Randomized complexity classes

Today: \textit{RP}, \textit{coRP}, and \textit{ZPP}
(what a zoo!)
Today

- Randomized Turing machines
- One-sided error: $\text{RP}$, $\text{coRP}$
- No error: $\text{ZPP}$
- Next time: two-sided error $\text{BPP}$
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
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).
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

❖ One read-only tapes
❖ As many work tapes as you need (but only a constant number!)
❖ (Possibly) one write-only output tape

\[ r \rightarrow \text{random tape} \]
\[ x \rightarrow \text{input tape} \]
Technical points 1/2

- We draw the random tape $r$ uniformly at random
Technical points 1/2

- We draw the random tape $r$ uniformly at random
- We will be interested in probabilities, e.g. $\Pr_r [\mathcal{M}(x,r) \text{ accepts}]$
Technical points 1/2

- We draw the random tape \( r \) uniformly at random.

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

- Random tape must not just be read-only: we impose that \textbf{no bit on} \( r \) \textbf{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: RP

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

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

- $M(x,r)$ 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 **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 $x \in L$ 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 $|r| = p(n)$)
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 $\operatorname{Pr}_r [M(x,r) \text{ accepts}] \geq 1/2$

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

probability taken over all $r \in \{0,1\}^{p(n)}$
**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 $\Pr_r [M(x,r) \text{ accepts}] \geq 1/2$
  - 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 $r \in \{0,1\}^{p(n)}$
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., $\Pr_r[M(x, r) \text{ accepts}] = 0$).

- One-sided error: we make no error if $x \notin L$.

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 $\text{RP}$ 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., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

**One-sided error:** we make no error if $x \notin L$

Perhaps paradoxically, that means that we make no error if $M(x,r)$ accepts (so please do not confuse acceptance with being in the language!)

i.e. there is also a polynomial $p(n)$ such that $M(x,r)$ 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 $\text{RP}$ 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 **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 $\text{RP}$ 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: $\text{RP}$-languages are not defined by « $\text{RP}$-machines » (there is no such notion)

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

- $L$ is in coRP iff complement $L^c$ is in RP, hence:
  - $L$ is in 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 $\Pr_r [M(x,r) \text{ accepts}] \geq \frac{1}{2}$ $M(x,r)$ accepts for every $r$
    - if $x \notin L$ then $M(x,r)$ accepts for no $r$ $\Pr_r [M(x,r) \text{ accepts}] \leq \frac{1}{2}$
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 $\mathbb{P}$
  (now solved: indeed in $\mathbb{P}$ [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 $\mathbb{P}$
(now solved: indeed in $\mathbb{P}$ [Agrawal,Kayal,Saxena 2004])

❖ In $\text{coNP}$ (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 \( \mathbb{P} \)
(now solved: indeed in \( \mathbb{P} \) [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 \( \text{P} \)
  (now solved: indeed in \( \text{P} \) [Agrawal, Kayal, Saxena 2004])

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

❖ In \( \text{NP} \) [Pratt 1975]

❖ Can also be solved efficiently with randomization…
Fermat’s little theorem

- **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 \( r \) \( (1 \leq r < p) \),
  \[ r^{p-1} = 1 \mod p. \]

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

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

- $\Rightarrow$ 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 $O(\log^{1+\varepsilon} p)$)
Fermat’s little theorem

- **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 $O(\log^{1+\varepsilon} p)$)

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

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

- Draw $r$ at random in $[2, p-2]$; accept if $r^{p-1} = 1 \mod p$. 
Fermat’s little theorem in practice

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

- \( \Rightarrow \) draw \( r \) at random 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 \( r \) \( (1 \leq r < p) \),
  \[ rp^{p-1} = 1 \mod p. \]

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

- 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 ≤ $r$ < $p$), $r^{p-1} = 1 \mod p$.

- ⇒ 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

- $r^{86} = 16 \mod 87$
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 $r$ at random in $[2, p-2]$; accept if $r^{p-1} \equiv 1 \pmod{p}$.

- Is 87 prime?

- Draw $r$ at random… say 25

- $r^{86} = 16 \pmod{87}$

- $\Rightarrow$ 87 is **not prime** (definitely)
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 $r$ 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 $r \ (1 \leq r < p)$,
  
  $r^{p-1} = 1 \mod p$.

- ⇒ 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 $r$ ($1 \leq r < p$), $r^{p-1} \equiv 1 \mod p$.

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

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

If $p$ is prime, will succeed for every $r$.
Fermat’s little theorem in practice

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

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

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

- Else, will fail with (hopefully) high probability (0.024 in the example, looks good); but…
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 fail miserably
Carmichael numbers

- A Carmichael number is a number $p$:
  - that is not prime
  - but passes all Fermat tests ($r^{p-1} \equiv 1 \pmod{p}$ for every $r$)
- I.e., on which our hopes of low error 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 ($r^{p-1} \equiv 1 \pmod{p}$ for every $r$)
- I.e., on which our hopes of low error 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):

$$
egin{array}{ccccccc}
    r^q & \ldots & r^{(2^i-1)q} & r^{(2i)q} & \ldots & r^{(2^k-1)q} & r^{(2^k)q} & \mod p \\
    \text{(don’t care \ldots \ldots don’t care)} & -1 & 1 & \ldots & 1 & 1 & 1
\end{array}
$$

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

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):

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

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

For some $i$, or:

$$ \begin{array}{cccccccc}
    r^q & \ldots & r^{(2^{i-1} q)} & r^{(2^i q)} & \ldots & r^{(2^{k-1} q)} & r^{(2^k q)} \\
    1 & \ldots & 1 & 1 & \ldots & 1 & 1 \\
\end{array} \mod p $$
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>$r^{2^i q}$</th>
<th>$r^{2^{i+1} q}$</th>
<th>$r^{2^{i+2} q}$</th>
<th>$\ldots$</th>
<th>$r^{2^{i+k-1} q}$</th>
<th>$r^{2^k q}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\ldots$</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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

$\pmod{p}$

(see also $r^q \equiv 1 \pmod{p}$ and $r^{2^i q} \equiv -1 \pmod{p}$ for some $i$, or:)

1. $r^q \equiv 1 \pmod{p}$
2. $r^{2^i q} \equiv -1 \pmod{p}$
3. $r^{2^{i+1} q} \equiv r^{2^{i+2} q} \equiv \ldots \equiv r^{2^{i+k-1} q} \equiv r^{2^k q} \equiv 1 \pmod{p}$
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 \((p \text{ probably prime})\)
  — otherwise reject \((p \text{ definitely not prime})\)

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

  - **Theorem.** PRIMALITY is in \( \text{coRP} \).

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

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

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., $\Pr_r [M(x,r) \text{ accepts}]=0$).

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 reduction

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

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

❖ A language \( L \) is in \( \text{RP} \) if and only if there is a \textbf{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 \textbf{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 = 1 – 1/2
(= 1/2 here)

error = 1 – 1/2
(= 1/2 here)

error = \varepsilon
Error reduction

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

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

- Error = 1 - 1/2 (= 1/2 here)
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

- 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 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 $\not= \text{RP}$)
  $\text{RP}(1)=\{\text{all languages}\}$ (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., $\Pr_r [M(x,r) \text{ accepts}] = 0$, $\text{error} = \epsilon$)
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:

- 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 $= \varepsilon$)
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 $L$. 
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.

Remember: if \( M(x, r) \) accepts, then \( x \) must be in \( L \).
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 $\text{RP}(\varepsilon)$ 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 $L$. 
Repeating experiments (pretty) formally

- We have defined a new randomized TM $M'(x, r[1]#...#r[K])$ by:

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

Remember: if $M(x, r[i])$ 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., $\Pr_r [M(x,r) \text{ accepts}] = 0$), $\text{error} = \varepsilon$
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-timeTM $M$ 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 $M'(x, r[1]# \ldots # r[K])$ 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 \( r = r[1]\#\ldots\#r[K] \), \( \Pr_r(\mathcal{M}'(x, r) \text{ rejects}) \)
  \[= \Pr_r(\forall i = 1..K, \mathcal{M}(x, r[i]) \text{ rejects}) \]

- 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 \( \mathcal{M}(x,r) \text{ accepts for no } r \) (i.e., \( \Pr_r[\mathcal{M}(x,r) \text{ accepts}] = 0 \), \( \varepsilon \text{ error} = \varepsilon \))

- 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 accept;
    - reject.
Acceptance: 1. if $x \in L$

- If $x \in L$ (recall $L$ in $\text{RP}(\epsilon)$), 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)

- 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., $\Pr_r [M(x,r) \text{ accepts}] = 0$, error $= \epsilon$)

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

- 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 \( M(x, r) \text{ accepts} \) for no \( r \)
    (i.e., \( \Pr_r[M(x, r) \text{ accepts}] = 0 \), \( \text{error} = \varepsilon \))

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

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

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a polynomial-time TM $M$ 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 $x \notin L$; Complexity

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

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

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 $x \in L$ then
  $\Pr_r(M'(x, r) \text{ accepts}) \geq 1 - \epsilon^K$

- If $x \not\in L$, then
  $M'(x, r) \text{ accepts for no } r$

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

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 $x \not\in L$ 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:

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

- If \( x \in L \) 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 \textbf{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., \( \Pr_r[\{M(x,r) \text{ accepts}] \leq \varepsilon \))

- 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.
The hard direction continued

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

- A language $L$ is in $\text{RP}(\varepsilon)$ and only if there is a **polynomial-time** TM $M$ 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 continued

- If $x \in L$ (recall $L$ in $\text{RP}(\varepsilon)$) then $\Pr_r(\mathcal{M}'(x, r) \text{ accepts}) \geq 1 - \varepsilon^K$
- If $x \not\in L$, 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 $x \not\in L$ 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.
The hard direction continued

- 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., $\Pr_r [M(x, r) \text{ accepts}] = 0$)

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 continued

- 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., $\Pr_r [M(x, r) \text{ accepts}] = 0$, 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.
The hard direction: the end

- 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., $\Pr_r [M(x,r) \text{ accepts}] = 0$), $\text{error} = \epsilon$

- 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.
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., $\Pr_r [M(x,r) \text{ accepts}] = 0$, $\text{error} = \varepsilon$)

- 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.
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 $\Pr_r [M(x,r) \text{ accepts}] \geq 1 - \varepsilon$
  - 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 $M'(x, r[1] \# \ldots \# r[K])$ 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 \( \textbf{RP}(\varepsilon) \) 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 = \( \varepsilon \))
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 $\varepsilon = 0.9999999999999999$!

- 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., $\Pr_r [M(x,r) \text{ accepts}] = 0$).

  \[ \text{error} = \varepsilon \]
Can we do even better?

- Hence we define the same class with error $\epsilon = 0.000000000000001$
- ... or with error $\epsilon = 0.99999999999999!$
- Can we make $\epsilon$ go to 0 as $n \to \infty$?

- 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., $\Pr_r [M(x,r) \text{ accepts}] = 0$).
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., \( \Pr_r[M(x,r) \text{ accepts}] = 0 \)) 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.
Let us take $K = a$ 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 $M'(x, r)$ 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.
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
  $M'(x, r) \text{ accepts for no } r$

- 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 $M(x, r)$ accepts for no $r$ (i.e., $\Pr_r [M(x, r) \text{ accepts}] = 0$).

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

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

$\text{for } i=1 \text{ to } K:$

- If $M(x, r[i]) \text{ accepts, then exit the loop and accept;}$
- reject.

$=O(q(n)p(n))$, still polynomial time
The hard direction revisited

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

Let us take $K = a$ polynomial $q(n)$

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., $\Pr_r [M(x, r) \text{ accepts}] = 0$)

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.

$\text{error } \epsilon^K = \epsilon q(n)$
(exponentially small)

$= O(q(n)p(n))$, 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 \( x \notin L \) then \( M(x,r) \) accepts for no \( r \) (i.e., \( \Pr_r [M(x,r) \text{ accepts}] = 0 \))

\( \varepsilon \) error = \( \varepsilon \)
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., $\Pr_r [M(x,r) \text{ accepts} | x \notin L] = 0$)

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

- **Proof.** First,
  \[ P = \text{RP}(0) \subseteq \text{RP}(1/2) = \text{RP} \]

- 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 \( M(x,r) \text{ accepts for no } r \)
    (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 \( L \in \text{RP} \).

- 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., \( \Pr_r [M(x,r) \text{ accepts}] = 0 \)).
Theorem. \( \mathcal{P} \subseteq \mathcal{RP} \subseteq \mathcal{NP} \).

Proof. First,
\( \mathcal{P} = \mathcal{RP}(0) \subseteq \mathcal{RP}(1/2) = \mathcal{RP} \).

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

If \( x \in L \Rightarrow \) for some \( r \),
\( M(x, r) \) accepts.

A language \( L \) is in \( \mathcal{RP} \) 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., \( \Pr_r [M(x, r) \text{ accepts}] = 0 \)).

(in fact, for at least half of them!)
Relation to ordinary classes

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

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

- Second, let $L \in RP$.
  
  - If $x \in L \Rightarrow$ for some $r$, $M(x, r)$ accepts
  
  - If $x \notin L \Rightarrow$ for no $r$.

- 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., $Pr_r [M(x, r) \text{ accepts}] = 0$).

  (in fact, for at least half of them!)
Theorem. $P \subseteq RP \subseteq NP$.

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

Second, let $L \in RP$.

- If $x \in L \Rightarrow$ for some $r$, $M(x, r)$ accepts
- If $x \notin L \Rightarrow$ for no $r$. (in fact, for at least half of them!)

Hence $L = \{x \mid \exists r, M(x, r) \text{ accepts}\}$ is in $NP$. \Box

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., $Pr_r [M(x, r) \text{ accepts}] = 0$).
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: $\text{ZPP} = \text{RP} \cap \text{coRP}$
- Not clear that those two definitions are equivalent, right?
Let us start simple:
Let us start simple:

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

Definition. $\textbf{ZPP} = \textbf{RP} \cap \textbf{coRP}$

I.e., $L$ is in $\textbf{ZPP}$ 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 = RP ∩ 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.** \( \text{ZPP} = \text{RP} \cap \text{coRP} \)

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]
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 $\text{ZPP} = \text{ZPP}'$. 

**ZPP, alternate form**
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)
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
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

… but before that, we explain why (intuitively) \( \text{ZPP} \subseteq \text{ZPP}' \).
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

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

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.$\geq1/2$

- if $x \notin L$ then $M_1(x,r)$ accepts with prob.$\leq1/2$
  $M_2(x,r)$ 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 \textbf{reject}
  - if \( M_2(x,\ldots) \) accepts: stop and \textbf{accept}

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

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 $x \not\in L$ 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 $\leq 1/2$...

why?
(if you tell me that this is even $\leq 1/4$, you are wrong)

Hence this machine never makes any mistake
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 *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

❖ An 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

- An 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

- 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 »

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

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

\[
\mathbb{E}(X) = \sum_{n \geq 1} n \cdot \Pr(X = n) = \sum_{n \geq 1} \Pr(X \geq n) \leq \sum_{n \geq 1} (1 - \alpha)^{n-1} = 1/\alpha. \quad \square
\]
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 $\leq \frac{4}{\pi}$
  $\sim 1.27324$ 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 \( ZPP = \text{RP} \cap \text{coRP} \) 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 $L$)

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

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

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

Hence this machine never makes any mistake
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 $M_1(x,\ldots)$ accepts 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 $L$)

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

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

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]
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 \( a \geq 0 \):
  \[ \Pr(X \geq a \cdot E(X)) \leq 1/a. \]

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 \( ZPP = ZPP' \).
Markov’s inequality

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

- 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 \( a \geq 0 \):
  \[ \Pr(X \geq a \cdot E(X)) \leq \frac{1}{a}. \]
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 $a \cdot E(X)$. $\square$
Markov’s inequality

❖ **Theorem (Markov’s inequality).**
Let $X$ be a **non-negative real-valued** random variable with **finite** expectation $E(X)$. For every $a > 0$:

$$\Pr(X \geq a \cdot E(X)) \leq \frac{1}{a}.$$

❖ **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 poly. time $p(n)$ with no error.

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

  (and random tape $r$ of size $a. p(n)$)

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

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

- Markov on r.v. $X = \text{time taken by } M \text{ on } x$; 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 $x$

- (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 $x \notin L \Rightarrow$ error $= \Pr_r(M_1(x,r) \text{ accepts})$
  
  \[ = \Pr(X \geq a \cdot p(n)) \quad (M \text{ makes no mistake}) \]
  \[ \leq \Pr(X \geq a \cdot E(X)) \quad (E(X) \leq p(n)) \]
  \[ \leq \frac{1}{a} = \frac{1}{2} \quad (\text{Markov}) \]

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 $\mathcal{M}$ on $x$; also let $a=2$.
- $E(X) \leq p(n)$ **finite** OK
- If $x \notin L \Rightarrow$ error $= \Pr_r(\mathcal{M}_1(x,r) \text{ accepts})$
  $= \Pr(X \geq a \cdot p(n))$ ($\mathcal{M}$ makes no mistake)
  $\leq \Pr(X \geq a \cdot E(X))$ ($E(X) \leq p(n)$)
  $\leq 1/a = 1/2$ (Markov)
- If $x \in L \Rightarrow \mathcal{M}_1(x,r)$ 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 $\text{coRP}$.
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 \( x \in L \implies \text{error} = \Pr_r(M_2(x,r) \text{ accepts rejects}) = \Pr(X \geq a \cdot p(n)) \) (\( M \) makes no mistake)
  \[ \leq \Pr(X \geq a \cdot E(X)) \] (\( E(X) \leq p(n) \))
  \[ \leq 1/a = 1/2 \] (Markov)
- If \( x \notin L \implies M_2(x,r) \text{ must accept reject.} \)
- Hence \( L \) is in \( \text{coRP} \).
The reverse inclusion $\text{ZPP}' \subseteq \text{ZPP}$

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

Symmetrically:

Let $L$ in $\text{ZPP}'$, decided by $M$ running in average poly. time $p(n)$ with no error.
Define $M_2$ 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 reject (that may be an error).

Hence $L$ is both in $\text{RP}$ and in $\text{coRP}$, namely in $\text{ZPP}$.  \[\Box\]
Definition. $\text{ZPP} = \text{RP} \cap \text{coRP}$

Theorem. 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:**
$\Pr_r [M(x,r) \text{ errs}] \leq 1/3$