Randomized complexity classes

Today: RP, coRP, and ZPP (what a zoo!)
Today

- Randomized Turing machines
- One-sided error: RP, coRP
- No error: ZPP
- Next time: two-sided error BPP
Randomized Turing machines
Ordinary Turing machines

- One **read-only** input tape $x$ (size $|x| = n$)
- As many **work tapes** as you need (but only a constant number!)
- (Possibly) one **write-only** output tape
Drawing strings at random

- We will study **probabilistic** complexity classes, where our TMs can now **draw** strings of bits at random.
We will study **probabilistic** complexity classes, where our TMs can now **draw** strings of bits at random.

- No need to invent a new TM model.
Drawing strings at random

- We will study **probabilistic** complexity classes, where our TMs can now **draw** strings of bits at random.
- No need to invent a new TM model.
- **Choice 1**: use a **non-deterministic** TM model and draw execution branch at random (we won’t do that; hard to do it right).
Drawing strings at random

- We will study **probabilistic** complexity classes, where our TMs can now **draw** strings of bits at random.
- No need to invent a new TM model.
- **Choice 1**: use a **non-deterministic** TM model and draw execution branch at random (we won’t do that; hard to do it right).
- **Choice 2**: … next slide.
Randomized Turing machines

- Two tapes
  - One \textit{read-only} tapes
    - As many \textit{work} tapes as you need
      (but only a constant number!)
  - (Possibly) one \textit{write-only} output tape
- \textit{random} tape
- \textit{input} tape

$(size=n)$
We draw the random tape $r$ uniformly at random
We draw the random tape $r$ uniformly at random.

We will be interested in probabilities, e.g. $\Pr_r [\mathcal{M}(x,r) \text{ accepts}]$. 
We draw the random tape $r$ uniformly at random.

We will be interested in probabilities, e.g. $\Pr_{r} [M(x,r) \text{ accepts}]$.

Random tape must not just be read-only: we impose that no bit on $r$ is ever read twice (otherwise bits read are not independent).
we need $r$ to contain at least $f(n)$ bits, where $f(n)$ is an upper bound on the time taken by the TM.
we need \( r \) to contain at least \( f(n) \) bits, where \( f(n) \) is an upper bound on the time taken by the TM.

We will always assume that \( r \) is large enough.
we need $r$ to contain at least $f(n)$ bits, where $f(n)$ is an upper bound on the time taken by the TM.

We will always assume that $r$ is large enough

OK for classes defined by worst-case time, will cause problems for classes defined with no a priori upper bound on time (e.g., ZPP)
Our first probabilistic class: \( \text{RP} \)

(also sometimes known as the class of *Monte Carlo* languages)
A language $L$ is in \textbf{RP} if and only if there is a \textit{polynomial-time} TM $M$ such that for every input $x$ (of size $n$):
A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

- i.e. there is also a polynomial $p(n)$ / $M(x,r)$ terminates in time $\leq p(n)$, where $n = |x|$, in the worst case (and for any value of $r$)
A language $L$ is in $\textbf{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

\[ M(x,r) \text{ terminates in time } \leq p(n), \]

where $n = |x|$, in the worst case (and for any value of $r$)

\[ \text{hence, implicitly, we require } |r| \geq p(n), \]

(let us say $|r| = p(n)$)
RP: **Randomized Polynomial time**

- A language $L$ is in **RP** if and only if there is a **polynomial-time** TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1/2$

---

i.e. there is also a **polynomial** $p(n)$ / $M(x, r)$ terminates in time $\leq p(n)$, where $n = |x|$, in the worst case (and for any value of $r$)

---

... hence, implicitly, we require $|r| \geq p(n)$ (let us say $|r| = p(n)$)
RP: Randomized Polynomial time

- A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$

i.e. there is also a polynomial $p(n)$ / $M(x,r)$ terminates in time $\leq p(n)$, where $n = |x|$, in the worst case (and for any value of $r$)

... hence, implicitly, we require $|r| \geq p(n)$ (let us say $|r| = p(n)$)

probability taken over all $r \in \{0,1\}^{p(n)}$
A language $L$ is in **RP** if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

- if $x \in L$ then $Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
- if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $Pr_r [M(x,r) \text{ accepts}] = 0$).

... hence, implicitly, we require $|r| \geq p(n)$ (let us say $|r| = p(n)$)

probability taken over all $r \in \{0,1\}^{p(n)}$

i.e. there is also a polynomial $p(n)$ / $M(x,r)$ terminates in time $\leq p(n)$, where $n = |x|$, in the worst case (and for any value of $r$)
RP: Randomized Polynomial time

- A language $L$ is in **RP** if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

  i.e. there is also a polynomial $p(n)$ / $M(x,r)$ terminates in time $\leq p(n)$, where $n = |x|$, in the worst case (and for any value of $r$)

  ... hence, implicitly, we require $|r| \geq p(n)$ (let us say $|r| = p(n)$)

  probability taken over all $r \in \{0,1\}^{p(n)}$

  one-sided error: we make no error if $x \notin L$
A language $L$ is in **RP** if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

- if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
- if $x \not\in L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

**One-sided error:** we make no error if $x \not\in L$.
RP: Randomized Polynomial time

- A language $L$ is in **RP** if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\text{Pr}_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\text{Pr}_r [M(x,r) \text{ accepts}]=0$).
RP: Randomized Polynomial time

- A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

Note: RP-languages are not defined by « RP-machines » (there is no such notion).
**RP: Randomized Polynomial time**

- A language $L$ is in **RP** if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

---

Note: **RP-languages** are **not** defined by « **RP-machines» » (there is no such notion)

---

... but if we wanted to define « **RP-machines» », those would be machines $M$ such that, for every $x$,
- either $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
- or $\Pr_r [M(x,r) \text{ accepts}] = 0$
coRP

- $L$ is in $\text{coRP}$ iff complement $L^c$ is in $\text{RP}$, hence:
coRP

- \( L \) is in \( \text{coRP} \) iff complement \( L^c \) is in \( \text{RP} \), hence:
- \( L \) is in \( \text{coRP} \) if and only if
  - there is a polynomial-time TM \( M \)
  - such that for every input \( x \) (of size \( n \)):
    - if \( x \in L \) then \( \Pr_r [M(x,r) \text{ accepts}] \geq 1/2 \)
    - if \( x \not\in L \) then \( M(x,r) \) accepts for no \( r \)
    - \( \Pr_r [M(x,r) \text{ accepts}] \leq 1/2 \)
A motivating example for (co)RP

❖ PRIMALITY
INPUT: a natural number \( p \), in binary
Q: is \( p \) prime?
A motivating example for (co)RP

PRIMALITY
INPUT: a natural number $p$, in binary
Q: is $p$ prime?

For a long time, not known to be in $P$
(now solved: indeed in $P$ [Agrawal, Kayal, Saxena 2004])
A motivating example for (co)RP

❖ PRIMALITY
INPUT: a natural number $p$, in binary
Q: is $p$ prime?

❖ For a long time, not known to be in $P$ (now solved: indeed in $P$ [Agrawal, Kayal, Saxena 2004])

❖ In coNP (guess a proper divisor)
A motivating example for (co)RP

❖ PRIMALITY
INPUT: a natural number \( p \), in binary
Q: is \( p \) prime?

❖ For a long time, not known to be in \( \text{P} \)
(now solved: indeed in \( \text{P} \) [Agrawal, Kayal, Saxena 2004])

❖ In \( \text{coNP} \) (guess a proper divisor)

❖ In \( \text{NP} \) [Pratt 1975]
A motivating example for (co)RP

- **PRIMALITY**
  - INPUT: a natural number $p$, in binary
  - Q: is $p$ prime?

- For a long time, not known to be in $\mathbf{P}$
  (now solved: indeed in $\mathbf{P}$ [Agrawal, Kayal, Saxena 2004])

- In $\text{coNP}$ (guess a proper divisor)

- In $\mathbf{NP}$ [Pratt 1975]

- Can also be solved efficiently with randomization…
Thm (Fermat). If $p$ is prime, then for every $r$ $(1 \leq r < p)$, $r^{p-1} = 1 \mod p$. 
Fermat’s little theorem

- **Thm (Fermat).** If \( p \) is prime, then for every \( r \) (\( 1 \leq r < p \)),
  \[ r^{p-1} = 1 \pmod{p}. \]

- \( \Rightarrow \) draw \( r \) at random in \([2, p-2]\); accept if \( r^{p-1} = 1 \pmod{p} \).
Fermat’s little theorem

- **Thm (Fermat).** If $p$ is prime, then for every $r$ ($1 \leq r < p$),
  \[ r^{p-1} \equiv 1 \mod p. \]

- **⇒** draw $r$ at random in $[2, p-2]$; accept if $r^{p-1} \equiv 1 \mod p$.

- **Note:** computing mod $p$ is **efficient**:
  - size of all numbers **bounded** by $\text{size}(p) = O(\log p)$.
  - addition mod $p$ in time $O(\log p)$
  - mult. mod $p$ in time $O(\log^2 p)$ (even $O(\log^{1+\varepsilon} p)$)
Thm (Fermat). If \( p \) is prime, then for every \( r \) (\( 1 \leq r < p \)), 
\[ r^{p-1} = 1 \mod p. \]

\[ \Rightarrow \] draw \( r \) at random in \([2, p-2]\); accept if \( r^{p-1} = 1 \mod p \).

Note: computing mod \( p \) is efficient:
size of all numbers bounded by \( \text{size}(p) = O(\log p) \).
— addition mod \( p \) in time \( O(\log p) \)
— mult. mod \( p \) in time \( O(\log^2 p) \) (even \( O(\log^{1+\epsilon} p) \))

An experiment… (next slide)
Fermat’s little theorem in practice

- **Thm (Fermat).** If $p$ is prime, then for every $r \ (1 \leq r < p)$, 
  $$r^{p-1} = 1 \mod p.$$ 

- $\Rightarrow$ draw $r$ at random in $[2, p-2]$; accept if $r^{p-1} = 1 \mod p$. 
Thm (Fermat). If \( p \) is prime, then for every \( r \) (\( 1 \leq r < p \)),
\[ r^{p-1} = 1 \pmod{p}. \]

⇒ draw \( r \) at random in \([2, p-2]\); accept if \( r^{p-1} = 1 \pmod{p} \).

Is 87 prime?
Fermat’s little theorem in practice

- **Thm (Fermat).** If $p$ is prime, then for every $r$ ($1 \leq r < p$),
  \[ r^{p-1} \equiv 1 \pmod{p}. \]

- \[ \Rightarrow \text{draw } r \text{ at random in } [2, p-2]; \text{ accept if } r^{p-1} \equiv 1 \pmod{p}. \]

- Is 87 prime?

- Draw $r$ at random… say 25
Fermat’s little theorem in practice

- **Thm (Fermat).** If \( p \) is prime, then for every \( r \) (\( 1 \leq r < p \)),
  \[ r^{p-1} = 1 \mod p. \]

- \( \Rightarrow \) draw \( r \) at random in \([2, p-2]\); accept if \( r^{p-1} = 1 \mod p \).

- Is 87 prime?

- Draw \( r \) at random… say 25

- \( r^{86} = 16 \mod 87 \)
Fermat’s little theorem in practice

- **Thm (Fermat).** If \( p \) is prime, then for every \( r \) (\( 1 \leq r < p \)),
  \[ r^{p-1} = 1 \mod p. \]

- ⇒ draw \( r \) at random in \([2, p-2]\); accept if \( r^{p-1} = 1 \mod p \).

- Is 87 prime?

- Draw \( r \) at random… say 25

- \( r^{86} = 16 \mod 87 \)

- ⇒ 87 is **not prime** (definitely)
Fermat’s little theorem in practice

- **Thm (Fermat).** If $p$ is prime, then for every $r$ (1 ≤ $r$ < $p$),
  $r^{p-1} = 1 \mod p$.

- ⇒ draw $r$ in $[2, p-2]$; accept if $r^{p-1} = 1 \mod p$.

- Is 87 prime?
Fermat’s little theorem in practice

- **Thm (Fermat).** If $p$ is prime, then for every $r \ (1 \leq r < p)$,
  $$r^{p-1} = 1 \mod p.$$  

- Imply draw $r$ in $[2, p-2]$; accept if $r^{p-1} = 1 \mod p$.

- Is 87 prime?

- The probability (over $r$) of error is:
  $$2/84 \approx 0.024$$
Fermat’s little theorem in practice

- Thm (Fermat). If \( p \) is prime, then for every \( r \) (1 \( \leq \) \( r \) \( < \) \( p \)),
  \[ rp^{-1} = 1 \mod p. \]

- \( \Rightarrow \) draw \( r \) in \([2, p-2]\); accept if \( rp^{-1} = 1 \mod p. \)
Fermat’s little theorem in practice

- **Thm (Fermat).** If \( p \) is prime, then for every \( r \) \((1 \leq r < p)\), \( r^{p-1} = 1 \text{ mod } p \).

- ⇒ draw \( r \) in \([2, p-2]\); accept if \( r^{p-1} = 1 \text{ mod } p \).

- If \( p \) is prime, will succeed for every \( r \)
Fermat’s little theorem in practice

- **Thm (Fermat).** If $p$ is prime, then for every $r$ ($1 \leq r < p$), $r^{p-1} \equiv 1 \pmod{p}$.

- $\Rightarrow$ draw $r$ in $[2, p-2]$; accept if $r^{p-1} \equiv 1 \pmod{p}$.

- If $p$ is prime, will succeed for every $r$.

- Else, will fail with (hopefully) high probability (0.024 in the example, looks good); but…
A **Carmichael number** is a number \( p \):
— that is **not** prime
— but passes all Fermat tests \( r^{p-1} \equiv 1 \mod p \) for every \( r \)

I.e., on which our hopes of low error rate fail miserably
Carmichael numbers

- A Carmichael number is a number \( p \):
  - that is not prime
  - but passes all Fermat tests \( r^{p-1}=1 \mod p \) for every \( r \)
- I.e., on which our hopes of low error rate fail miserably
- Infinitely many of them [Alford, Granville, Pomerance 1994]: 561, 1105, 1729, 2465, 2821, 6601, 8911, 10585, 15841, etc.
Carmichael numbers

- A Carmichael number is a number $p$:
  — that is not prime
  — but passes all Fermat tests ($r^{p-1}=1 \mod p$ for every $r$)
- I.e., on which our hopes of low error rate fail miserably
- Infinitely many of them [Alford, Granville, Pomerance 1994]: 561, 1105, 1729, 2465, 2821, 6601, 8911, 10585, 15841, etc.
- Frustrating: if $p$ is not prime and passes at least one Fermat test, then it passes at least half of them...
The Miller-Rabin test (1/2)

- We use another basic fact: if \( p \) is prime, then the only square roots of 1 mod \( p \) are 1 and \(-1\).
The Miller-Rabin test (1/2)

- We use another basic fact: if $p$ is prime, then the only square roots of 1 mod $p$ are 1 and $-1$.

- Hence, if $p$ is prime and odd (so $p - 1 = 2^k q$, $q$ odd):

  \[
  r^q \quad \ldots \quad r^{(2^{i-1}q)} \quad r^{(2^i q)} \quad \ldots \quad r^{(2^{k-1}q)} \quad r^{(2^k q)} \quad \mod p
  \]

  (read from right to left: $\leftarrow$)

  (don’t care ... .... don’t care) $-1$ $1$ $\ldots$ $1$ $1$ for some $i$, or:
The Miller-Rabin test (1/2)

- We use another basic fact: if \( p \) is prime, then the only square roots of 1 mod \( p \) are 1 and –1

- Hence, if \( p \) is prime and odd (so \( p–1 = 2^k q \), \( q \) odd):

\[
\begin{array}{ccccccc}
    r^q & \ldots & r^{(2^{i-1} q)} & r^{(2^i q)} & \ldots & r^{(2^{k-1} q)} & r^{(2^k q)} \\
    \text{mod } p \\
    \text{(don’t care ... .... don’t care)} & -1 & 1 & \ldots & 1 & 1 & 1
\end{array}
\]

(Read from right to left : \( \leftarrow \)) for some \( i \), or:

\[
\begin{array}{ccccccc}
    1 & \ldots & 1 & 1 & \ldots & 1 & 1 \\
    \text{mod } p 
\end{array}
\]
The Miller-Rabin test (2/2)

- On input $p$, draw $r$ at random:
  - if the test shown here: succeeds, then accept ($p$ probably prime)
  - otherwise reject ($p$ definitely not prime)
The Miller-Rabin test (2/2)

- On input $p$, draw $r$ at random:
  - if the test shown here:
    - succeeds, then accept ($p$ probably prime)
    - otherwise reject ($p$ definitely not prime)
  
- Probability of error $\leq 1/4$. Excellent! Hence:

Hence, if $p$ is prime and odd (so $p-1 = 2^k q$, $q$ odd):

<table>
<thead>
<tr>
<th>$r^q$</th>
<th>$r^{q(2^i q)}$</th>
<th>$r^{q(2^i q)}$</th>
<th>$r^{q(2^i q)}$</th>
<th>$r^{q(2^i q)}$</th>
<th>$r^{q(2^i q)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td></td>
<td>$-1$</td>
<td>$1$</td>
<td>$1$</td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$1$</td>
<td>$\ldots$</td>
<td>$1$</td>
<td>$1$</td>
<td>$1$</td>
</tr>
</tbody>
</table>

(Read from right to left: $\rightarrow$)`
The Miller-Rabin test (2/2)

- On input $p$, draw $r$ at random:
  - if the test shown here succeeds, then accept ($p$ probably prime)
  - otherwise reject ($p$ definitely not prime)

- Probability of error $\leq 1/4$. Excellent! Hence:

- Theorem. PRIMALITY is in coRP.
The Miller-Rabin test (2/2)

- On input \( p \), draw \( r \) at random:
  — if the test shown here: succeeds, then accept (\( p \) probably prime)
  — otherwise reject (\( p \) definitely not prime)

- Probability of error \( \leq 1/4 \). Excellent! Hence:

  - **Theorem. PRIMALITY is in coRP.**

  - (Superseded by [AKS04]… but Miller-Rabin works in log space, not [AKS04]!)

Hence, if \( p \) is prime and odd (so \( p-1 = 2^k q, q \) odd):

\[
\begin{array}{cccccccc}
  r^q & \ldots & r^{(2^i-1)q} & r^{(2^i q)} & \ldots & r^{(2^{i+1}-1)q} & r^{(2^{i+1} q)} \\
  (-1) & 1 & \ldots & 1 & 1 & \ldots & 1 \\
\end{array}
\mod p
\]

(read from right to left: \( \rightarrow \))
Notes on Primality Testing
And Public Key Cryptography
Part 1: Randomized Algorithms
Miller–Rabin and Solovay–Strassen Tests

Jean Gallier and Jocelyn Quaintance
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA
e-mail: jean@cis.upenn.edu

© Jean Gallier
February 27, 2019

https://www.cis.upenn.edu/~jean/RSA-primality-testing.pdf
Error reduction

- What is so special about error 1/2?

- A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r[M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r[M(x,r) \text{ accepts}]=0$).

- $\text{error} = 1 - 1/2$ (= 1/2 here)
Error reduction

- What is so special about error $1/2$?
- Nothing!

A language $L$ is in RP if and only if there is a *polynomial-time* TM $M$ such that for every input $x$ (of size $n$):
- if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
- if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).
Error reduction

- What is so special about error $1/2$?
- Nothing!

A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
- if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1/2$
- if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

A language $L$ is in $\text{RP}(\epsilon)$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
- if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1-\epsilon$
- if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

$\text{error} = 1 - 1/2$ (= $1/2$ here)
Error reduction

- What is so special about error $1/2$?
  - Nothing!

- **Theorem.** $\forall \varepsilon \in ]0, 1[,$ $\text{RP} = \text{RP}(\varepsilon)$. 

- A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

- $\text{error} = 1 - 1/2$ ($= 1/2$ here)
Error reduction

- What is so special about error 1/2?
  - Nothing!

- **Theorem.** ∀ ε ∈ ]0, 1[, $\text{RP} = \text{RP}(ε)$.

- **Note:** $\text{RP} = \text{RP}(1/2)$ (def.)

- A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\text{Pr}_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \not\in L$ then $M(x,r)$ accepts for no $r$ (i.e., $\text{Pr}_r [M(x,r) \text{ accepts}] = 0$).

- A language $L$ is in $\text{RP}(ε)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\text{Pr}_r [M(x,r) \text{ accepts}] \geq 1 - ε$
  - if $x \not\in L$ then $M(x,r)$ accepts for no $r$ (i.e., $\text{Pr}_r [M(x,r) \text{ accepts}] = 0$).
Error reduction: the easy direction

- Clearly, if $\eta \leq \varepsilon$ then $\text{RP}(\eta) \subseteq \text{RP}(\varepsilon)$

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).
Error reduction: the easy direction

- Clearly, if $\eta \leq \varepsilon$ then $\text{RP}(\eta) \subseteq \text{RP}(\varepsilon)$

- Proof: take any $L \in \text{RP}(\eta)$

... I’ll let you finish the argument

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

- error $= \varepsilon$
Error reduction: the easy direction

- Clearly, if $\eta \leq \varepsilon$ then $\text{RP}(\eta) \subseteq \text{RP}(\varepsilon)$

- Proof: take any $L \in \text{RP}(\eta)$
  ... I’ll let you finish the argument

- Note: $\text{RP}(0) = \text{P}$ (believed $\neq \text{RP}$)
  $\text{RP}(1) = \{\text{all languages}\}$ (why?)

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$)
The hard direction: repeating experiments

- Let $L \in \text{RP}(\epsilon)$, $0<\eta<\epsilon<1$

- On input $x$, let us do the following (at most) $K$ times:
  - A language $L$ is in $\text{RP}(\epsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
    - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1-\epsilon$
    - if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [\text{M(x, r) accepts}] = 0$, $\text{error} = \epsilon$)
The hard direction: repeating experiments

- Let $L \in \text{RP}(\varepsilon)$, $0<\eta<\varepsilon<1$
- On input $x$, let us do the following (at most) $K$ times:
  - Draw $r$ at random, simulate $M(x, r)$ and:

\[
\text{error} = \varepsilon
\]

A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

- if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1-\varepsilon$
- if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$)
The hard direction: repeating experiments

- Let $L \in \text{RP}(\varepsilon)$, $0<\eta<\varepsilon<1$

- On input $x$, let us do the following (at most) $K$ times:
  - Draw $r$ at random, simulate $M(x, r)$ and:
    - If $M(x, r)$ accepts, then exit the loop and accept;

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

Remember: if $M(x, r)$ accepts, then $x$ must be in $L$. 
Let $L \in \text{RP}(\varepsilon)$, $0 < \eta < \varepsilon < 1$

On input $x$, let us do the following (at most) $K$ times:

- Draw $r$ at random, simulate $M(x, r)$ and:
  - If $M(x, r)$ accepts, then exit the loop and accept;
  - Otherwise, proceed and loop.

Remember: if $M(x, r)$ accepts, then $x$ must be in $L$. 

A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

- if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon$
- if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

$\text{error} = \varepsilon$
The hard direction: repeating experiments

- Let $L \in \text{RP}(\varepsilon)$, $0 < \eta < \varepsilon < 1$
- On input $x$, let us do the following (at most) $K$ times:
  - Draw $r$ at random, simulate $M(x, r)$ and:
    - If $M(x, r)$ accepts, then exit the loop and accept;
    - Otherwise, proceed and loop.
  - At the end of the loop, reject.

A language $L$ is in $\text{RP}(\varepsilon)$ if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

Remember: if $M(x, r)$ accepts, then $x$ must be in $L$. 
Repeating experiments (pretty) formally

- We have defined a new randomized TM $M'(x, r[1]\# \ldots \# r[K])$ by:
  - for $i=1$ to $K$:
    - If $M(x, r[i])$ accepts, then exit the loop and accept;
    - reject.

Remember: if $M(x, r[i])$ accepts, then $x$ must be in $L$. 

- A language $L$ is in **RP($\varepsilon$)** and only if there is a **polynomial-time** TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

Error $= \varepsilon$
Acceptance: 1. if $x \in L$

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$), then letting $r=r[1]\#\ldots\#r[K]$, $\Pr_r(M'(x, r) \text{ rejects})$

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r[M(x,r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r[M(x,r) \text{ accepts}] = 0$), error $= \varepsilon$

- Define $M'(x, r[1]\#\ldots\#r[K])$ by:
  - for $i=1$ to $K$:
    - If $M(x, r[i])$ accepts, then exit the loop and accept;
    - reject.
Acceptance: 1. if $x \in L$

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$), then letting $r=r[1]#...#r[K]$, $\Pr_r(M'(x, r) \text{ rejects})$
  - $= \Pr_r(\forall i=1..K, M(x, r[i]) \text{ rejects})$

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$
    (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$, $\text{error} = \varepsilon$

Define $M'(x, r[1]#...#r[K])$ by:

- for $i=1$ to $K$:
  - If $M(x, r[i])$ accepts, then exit the loop and accept;
  - reject.
Acceptance: 1. if $x \in L$

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$), then letting $r=r[1] \# \ldots \# r[K]$, $\text{Pr}_r(M'(x, r) \text{ rejects})$
  - $= \text{Pr}_r(\forall i=1..K, M(x, r[i]) \text{ rejects})$
  - $= \Pi_{i=1..K} \text{Pr}_{r[i]}(M(x, r[i]) \text{ rejects})$ (independence)

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\text{Pr}_r[M(x,r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\text{Pr}_r[M(x,r) \text{ accepts}] = 0$)

- Define $M'(x, r[1] \# \ldots \# r[K])$ by:
  - for $i=1$ to $K$:
    - If $M(x, r[i])$ accepts, then exit the loop and accept;
    - reject.
Acceptance: 1. if $x \in L$

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$), then letting $r=r[1]# \ldots # r[K]$,
  $\Pr_r(M'(x, r) \text{ rejects})$
  
  - $= \Pr_r(\forall i=1..K, M(x, r[i]) \text{ rejects})$
  
  - $= \prod_{i=1..K} \Pr_{r[i]}(M(x, r[i]) \text{ rejects})$
    (independence)
  
  - $\leq \varepsilon^K$

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x, r) \text{ accepts for no } r$
    (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$), error $= \varepsilon$

Define $M'(x, r[1]# \ldots # r[K])$ by:

- for $i=1$ to $K$:
  - If $M(x, r[i]) \text{ accepts, then exit the loop and accept; }$
  - reject.
Acceptance: 1. if \( x \in L \)

- If \( x \in L \) (recall \( L \) in \( \text{RP}(\varepsilon) \)), then
  letting \( r=r[1]\#\ldots\#r[K] \),
  \( \Pr_r(\mathcal{M}'(x, r) \text{ rejects}) \)
  \[ = \Pr_r(\forall i=1..K, \mathcal{M}(x, r[i]) \text{ rejects}) \]
  \[ = \prod_{i=1..K} \Pr_{r[i]}(\mathcal{M}(x, r[i]) \text{ rejects}) \] (independence)
  \[ \leq \varepsilon^K \]

- \( \Rightarrow \) If \( x \in L \) then
  \( \Pr_r(\mathcal{M}'(x, r) \text{ accepts}) \geq 1-\varepsilon^K \)

- A language \( L \) is in \( \text{RP}(\varepsilon) \) and only if
  there is a polynomial-time TM \( \mathcal{M} \)
  such that for every input \( x \) (of size \( n \)):
  - if \( x \in L \) then \( \Pr_r [\mathcal{M}(x,r) \text{ accepts}] \geq 1-\varepsilon \)
  - if \( x \not\in L \) then \( \mathcal{M}(x,r) \text{ accepts for no } r \)
    (i.e., \( \Pr_r [\mathcal{M}(x,r) \text{ accepts}] = 0 \), \( \text{error} = \varepsilon \))

- Define \( \mathcal{M}'(x, r[1]\#\ldots\#r[K]) \) by:
  - for \( i=1 \) to \( K \):
    - If \( \mathcal{M}(x, r[i]) \text{ accepts} \), then exit the loop and \text{accept};
    - \text{reject.}
Acceptance: 2. if \( x \notin L \); Complexity

- If \( x \in L \) (recall \( L \) in \( \text{RP}(\varepsilon) \)) then \( \Pr_r(\mathcal{M}'(x, r) \text{ accepts}) \geq 1 - \varepsilon^K \)
- A language \( L \) is in \( \text{RP}(\varepsilon) \) and only if there is a polynomial-time TM \( \mathcal{M} \) such that for every input \( x \) (of size \( n \)):
  - if \( x \in L \) then \( \Pr_r(\mathcal{M}(x, r) \text{ accepts}) \geq 1 - \varepsilon \)
  - if \( x \notin L \) then \( \mathcal{M}(x, r) \) accepts for no \( r \) (i.e., \( \Pr_r(\mathcal{M}(x, r) \text{ accepts}) = 0 \), error = \( \varepsilon \))

Define \( \mathcal{M}'(x, r[1]\#\ldots\#r[K]) \) by:

- for \( i=1 \) to \( K \):
  - If \( \mathcal{M}(x, r[i]) \) accepts, then exit the loop and accept;
  - reject.
Acceptance: 2. if \( x \notin L \); Complexity

- If \( x \in L \) (recall \( L \) in \( \text{RP}(\varepsilon) \)) then
  \[ \Pr_r(\mathcal{M}'(x, r) \text{ accepts}) \geq 1 - \varepsilon^K \]

- If \( x \notin L \), then
  \( \mathcal{M}'(x, r) \) accepts for no \( r \)

- A language \( L \) is in \( \text{RP}(\varepsilon) \) and only if there is a polynomial-time TM \( \mathcal{M} \) such that for every input \( x \) (of size \( n \)):
  - if \( x \in L \) then \( \Pr_r [\mathcal{M}(x, r) \text{ accepts}] \geq 1 - \varepsilon \)
  - if \( x \notin L \) then \( \mathcal{M}(x, r) \) accepts for no \( r \)
    (i.e., \( \Pr_r [\mathcal{M}(x, r) \text{ accepts}] = 0 \))
    \( \text{error} = \varepsilon \)

- Define \( \mathcal{M}'(x, r[1] \ldots \#r[K]) \) by:
  - for \( i = 1 \) to \( K \):
    - If \( \mathcal{M}(x, r[i]) \) accepts, then exit the loop and accept;
    - reject.
Acceptance: 2. if $x \notin L$; Complexity

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$) then $\Pr_r(M'(x, r) \text{ accepts}) \geq 1 - \varepsilon^K$
- If $x \notin L$, then $M'(x, r)$ accepts for no $r$
- If $M$ runs in time $p(n)$, then $M'$ runs in time $O(Kp(n))$

A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

- if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon$
- if $x \notin L$ then $M(x, r)$ accepts for no $r$
  (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$
- error = $\varepsilon$

Define $M'(x, r[1]\#\ldots\#r[K])$ by:
- for $i=1$ to $K$:
  - If $M(x, r[i])$ accepts, then exit the loop and accept;
  - reject.
Acceptance: 2. if $x \not\in L$; Complexity

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$) then $\Pr_r(M'(x, r) \text{ accepts}) \geq 1 - \varepsilon^K$
- If $x \not\in L$, then $M'(x, r)$ accepts for no $r$
- If $M$ runs in time $p(n)$, then $M'$ runs in time $O(KP(n))$
- Hence $L$ is in $\text{RP}(\varepsilon^K)$

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \not\in L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$)

- Define $M'(x, r[1]\#\ldots\#r[K])$ by:
  - for $i = 1$ to $K$:
    - If $M(x, r[i])$ accepts, then exit the loop and accept;
    - reject.
The hard direction: the end

- We have shown that every language $L$ in $\text{RP}(\varepsilon)$ is in $\text{RP}(\varepsilon^K)$ (for any $\varepsilon \in [0,1]$, $K \geq 1$)

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$)

- Define $M'(x, r[1] \ldots r[K])$ by:
  - for $i = 1$ to $K$:
    - If $M(x, r[i])$ accepts, then exit the loop and accept;
    - reject.
The hard direction: the end

- We have shown that every language \( L \) in \( \text{RP}(\varepsilon) \) is in \( \text{RP}(\varepsilon^K) \) (for any \( \varepsilon \in [0,1] \), \( K \geq 1 \))

- If \( 0 < \eta < \varepsilon < 1 \), choose \( K \) large enough so that \( \varepsilon^K \leq \eta \) (explicitly, \( K \geq \eta / \log \varepsilon \))

- A language \( L \) is in \( \text{RP}(\varepsilon) \) and only if there is a polynomial-time TM \( M \) such that for every input \( x \) (of size \( n \)):
  - if \( x \in L \) then \( \Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon \)
  - if \( x \notin L \) then \( M(x, r) \) accepts for no \( r \) (i.e., \( \Pr_r [M(x, r) \text{ accepts}] = 0 \))

- Define \( M'(x, r[1] \# \ldots \# r[K]) \) by:
  - for \( i = 1 \) to \( K \):
    - If \( M(x, r[i]) \) accepts, then exit the loop and accept;
    - reject.
The hard direction: the end

- We have shown that every language $L$ in $\text{RP}(\varepsilon)$ is in $\text{RP}(\varepsilon^K)$ (for any $\varepsilon \in [0,1]$, $K\geq 1$).

- If $0<\eta<\varepsilon<1$, choose $K$ large enough so that $\varepsilon^K \leq \eta$ (explicitly, $K\geq \eta / \log \varepsilon$).

- Then $L$ is in $\text{RP}(\eta)$. ☐

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\text{Pr}_r [M(x,r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\text{Pr}_r [M(x,r) \text{ accepts}] = 0$, $\text{error} = \varepsilon$)

- Define $M'(x, r[1]\#\ldots\#r[K])$ by:
  - for $i=1$ to $K$:
    - If $M(x, r[i])$ accepts, then exit the loop and accept; reject.
Can we do even better?

- A language $L$ is in $\text{RP}(\epsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1-\epsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$)
Can we do even better?

- Hence we define the same class with error $\varepsilon = 0.000000000001$

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$).
Can we do even better?

- Hence we define the same class with error $\varepsilon = 0.000000000001$
- ... or with error $\varepsilon = 0.99999999$!

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$)
Can we do even better?

- Hence we define the same class with error $\varepsilon = 0.000000000001$
- ... or with error $\varepsilon = 0.99999999$!
- Can we make $\varepsilon$ go to 0 as $n \to \infty$?

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1 - \varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$)
The hard direction revisited

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$) then $\Pr_r(M'(x, r) \text{ accepts}) \geq 1 - \varepsilon^K$
- If $x \notin L$, then $M'(x, r)$ accepts for no $r$
- If $M$ runs in time $p(n)$, then $M'$ runs in time $O(Kp(n))$
- Hence $L$ is in $\text{RP}(\varepsilon^K)$.

A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a **polynomial-time** TM $M$ such that for every input $x$ (of size $n$):
- if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon$
- if $x \notin L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

Define $M'(x, r[1]\#\ldots\#r[K])$ by:
- for $i=1$ to $K$:
  - If $M(x, r[i])$ accepts, then exit the loop and accept;
  - reject.

error $= \varepsilon$
The hard direction revisited

Let us take \( K = \text{a polynomial} \ q(n) \)

- If \( x \in L \) (recall \( L \) in \( \text{RP}(\varepsilon) \)) then
  \( \Pr_r(M'(x, r) \text{ accepts}) \geq 1-\varepsilon^K \)
- If \( x \notin L \), then
  \( M'(x, r) \text{ accepts for no } r \)
- If \( M \) runs in time \( p(n) \), then
  \( M' \) runs in time \( O(Kp(n)) \)
- Hence \( L \) is in \( \text{RP}(\varepsilon^K) \).

A language \( L \) is in \( \text{RP}(\varepsilon) \) and only if there is a polynomial-time TM \( M \) such that for every input \( x \) (of size \( n \)):

- if \( x \in L \) then \( \Pr_r [M(x, r) \text{ accepts}] \geq 1-\varepsilon \)
- if \( x \notin L \) then \( M(x, r) \text{ accepts for no } r \) (i.e., \( \Pr_r [M(x, r) \text{ accepts}] = 0 \)).

Define \( M'(x, r[1]\#\ldots\#r[K]) \) by:

- for \( i=1 \) to \( K \):
  - If \( M(x, r[i]) \text{ accepts} \), then exit the loop and accept;
  - reject.
The hard direction revisited

- If \( x \in L \) (recall \( L \) in \( \text{RP}(\varepsilon) \)) then
  \[ \Pr_r(M'(x, r) \text{ accepts}) \geq 1 - \varepsilon^K \]

- If \( x \notin L \), then
  \( M'(x, r) \) accepts for no \( r \)

- If \( M \) runs in time \( p(n) \), then
  \( M' \) runs in time \( O(Kp(n)) \)

- Hence \( L \) is in \( \text{RP}(\varepsilon^K) \).
The hard direction revisited

- If \( x \in L \) (recall \( L \) in \( \text{RP}(\varepsilon) \)) then
  \( \Pr_r(\mathcal{M}'(x, r) \text{ accepts}) \geq 1-\varepsilon^K \)
- If \( x \notin L \), then
  \( \mathcal{M}'(x, r) \) accepts for no \( r \)
- If \( \mathcal{M} \) runs in time \( p(n) \), then
  \( \mathcal{M}' \) runs in time \( O(Kp(n)) \)
- Hence \( L \) is in \( \text{RP}(\varepsilon^K) \).

Let us take \( K = \text{a polynomial } q(n) \)

- A language \( L \) is in \( \text{RP}(\varepsilon) \) and only if there is a polynomial-time TM \( \mathcal{M} \) such that for every input \( x \) (of size \( n \)):
  - if \( x \in L \) then \( \Pr_r[\mathcal{M}(x, r) \text{ accepts}] \geq 1-\varepsilon \)
  - if \( x \notin L \) then \( \mathcal{M}(x, r) \) accepts for no \( r \) (i.e., \( \Pr_r[\mathcal{M}(x, r) \text{ accepts}] = 0 \)).

Error \( \varepsilon^K = \varepsilon^{q(n)} \) (exponentially small)

Define \( \mathcal{M}'(x, r[1] \# \ldots \# r[K]) \) by:
- for \( i=1 \) to \( K \):
  - If \( \mathcal{M}(x, r[i]) \) accepts, then exit the loop and accept;
  - reject.

Error = \( \varepsilon \)

\( \gamma = O(q(n)p(n)) \), still polynomial time
Let $\varepsilon = 1/2$. We have proved:

**Theorem.** \( \text{RP} = \text{RP}(1/2^{q(n)}) \) for every polynomial \( q(n) \).

I.e., error can be made exponentially small.

(Note: \( \text{RP}(\varepsilon) \) called \( \cup_{p(n)} \text{RTIME}(p(n), p(n), 0, \varepsilon) \) in the notes: ignore the complication)

A language \( L \) is in \( \text{RP}(\varepsilon) \) and only if there is a polynomial-time TM \( M \) such that for every input \( x \) (of size \( n \)):

- if \( x \in L \) then \( \Pr_r [M(x, r) \text{ accepts}] \geq 1 - \varepsilon \)
- if \( x \not\in L \) then \( M(x, r) \) accepts for no \( r \) (i.e., \( \Pr_r [M(x, r) \text{ accepts}] = 0 \)).
The hard direction revisited

- Let $\varepsilon = 1/2$. We have proved:
  - **Theorem.** $\text{RP} = \text{RP}(1/2^{q(n)})$ for every polynomial $q(n)$.
  - I.e., error can be made exponentially small.
  - (Note: $\text{RP}(\varepsilon)$ called $\cup_{p(n)} \text{RTIME}(p(n),p(n),0,\varepsilon)$ in the notes: ignore the complication)

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a **polynomial-time** TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 1-\varepsilon$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $\Pr_r [M(x,r) \text{ accepts}] = 0$)

- Exercise: show that, conversely:
  - **Theorem.** $\text{RP} = \text{RP}(1-1/q(n))$ for every polynomial $q(n)$.
  - I.e., error can be assumed « polynomially large » as well
Relation to ordinary classes

❖ Theorem. $P \subseteq RP \subseteq NP$.

❖ Proof. First, $P = RP(0) \subseteq RP(1/2) = RP$.

❖ A language $L$ is in $RP$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $Pr_r [M(x,r) \text{ accepts}] = 0$).
Relation to ordinary classes

- **Theorem.** $P \subseteq RP \subseteq NP$.

- **Proof.** First,
  
  $P=RP(0) \subseteq RP(1/2) = RP$

- Second, let $L \in RP$.

- A language $L$ is in $RP$ if and only if there is a **polynomial-time** TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
  - if $x \notin L$ then $M(x,r)$ accepts for no $r$ (i.e., $Pr_r [M(x,r) \text{ accepts}] = 0$).
Relation to ordinary classes

❖ Theorem. \( P \subseteq \text{RP} \subseteq \text{NP} \).

❖ Proof. First,

\[ P = \text{RP}(0) \subseteq \text{RP}(1/2) = \text{RP} \]

Second, let \( L \in \text{RP} \).

❖ If \( x \in L \) \( \Rightarrow \) for some \( r \), 

\[ M(x, r) \text{ accepts} \]
Theorem. $P \subseteq RP \subseteq NP$.

Proof. First,

$$P = \text{RP}(0) \subseteq \text{RP}(1/2) = \text{RP}$$

Second, let $L \in \text{RP}$.

- If $x \in L \Rightarrow$ for some $r$, $M(x, r)$ accepts
- If $x \not\in L \Rightarrow$ for no $r$.

A language $L$ is in $\text{RP}$ if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):

- if $x \in L$ then $\Pr_r [M(x, r) \text{ accepts}] \geq 1/2$
- if $x \not\in L$ then $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).
Relation to ordinary classes

- **Theorem.** \( P \subseteq \text{RP} \subseteq \text{NP} \).

- **Proof.** First,
  \[ P = \text{RP}(0) \subseteq \text{RP}(1/2) = \text{RP} \]
- Second, let \( L \in \text{RP} \).
  - If \( x \in L \implies \) for some \( r \), \( M(x, r) \) accepts
  - (in fact, for at least half of them!)
  - If \( x \notin L \implies \) for no \( r \).
- Hence \( L = \{x \mid \exists r, M(x, r) \text{ accepts}\} \) is in \( \text{NP} \). \( \square \)
Our second probabilistic class: ZPP

(also known as the class of Las Vegas languages)
ZPP

- ZPP = Zero Probability of error Polynomial-time
ZPP

- **ZPP = Zero Probability of error Polynomial-time**

- Usually defined as the class of languages $L$ which we can decide in **average** polynomial-time (not worst-case!) with probability **zero** of making a mistake.
ZPP

- ZPP = Zero Probability of error Polynomial-time
- Usually defined as the class of languages $L$ which we can decide in average polynomial-time (not worst-case!) with probability zero of making a mistake.
- Alternate definition: $\text{ZPP} = \text{RP} \cap \text{coRP}$
ZPP

- ZPP = Zero Probability of error Polynomial-time
- Usually defined as the class of languages \( L \) which we can decide in average polynomial-time (not worst-case!) with probability zero of making a mistake.
- Alternate definition:
  \[ ZPP = RP \cap coRP \]
- Not clear that those two definitions are equivalent, right?
Let us start simple:
Let us start simple:

Definition. $\text{ZPP} = \text{RP} \cap \text{coRP}$
Let us start simple:

Definition. ZPP = RP \cap \text{coRP}

I.e., \( L \) is in ZPP iff there are two poly-time rand. TMs \( M_1 \) and \( M_2 \) such that:

- if \( x \in L \) then \( M_1(x,r) \) accepts for every \( r \) [no error] \( M_2(x,r) \) accepts with prob. \( \geq 1/2 \)
- if \( x \notin L \) then \( M_1(x,r) \) accepts with prob. \( \leq 1/2 \) \( M_2(x,r) \) rejects for every \( r \) [no error]
Let us start simple:

**Definition.** \( ZPP = \text{RP} \cap \text{coRP} \)

I.e., \( L \) is in \( ZPP \) iff there are two poly-time rand. TMs \( M_1 \) and \( M_2 \) such that:

- if \( x \in L \) then \( M_1(x,r) \) accepts for every \( r \) [no error]
  \( M_2(x,r) \) accepts with prob. \( \geq 1/2 \)

- if \( x \notin L \) then \( M_1(x,r) \) accepts with prob. \( \leq 1/2 \)
  \( M_2(x,r) \) rejects for every \( r \) [no error]
Let us start simple:

Definition. \( ZPP = \text{RP} \cap \text{coRP} \)

I.e., \( L \) is in \( ZPP \) iff there are two poly-time rand. TMs \( M_1 \) and \( M_2 \) such that:

- if \( x \in L \) then \( M_1(x,r) \) accepts for every \( r \) [no error] \( M_2(x,r) \) accepts with prob. \( \geq 1/2 \)
- if \( x \notin L \) then \( M_1(x,r) \) accepts with prob. \( \leq 1/2 \) \( M_2(x,r) \) rejects for every \( r \) [no error]
Let us define \( ZPP' \) (for now) as the class of languages \( L \) which we can decide in \textit{average} polynomial-time with probability \textit{zero} of making a mistake.

I claim that \( ZPP = ZPP' \).
Let us define \( \text{ZPP}' \) (for now) as the class of languages \( L \) which we can decide in average polynomial-time with probability \( \text{zero} \) of making a mistake.

I claim that \( \text{ZPP} = \text{ZPP}' \).

The definition of \( \text{ZPP}' \) has a few technical problems… (see next slides)
Let us define $\text{ZPP}'$ (for now) as the class of languages $L$ which we can decide in average polynomial-time with probability zero of making a mistake.

I claim that $\text{ZPP} = \text{ZPP}'$.

The definition of $\text{ZPP}'$ has a few technical problems… (see next slides)

we will need something called Markov’s inequality too
Let us define \( ZPP' \) (for now) as the class of languages \( L \) which we can decide in average polynomial-time with probability zero of making a mistake.

I claim that \( ZPP = ZPP' \).

The definition of \( ZPP' \) has a few technical problems… (see next slides)

we will need something called Markov’s inequality too

… but before that, we explain why (intuitively) \( ZPP \subseteq ZPP' \).
Deciding $L$ in $\text{ZPP} = \text{RP} \cap \text{coRP}$ with no error

- Assume $M_1$ and $M_2$ such as here→
- Now run the following on input $x$:
  
  forever:
  if $M_1(x,\ldots)$ rejects: stop and reject
  if $M_2(x,\ldots)$ accepts: stop and accept
Deciding $L$ in $\text{ZPP} = \text{RP} \cap \text{coRP}$ with no error

- Assume $M_1$ and $M_2$ such as here→
- Now run the following on input $x$:
  
  forever:
  
  if $M_1(x,\ldots)$ rejects: stop and reject
  
  if $M_2(x,\ldots)$ accepts: stop and accept

I.e., $L$ is in ZPP iff there are two poly-time rand. TMs $M_1$ and $M_2$ such that:

- if $x \in L$ then $M_1(x,r)$ accepts for every $r$ [no error]
  $M_2(x,r)$ accepts with prob.$\geq 1/2$

- if $x \notin L$ then $M_1(x,r)$ accepts with prob.$\leq 1/2$
  $M_2(x,r)$ rejects for every $r$ [no error]

then $x$ cannot be in $L$ (sure)
Deciding $L$ in $\text{ZPP} = \text{RP} \cap \text{coRP}$ with no error

- Assume $M_1$ and $M_2$ such as here:
- Now run the following on input $x$:
  forever:
  - if $M_1(x,\ldots)$ rejects: stop and reject
  - if $M_2(x,\ldots)$ accepts: stop and accept

I.e., $L$ is in ZPP iff there are two poly-time rand. TMs $M_1$ and $M_2$ such that:
- if $x \in L$ then $M_1(x,r)$ accepts for every $r$ [no error]
  $M_2(x,r)$ accepts with prob. $\geq 1/2$
- if $x \notin L$ then $M_1(x,r)$ accepts with prob. $\leq 1/2$
  $M_2(x,r)$ rejects for every $r$ [no error]
Deciding \( L \) in \( \text{ZPP} = \text{RP} \cap \text{coRP} \) with no error

- Assume \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \) such as here→

- Now run the following on input \( x \):
  forever:
  - if \( \mathcal{M}_1(x,\ldots) \) rejects: stop and reject
  - if \( \mathcal{M}_2(x,\ldots) \) accepts: stop and accept

I.e., \( L \) is in \( \text{ZPP} \) iff there are two poly-time rand. TMs \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \) such that:
- if \( x \in L \) then \( \mathcal{M}_1(x,r) \) accepts for every \( r \) [no error]
  - \( \mathcal{M}_2(x,r) \) accepts with prob.\( \geq 1/2 \)
- if \( x \notin L \) then \( \mathcal{M}_1(x,r) \) accepts with prob.\( \leq 1/2 \)
  - \( \mathcal{M}_2(x,r) \) rejects for every \( r \) [no error]

then \( x \) cannot be in \( L \) (sure)

then \( x \) must be in \( L \) (sure)

Hence this machine never makes any mistake
Deciding $L$ in ZPP = $\text{RP} \cap \text{coRP}$ with no error

- Assume $\mathcal{M}_1$ and $\mathcal{M}_2$ such as here→
- Now run the following on input $x$:
  
  
  forever:
  
  if $\mathcal{M}_1(x,\ldots)$ rejects: stop and reject
  if $\mathcal{M}_2(x,\ldots)$ accepts: stop and accept

  then $x$ cannot be in $L$ (sure)
  then $x$ must be in $L$ (sure)

  Hence this machine never makes any mistake

It may be that $\mathcal{M}_1(x,\ldots)$ accepted and $\mathcal{M}_2(x,\ldots)$ rejected,
— in which case we loop
— and that happens with probability $\leq 1/2$...
  why?
  (if you tell me that this is even $\leq 1/4$, you are wrong)
Deciding $L$ in $\text{ZPP} = \text{RP} \cap \text{coRP}$ with no error

- Assume $M_1$ and $M_2$ such as here→
- Now run the following on input $x$:
  forever:
    if $M_1(x,\ldots)$ rejects: stop and reject
    if $M_2(x,\ldots)$ accepts: stop and accept

It may be that $M_1(x,\ldots)$ accepted and $M_2(x,\ldots)$ rejected,
— in which case we loop
— and that happens with probability $\leq 1/2$...
why?
(if you tell me that this is even $\leq 1/4$, you are wrong)

Hence this machine never makes any mistake

We will see that this implies that the machine terminates in
$\leq 2$ turns of the loop on average
A technical problem

- All this requires us to draw \textit{arbitrarily long} bitstrings
A technical problem

- All this requires us to draw **arbitrarily long** bitstrings
- In fact, even **infinite** bit strings (for those computations that do not terminate)
A technical problem

- All this requires us to draw arbitrarily long bitstrings
- In fact, even infinite bit strings (for those computations that do not terminate)
- Requires measure theory:
  there is a unique measure $\mu$ on $\{0,1\}^\omega$
  with $\sigma$-algebra generated by cylinders $w.\{0,1\}^\omega$
  such that $\mu(w.\{0,1\}^\omega) = 1/2^{\|w\|}$ (Carathéodory)
A technical problem

- All this requires us to draw arbitrarily long bitstrings.
- In fact, even infinite bit strings (for those computations that do not terminate).
- Requires measure theory: there is a unique measure $\mu$ on $\{0,1\}^\omega$ with $\sigma$-algebra generated by cylinders $w.\{0,1\}^\omega$ such that $\mu(w.\{0,1\}^\omega) = 1/2^{|w|}$ (Carathéodory).
- We will happily ignore this.
Rejection sampling

- A classic probabilistic procedure (rejection sampling):
  forever:
    compute something (with some random data \( r \)), \( x \);
    if \( P(x) \) holds: stop and return \( x \)

- Trick. If:
  — the random bits are independent across turns of the loop
  — and \( P(x) \) holds with prob. \( \geq \alpha \) at each turn
  then rejection sampling terminates in
    \( 1/\alpha \) turns of the loop on average.
Rejection sampling

- A classic probabilistic procedure (rejection sampling):
  forever:
  
  - compute something (with some random data $r$), $x$;
  - if $P(x)$ holds: stop and return $x$

- **Trick.** If:
  - the random bits are independent across turns of the loop
  - and $P(x)$ holds with prob. $\geq \alpha$ at each turn
  then rejection sampling terminates in

  $\frac{1}{\alpha}$ turns of the loop on average.
Rejection sampling

- An classic probabilistic procedure (rejection sampling):
  forever:
    compute something (with some random data $r$), $x$;
    if $P(x)$ holds: stop and return $x$

- Proof. Let $X$ be the random variable « # turns through the loop »
Rejection sampling

- An classic probabilistic procedure (rejection sampling): forever:
  compute something (with some random data \( r \)), \( x \);
  if \( P(x) \) holds: stop and return \( x \)

- Proof. Let \( X \) be the random variable « # turns through the loop »

- \( \Pr(X \geq n) = \Pr(P \text{ failed at turns 1, } \ldots, n-1) \)
  \( \leq (1 - \alpha)^{n-1} \)  (by independence)
Rejection sampling

- An classic probabilistic procedure (**rejection sampling**): forever:
  - compute something (with some random data $r$), $x$;
  - if $P(x)$ holds: stop and return $x$

- Proof. Let $X$ be the random variable « # turns through the loop »

- $\Pr(X \geq n) = \Pr(P \text{ failed at turns } 1, \ldots, n-1)$
  - $\leq (1-\alpha)^{n-1}$ (by independence)

- $\mathbb{E}(X) = \sum_{n \geq 1} n \cdot \Pr(X = n) = \sum_{n \geq 1} \Pr(X \geq n) \leq \sum_{n \geq 1} (1-\alpha)^{n-1} = 1/\alpha$. ☐

Expectation (average)
Rejection sampling: a typical application

- Draw a point inside the disc:
- Repeatedly draw a point inside the inscribing square
- If it is in the disc, return it.
Rejection sampling: a typical application

- Draw a point inside the disc:
- Repeatedly draw a point inside the inscribing square
  - If it is in the disc, return it.
- Terminates in $\leq 4/\pi$
  $\sim 1.27324$ turns
Rejection sampling: a typical application

- Draw a point inside the disc:
- Repeatedly draw a point inside the inscribing square
  - If it is in the disc, return it.
- Terminates in $\leq \frac{4}{\pi} \approx 1.27324$ turns

(Used as first step in the Box-Muller procedure drawing two independent numbers with a normal distribution)
Deciding $L$ in $\text{ZPP} = \text{RP} \cap \text{coRP}$ with no error

- Assume $M_1$ and $M_2$ such as here:

- Now run the following on input $x$:
  
  ```
  forever:
  if $M_1(x,\ldots)$ rejects: stop and reject
  if $M_2(x,\ldots)$ accepts: stop and accept
  ```

  It may be that $M_1(x,\ldots)$ accepted and $M_2(x,\ldots)$ rejected, — in which case we loop — and that happens with probability $\leq 1/2$… (two cases: $x$ in $L$, $x$ not in $L$)

- Hence this machine never makes any mistake

I.e., $L$ is in $\text{ZPP}$ iff there are two poly-time rand. TMs $M_1$ and $M_2$ such that:

- if $x \in L$ then $M_1(x,r)$ accepts for every $r$ [no error]
  $M_2(x,r)$ accepts with prob.$\geq 1/2$

- if $x \notin L$ then $M_1(x,r)$ accepts with prob.$\leq 1/2$
  $M_2(x,r)$ rejects for every $r$ [no error]
Deciding \( L \) in \( \text{ZPP} = \text{RP} \cap \text{coRP} \) with no error

- Assume \( M_1 \) and \( M_2 \) such as here:

- Now run the following on input \( x \):
  - forever:
    - if \( M_1(x,\ldots) \) rejects: stop and reject
    - if \( M_2(x,\ldots) \) accepts: stop and accept

  It may be that \( M_1(x,\ldots) \) accepted and \( M_2(x,\ldots) \) rejected, — in which case we loop — and that happens with probability \( \leq 1/2 \)… (two cases: \( x \) in \( L \), \( x \) not in \( L \))

Hence this machine never makes any mistake

This is rejection sampling: stops in \( \leq 2 \) turns on average hence in polytime on average.

I.e., \( L \) is in \( \text{ZPP} \) iff there are two poly-time rand. TMs \( M_1 \) and \( M_2 \) such that:
  - if \( x \in L \) then \( M_1(x,r) \) accepts for every \( r \) [no error]
    - \( M_2(x,r) \) accepts with prob.\( \geq 1/2 \)
  - if \( x \notin L \) then \( M_1(x,r) \) accepts with prob.\( \leq 1/2 \)
    - \( M_2(x,r) \) rejects for every \( r \) [no error]
Markov’s inequality

- Hence:
  \[ ZPP \ (= \ RP \cap \ coRP) \subseteq ZPP' \]

- In order to show the reverse inclusion, we use:

- **Theorem (Markov’s inequality).**
  Let \( X \) be a **non-negative real-valued** random variable with **finite** expectation \( E(X) \). For every \( a > 0 \):
  \[ \Pr(X \geq a \cdot E(X)) \leq \frac{1}{a}. \]
Markov’s inequality

- Hence:
  \( \text{ZPP} (= \text{RP} \cap \text{coRP}) \subseteq \text{ZPP}' \)

- In order to show the reverse inclusion, we use:

  - Theorem (Markov’s inequality).
    Let \( X \) be a non-negative real-valued random variable with finite expectation \( E(X) \). For every \( a > 0 \):
    \[ \Pr(X \geq a \cdot E(X)) \leq \frac{1}{a}. \]
Markov’s inequality

Theorem (Markov’s inequality). Let $X$ be a non-negative real-valued random variable with finite expectation $E(X)$. For every $a > 0$: $\Pr(X \geq a \cdot E(X)) \leq 1/a$.

Proof. $E(X) = \int_t \Pr(X \geq t) \, dt$

\geq \text{area of the blue rectangle}

= a \cdot E(X) \cdot \Pr(X \geq a \cdot E(X))$

Then divide out by $a \cdot E(X)$. $\square$
The reverse inclusion $\text{ZPP}' \subseteq \text{ZPP}$

- Let $L$ in $\text{ZPP}'$, decided by $M$ running in average polynomial-time with no error.
- Define $M_1$ as follows: on input $x$
  (and random tape $r$ of size $a \cdot p(n)$)
  simulate $M$ on $x$ for at most $a \cdot p(n)$ steps (timeout).
  If timeout reached, then accept (that may be an error).
The reverse inclusion $\text{ZPP}' \subseteq \text{ZPP}$

- Markov on r.v. $X =$ time taken by $M$ on $x$; also let $a=2$.
- $E(X) \leq p(n)$ finite OK

Let $L$ in $\text{ZPP}'$, decided by $M$ running in average poly. time $p(n)$ with no error.

Define $M_1$ as follows: on input $x$

(simulate $M$ on $x$ for at most $a \cdot p(n)$ steps (timeout).

If timeout reached, then accept (that may be an error).
The reverse inclusion $\text{ZPP}' \subseteq \text{ZPP}$

- Markov on r.v. $X =$ time taken by $\mathcal{M}$ on $x$; also let $a=2$.
- $E(X) \leq p(n)$ finite OK
- If $x \notin L \Rightarrow \text{error} = \Pr_r(\mathcal{M}_1(x, r) \text{ accepts})$
  $$= \Pr(X \geq a \cdot p(n)) \quad (\mathcal{M} \text{ makes no mistake})$$
  $$\leq \Pr(X \geq a \cdot E(X)) \quad (E(X) \leq p(n))$$
  $$\leq 1/a = 1/2 \quad (\text{Markov})$$
The reverse inclusion $\text{ZPP}' \subseteq \text{ZPP}$

- Markov on r.v. $X =$ time taken by $\mathcal{M}$ on $x$; also let $a=2$.
- $E(X) \leq p(n)$ finite OK
- If $x \notin L \Rightarrow$ error $= \Pr_r(\mathcal{M}_1(x,r) \text{ accepts})$
  \[ = \Pr(X \geq a \cdot p(n)) \quad (\mathcal{M} \text{ makes no mistake}) \]
  \[ \leq \Pr(X \geq a \cdot E(X)) \quad (E(X) \leq p(n)) \]
  \[ \leq 1/a = 1/2 \quad (\text{Markov}) \]
- If $x \in L \Rightarrow \mathcal{M}_1(x,r)$ must accept.
The reverse inclusion $\text{ZPP}' \subseteq \text{ZPP}$

- Markov on r.v. $X =$ time taken by $M$ on $x$;
  also let $a=2$.
- $E(X) \leq p(n)$ finite OK
- If $x \notin L \Rightarrow$ error $= \Pr_r(M_1(x,r) \text{ accepts})$
  $= \Pr(X \geq a \cdot p(n))$ (M makes no mistake)
  $\leq \Pr(X \geq a \cdot E(X))$ (E(X) $\leq p(n)$)
  $\leq 1/a = 1/2$ (Markov)
- If $x \in L \Rightarrow M_1(x,r)$ must accept.
- Hence $L$ is in $\text{coRP}$.

Let $L$ in $\text{ZPP}'$, decided by $M$ running in average poly. time $p(n)$ with no error.
Define $M_1$ as follows: on input $x$
and random tape $r$ of size $a \cdot p(n)$
simulate $M$ on $x$ for at most $a \cdot p(n)$ steps (timeout).
If timeout reached, then accept (that may be an error).
The reverse inclusion \( \text{ZPP}' \subseteq \text{ZPP} \)

- Markov on r.v. \( X = \) time taken by \( M \) on \( x \); also let \( a=2 \).
- \( E(X) \leq p(n) \) finite OK
- If \( x \in L \Rightarrow \text{error} = \Pr_r(\text{M}_2(x,r) \text{ accepts rejects}) \)
  \[ = \Pr(X \geq a \cdot p(n)) \quad (M \text{ makes no mistake}) \]
  \[ \leq \Pr(X \geq a \cdot E(X)) \quad (E(X) \leq p(n)) \]
  \[ \leq 1/a = 1/2 \quad (\text{Markov}) \]
- If \( x \notin L \Rightarrow \text{M}_2(x,r) \text{ must accept reject.} \)
- Hence \( L \) is in \( \text{coRP} \cap \text{RP} \).
The reverse inclusion $ZPP' \subseteq ZPP$

- Markov on r.v. $X =$ time taken by $M$ on $x$; also let $a=2$.
- $E(X) \leq p(n)$ finite OK
- If $x \in L \Rightarrow$ error $= \Pr_r(M_2(x,r) \text{ accepts } \text{ rejects})$
  
  
  $= \Pr(X \geq a \cdot p(n))$ (M makes no mistake)
  
  $\leq \Pr(X \geq a \cdot E(X))$ ($E(X) \leq p(n)$)
  
  $\leq 1/a = 1/2$ (Markov)
- If $x \notin L \Rightarrow M_2(x,r)$ must accept reject.
- Hence $L$ is in $\text{coRP} \cap \text{RP}$.

Symmetrically:

Let $L$ in $ZPP'$, decided by $M$ running in average poly. time $p(n)$ with no error.
Define $M_2$ as follows: on input $x$

 simulate $M$ on $x$ for at most $a \cdot p(n)$ steps (timeout).
  
  If timeout reached, then reject (that may be an error).

Hence $L$ is both in $\text{RP}$ and $\text{coRP}$, namely in $ZPP$. $\square$
Summary on ZPP

- **Definition.** $\text{ZPP} = \text{RP} \cap \text{coRP}$

- **Theorem.** $\text{ZPP}$ is the class of languages $L$ which we can decide in average polynomial-time with probability zero of making a mistake.
Next time...
**BPP: Bounded Prob. of Error Polynomial time**

- A language $L$ is in **BPP** if and only if there is a polynomial-time TM $M$ such that for every input $x$ (of size $n$):
  - if $x \in L$ then $\Pr_r [M(x,r) \text{ accepts}] \geq 2/3$
  - if $x \notin L$ then $\Pr_r [M(x,r) \text{ accepts}] \leq 1/3$.

**two-sided error:** $\Pr_r [M(x,r) \text{ errs}] \leq 1/3$