Randomized Computation

Antonis Antonopoulos

CoReLab - NTUA

January 2012

Warmup: Randomized Quicksort

Deterministic Quicksort

Input: A list L of integers;

If $n \leq 1$ then return L.

Else {

- let $i = 1$;
- let L_1 be the sublist of L whose elements are $< a_i$;
- let L_1 be the sublist of L whose elements are $= a_i$;
- let L_1 be the sublist of L whose elements are $> a_i$;
- Recursively Quicksort L_1 and L_3;
- return $L = L_1 L_2 L_3$;

Warmup: Randomized Quicksort

Randomized Quicksort

Input: A list L of integers;

If $n \leq 1$ then return L.

Else {

- choose a random integer i, $1 \leq i \leq n$;
- let L_1 be the sublist of L whose elements are $< a_i$;
- let L_1 be the sublist of L whose elements are $= a_i$;
- let L_1 be the sublist of L whose elements are $> a_i$;
- Recursively Quicksort L_1 and L_3;
- return $L = L_1 L_2 L_3$;

Warmup: Randomized Quicksort

- Let T_d the *max* number of comparisons for the Deterministic Quicksort:

$$T_d \geq T_d(n-1) + \mathcal{O}(n)$$
$$\Downarrow$$
$$T_d(n) = \Omega(n^2)$$

Warmup: Randomized Quicksort

- Let T_d the *max* number of comparisons for the Deterministic Quicksort:

$$T_d \geq T_d(n-1) + \mathcal{O}(n)$$

$$\Downarrow$$

$$T_d(n) = \Omega(n^2)$$

- Let T_r the *expected* number of comparisons for the Randomized Quicksort:

$$T_r \geq \frac{1}{n}\sum_{j=0}^{n-1}[T_r(j) - T_r(n-1-j)] + \mathcal{O}(n)$$

$$\Downarrow$$

$$T_r(n) = \mathcal{O}(n \log n)$$

Warmup: Polynomial Identity Testing

1. Two polynomials are equal if they have the same coefficients for corresponding powers of their variable.
2. A polynomial is *identically zero* if all its coefficients are equal to the additive identity element.
3. How we can test if a polynomial is identically zero?

Warmup: Polynomial Identity Testing

1. Two polynomials are equal if they have the same coefficients for corresponding powers of their variable.
2. A polynomial is *identically zero* if all its coefficients are equal to the additive identity element.
3. How we can test if a polynomial is identically zero?
4. We can choose uniformly at random $r_1, \dots, r_n$ from a set $S \subseteq \mathbb{F}$.
5. We are wrong with a probability at most:

Theorem (Schwartz-Zippel Lemma)

Let $Q(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$ be a multivariate polynomial of total degree d. Fix any finite set $S \subseteq \mathbb{F}$, and let $r_1, \dots, r_n$ be chosen indepedently and uniformly at random from S. Then:

$$\mathbf{Pr}[Q(r_1, \dots, r_n) = 0 | Q(x_1, \dots, x_n) \neq 0] \leq \frac{d}{|S|}$$

Warmup: Polynomial Identity Testing

Proof:

(*By Induction on n*)

- For $n = 1$: $\mathbf{Pr}[Q(r) = 0 | Q(x) \neq 0] \leq d/|S|$
- For n:

$$Q(x_1, \ldots, x_n) = \sum_{i=0}^{k} Q_i(x_2, \ldots, x_n)$$

where $k \leq d$ is the smallest exponent of x_1 in Q.
$deg(Q_k) \leq d - k \Rightarrow \mathbf{Pr}[Q_k(r_2, \ldots, r_n)] \leq (d-k)/|S|$
Suppose that $Q_k(r_1, \ldots, r_n) \neq 0$. Then:

$$q(x_1) = Q(x_1, r_2, \ldots, r_n) = \sum_{i=0}^{k} x_1^i Q_i(r_2, \ldots, r_n)$$

$deg(q(x_1)) = k$, and $q(x_1) \neq 0$!

Warmup: Polynomial Identity Testing

Proof (cont'd):
The base case now implies that:

$$\mathbf{Pr}[q(r_1) = Q(r_1, \ldots, r_n) = 0] \leq k/|S|$$

Thus, we have shown the following two equalities:

$$\mathbf{Pr}[Q_k(r_2, \ldots, r_n) = 0] \leq \frac{d-k}{|S|}$$

$$\mathbf{Pr}[Q_k(r_1, r_2, \ldots, r_n) = 0 | Q_k(r_2, \ldots, r_n) \neq 0] \leq \frac{k}{|S|}$$

Using the following identity: $\mathbf{Pr}[\mathcal{E}_1] \leq \mathbf{Pr}[\mathcal{E}_1|\overline{\mathcal{E}}_2] + \mathbf{Pr}[\mathcal{E}_2]$ we obtain that the requested probability is no more than the sum of the above, which proves our theorem! $\square$

Probabilistic Turing Machines

- A Probabilistic Turing Machine is a TM as we know it, but with access to a "random source", that is an extra (read-only) tape containing *random-bits*!
- Randomization on:
  - **Output** (one or two-sided)
  - **Running Time**

Definition (Probabilistic Turing Machines)

A Probabilistic Turing Machine is a TM with two transition functions δ_0, δ_1. On input x, we choose in each step with probability $1/2$ to apply the transition function δ_0 or δ_1, indepedently of all previous choices.

- We denote by $M(x)$ the *random variable* corresponding to the output of M at the end of the process.
- For a function $T : \mathbb{N} \rightarrow \mathbb{N}$, we say that M runs in $T(|x|)$-time if it halts on x within $T(|x|)$ steps (*regardless of the random choices it makes*).

BPP Class

Definition (BPP Class)

For $T : \mathbb{N} \to \mathbb{N}$, let $\mathbf{BPTIME}(T(n))$ the class of languages L such that there exists a PTM which halts in $\mathcal{O}(T(|x|))$ time on input x, and $\mathbf{Pr}[M(x) = L(x)] \geq 2/3$.
We define:

$$\mathbf{BPP} = \bigcup_{c \in \mathbb{N}} \mathbf{BPTIME}(n^c)$$

- The class **BPP** represents our notion of efficient (randomized) computation!
- We can also define **BPP** using certificates:

BPP Class

Definition (Alternative Definition of BPP)

A language $L \in \mathbf{BPP}$ if there exists a poly-time TM M and a polynomial $p \in poly(n)$, such that for every $x \in \{0,1\}^*$:

$$\mathbf{Pr}_{r \in \{0,1\}^{p(n)}}[M(x,r) = L(x)] \geq \frac{2}{3}$$

- $\mathbf{P} \subseteq \mathbf{BPP}$
- $\mathbf{BPP} \subseteq \mathbf{EXP}$
- The "**P** vs **BPP**" question.

Quantifier Characterizations

- Proper formalism (*Zachos et al.*):

Definition (Majority Quantifier)

Let $R : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ be a predicate, and ε a rational number, such that $\varepsilon \in \left(0, \frac{1}{2}\right)$. We denote by $(\exists^+ y, |y| = k)R(x,y)$ the following predicate:

> *"There exist at least $\left(\frac{1}{2} + \varepsilon\right) \cdot 2^k$ strings y of length m for which $R(x,y)$ holds."*

We call $\exists^+$ the *overwhelming majority* quantifier.

- $\exists_r^+$ means that the fraction r of the possible certificates of a certain length satisfy the predicate for the certain input.

Quantifier Characterizations

Definition

We denote as $\mathcal{C} = (Q_1/Q_2)$, where $Q_1, Q_2 \in \{\exists, \forall, \exists^+\}$, the class $\mathcal{C}$ of languages L satisfying:

- $x \in L \Rightarrow Q_1 y \; R(x,y)$
- $x \notin L \Rightarrow Q_2 y \; \neg R(x,y)$

- $\mathbf{P} = (\forall/\forall)$
- $\mathbf{NP} = (\exists/\forall)$
- $co\mathbf{NP} = (\forall/\exists)$
- $\mathbf{BPP} = (\exists^+/\exists^+) = co\mathbf{BPP}$

RP Class

- In the same way, we can define classes that contain problems with one-sided error:

Definition

The class $\textbf{RTIME}(T(n))$ contains every language L for which there exists a PTM M running in $\mathcal{O}(T(|x|))$ time such that:

- $x \in L \Rightarrow \textbf{Pr}[M(x) = 1] \geq \frac{2}{3}$
- $x \notin L \Rightarrow \textbf{Pr}[M(x) = 0] = 1$

We define

$$\textbf{RP} = \bigcup_{c \in \mathbb{N}} \textbf{RTIME}(n^c)$$

- Similarly we define the class *co***RP**.

RP Class

- $\mathbf{RP} \subseteq \mathbf{NP}$, since every accepting "branch" is a certificate!
- $\mathbf{RP} \subseteq \mathbf{BPP}$, $co\mathbf{RP} \subseteq \mathbf{BPP}$
- $\mathbf{RP} = (\exists^+/\forall)$

RP Class

- $\mathbf{RP} \subseteq \mathbf{NP}$, since every accepting "branch" is a certificate!
- $\mathbf{RP} \subseteq \mathbf{BPP}$, $co\mathbf{RP} \subseteq \mathbf{BPP}$
- $\mathbf{RP} = (\exists^+/\forall) \subseteq (\exists/\forall) = \mathbf{NP}$

RP Class

- $\mathbf{RP} \subseteq \mathbf{NP}$, since every accepting "branch" is a certificate!
- $\mathbf{RP} \subseteq \mathbf{BPP}$, $co\mathbf{RP} \subseteq \mathbf{BPP}$

- $\mathbf{RP} = (\exists^+/\forall) \subseteq (\exists/\forall) = \mathbf{NP}$
- $co\mathbf{RP} = (\forall/\exists^+) \subseteq (\forall/\exists) = co\mathbf{NP}$

RP Class

- $\mathbf{RP} \subseteq \mathbf{NP}$, since every accepting "branch" is a certificate!
- $\mathbf{RP} \subseteq \mathbf{BPP}$, $co\mathbf{RP} \subseteq \mathbf{BPP}$

- $\mathbf{RP} = (\exists^+/\forall) \subseteq (\exists/\forall) = \mathbf{NP}$
- $co\mathbf{RP} = (\forall/\exists^+) \subseteq (\forall/\exists) = co\mathbf{NP}$

Theorem (Decisive Characterization of BPP)

$$\mathbf{BPP} = (\exists^+/\exists^+) = (\exists^+\forall/\forall\exists^+) = (\forall\exists^+/\exists^+\forall)$$

ZPP Class

- And now something completely different:
- What is the random variable was the running time and not the output?

ZPP Class

- And now something completely different:
- What is the random variable was the running time and not the output?
- We say that M has expected running time $T(n)$ if the expectation $\mathbf{E}[T_{M(x)}]$ is at most $T(|x|)$ for every $x \in \{0,1\}^*$. ($T_{M(x)}$ is the running time of M on input x, and it is a **random variable**!)

Definition

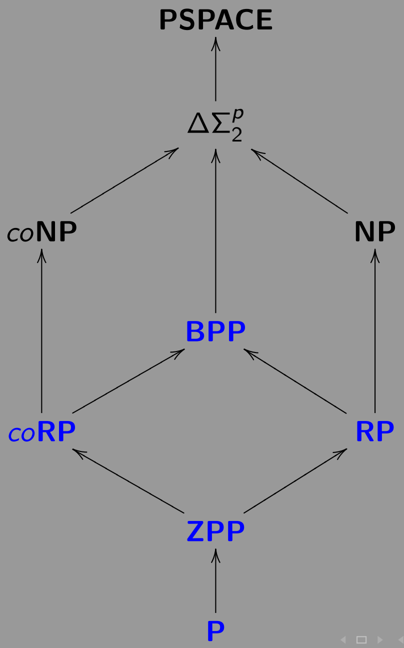
The class $\mathbf{ZPTIME}(T(n))$ contains all languages L for which there exists a machine M that runs in an expected time $\mathcal{O}(T(|x|))$ such that for every input $x \in \{0,1\}^*$, whenever M halts on x, the output $M(x)$ it produces is exactly $L(x)$. We define:

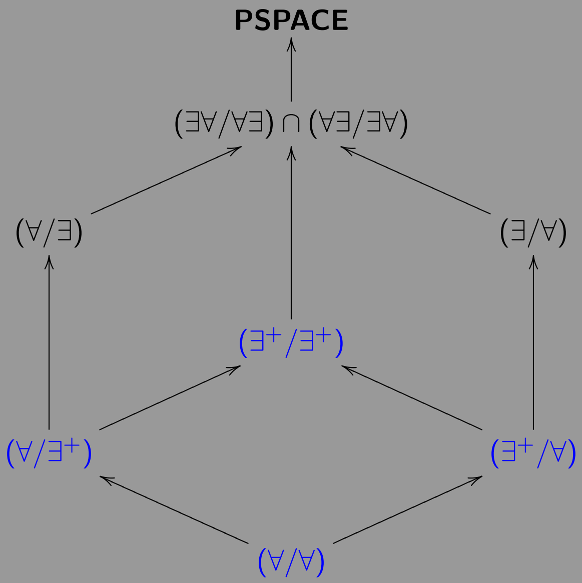
$$\mathbf{ZPP} = \bigcup_{c \in \mathbb{N}} \mathbf{ZTIME}(n^c)$$

ZPP Class

- The output of a **ZPP** machine is always correct!
- The problem is that we aren't sure about the running time.
- We can easily see that $\mathbf{ZPP} = \mathbf{RP} \cap co\mathbf{RP}$.

- The next Hasse diagram summarizes the previous inclusions:
  (Recall that $\Delta\Sigma_2^p = \Sigma_2^p \cap \Pi_2^p = \mathbf{NP}^{\mathbf{NP}} \cap co\mathbf{NP}^{\mathbf{NP}}$)


PSPACE
$\Delta\Sigma_2^p$
coNP
NP
BPP
coRP
RP
ZPP
P


PSPACE
$(\exists\forall/\forall\exists) \cap (\forall\exists/\exists\forall)$
$(\forall/\exists)$
$(\exists/\forall)$
$(\exists^+/\exists^+)$
$(\forall/\exists^+)$
$(\exists^+/\forall)$
$(\forall/\forall)$

Error Reduction for BPP

Theorem (Error Reduction for BPP)

Let $L \subseteq \{0,1\}^*$ be a language and suppose that there exists a poly-time PTM M such that for every $x \in \{0,1\}^*$:

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

Then, for every constant $d > 0$, $\exists$ poly-time PTM M' such that for every $x \in \{0,1\}^*$:

$$\mathbf{Pr}[M(x) = L(x)] \geq 1 - 2^{-|x|^d}$$

Proof: The machine M' does the following:

- `Run` $M(x)$ `for every input` x `for` $k = 8|x|^{2c+d}$ `times, and obtain outputs` $y_1, y_2, \dots, y_k \in \{0, 1\}$.
- `If the majority of these outputs is 1, return 1`
- `Otherwise, return 0.`

We define the r.v. X_i for every $i \in [k]$ to be 1 if $y_i = L(x)$ and 0 otherwise. $X_1, X_2, \dots, X_k$ are indepedent Boolean r.v.'s, with:

$$\mathbf{E}[X_i] = \mathbf{Pr}[X_i = 1] \geq \frac{1}{2} + |x|^{-c}$$

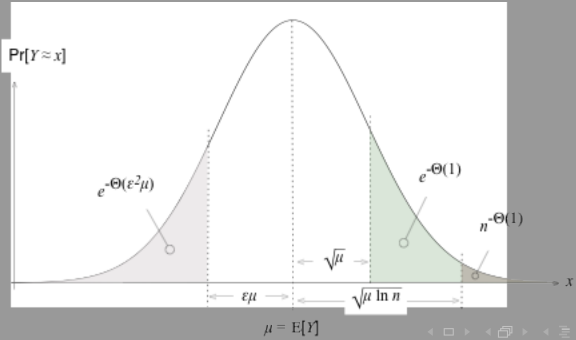
Applying a Chernoff Bound we obtain:

$$\mathbf{Pr}\left[|\sum_{i=1}^{k} X_i - pk| > \delta pk \right] < e^{-\frac{\delta^2}{4} pk} = e^{-\frac{1}{4|x|^{2c}} \frac{1}{2} 8|x|^{2c+d}} \leq 2^{-|x|^d}$$

□

Intermission: Chernoff Bounds

- *How many* samples do we need in order to estimate μ up to an *error* of $\pm\varepsilon$ with *probability* at least $1-\delta$?
- Chernoff Bound tells us that this number is $\mathcal{O}(\rho/\varepsilon^2)$, where $\rho = \log(1/\delta)$.
- The probability that k is $\rho\sqrt{n}$ far from μn decays **exponentially** with ρ.



Intermission: Chernoff Bounds

$$\mathbf{Pr}\left[\sum_{i=1}^{n} X_i \geq (1+\delta)\mu\right] \leq \left[\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right]^{\mu}$$

$$\mathbf{Pr}\left[\sum_{i=1}^{n} X_i \leq (1-\delta)\mu\right] \leq \left[\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right]^{\mu}$$

Other useful form is:

$$\mathbf{Pr}\left[\left|\sum_{i=1}^{n} X_i - \mu\right| \geq c\mu\right] \leq 2e^{-\min\{c^2/4, c/2\}\cdot\mu}$$

- This probability is bounded by $2^{-\Omega(\mu)}$.

Error Reduction for BPP

- From the above we can obtain the following interesting corollary:

Corollary

For $c > 0$, let $\mathbf{BPP}_{1/2+n^{-c}}$ denote the class of languages L for which there is a polynomial-time PTM M satisfying $\mathbf{Pr}[M(x) = L(x)] \geq 1/2 + |x|^{-c}$ for every $x \in \{0,1\}^*$. Then:

$$\mathbf{BPP}_{1/2+n^{-c}} = \mathbf{BPP}$$

- Obviously, $\exists^+ = \exists^+_{1/2+\varepsilon} = \exists^+_{2/3} = \exists^+_{3/4} = \exists^+_{0.99} = \exists^+_{1-2^{-p(|x|)}}$

Complete Problems for BPP?

- The defining property of **BPTIME** machines is **semantic**!
- We cannot test whether a TM can accept every input string with probability $\geq 2/3$ or with $\leq 1/3$ (**why?**)
- In contrast, the defining property of **NP** is **syntactic**!
- We have:
  - **Syntactic Classes**
  - **Semantic Classes**
- If finally **P** $=$ **BPP**, then **BPP** will have complete problems!!

Complete Problems for BPP?

- The defining property of **BPTIME** machines is **semantic**!
- We cannot test whether a TM can accept every input string with probability $\geq 2/3$ or with $\leq 1/3$ (**why?**)
- In contrast, the defining property of **NP** is **syntactic**!
- We have:
  - **Syntactic Classes**
  - **Semantic Classes**
- If finally $\mathbf{P} = \mathbf{BPP}$, then **BPP** will have complete problems!!
- For the same reason, in semantic classes we cannot prove Hierarchy Theorems using Diagonalization.

The Class PP

Definition

A language $L \in \mathbf{PP}$ if there exists a poly-time TM M and a polynomial $p \in poly(n)$, such that for every $x \in \{0,1\}^*$:

$$\mathbf{Pr}_{r \in \{0,1\}^{p(n)}}[M(x,r) = L(x)] \geq \frac{1}{2}$$

- Or, more "syntactically":

Definition

A language $L \in \mathbf{PP}$ if there exists a poly-time TM M and a polynomial $p \in poly(n)$, such that for every $x \in \{0,1\}^*$:

$$x \in L \Leftrightarrow \left|\left\{y \in \{0,1\}^{p(|x|)} : M(x,y) = 1\right\}\right| \geq \frac{1}{2} \cdot 2^{p(|x|)}$$

The Class PP

- Due to the lack of a gap between the two cases, we cannot amplify the probability with polynomially many repetitions, as in the case of **BPP**.
- **PP** is closed under complement.
- A breakthrough result of R. Beigel, N. Reingold and D. Spielman is that **PP** is closed under *intersection*!

The Class PP

- Due to the lack of a gap between the two cases, we cannot amplify the probability with polynomially many repetitions, as in the case of **BPP**.
- **PP** is closed under complement.
- A breakthrough result of R. Beigel, N. Reingold and D. Spielman is that **PP** is closed under *intersection*!
- The syntactic definition of **PP** gives the possibility for *complete problems*:
- Consider the problem `MAJSAT`:
  Given a Boolean Expression, is it true that the majority of the 2^n truth assignments to its variables (that is, at least $2^{n-1} + 1$ of them) satisfy it?

The Class PP

Theorem

MAJSAT is **PP***-complete!*

- MAJSAT is not likely in **NP**, since the (*obvious*) certificate is not very succinct!

The Class PP

Theorem

MAJSAT is **PP***-complete!*

- `MAJSAT` is not likely in **NP**, since the (*obvious*) certificate is not very succinct!

Theorem

$$\mathbf{NP} \subseteq \mathbf{PP} \subseteq \mathbf{PSPACE}$$

The Class PP

Theorem

MAJSAT is **PP***-complete!*

- MAJSAT is not likely in **NP**, since the (*obvious*) certificate is not very succinct!

Theorem

$$\mathbf{NP} \subseteq \mathbf{PP} \subseteq \mathbf{PSPACE}$$

Proof:

It is easy to see that **PP** $\subseteq$ **PSPACE**:
We can simulate any **PP** machine by enumerating all strings y of length $p(n)$ and verify whether **PP** machine accepts. The **PSPACE** machine accepts if and only if there are more than $2^{p(n)-1}$ such y's (by using a counter).

The Class PP

Proof (cont'd):

Now, for **NP** $\subseteq$ **PP**, let $A \in$ **NP**. That is, $\exists p \in poly(n)$ and a poly-time and balanced predicate R such that:

$$x \in A \;\Leftrightarrow\; (\exists y, |y| = p(|x|)) : R(x, y)$$

Consider the following TM:

> *M accepts input (x, by), with $|b| = 1$ and $|y| = p(|x|)$, if and only if $R(x, y) = 1$ or $b = 1$.*

- If $x \in A$, then $\exists$ at least one y s.t. $R(x, y)$.
  Thus, $\mathbf{Pr}[M(x) \text{ accepts}] \geq 1/2 + 2^{-(p(n)+1)}$.
- If $x \notin A$, then $\mathbf{Pr}[M(x) \text{ accepts}] = 1/2$.

$\square$

Other Results

Theorem

If $\mathbf{NP} \subseteq \mathbf{BPP}$, *then* $\mathbf{NP} = \mathbf{RP}$.

Other Results

Theorem

If $\mathbf{NP} \subseteq \mathbf{BPP}$, *then* $\mathbf{NP} = \mathbf{RP}$.

Proof:

- It suffices to show that if $\mathtt{SAT} \in \mathbf{BPP}$, then $\mathtt{SAT} \in \mathbf{RP}$.
- Recall that SAT has the **self-reducibility** property: $\phi(x_1, \ldots, x_n)$: $\phi \in \mathtt{SAT} \Leftrightarrow (\phi|_{x_1=0} \in \mathtt{SAT} \ \vee \ \phi|_{x_1=1} \in \mathtt{SAT})$.
- $\mathtt{SAT} \in \mathbf{BPP}$: $\exists$ PTM M computing SAT with error probability bounded by $2^{-|\phi|}$.
- We can use the *self-reducibility* of SAT to produce a truth assignment for ϕ as follows:

Other Results

Proof (cont'd):

`Input:` `A Boolean formula` ϕ `with` n `variables`
If $M(\phi) = 0$ **then** `reject` ϕ;
For $i = 1$ `to` n
$\rightarrow$ **If** $M(\phi|_{x_1=\alpha_1,\ldots,x_{i-1}=\alpha_{i-1}x_i=0}) = 1$ **then** `let` $\alpha_i = 0$
$\rightarrow$ **ElseIf** $M(\phi|_{x_1=\alpha_1,\ldots,x_{i-1}=\alpha_{i-1}x_i=1}) = 1$ **then** `let` $\alpha_i = 1$
$\rightarrow$ **Else** `reject` ϕ `and halt;`
If $\phi|_{x_1=\alpha_1,\ldots,x_n=\alpha_n} = 1$ **then** `accept` F
Else `reject` F

Other Results

Proof (cont'd):

`Input:` `A Boolean formula` ϕ `with` n `variables`
If $M(\phi) = 0$ **then** `reject` ϕ;
For $i = 1$ `to` n
$\rightarrow$ **If** $M(\phi|_{x_1=\alpha_1,\dots,x_{i-1}=\alpha_{i-1}x_i=0}) = 1$ **then** `let` $\alpha_i = 0$
$\rightarrow$ **ElseIf** $M(\phi|_{x_1=\alpha_1,\dots,x_{i-1}=\alpha_{i-1}x_i=1}) = 1$ **then** `let` $\alpha_i = 1$
$\rightarrow$ **Else** `reject` ϕ `and halt;`
If $\phi|_{x_1=\alpha_1,\dots,x_n=\alpha_n} = 1$ **then** `accept` F
Else `reject` F

- Note that M_1 accepts ϕ *only if* a t.a. $t(x_i) = \alpha_i$ is found.
- Therefore, M_1 never makes mistakes if $\phi \notin \texttt{SAT}$.
- If $\phi \in \texttt{SAT}$, then M rejects ϕ on each iteration of the loop w.p. $2^{-|\phi|}$.
- So, $\mathbf{Pr}[M_1 \text{ accepting } x] = (1 - 2^{-|\phi|})^n$, which is greater than $1/2$ if $|\phi| \geq n > 1$. $\square$

Relativized Results

Theorem

Relative to a random oracle A, $\mathbf{P}^A = \mathbf{BPP}^A$. That is,

$$\mathbf{Pr}_A[\mathbf{P}^A = \mathbf{BPP}^A] = 1$$

Also,

- $\mathbf{BPP}^A \subsetneq \mathbf{NP}^A$, relative to a *random* oracle A.
- There exists an A such that: $\mathbf{P}^A \neq \mathbf{RP}^A$.
- There exists an A such that: $\mathbf{RP}^A \neq co\mathbf{RP}^A$
- There exists an A such that: $\mathbf{RP}^A \neq \mathbf{NP}^A$.

Relativized Results

Theorem

Relative to a random oracle A, $\mathbf{P}^A = \mathbf{BPP}^A$. That is,

$$\mathbf{Pr}_A[\mathbf{P}^A = \mathbf{BPP}^A] = 1$$

Also,

- $\mathbf{BPP}^A \subsetneq \mathbf{NP}^A$, relative to a *random* oracle A.
- There exists an A such that: $\mathbf{P}^A \neq \mathbf{RP}^A$.
- There exists an A such that: $\mathbf{RP}^A \neq co\mathbf{RP}^A$
- There exists an A such that: $\mathbf{RP}^A \neq \mathbf{NP}^A$.

Corollary

There exists an A such that:

$$\mathbf{P}^A \neq \mathbf{RP}^A \neq \mathbf{NP}^A \not\subseteq \mathbf{BPP}^A$$

Further Reading

- Sanjeev Arora and Boaz Barak, *Computational Complexity: A Modern Approach*, Cambridge University Press, 2009
- Ding Zhu Du, Ker-I Ko *Theory of Computational Complexity*, John Wiley & Sons Inc, 2000
- Rajeev Motwani, Prabhakar Raghavan *Randomized Algorithms*, Cambridge University Press, 1995
- Christos Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.
- S. Zachos, *Probabilistic quantifiers, adversaries, and complexity classes: an overview*. In Proc. of the conference on Structure in complexity theory, pages 383-400, New York, NY, USA, 1986. Springer-Verlag New York, Inc.
- S. Zachos and H. Heller, *A decisive characterization of BPP*. Information and Control, 69(1-3):125-135, 1986

Thank You!