1\}^n$
- We can use r_0 as a advice, so L $\in$ P/poly

BPP is in PH

Theorem

$BPP \subseteq \Sigma_2 \cap \Pi_2$

- First, we prove BPP $\subseteq \Sigma_2$
- Suppose L $\in$ BPP, then $\exists$ ppt M for L s.t.

$$x \in L \Rightarrow Pr[M(x,r) = 1] \geq 1 - 2^{-n}$$

$$x \notin L \Rightarrow Pr[M(x,r) = 1] \leq 2^{-n}$$

$|r| = m = poly(n)$

- for x $\in \{0,1\}^n$, let S_x denote the set of r's for which M(x,r) $=$ 1. Then:

$$x \in L \Rightarrow |S_x| \geq (1 - 2^{-n})2^m$$

$$x \notin L \Rightarrow |S_x| \leq (2^{-n})2^m$$

BPP is in PH

- For a set S $\subseteq \{0,1\}^m$ and a string u $\in \{0,1\}^m$. Define $S+u=\{x+u : x \in S\}$. Let $k=\lceil \frac{m}{n} \rceil +1$, we have the following two claims.
- Claim 1: For every set S $\subseteq \{0,1\}^m$ with $|S| \leq 2^{m-n}$ and every k vectors $u_1, ..., u_k$, $\cup_{i=1}^{k}(S+u_i) \neq \{0,1\}^m$
- Proof of Claim 1:
  - Since |S+u_i| = |S|, $|\cup_{i=1}^{k}(S+u_i)| \leq$ k|S+u_i| = k|S| $< 2^m$(for sufficiently large n)
- Claim 2: For every set S $\subseteq \{0,1\}^m$ with $|S| \geq (1-2^{-n})2^m$, there exists $u_1, ..., u_k$, s.t. $\cup_{i=1}^{k}(S+u_i) = \{0,1\}^m$

BPP is in PH

Proof of Claim 2:

- We claim if $u_1, ..., u_k$ are chosen independently at random then $\Pr[\cup_{i=1}^{k}(S + u_i) = \{0,1\}^m] > 0$
- For r $\in \{0,1\}^m$, let B_r denote the "bad event" r $\notin \cup_{i=1}^{k}(S + u_i)$. It's suffice to show $Pr[\exists r \in \{0,1\}^m \; B_r] < 1$
- $Pr[\exists r \in \{0,1\}^m \; B_r] \leq \sum_{r\in\{0,1\}^m} Pr[B_r] \leq 2^m max\{Pr[B_r]\}$ we should only prove $\forall$ r, $Pr[B_r] < 2^{-m}$
- Let B_r^i denote the event r $\notin S + u_i$(equivalently, $r + u_i \notin S$), then $B_r = \cap_{i\in[k]} B_r^i$
- $r + u_i$ is a uniform element in $\{0,1\}^m$, it will be in S with probability at least 1-2^{-n}. So,

$$Pr[B_r^i] = Pr[r + u_i \notin S] = 1 - Pr[r + u_i \notin S] \leq 2^{-n}$$

- Since the events B_r^i are independent, we have

$$Pr[Br] = Pr[\cap_{i\in[k]} B_r^i] = Pr[B_r^i]^k \leq 2^{-nk} < 2^{-m}$$

BPP is in PH

- Claim 1: For every set S $\subseteq \{0,1\}^m$ with $|S| \leq 2^{m-n}$ and every k vectors $u_1, \ldots, u_k$, $\cup_{i=1}^{k}(S + u_i) \neq \{0,1\}^m$
- Claim 2: For every set S $\subseteq \{0,1\}^m$ with $|S| \leq (1 - 2^{-n})2^m$, there exists $u_1, \ldots, u_k$, s.t. $\cup_{i=1}^{k}(S + u_i) = \{0,1\}^m$
- Together with Claim 1 and 2, we know:
  - x $\in$ L $\Rightarrow |S_x| \geq (1 - 2^{-n})2^m$
    $\Rightarrow \exists\ u_1, \ldots, u_k \in \{0,1\}^m\ \forall$ r $\in \{0,1\}^m$ r $\in \cup_{i=1}^{k}(S_x + u_i)$
  - x $\notin$ L $\Rightarrow |S_x| \leq 2^{m-n}$
    $\Rightarrow \forall\ u_1, \ldots, u_k \in \{0,1\}^m\ \exists$ r $\in \{0,1\}^m$ r $\notin \cup_{i=1}^{k}(S_x + u_i)$
  - Whether r $\in \cup_{i=1}^{k}(S_x + u_i)$ or not can be decided in polynomial time.
- So, L $\in \Sigma_2$, BPP $\subseteq \Sigma_2$
- L $\in$ BPP $\Rightarrow \overline{L} \in coBPP = BPP \Rightarrow \overline{L} \in \Sigma_2 \Rightarrow L \in co\Sigma_2 = \Pi_2$
- So, BPP $\subseteq \Pi_2$, BPP $\subseteq \Sigma_2 \cap \Pi_2$

Unsolved Problem about BPP

- BPP $=$ P?
- Complete problem for BPP?
- Does BPTIME have a hierarchy theorem?

The End