time.

The class of all languages with polynomial time

- If \( x \notin L \), then all computation paths halt with "no."
- If \( x \in L \), then at least half of the \( 2^{|x|} \) computation paths of \( N \) on \( x \) halt with "yes."

for a language \( L \) the following conditions hold:

- \( N \) is a non-deterministic machine
- \( N \) is a polynomial-time machine
- \( N \) is a non-deterministic machine
- \( N \) is a polynomial-time machine

Randomized Complexity Classes
\[(u) = (\tfrac{2}{1})O = \frac{\log_2 \frac{1}{1}}{1} -
\]

the order 1/ for some polynomial d/ (u)

In fact, e can be arbitrarily close to 0 as long as it is of

\[\left| \frac{\log_2 \frac{1}{1}}{1} - e \right| = \text{Now pick } \epsilon -
\]

of false negatives can be reduced to \((1 - \epsilon)\)

By repeating the algorithm \(k\) times, the probability

Any constant \(0 < \epsilon < 1\) can replace 0.5.

The probability of false negatives is at most 0.5.

There are no false positive answers.

Non-deterministic steps can be seen as fair coin flips.

Comments on RP
computation.

RP ⊆ coRP is a “plausible” notion of efficient

$R P \cap co R P$ is

$P \neq \text{ coRP }$

$\text{PRIMES } \subseteq \text{ RP }$

$\text{COMPOSITE } \subseteq \text{ RP }$

accepting paths.

$N T M$ with extra demands on the number of

- A polynomial Monte Carlo $T M$ is a polynomial-time

- A polynomial-time deterministic $T M$ is like a

$P \subseteq \text{ RP } \subseteq \text{ NP }$

Where RP Fits
The algorithm is called Las Vegas.

- A negative answer from the one without false negatives.
- A positive answer from the one without false positives.

One definite answer will come (unlike RP).

If we repeatedly run both Monte Carlo algorithms, eventually

with no false positives and another with no false negatives.

A language in ZPP has two Monte Carlo algorithms, one

The class \text{ZPP} is defined as \text{RP} \cup \text{coRP}.

\text{ZPP} \text{a (Zero Probabilistic Polynomial)}
The ZPP Algorithm

1: while true do
   {negatives.
   \[ N_1 \text{ has no false positives, and } N_2 \text{ has no false}\]
   \{Suppose that } T \in \text{ZPP}\]

2: if \( N_1 \) then
   \text{return "yes"}
else
   \text{return "no"}
end if

3: end while

4: while true do
   \text{end while and if}

5: end if

6: \text{end if and if}

7: \text{if } N_1 \text{ then }
   \text{return "no"}
\text{if } N_2 \text{ then }
   \text{return "yes"}
Primes $\in \text{Zpp}$ (whose proof remains inaccessible).

\[
(u)d\mathcal{A} = (u)d\mathcal{A} \sum_{i=1}^{\infty} g_{i} 0.5\frac{\Gamma}{\infty}
\]

Thus

- The expected running time for a definite answer is
- Let $p(u)$ be the running time of each run.
- Not generate a definite answer is 0.5.
- The probability that a run of the 2 algorithms does polynomial.

- The expected running time for it to happen is

Zpp (continued)
? \not \in T \nRightarrow x \text{ when } x \\
\bullet 
\text{But how to get a "no" when } x \\
\cdot 
\text{You eventually get a "yes" if } x \in T \\

8: \text{end while}
{\text{What to do here?}}
7: \text{if}
6: \text{end if}
5: \text{return "yes" if }
4: \text{\textbf{yes} } = (x)^N \text{ if }
3: \text{while true do}
\{\text{decides } T \text{ without false positives.}\}
2: \text{Suppose that } T \in \text{ RP}
1: \text{Me Too, RP?}
\[ pp \text{ is closed under complement.} \]

\[ \text{MAJSAT is } pp\text{-complete.} \]

Statements to \( \phi \)'s \( u \) variables satisfy it:

\[ \text{MAJSAT: is it true that the majority of the } 2^n \text{ truth} \]

- \( \text{We say that } N \text{ decides by majority.} \)
- \( \text{input } x \text{ end up with a } \text{"yes:} \)
- \( \text{input } x \text{ end up with } \text{"yes."} \)
- \( \text{For all inputs } x, x \in \mathcal{U} \text{ if and only if more than half} \)
- \( \text{polynomial-time promise NTM } \text{N such that:} \)
- \( \text{A language } L \text{ is in the class } pp \text{ if there is a} \)

\[ pp \]
produces \( 2^{|x|}d \) computation paths.

Suppose that \( N \) on input \( x \) computes for \( |x| \) steps and

Consider an input \( x \).

simply accepts (after \( |x|d \) steps).

\( N \) starts at \( s \) and either branches to \( N \)'s program or

\( N \) has one more extra state than \( N \).

Construct a new NTM \( N' \).

Suppose that \( L \) \( \in \) NP is decided by an NTM \( N \).

\( NP \subset \bar{NP} \subset \bar{PP} \subset PP \)

NP vs. PP
So \( N \) accepts \( T \) by majority and \( T \in \text{P} \).

That is, it and only if \( x \in T \).

If at least one path of \( N \) accepts \( x \),

Thus a majority of the paths of \( N \) accept \( x \) if and only

Half of these will always halt with "Yes."

Then \( N \) has \( 2^p(\lfloor |x|/p \rfloor + 1) \) computation paths.

The Proof (continued)
Question: Can you quantify the confidence?

Answer: Flip the coin many times and pick the side that appeared the most times.

How to decide which side is the more likely—with high confidence?

But you do not know which is which.

For some $0 > \varepsilon > 1$, $0.5 - \varepsilon$ to appear and the other side has probability $0.5 + \varepsilon$.

You have a biased coin.

Theory of Large Deviations
The Chernoff bound is asymptotically optimal. 

The probability that the deviation of a random variable from its expected value decreases exponentially with the deviation.

\[
\Pr\left[ u d(\theta + 1) \leq X \right] \geq \frac{e^{-\theta}}{u d^{\theta}}
\]

Let \( \theta \geq 0 \) for all \( i \), then for all \( i \), \( X_i \). Let \( X \) be a random variable taking the values \( 1 \) and \( 0 \) with probabilities \( p \) and \( 1 - p \), respectively. Suppose that \( x_1, x_2, \ldots, x_n \) are independent random variables. 

Theorem 62 (Chernoff, 1952)
\[
\mathbb{E} \left[ ud(\theta + 1) \right] - \mathbb{E} \left[ ud(\theta) \right] \geq \mathbb{E} [X] \text{prob}
\]

With \( \gamma \) we have

\[
\mathbb{E} / \mathbb{E} [X] \text{prob} \geq \mathbb{E} [X] \text{prob}
\]

random variables says that Markov's inequality (p. 282) generalized to real-valued

\[
\mathbb{E} [ ud(\theta + 1) ] \geq \mathbb{E} [ X ] \text{prob} = \mathbb{E} [ ud(\theta + 1) ] \geq \mathbb{E} [ X ] \text{prob}
\]

Then \( \gamma \) be any positive real number.

The Proof
\[ u < 0 \text{ for all } u \in \mathbb{R} \Rightarrow u(\theta + 1) \geq 0, \]

\[ (1 - \varphi)ud\theta ud(\theta + 1) \geq \theta \geq u[(I - \varphi)d + 1]ud(\theta + 1) \geq \left[ ud(\theta + 1) \leq X \right] \]

Substituting we obtain

\[ u[(1 - \varphi)d + 1] = u(\left[ x_1 \varphi \right] \mathcal{H}) = \left[ x_1 \varphi \right] \mathcal{H} \]

Because \( x_1 \) and \( x_2 \) are independent

\[ \mathcal{X} \]

The Proof (continued)
\[
\frac{3}{\varepsilon \theta} = \left( \frac{9}{I} + \frac{2}{I} \right) \varepsilon \theta \geq \left( \frac{9}{\theta} + \frac{2}{I} \right) \varepsilon \theta \geq \frac{9}{\varepsilon \theta} + \frac{2}{\varepsilon \theta}
\]

which is less than \( \bar{I} \geq \theta \geq 0 \) for \( \cdots + \frac{2}{I} \varepsilon \theta - \frac{9}{\varepsilon \theta} + \frac{2}{\varepsilon \theta} \) to expands the exponent. With the choice of \( \theta = \text{in}(I) \), the above becomes

\[\text{Proof}\]

\[\text{The Proof (continued)}\]
with probability at most $2^{-f}$ with the majority rule.

Our original problem (p. 329) hence demands $1.4/\varepsilon^2$.

The textbook’s corollary to Lemma 1.9 seems incorrect.

$$
\Pr_{/u, \varepsilon} \geq \left[ \frac{z}{u} \geq \sum_{x=1}^{i} \right] 
$$

Corollary 63: For some $1/2$, then $1/2 \geq \varepsilon \geq 0$ for.

From this it follows:

**Effectiveness of the Majority Rule**
\[ N \text{ accepts or rejects by a clear majority:} \]

- If \( x \not\in L \), then at least \( 3/4 \) of the computation paths of \( N \) on \( x \) reject.
- If \( x \in L \), then at least \( 3/4 \) of the computation paths of \( N \) on \( x \) accept, and

\[ \exists \text{ a precise polynomial-time TM } N \text{ such that:} \]

- The class \( \text{BPP} \) (Bounded Probabilistic Polynomial)
between $1/2$ and $1$ can be used.

$$(u)d / (0.5 + 1)$$

In fact, any $0.5$ plus inverse polynomial.

Without affecting the class $\text{BPP}$.

Any constant strictly between $1/2$ and $1$ can be used.

The number $3/4$ bounds the probability of a right answer away from $1/2$.

$\text{Magic} 3/4$
The Majority Vote Algorithm

1. Suppose that $T$ is decided by $N$ by majority $(1/2) + \epsilon$.

2. Run $N$ on input $x$.

3. end for

4. If "yes" is the majority answer then

5. "yes"

6. else

7. "no"

8. end if
because $k$ remains polynomial in $n$. 

As with the RP case, $e$ can be any inverse polynomial, 

\[ \frac{1}{4} \]

By taking $\gamma = \frac{2}{\sqrt{e^2}},$ the error probability is at most 

answer is at most $e^{-e^2}$. 

By Corollary 63 (p. 334), the probability of a false 

times $N$'s running time. 

The running time remains polynomial, being $2k + 1$ 

Analysis
Whether \( \text{BPP} \subseteq (\text{NP} \cap \text{coNP}) \) is unknown.

If a problem is in \( \text{BPP} \), we take it to mean that the problem can be solved efficiently.

- \( \text{BPP} \) is the most comprehensive yet plausible notion of efficient computation.

Aspects of \( \text{BPP} \):
A Review of Classes
NP either.

This approach does not work for \( \text{RP} \) (it did not work for \( \text{NP} \)) either.

Hence \( \text{BPP} = \text{coBPP} \).

By reversing the answer.

An algorithm for \( L \in \text{BPP} \) becomes one for \( \overline{L} \in \text{coBPP} \).

The definition of \( \text{BPP} \) is symmetric: acceptance by clear majority and rejection by clear minority.

\( \text{coBPP} \)
circuit for each possible input length $n$.

To relate circuits with arbitrary languages, we need one circuit with $n$ inputs that accepts certain strings in $\{0, 1\}^n$. By identity true with 1 and false with 0, a boolean function of $n$ variables.

A boolean circuit with $n$ inputs computes a boolean circuit of Turing machines.

Circuit complexity is based on boolean circuit complexity.
\[
\text{For all inputs } 0, 1, \text{ the circuit } C \text{ accepts } 0 \cup \mathcal{T} \text{ if and only if:}
\]

- For input } x \in \{0, 1\} \text{, the output } |x| \text{ is at most } p(\mu) \text{ for some fixed polynomial } p.

- The size of the family } C \text{ such that:}

  - There is a family } C \text{ has polynomial circuits if there is a family of circuits } C \text{ has } n \text{ boolean inputs.}

  - \text{A family } C \text{ of circuits is an infinite sequence } C = (C_0, C_1, \ldots)\text{ of boolean circuits, where } C_n \text{ has size } n \text{ of circuits is the number of gates in it.}

\text{Formal Definitions}
The size of the circuit is polynomial in \( n \).

The size of the circuit depends only on \( T \) and the length of the input.

\( \mathcal{u} \{ 0, 1 \} \cup T \) gates that accepts \( u \) gives, for any input of size \( n \), a circuit with \( O(2^n) \).

The construction in the proof of Theorem 25 (p. 169) proves that \( \mathcal{P} \) be decided by a \( \mathcal{TM} \) in time \( p(n) \).

Proposition 6.4. All languages in \( \mathcal{P} \) have polynomial

The Circuit Complexity of \( \mathcal{P} \)
The family of circuits \( (C_0, C_1, \ldots) \) is polynomial in size.

- \( \bigwedge \) and trivially true if \( u \in \bigwedge \).
- \( u \bigwedge \not\in \bigwedge \).
- \( \bigwedge \{1\} \cup \bigwedge \) must be undecidable.
- \( \{0, 1\} \subseteq \bigwedge \) be an undecidable language.
- \( \bigwedge \) circuits.

There are undecidable languages that have polynomial circuits.

- \( \bigwedge \) is unique that polynomial circuits accept only languages in

languages that polynomial circuits accept.
The effective and efficient constructibility of $C_0, C_1$...

What gives?

- Efficient computation.

- Polynomial circuits are not a plausible notion of computation.

- Circuits are not a realistic model of computation.

Despite their simplicity,

A Patch
A family \((C_0, C_1, \ldots)\) of circuits is uniform if there is a
log\(n\)-space bounded TM which on input \(1^n\) outputs \(C_n\).

- Circuits now cannot accept undecidable languages.

Uniformity

- A language has uniformly polynomial circuits if there is a uniform family of polynomial circuits that decides it.