Complexity Theory

Jan Křetínský

Chair for Foundations of Software Reliability
and Theoretical Computer Science
Technical University of Munich

Summer 2016

Based on slides by Jörg Kreiker

Lecture 12–13

Randomization and Polynomial Time

“Realistic computation somewhere between P and NP”

Agenda

- Motivation: From **NP** to a more realistic class by randomization
  - Choosing the certificate at random
  - Error reduction by rerunning
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP**
- Power of randomization with two-sided error: **PP**, **BPP**

Recap **P**

Definition (P)

For every $L \subseteq \{0,1\}^*$:
$L \in$ **P** if there is a poly-time TM M such that for every $x \in \{0,1\}^*$:

$$x \in L \Leftrightarrow M(x) = 1.$$

- "poly-time TM M":
  - M deterministic
  - M outputs $\{0,1\}$
  - There is a polynomial $T(n)$ s.t. M halts on every x within $T(|x|)$ steps.
- Problems in **P** are deemed "tractable".

Recap NP

Theorem (Certificates)

For every $L \subseteq \{0,1\}^*$:
*$L \in$ **NP** if and only if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a poly-time TM M such that for every $x \in \{0,1\}^*$*

$$x \in L \Leftrightarrow \exists u \in \{0,1\}^{p(|x|)} \; : \; M(x,u) = 1$$

- Certificate u: satisfying assignment, independent set, 3-coloring, etc.
- **NP** captures the class of possibly (not) tractable computations:
  - Don't know how to compute u in poly-time, but
  - if there is a u, then $|u|$ is polynomial in $|x|$, and
  - we can check in poly-time if a u is a certificate/solution.

Recap NP

Theorem (Certificates)

For every $L \subseteq \{0,1\}^*$*:*
$L \in \mathbf{NP}$ *if and only if there exists a polynomial* $p : \mathbb{N} \to \mathbb{N}$ *and a poly-time TM* M *such that for every* $x \in \{0,1\}^*$

$$x \in L \Leftrightarrow \exists u \in \{0,1\}^{p(|x|)} \; : \; M(x,u) = 1$$

- Certificate u: satisfying assignment, independent set, 3-coloring, etc.
- **NP** captures the class of possibly (not) tractable computations:
  - Don't know how to compute u in poly-time, but
  - if there is a u, then $|u|$ is polynomial in $|x|$, and
  - we can check in poly-time if a u is a certificate/solution.
- NDTMs can check all $2^{p(|x|)}$ possible us in parallel.
- Seems unrealistic. Common conjecture: $\mathbf{P} \neq \mathbf{NP}$.

Recap NP

Theorem (Certificates)

For every $L \subseteq \{0,1\}^*$:
*$L \in$ **NP** if and only if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a poly-time TM M such that for every $x \in \{0,1\}^*$*

$$x \in L \Leftrightarrow \exists u \in \{0,1\}^{p(|x|)} \; : \; M(x,u) = 1$$

- Certificate u: satisfying assignment, independent set, 3-coloring, etc.
- **NP** captures the class of possibly (not) tractable computations:
  - Don't know how to compute u in poly-time, but
  - if there is a u, then $|u|$ is polynomial in $|x|$, and
  - we can check in poly-time if a u is a certificate/solution.
- NDTMs can check all $2^{p(|x|)}$ possible us in parallel.
- Seems unrealistic. Common conjecture: **P** $\neq$ **NP**.
- Goal: Obtain from **NP** a more realistic class by randomization:

Choose u uniformly at random from $\{0,1\}^{p(|x|)}$.

Randomizing NP

Definition (Accept/Reject certificates and probabilities)

Fix some $L \in \mathbf{NP}$ decided by M using certificates u of length $p(\cdot)$:

$$A_{M,x} := \{u \in \{0,1\}^{p(|x|)} \mid M(x,u) = 1\} \text{ and } R_{M,x} := \{0,1\}^{p(|x|)} \setminus A_{M,x}.$$

Randomizing NP

Definition (Accept/Reject certificates and probabilities)

Fix some $L \in \mathbf{NP}$ decided by M using certificates u of length $p(\cdot)$:

$$A_{M,x} := \{u \in \{0,1\}^{p(|x|)} \mid M(x,u) = 1\} \text{ and } R_{M,x} := \{0,1\}^{p(|x|)} \setminus A_{M,x}.$$

- If we choose $u \in \{0,1\}^{p(|x|)}$ uniformly at random:
  - $A_{M,x}$ is the event that u "says accept x".
  - $R_{M,x}$ is the event that u "says reject x".

Randomizing NP

Definition (Accept/Reject certificates and probabilities)

Fix some $L \in \mathbf{NP}$ decided by M using certificates u of length $p(\cdot)$:

$$A_{M,x} := \{u \in \{0,1\}^{p(|x|)} \mid M(x,u) = 1\} \text{ and } R_{M,x} := \{0,1\}^{p(|x|)} \setminus A_{M,x}.$$

- If we choose $u \in \{0,1\}^{p(|x|)}$ uniformly at random:
  - $A_{M,x}$ is the event that u “says accept x”.
  - $R_{M,x}$ is the event that u “says reject x”.

Definition (Accept/Reject certificates and probabilities (cont’d))

$$\Pr[A_{M,x}] := \frac{|A_{M,x}|}{2^{p(|x|)}} \text{ and } \Pr[R_{M,x}] := \frac{|R_{M,x}|}{2^{p(|x|)}} = 1 - \Pr[A_{M,x}].$$

Randomizing NP

Definition (Accept/Reject certificates and probabilities)

Fix some $L \in \mathbf{NP}$ decided by M using certificates u of length $p(\cdot)$:

$$A_{M,x} := \{u \in \{0,1\}^{p(|x|)} \mid M(x,u) = 1\} \text{ and } R_{M,x} := \{0,1\}^{p(|x|)} \setminus A_{M,x}.$$

- If we choose $u \in \{0,1\}^{p(|x|)}$ uniformly at random:
  - $A_{M,x}$ is the event that u "says accept x".
  - $R_{M,x}$ is the event that u "says reject x".

Definition (Accept/Reject certificates and probabilities (cont'd))

$$\Pr[A_{M,x}] := \frac{|A_{M,x}|}{2^{p(|x|)}} \text{ and } \Pr[R_{M,x}] := \frac{|R_{M,x}|}{2^{p(|x|)}} = 1 - \Pr[A_{M,x}].$$

$L \in \mathbf{NP}$ iff $\forall x \in \{0,1\}^*$:

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 2^{-p(|x|)} \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0.$$

Randomizing NP: Example SAT

- Input: CNF-formula ϕ with n variables.
- Output: Choose truth assignment $u \in \{0, 1\}^n$ uniformly at random.
    - If u satisfies ϕ, output yes, $\phi \in$ SAT.
    - Else, output probably, $\phi \notin$ SAT.

- If output is yes, $\phi \in$ SAT, then we know $\phi \in$ SAT for sure.
- But what if output is probably, $\phi \notin$ SAT?

Randomizing NP: Example SAT

- Input: CNF-formula ϕ with n variables.
- Output: Choose truth assignment $u \in \{0,1\}^n$ uniformly at random.
  - If u satisfies ϕ, output yes, $\phi \in$ SAT.
  - Else, output probably, $\phi \notin$ SAT.

- If output is yes, $\phi \in$ SAT, then we know $\phi \in$ SAT for sure.
- But what if output is probably, $\phi \notin$ SAT?
- Consider $\phi = x_1 \wedge x_2 \wedge \ldots \wedge x_n \in$ SAT:
  - Probability of probably, $\phi \notin$ SAT: $\Pr[R_{M,x}] = 1 - 2^{-n}$
  - Called false negative.

Randomizing NP: Example SAT

- Input: CNF-formula ϕ with n variables.
- Output: Choose truth assignment $u \in \{0, 1\}^n$ uniformly at random.
  - If u satisfies ϕ, output yes, $\phi \in$ SAT.
  - Else, output probably, $\phi \notin$ SAT.

- If output is yes, $\phi \in$ SAT, then we know $\phi \in$ SAT for sure.
- But what if output is probably, $\phi \notin$ SAT?
- Consider $\phi = x_1 \wedge x_2 \wedge \ldots \wedge x_n \in$ SAT:
  - Probability of probably, $\phi \notin$ SAT: $\Pr[R_{M,x}] = 1 - 2^{-n}$
  - Called false negative.
- If we run this algorithm r-times,
  prob. of false negative decreases to: $(1 - 2^{-n})^r \approx e^{-r/2^n}$.

Randomizing NP: Example SAT

- Input: CNF-formula ϕ with n variables.
- Output: Choose truth assignment $u \in \{0,1\}^n$ uniformly at random.
  - If u satisfies ϕ, output yes, $\phi \in$ SAT.
  - Else, output probably, $\phi \notin$ SAT.

- If output is yes, $\phi \in$ SAT, then we know $\phi \in$ SAT for sure.
- But what if output is probably, $\phi \notin$ SAT?
- Consider $\phi = x_1 \wedge x_2 \wedge \ldots \wedge x_n \in$ SAT:
  - Probability of probably, $\phi \notin$ SAT: $\Pr[R_{M,x}] = 1 - 2^{-n}$
  - Called false negative.
- If we run this algorithm r-times,
  prob. of false negative decreases to: $(1 - 2^{-n})^r \approx e^{-r/2^n}$.
- Exponential number $r \sim 2^n$ required to reduce this to any tolerable error bound like $1/4$ or $1/10$.
- Not that helpful as SAT $\in$ **EXP** (zero prob. of false negative).

Randomizing NP: Conclusion

- Not enough to only choose certificate u at random,
  we need to require that $\Pr[A_{M,x}]$ is significantly larger than $2^{-p(|x|)}$;
  otherwise we'll stay in **NP**.

Randomizing **NP**: Conclusion

- Not enough to only choose certificate u at random,
  we need to require that $\Pr[A_{M,x}]$ is significantly larger than $2^{-p(|x|)}$;
  otherwise we'll stay in **NP**.
- Goal:
  Polynomial number $r(|x|)$ of reruns should make prob. of false negatives arbitrary small.

Randomizing NP: Conclusion

- Not enough to only choose certificate u at random,
  we need to require that $\Pr[A_{M,x}]$ is significantly larger than $2^{-p(|x|)}$;
  otherwise we'll stay in **NP**.
- Goal:
  Polynomial number $r(|x|)$ of reruns should make prob. of false negatives arbitrary small.
- This holds if $\Pr[A_{M,x}] \geq n^{-k}$ for some $k > 0$:

$$(1 - \Pr[A_{M,x}])^{c|x|^{k+d}} \geq \left(1 - 1/|x|^k\right)^{c|x|^{k+d}} \approx e^{-c|x|^d}$$

as $\lim_{m\to\infty}(1 - 1/m)^m = e^{-1}$.

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
  - Choosing the certificate at random ✓
  - Error reduction by rerunning ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP**
  - Definitions
  - Monte Carlo and Las Vegas algorithms
  - Examples: ZEROP and perfect matchings
- Power of randomization with two-sided error: **PP**, **BPP**

Definition of RP

Definition (Randomized P (RP))

$L \in \mathbf{RP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0.$$

- $\mathbf{P} \subseteq \mathbf{RP} \subseteq \mathbf{NP}$
- $\mathbf{coRP} := \{\overline{L} \mid L \in \mathbf{RP}\}$
- **RP** unchanged if we replace $\geq 3/4$ by $\geq n^{-k}$ or $\geq 1 - 2^{-n^k}$ ($k > 0$).

Definition of RP

Definition (Randomized P (RP))

$L \in \mathbf{RP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0.$$

- $\mathbf{P} \subseteq \mathbf{RP} \subseteq \mathbf{NP}$
- $\mathbf{coRP} := \{\overline{L} \mid L \in \mathbf{RP}\}$
- **RP** unchanged if we replace $\geq 3/4$ by $\geq n^{-k}$ or $\geq 1 - 2^{-n^k}$ $(k > 0)$.
- Realistic model of computation? How to obtain random bits?
    - "Slightly random sources": see e.g. Papadimitriou p. 261

Definition of RP

Definition (Randomized P (RP))

$L \in$ **RP** if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0.$$

- **P** $\subseteq$ **RP** $\subseteq$ **NP**
- **coRP** $:= \{\overline{L} \mid L \in \textbf{RP}\}$
- **RP** unchanged if we replace $\geq 3/4$ by $\geq n^{-k}$ or $\geq 1 - 2^{-n^k}$ $(k > 0)$.
- Realistic model of computation? How to obtain random bits?
  - "Slightly random sources": see e.g. Papadimitriou p. 261
- One-sided error probabiliy for **RP**:
  - False negatives: if $x \in L$, then $\Pr[R_{M,x}] \leq 1/4$.
  - If $M(x, u) = 1$, output $x \in L$; else output probably, $x \notin L$
  - Error reduction by rerunning a polynomial number of times.

coRP, ZPP

Lemma (coRP)

*$L \in$ **coRP** if and only if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$*

$$x \in L \Rightarrow Pr[A_{M,x}] = 1 \textit{ and } x \notin L \Rightarrow Pr[A_{M,x}] \leq 1/4.$$

- One-sided error probability for **coRP**:
  - False positives: if $x \notin L$, then $\Pr[A_{M,x}] \leq 1/4$.
  - If $M(x, u) = 1$, output probably, $x \in L$; else output $x \notin L$

coRP, ZPP

Lemma (coRP)

*$L \in$ **coRP** if and only if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0,1\}^*$*

$$x \in L \Rightarrow Pr[A_{M,x}] = 1 \text{ and } x \notin L \Rightarrow Pr[A_{M,x}] \leq 1/4.$$

- One-sided error probability for **coRP**:
  - False positives: if $x \notin L$, then $\Pr[A_{M,x}] \leq 1/4$.
  - If $M(x, u) = 1$, output probably, $x \in L$; else output $x \notin L$

Definition (“Zero Probability of Error”-P (ZPP))

$$\mathbf{ZPP} := \mathbf{RP} \cap \mathbf{coRP}$$

- If $L \in$ **ZPP**, then we have both an **RP**- and a **coRP**-TM for L.

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP**
  - Definitions ✓
  - Monte Carlo and Las Vegas algorithms
  - Examples: ZEROP and perfect matchings
- Power of randomization with two-sided error: **PP**, **BPP**

RP-algorithms

- Assume $L \in \mathbf{RP}$ decided by TM $M(\cdot, \cdot)$.
- Given input x:
  - Choose $u \in \{0, 1\}^{p(|x|)}$ uniformly at random.
  - Run $M(x, u)$.
  - If $M(x, u) = 1$, output: yes, $x \in L$.
  - If $M(x, u) = 0$, output: probably, $x \notin L$.
- Called Monte Carlo algorithm.

RP-algorithms

- Assume $L \in$ **RP** decided by TM $M(\cdot, \cdot)$.
- Given input x:
  - Choose $u \in \{0, 1\}^{p(|x|)}$ uniformly at random.
  - Run $M(x, u)$.
  - If $M(x, u) = 1$, output: yes, $x \in L$.
  - If $M(x, u) = 0$, output: probably, $x \notin L$.
- Called Monte Carlo algorithm.
- If we rerun this algorithm exactly k-times:
  - If $x \in L$, probability that at least once yes, $x \in L$

$$\geq 1 - (1 - 3/4)^k = 1 - 4^{-k}$$

  - but if $x \notin L$, we will never know for sure.

RP-algorithms

- Assume $L \in$ **RP** decided by TM $M(\cdot,\cdot)$.
- Given input x:
  - Choose $u \in \{0,1\}^{p(|x|)}$ uniformly at random.
  - Run $M(x,u)$.
  - If $M(x,u) = 1$, output: yes, $x \in L$.
  - If $M(x,u) = 0$, output: probably, $x \notin L$.
- Called Monte Carlo algorithm.
- If we rerun this algorithm exactly k-times:
  - If $x \in L$, probability that at least once yes, $x \in L$

$$\geq 1 - (1 - 3/4)^k = 1 - 4^{-k}$$

  - but if $x \notin L$, we will never know for sure.
- Expected running time if we rerun till output yes, $x \in L$:
  - If $x \in L$:
    - Number of reruns geometrically distributed with success prob. $\geq 3/4$, i.e.,
    - the expected number of reruns is at most $4/3$.
    - Expected running time also polynomial.
  - If $x \notin L$:
    - We run forever.

ZPP-algorithms

- Assume $L \in$ **ZPP**.
- Then we have Monte Carlo algorithms for both $x \in L$ and $x \in \overline{L}$.
- Given x:
  - Run both algorithms once.
  - If both reply probably, then output don't know.
  - Otherwise forward the (unique) yes-reply.
- Called Las Vegas algorithm

ZPP-algorithms

- Assume $L \in$ **ZPP**.
- Then we have Monte Carlo algorithms for both $x \in L$ and $x \in \overline{L}$.
- Given x:
  - Run both algorithms once.
  - If both reply probably, then output don't know.
  - Otherwise forward the (unique) yes-reply.
- Called Las Vegas algorithm
- If we rerun this algorithm exactly k-times:
  - If $x \in L$ ($x \in \overline{L}$), probability that at least once yes, $x \in L$ (yes, $x \in \overline{L}$)

$$\geq 1 - (1 - 3/4)^k = 1 - 4^{-k}$$

ZPP-algorithms

- Assume $L \in$ **ZPP**.
- Then we have Monte Carlo algorithms for both $x \in L$ and $x \in \overline{L}$.
- Given x:
  - Run both algorithms once.
  - If both reply probably, then output don't know.
  - Otherwise forward the (unique) yes-reply.
- Called Las Vegas algorithm
- If we rerun this algorithm exactly k-times:
  - If $x \in L$ ($x \in \overline{L}$), probability that at least once yes, $x \in L$ (yes, $x \in \overline{L}$)

$$\geq 1 - (1 - 3/4)^k = 1 - 4^{-k}$$

- Expected running time if we rerun till output yes:
  - In both cases expected number of reruns at most $4/3$.
  - So, randomized algorithm which decides L in expected polynomial time.

ZPP-algorithms

- Assume $L \in$ **ZPP**.
- Then we have Monte Carlo algorithms for both $x \in L$ and $x \in \overline{L}$.
- Given x:
  - Run both algorithms once.
  - If both reply probably, then output don't know.
  - Otherwise forward the (unique) yes-reply.
- Called Las Vegas algorithm
- If we rerun this algorithm exactly k-times:
  - If $x \in L$ ($x \in \overline{L}$), probability that at least once yes, $x \in L$ (yes, $x \in \overline{L}$)

$$\geq 1 - (1 - 3/4)^k = 1 - 4^{-k}$$

- Expected running time if we rerun till output yes:
  - In both cases expected number of reruns at most $4/3$.
  - So, randomized algorithm which decides L in expected polynomial time.
- More on expected running time vs. exact running time later on.

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP**
  - Definitions ✓
  - Monte Carlo and Las Vegas algorithms ✓
  - Examples: ZEROP and perfect matchings
- Power of randomization with two-sided error: **PP**, **BPP**

ZEROP

- Given: Multivariate polynomial $p(x_1, \ldots, x_k)$, not necessarily expanded, but evaluable in polynomial time.
- Wanted: Decide if $p(x_1, \ldots, x_k)$ is the zero polynomial.

$$\left|\begin{pmatrix} 0 & y^2 & xy \\ z & 0 & y \\ 0 & yz & xz \end{pmatrix}\right| = -y^2(z \cdot xz - 0) + xy(z \cdot yz - 0) = -xy^2z^2 + xy^2z^2 = 0$$

- ZEROP := “All zero polynomials evaluable polynomial time”.
- E.g. determinant: substitute values for variables, then use Gauß-elemination.
- Not known to be in **P**.

Lemma (cf. Papadimitriou p. 243)

Let $p(x_1, \ldots, x_k)$ be a nonzero polynomial with each variable x_i of degree at most d. Then for $M \in \mathbb{N}$:

$$\left|\{(x_1, \ldots, x_k) \in \{0, 1, \ldots, M-1\}^k \mid p(x_1, \ldots, x_k) = 0\}\right| \leq kdM^{k-1}.$$

ZEROP

Lemma (cf. Papadimitriou p. 243)

Let $p(x_1, \ldots, x_k)$ be a nonzero polynomial with each variable x_i of degree at most d. Then for $M \in \mathbb{N}$:

$$\left|\{(x_1, \ldots, x_k) \in \{0, 1, \ldots, M-1\}^k \mid p(x_1, \ldots, x_k) = 0\}\right| \leq kdM^{k-1}.$$

Let $X_1, \ldots, X_k$ be independent random variables, each uniformly distributed on $\{0, 1, \ldots, M-1\}$. Then for $M = 4kd$:

$$p \notin \text{ZEROP} \Rightarrow \Pr\left[p(X_1, \ldots, X_k) = 0\right] \leq \frac{kdM^{k-1}}{M^k} = \frac{kd}{M} = \frac{1}{4}.$$

ZEROP

Lemma (cf. Papadimitriou p. 243)

Let $p(x_1, \ldots, x_k)$ be a nonzero polynomial with each variable x_i of degree at most d. Then for $M \in \mathbb{N}$:

$$\left|\{(x_1, \ldots, x_k) \in \{0, 1, \ldots, M-1\}^k \mid p(x_1, \ldots, x_k) = 0\}\right| \leq kdM^{k-1}.$$

Let $X_1, \ldots, X_k$ be independent random variables, each uniformly distributed on $\{0, 1, \ldots, M-1\}$. Then for $M = 4kd$:

$$p \notin \text{ZEROP} \Rightarrow \Pr\left[p(X_1, \ldots, X_k) = 0\right] \leq \frac{kdM^{k-1}}{M^k} = \frac{kd}{M} = \frac{1}{4}.$$

- So we can decide $p \in$ ZEROP in **coRP** if
  - we can evaluate $p(\cdot)$ in polynomial time, and
  - d is polynomial in the representation of p.

ZEROP

Lemma (cf. Papadimitriou p. 243)

Let $p(x_1, \ldots, x_k)$ be a nonzero polynomial with each variable x_i of degree at most d. Then for $M \in \mathbb{N}$:

$$\left|\{(x_1, \ldots, x_k) \in \{0, 1, \ldots, M-1\}^k \mid p(x_1, \ldots, x_k) = 0\}\right| \leq kdM^{k-1}.$$

Let $X_1, \ldots, X_k$ be independent random variables, each uniformly distributed on $\{0, 1, \ldots, M-1\}$. Then for $M = 4kd$:

$$p \notin \text{ZEROP} \Rightarrow \Pr[p(X_1, \ldots, X_k) = 0] \leq \frac{kdM^{k-1}}{M^k} = \frac{kd}{M} = \frac{1}{4}.$$

- So we can decide $p \in$ ZEROP in **coRP** if
  - we can evaluate $p(\cdot)$ in polynomial time, and
  - d is polynomial in the representation of p.
- See Arora p. 130 for work around if d is exponential
  - E.g. $p(x) = (\ldots((x-1)^2)^2 \ldots)^2$.

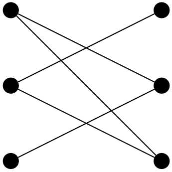
Perfect Matchings in Bipartite Graphs

- Given: bipartite graph $G = (U, V, E)$ with

$$|U| = |V| = n \text{ and } E \subseteq U \times V.$$

- Wanted: $M \subseteq E$ such that

$$\forall (u, v), (u', v') \in M : u \neq u' \wedge v \neq v'.$$



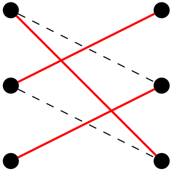
Perfect Matchings in Bipartite Graphs

- Given: bipartite graph $G = (U, V, E)$ with

$$|U| = |V| = n \text{ and } E \subseteq U \times V.$$

- Wanted: $M \subseteq E$ such that

$$\forall (u, v), (u', v') \in M : u \neq u' \wedge v \neq v'.$$



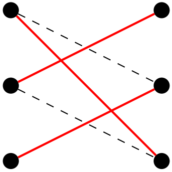
Perfect Matchings in Bipartite Graphs

- Given: bipartite graph $G = (U, V, E)$ with

$$|U| = |V| = n \text{ and } E \subseteq U \times V.$$

- Wanted: $M \subseteq E$ such that

$$\forall (u, v), (u', v') \in M : u \neq u' \wedge v \neq v'.$$



- Problem is known to be solvable in time $\mathcal{O}(n^5)$ (and better).
- So it is in **RP**.

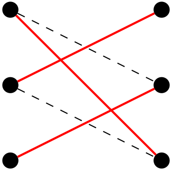
Perfect Matchings in Bipartite Graphs

- Given: bipartite graph $G = (U, V, E)$ with

$$|U| = |V| = n \text{ and } E \subseteq U \times V.$$

- Wanted: $M \subseteq E$ such that

$$\forall (u, v), (u', v') \in M : u \neq u' \wedge v \neq v'.$$



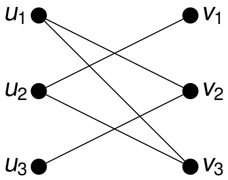
- Problem is known to be solvable in time $\mathcal{O}(n^5)$ (and better).
- So it is in **RP**.
- Still, some “easy” randomized algorithm relying on ZEROP.

Perfect Matchings in Bipartite Graphs

- For bipartite graph $G = (U, V, E)$ define square matrix M:

$$M_{ij} = \begin{cases} x_{ij} & \text{if } (u_i, v_j) \in E \\ 0 & \text{else .} \end{cases}$$

- Output:
  - “has perfect matching” if $\det(M) \notin$ ZEROP
  - “might not have perfect matching” if $\det(M) \in$ ZEROP



$$\left| \begin{pmatrix} 0 & x_{1,2} & x_{1,3} \\ x_{2,1} & 0 & x_{2,3} \\ 0 & x_{3,2} & 0 \end{pmatrix} \right| = -x_{1,3} x_{2,1} x_{3,2}$$

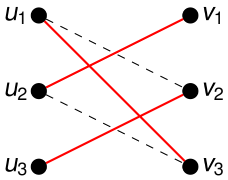
- Relies on Leibniz formula: $\det M = \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) \prod_{i=1}^{n} M_{i,\sigma(i)}$.

Perfect Matchings in Bipartite Graphs

- For bipartite graph $G = (U, V, E)$ define square matrix M:

$$M_{ij} = \begin{cases} x_{ij} & \text{if } (u_i, v_j) \in E \\ 0 & \text{else }. \end{cases}$$

- Output:
  - “has perfect matching” if $\det(M) \notin$ ZEROP
  - “might not have perfect matching” if $\det(M) \in$ ZEROP



$$\left|\begin{pmatrix} 0 & x_{1,2} & x_{1,3} \\ x_{2,1} & 0 & x_{2,3} \\ 0 & x_{3,2} & 0 \end{pmatrix}\right| = -x_{1,3}x_{2,1}x_{3,2}$$

- Relies on Leibniz formula: $\det M = \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) \prod_{i=1}^{n} M_{i,\sigma(i)}$.

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP** ✓
  - Definitions ✓
  - Monte Carlo and Las Vegas algorithms ✓
  - Examples: ZEROP and perfect matchings ✓
- Power of randomization with two-sided error: **PP**, **BPP**
  - Enlarging **RP** by false negatives and false positives
  - Comparison: **NP**, **RP**, **coRP**, **ZPP**, **BPP**, **PP**
  - Probabilistic Turing machines
  - Expected running time
  - Error reduction for **BPP**
  - Some kind of derandomization for **BPP**
  - **BPP** in the polynomial hierarchy

Probability of error for both $x \in L$ and $x \notin L$

- **RP** obtained from **NP** by
  - choosing certificate u uniformly at random
  - requiring a fixed fraction of accept-certificates if $x \in L$

$$x \in L \Rightarrow \Pr\left[A_{M,x}\right] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr\left[A_{M,x}\right] = 0.$$

- **RP**-algorithms can only make errors for $x \in L$.

Probability of error for both $x \in L$ and $x \notin L$

- **RP** obtained from **NP** by
  - choosing certificate u uniformly at random
  - requiring a fixed fraction of accept-certificates if $x \in L$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0.$$

- **RP**-algorithms can only make errors for $x \in L$.
- By allowing both errors for both cases, can we obtain a class that is
  - larger than **RP**,
  - but still more realistic than **NP**?

Probability of error for both $x \in L$ and $x \notin L$

- **RP** obtained from **NP** by
  - choosing certificate u uniformly at random
  - requiring a fixed fraction of accept-certificates if $x \in L$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0.$$

- **RP**-algorithms can only make errors for $x \in L$.
- By allowing both errors for both cases, can we obtain a class that is
  - larger than **RP**,
  - but still more realistic than **NP**?
- Assume we change the definition of **RP** to:

$$x \in L \Leftrightarrow \Pr[A_{M,x}] \geq 3/4.$$

- Two-sided error probabilities:
  - False negatives: If $x \in L$: $\Pr[R_{M,x}] \leq 1/4$
  - False positives: If $x \notin L$: $\Pr[A_{M,x}] < 3/4$
  - Outputs: probably, $x \in L$ and probably, $x \notin L$

Probabilistic Polynomial Time (PP)

Definition (PP)

$L \in$ **PP** if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Leftrightarrow \Pr[A_{M,x}] \geq 3/4.$$

Probabilistic Polynomial Time (PP)

Definition (PP)

$L \in$ **PP** if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Leftrightarrow \Pr\left[A_{M,x}\right] \geq 3/4.$$

- **RP** $\subseteq$ **PP** $\subseteq$ **EXP**

Probabilistic Polynomial Time (PP)

Definition (PP)

$L \in \mathbf{PP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Leftrightarrow \Pr[A_{M,x}] \geq 3/4.$$

- **RP** $\subseteq$ **PP** $\subseteq$ **EXP**
- One can show:
  - May replace $\geq$ by $>$.
  - May replace $3/4$ by $1/2$.
  - **PP** $=$ **coPP**

Probabilistic Polynomial Time (PP)

Definition (PP)

$L \in \mathbf{PP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Leftrightarrow \Pr[A_{M,x}] \geq 3/4.$$

- **RP** $\subseteq$ **PP** $\subseteq$ **EXP**
- One can show:
  - May replace $\geq$ by $>$.
  - May replace $3/4$ by $1/2$.
  - **PP** $=$ **coPP**
- **PP**: "$x \in L$ iff x is accepted by a majority"
  - If $x \notin L$, then x is not accepted by a majority ($\neq$ a majority rejects x!)

Probabilistic Polynomial Time (PP)

Definition (PP)

$L \in \mathbf{PP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0,1\}^*$

$$x \in L \Leftrightarrow \Pr[A_{M,x}] \geq 3/4.$$

- **RP** $\subseteq$ **PP** $\subseteq$ **EXP**
- One can show:
  - May replace $\geq$ by $>$.
  - May replace $3/4$ by $1/2$.
  - **PP** $=$ **coPP**
- **PP**: "$x \in L$ iff x is accepted by a majority"
  - If $x \notin L$, then x is not accepted by a majority ($\neq$ a majority rejects x!)
- Next: **PP** is at least as untractable as **NP**.

NP $\subseteq$ PP

Theorem

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

- Assume TM $M(x, u)$ for $L \in \mathbf{NP}$ uses certificates u of length $p(|x|)$.
- Consider TM $N(x, w)$ with $|w| = p(|x|) + 2$:
  - If $w = 00u$, define $N(x, w) := M(x, u)$.
  - Else $N(x, w) = 1$ iff $w \neq 11\ldots1$.

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

Theorem

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

- Assume TM $M(x, u)$ for $L \in \mathbf{NP}$ uses certificates u of length $p(|x|)$.
- Consider TM $N(x, w)$ with $|w| = p(|x|) + 2$:
  - If $w = 00u$, define $N(x, w) := M(x, u)$.
  - Else $N(x, w) = 1$ iff $w \neq 11 \dots 1$.
- Choose w uniformly on $\{0, 1\}^{p(|x|)+2}$ at random:
  - If $x \in L$: $\Pr[A_{N,x}] \geq$
  - If $x \notin L$: $\Pr[A_{N,x}] =$

NP $\subseteq$ PP

Theorem

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

- Assume TM $M(x, u)$ for $L \in \mathbf{NP}$ uses certificates u of length $p(|x|)$.
- Consider TM $N(x, w)$ with $|w| = p(|x|) + 2$:
  - If $w = 00u$, define $N(x, w) := M(x, u)$.
  - Else $N(x, w) = 1$ iff $w \neq 11 \ldots 1$.
- Choose w uniformly on $\{0, 1\}^{p(|x|)+2}$ at random:
  - If $x \in L$: $\Pr[A_{N,x}] \geq 3/4 - 2^{-p(|x|)-2} + 2^{-p(|x|)-2} = 3/4$
  - If $x \notin L$: $\Pr[A_{N,x}] =$

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

Theorem

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

- Assume TM $M(x, u)$ for $L \in \mathbf{NP}$ uses certificates u of length $p(|x|)$.
- Consider TM $N(x, w)$ with $|w| = p(|x|) + 2$:
  - If $w = 00u$, define $N(x, w) := M(x, u)$.
  - Else $N(x, w) = 1$ iff $w \neq 11\ldots1$.
- Choose w uniformly on $\{0, 1\}^{p(|x|)+2}$ at random:
  - If $x \in L$: $\Pr[A_{N,x}] \geq 3/4 - 2^{-p(|x|)-2} + 2^{-p(|x|)-2} = 3/4$
  - If $x \notin L$: $\Pr[A_{N,x}] = 3/4 - 2^{-p(|x|)-2} < 3/4$

“Bounded probability of error”-P (BPP)

- By the previous result:
    - **PP** does not seem to capture realistic computation.

“Bounded probability of error”-P (BPP)

- By the previous result:
  - **PP** does not seem to capture realistic computation.
- Proof relied on the dependency of the two error bounds:
  - We traded one-sided error probability

    $$x \in L \Rightarrow \Pr[A_{M,x}] \geq 2^{-p(|x|)} \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0$$

    for two-sided error probability

    $$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] < 3/4$$

    by adding enough accept-certificates, i.e.,

“Bounded probability of error”-P (BPP)

- By the previous result:
  - PP does not seem to capture realistic computation.
- Proof relied on the dependency of the two error bounds:
  - We traded one-sided error probability

    $$x \in L \Rightarrow \Pr[A_{M,x}] \geq 2^{-p(|x|)} \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0$$

    for two-sided error probability

    $$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] < 3/4$$

    by adding enough accept-certificates, i.e.,
  - we increased the probability for false positives,
  - while decreasing the probability for false negatives.

“Bounded probability of error”-P (BPP)

- By the previous result:
  - **PP** does not seem to capture realistic computation.
- Proof relied on the dependency of the two error bounds:
  - We traded one-sided error probability

    $$x \in L \Rightarrow \Pr[A_{M,x}] \geq 2^{-p(|x|)} \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] = 0$$

    for two-sided error probability

    $$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[A_{M,x}] < 3/4$$

    by adding enough accept-certificates, i.e.,
  - we increased the probability for false positives,
  - while decreasing the probability for false negatives.
- Possible fix:
  - Require bounds on both error probabilities.
  - “Bounded error probability of error”-**P**

BPP

Definition (BPP)

$L \in \mathbf{BPP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[R_{M,x}] \geq 3/4.$$

- $\mathbf{RP} \subseteq \mathbf{BPP} = \mathbf{coBPP} \subseteq \mathbf{PP}$
  - Reminder: if $L \in \mathbf{PP}$, then $x \notin L \Rightarrow \Pr[A_{M,x}] < 3/4$.

BPP

Definition (BPP)

$L \in$ **BPP** if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[R_{M,x}] \geq 3/4.$$

- **RP** $\subseteq$ **BPP** $=$ **coBPP** $\subseteq$ **PP**
  - Reminder: if $L \in$ **PP**, then $x \notin L \Rightarrow \Pr[A_{M,x}] < 3/4$.
- Two-sided error probabilities:
  - False negatives: If $x \in L$, then $\Pr[R_{M,x}] \leq 1/4$.
  - False positives: If $x \notin L$, then $\Pr[A_{M,x}] \leq 1/4$.
  - Outputs: probably, $x \in L$ and probably, $x \notin L$.
  - Error reduction to 2^{-n} by rerunning (later).

BPP

Definition (BPP)

$L \in \mathbf{BPP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[R_{M,x}] \geq 3/4.$$

- $\mathbf{RP} \subseteq \mathbf{BPP} = \mathbf{coBPP} \subseteq \mathbf{PP}$
  - Reminder: if $L \in \mathbf{PP}$, then $x \notin L \Rightarrow \Pr[A_{M,x}] < 3/4$.
- Two-sided error probabilities:
  - False negatives: If $x \in L$, then $\Pr[R_{M,x}] \leq 1/4$.
  - False positives: If $x \notin L$, then $\Pr[A_{M,x}] \leq 1/4$.
  - Outputs: probably, $x \in L$ and probably, $x \notin L$.
  - Error reduction to 2^{-n} by rerunning (later).
- It is unknown whether $\mathbf{BPP} = \mathbf{NP}$ or even $\mathbf{BPP} = \mathbf{P}$!
  - Under some non-trivial but “very reasonable” assumptions: $\mathbf{BPP} = \mathbf{P}$! (Arora p. 402)

BPP

Definition (BPP)

$L \in \mathbf{BPP}$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM $M(x, u)$ using certificates u of length $|u| = p(|x|)$ such that for every $x \in \{0, 1\}^*$

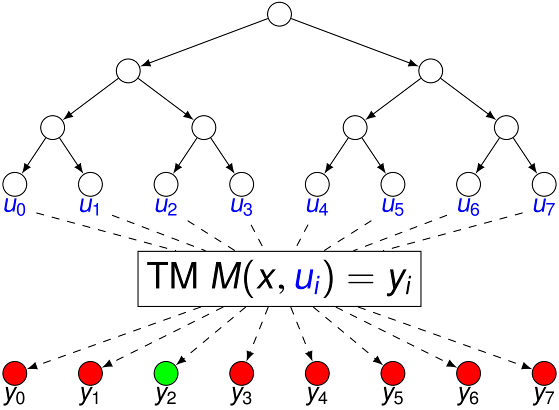
$$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[R_{M,x}] \geq 3/4.$$

- $\mathbf{RP} \subseteq \mathbf{BPP} = \mathbf{coBPP} \subseteq \mathbf{PP}$
  - Reminder: if $L \in \mathbf{PP}$, then $x \notin L \Rightarrow \Pr[A_{M,x}] < 3/4$.
- Two-sided error probabilities:
  - False negatives: If $x \in L$, then $\Pr[R_{M,x}] \leq 1/4$.
  - False positives: If $x \notin L$, then $\Pr[A_{M,x}] \leq 1/4$.
  - Outputs: probably, $x \in L$ and probably, $x \notin L$.
  - Error reduction to 2^{-n} by rerunning (later).
- It is unknown whether $\mathbf{BPP} = \mathbf{NP}$ or even $\mathbf{BPP} = \mathbf{P}$!
  - Under some non-trivial but "very reasonable" assumptions: $\mathbf{BPP} = \mathbf{P}$! (Arora p. 402)
- **BPP** = "most comprehensive, yet plausible notion of realistic computation" (Papadimitriou p. 259)

Agenda

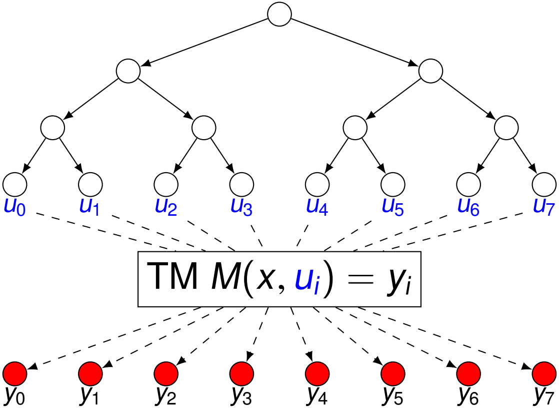
- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP** ✓
- Power of randomization with two-sided error: **PP**, **BPP**
  - Enlarging **RP** by false negatives and false positives ✓
  - Comparison: **NP**, **RP**, **coRP**, **ZPP**, **BPP**, **PP**
  - Probabilistic Turing machines
  - Expected running time
  - Error reduction for **BPP**
  - Some kind of derandomization for **BPP**
  - **BPP** in the polynomial hierarchy

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



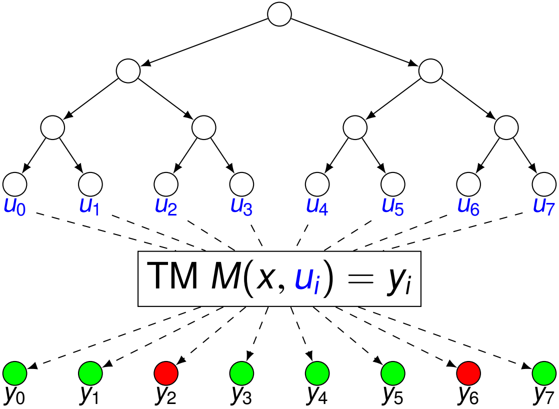
- $L \in$ **NP**:
  - if $x \in L$: at least one 🟢
  - if $x \notin L$: all 🔴

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



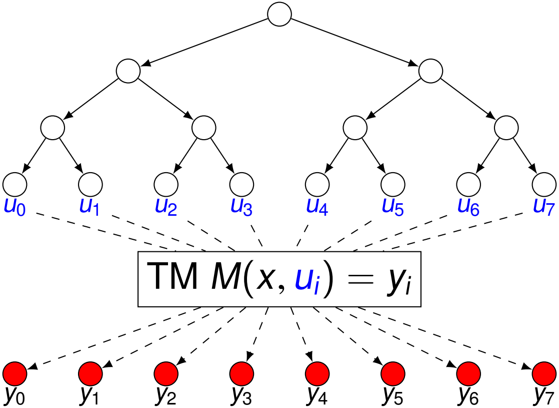
- $L \in$ **NP**:
  - if $x \in L$: at least one 🟢
  - if $x \notin L$: all 🔴

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



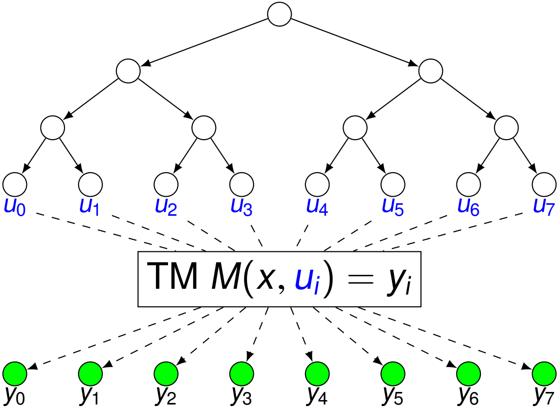
- $L \in$ **RP**:
  - if $x \in L$: at least 75% 🟢
  - if $x \notin L$: all 🔴

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



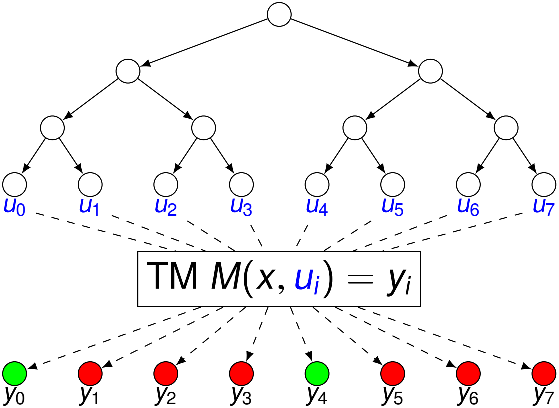
- $L \in$ **RP**:
  - if $x \in L$: at least 75% 🟢
  - if $x \notin L$: all 🔴

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



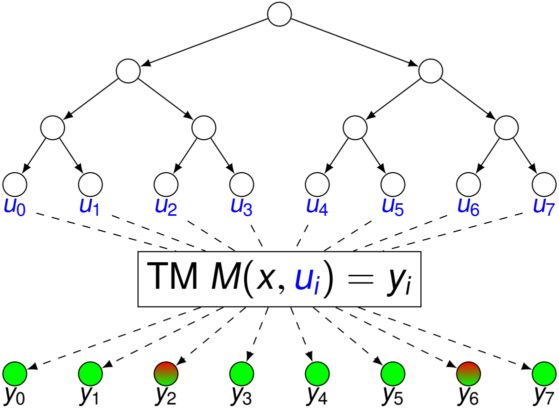
- $L \in$ **coRP**:
  - if $x \in L$: all ● (green)
  - if $x \notin L$: at least 75% ● (red)

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



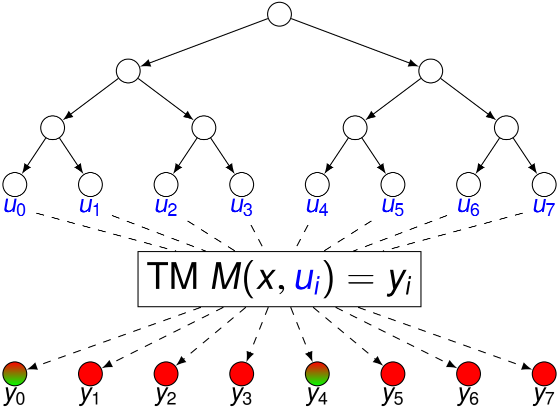
- $L \in$ **coRP**:
  - if $x \in L$: all 🟢
  - if $x \notin L$: at least 75% 🔴

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



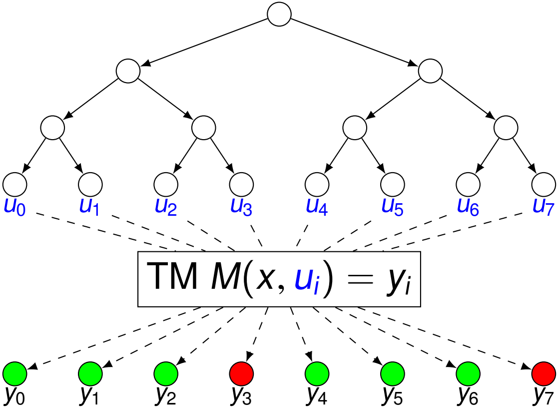
- $L \in$ **ZPP**:
  - if $x \in L$: no ●
  - if $x \notin L$: no ●

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



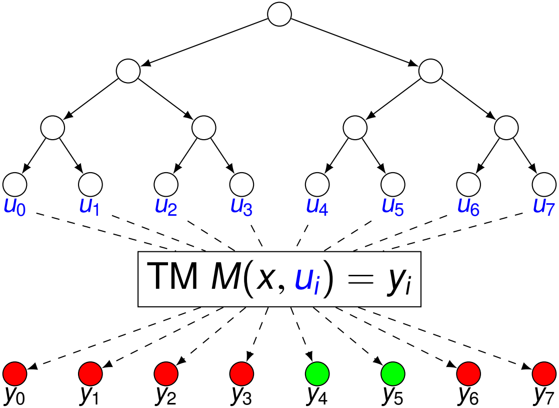
- $L \in$ **ZPP**:
  - if $x \in L$: no 🔴
  - if $x \notin L$: no 🟢

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



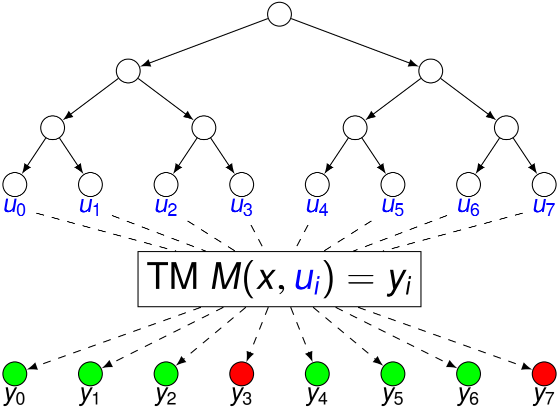
- $L \in$ **BPP**:
  - if $x \in L$: at least 75% 🟢
  - if $x \notin L$: at least 75% 🔴

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



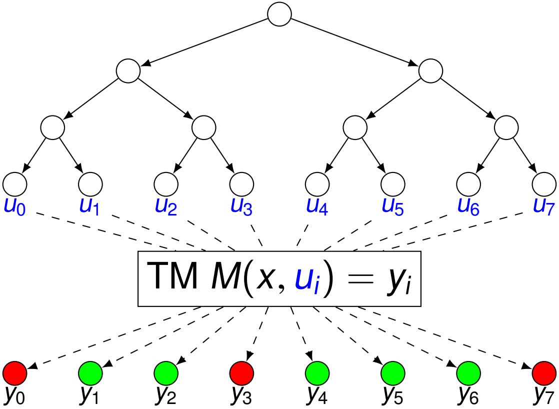
- $L \in$ **BPP**:
  - if $x \in L$: at least 75% 🟢
  - if $x \notin L$: at least 75% 🔴

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



- $L \in$ **PP**:
  - if $x \in L$: at least 75% ●
  - if $x \notin L$: less than 75% ●

NP vs. RP vs. coRP vs. ZPP vs. BPP vs. PP



- $L \in$ **PP**:
  - if $x \in L$: at least 75%
  - if $x \notin L$: less than 75%

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP** ✓
- Power of randomization with two-sided error: **PP**, **BPP**
  - Enlarging **RP** by false negatives and false positives ✓
  - Comparison: **NP**, **RP**, **coRP**, **ZPP**, **BPP**, **PP** ✓
  - Probabilistic Turing machines
  - Expected running time
  - Error reduction for **BPP**
  - Some kind of derandomization for **BPP**
  - **BPP** in the polynomial hierarchy

Probabilistic Turing Machines

Definition (PTM)

We obtain from an NDTM $M = (\Gamma, Q, \delta_1, \delta_2)$ a probabilistic TM (PTM) by choosing in every computation step the transition function uniformly at random, i.e., any given run of M on x of length exactly l occurs with probability 2^{-l}.
A PTM runs in time $T(n)$ if the underlying NDTM runs in time $T(n)$, i.e., if M halts on x within at most $T(|x|)$ steps regardless of the random choices it makes.

Probabilistic Turing Machines

Definition (PTM)

We obtain from an NDTM $M = (\Gamma, Q, \delta_1, \delta_2)$ a probabilistic TM (PTM) by choosing in every computation step the transition function uniformly at random, i.e., any given run of M on x of length exactly l occurs with probability 2^{-l}.
A PTM runs in time $T(n)$ if the underlying NDTM runs in time $T(n)$, i.e., if M halts on x within at most $T(|x|)$ steps regardless of the random choices it makes.

Corollary

*$L \in$ **RP** iff there is a poly-time PTM M s.t. for all $x \in \{0,1\}^*$:*

$$x \in L \Rightarrow Pr[M(x) = 1] \geq 3/4 \text{ and } x \notin L \Rightarrow Pr[M(x) = 1] = 0.$$

Probabilistic Turing Machines

Definition (PTM)

We obtain from an NDTM $M = (\Gamma, Q, \delta_1, \delta_2)$ a probabilistic TM (PTM) by choosing in every computation step the transition function uniformly at random, i.e., any given run of M on x of length exactly l occurs with probability 2^{-l}.
A PTM runs in time $T(n)$ if the underlying NDTM runs in time $T(n)$, i.e., if M halts on x within at most $T(|x|)$ steps regardless of the random choices it makes.

Corollary

*$L \in$ **coRP** iff there is a poly-time PTM M s.t. for all $x \in \{0,1\}^*$:*

$$x \in L \Rightarrow Pr[M(x) = 1] = 1 \textit{ and } x \notin L \Rightarrow Pr[M(x) = 1] \leq 1/4.$$

Probabilistic Turing Machines

Definition (PTM)

We obtain from an NDTM $M = (\Gamma, Q, \delta_1, \delta_2)$ a probabilistic TM (PTM) by choosing in every computation step the transition function uniformly at random, i.e., any given run of M on x of length exactly l occurs with probability 2^{-l}.
A PTM runs in time $T(n)$ if the underlying NDTM runs in time $T(n)$, i.e., if M halts on x within at most $T(|x|)$ steps regardless of the random choices it makes.

Corollary

*$L \in$ **BPP** iff there is a poly-time PTM M s.t. for all $x \in \{0,1\}^*$:*

$$x \in L \Rightarrow Pr[M(x) = 1] \geq 3/4 \textit{ and } x \notin L \Rightarrow Pr[M(x) = 1] \leq 1/4.$$

Probabilistic Turing Machines

Definition (PTM)

We obtain from an NDTM $M = (\Gamma, Q, \delta_1, \delta_2)$ a probabilistic TM (PTM) by choosing in every computation step the transition function uniformly at random, i.e., any given run of M on x of length exactly l occurs with probability 2^{-l}.
A PTM runs in time $T(n)$ if the underlying NDTM runs in time $T(n)$, i.e., if M halts on x within at most $T(|x|)$ steps regardless of the random choices it makes.

Corollary

*$L \in$ **PP** iff there is a poly-time PTM M s.t. for all $x \in \{0,1\}^*$:*

$$x \in L \Rightarrow Pr[M(x) = 1] \geq 3/4 \textit{ and } x \notin L \Rightarrow Pr[M(x) = 1] < 3/4.$$

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP** ✓
- Power of randomization with two-sided error: **PP**, **BPP**
  - Enlarging **RP** by false negatives and false positives ✓
  - Comparison: **NP**, **RP**, **coRP**, **ZPP**, **BPP**, **PP**✓
  - Probabilistic Turing machines✓
  - Expected running time
  - Error reduction for **BPP**
  - Some kind of derandomization for **BPP**
  - **BPP** in the polynomial hierarchy

Expected vs. Exact Running Time

- Recall: if $L \in$ **ZPP**
  - **RP**-algorithms for L and $\overline{L}$.
  - Rerun both algorithms on x until one outputs yes.
  - This decides L in expected polynomial time.
  - But might run infinitely long in the worst case.
- So, is expected time more powerful than exact time?

Expected Running Time

Definition (Expected running time of a PTM)

For a PTM M let $T_{M,x}$ be the random variable that counts the steps of a computation of M on x, i.e., $\Pr[T_{M,x} \leq t]$ is the probability that M halts on x within at most t steps.
We say that M runs in expected time $T(n)$ if $\mathbb{E}[T_{M,x}] \leq T(|x|)$ for every x.

Expected Running Time

Definition (Expected running time of a PTM)

For a PTM M let $T_{M,x}$ be the random variable that counts the steps of a computation of M on x, i.e., $\Pr[T_{M,x} \leq t]$ is the probability that M halts on x within at most t steps.
We say that M runs in expected time $T(n)$ if $\mathbb{E}[T_{M,x}] \leq T(|x|)$ for every x.

- Possibly infinite runs.

Expected Running Time

Definition (Expected running time of a PTM)

For a PTM M let $T_{M,x}$ be the random variable that counts the steps of a computation of M on x, i.e., $\Pr[T_{M,x} \leq t]$ is the probability that M halts on x within at most t steps.
We say that M runs in expected time $T(n)$ if $\mathbb{E}[T_{M,x}] \leq T(|x|)$ for every x.

- Possibly infinite runs.
- So, certificates would need to be unbounded.

Expected Running Time

Definition (Expected running time of a PTM)

For a PTM M let $T_{M,x}$ be the random variable that counts the steps of a computation of M on x, i.e., $\Pr[T_{M,x} \leq t]$ is the probability that M halts on x within at most t steps.
We say that M runs in expected time $T(n)$ if $\mathbb{E}[T_{M,x}] \leq T(|x|)$ for every x.

- Possibly infinite runs.
- So, certificates would need to be unbounded.

Definition (BPeP)

A language L is in **BPeP** if there is a polynomial $T : \mathbb{N} \to \mathbb{N}$ and a PTM M such that for every $x \in \{0,1\}^*$:

$$x \in L \Rightarrow \Pr[M(x) = 1] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[M(x) = 0] \geq 3/4$$

and $\mathbb{E}[T_{M,x}] \leq T(|x|)$.

Expected Running Time

- Assume $L \in$ **BPeP**.
  - PTM M deciding L within expected running time $T(n)$.

Expected Running Time

- Assume $L \in$ **BPeP**.
  - PTM M deciding L within expected running time $T(n)$.
- Probability that M does more than k steps on input x:

$$\Pr\left[T_{M,x} \geq k\right] \leq \frac{\mathbb{E}\left[T_{M,x}\right]}{k} \leq \frac{T(|x|)}{k}$$

by Markov's inequality.

Expected Running Time

- Assume $L \in$ **BPeP**.
  - PTM M deciding L within expected running time $T(n)$.
- Probability that M does more than k steps on input x:

$$\Pr\left[T_{M,x} \geq k\right] \leq \frac{\mathbb{E}\left[T_{M,x}\right]}{k} \leq \frac{T(|x|)}{k}$$

  by Markov's inequality.
- So, for $k = 10T(|x|)$ (polynomial in $|x|$):

$$\Pr\left[T_{M,x} \geq 10T(|x|)\right] \leq 0.1$$

  for every input x.

Expected Running Time

- New algorithm $\tilde{M}$:
  - Simulate M for at most $10T(|x|)$ steps.
  - If simulation termiantes, forward reply of M.
  - Otherwise, choose reply uniformly at random.

Expected Running Time

- New algorithm $\tilde{M}$:
  - Simulate M for at most $10T(|x|)$ steps.
  - If simulation termiantes, forward reply of M.
  - Otherwise, choose reply uniformly at random.
- Runs in (exact) polynomial time.

Expected Running Time

- New algorithm $\tilde{M}$:
  - Simulate M for at most $10T(|x|)$ steps.
  - If simulation termiantes, forward reply of M.
  - Otherwise, choose reply uniformly at random.
- Runs in (exact) polynomial time.
- Error probabilities:
  - Assume $x \in L$.
  - If simulation halts:
  - Otherwise:

Expected Running Time

- New algorithm $\tilde{M}$:
  - Simulate M for at most $10T(|x|)$ steps.
  - If simulation termiantes, forward reply of M.
  - Otherwise, choose reply uniformly at random.
- Runs in (exact) polynomial time.
- Error probabilities:
  - Assume $x \in L$.
  - If simulation halts: $\leq 1/4$
  - Otherwise:

Expected Running Time

- New algorithm $\tilde{M}$:
  - Simulate M for at most $10T(|x|)$ steps.
  - If simulation termiantes, forward reply of M.
  - Otherwise, choose reply uniformly at random.
- Runs in (exact) polynomial time.
- Error probabilities:
  - Assume $x \in L$.
  - If simulation halts: $\leq 1/4$
  - Otherwise: $= 1/2$

Expected Running Time

- New algorithm $\tilde{M}$:
  - Simulate M for at most $10T(|x|)$ steps.
  - If simulation termiantes, forward reply of M.
  - Otherwise, choose reply uniformly at random.
- Runs in (exact) polynomial time.
- Error probabilities:
  - Assume $x \in L$.
  - If simulation halts: $\leq 1/4$
  - Otherwise: $= 1/2$
  - In total: $1/4 \cdot \underbrace{\Pr[T_{M,x} \leq 10T(|x|)]}_{\leq 1} + 1/2 \cdot \underbrace{(1 - \Pr[T_{M,x} \leq 10T(|x|)])}_{\leq 0.1} \leq 0.3$

Expected Running Time

- New algorithm $\tilde{M}$:
  - Simulate M for at most $10T(|x|)$ steps.
  - If simulation termiantes, forward reply of M.
  - Otherwise, choose reply uniformly at random.
- Runs in (exact) polynomial time.
- Error probabilities:
  - Assume $x \in L$.
  - If simulation halts: $\leq 1/4$
  - Otherwise: $= 1/2$
  - In total: $1/4 \cdot \underbrace{\Pr[T_{M,x} \leq 10T(|x|)]}_{\leq 1} + 1/2 \cdot \underbrace{(1 - \Pr[T_{M,x} \leq 10T(|x|)])}_{\leq 0.1} \leq 0.3$

Lemma

$$\mathbf{BPP} = \mathbf{BPeP}$$

Lemma

*$L \in$ **ZPP** iff L is decided by some PTM in expected polynomial time.*

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP** ✓
- Power of randomization with two-sided error: **PP**, **BPP**
  - Enlarging **RP** by false negatives and false positives ✓
  - Comparison: **NP**, **RP**, **coRP**, **ZPP**, **BPP**, **PP**✓
  - Probabilistic Turing machines✓
  - Expected running time✓
  - Error reduction for **BPP**
  - Some kind of derandomization for **BPP**
  - **BPP** in the polynomial hierarchy

Error reduction

- Consider: $L \in$ **RP**:
  - Probability for error after r reruns:
  - if $x \notin L$: $= 0$
  - if $x \in L$: $\leq 4^{-r}$, i.e., r-times probably, $x \notin L$.

Error reduction

- Consider: $L \in$ **RP**:
  - Probability for error after r reruns:
  - if $x \notin L$: $= 0$
  - if $x \in L$: $\leq 4^{-r}$, i.e., r-times probably, $x \notin L$.
- Similarly for $L \in$ **coRP** and $L \in$ **ZPP**.

Error reduction

- Consider: $L \in$ **RP**:
  - Probability for error after r reruns:
  - if $x \notin L$: $= 0$
  - if $x \in L$: $\leq 4^{-r}$, i.e., r-times probably, $x \notin L$.
- Similarly for $L \in$ **coRP** and $L \in$ **ZPP**.
- What if $L \in$ **BPP**?
  - We cannot wait for a yes
  - Instead use the majority.

Error reduction for BPP

Definition (BPP(f))

Let $f : \mathbb{N} \to \mathbb{Q}$ be a function.
$L \in \mathbf{BPP}(f)$ if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM M such that for every $x \in \{0,1\}^*$

$$x \in L \Rightarrow \Pr[A_{M,x}] \geq f(|x|) \text{ and } x \notin L \Rightarrow \Pr[R_{M,x}] \geq f(|x|).$$

Theorem (Error reduction for BPP)

For any $c > 0$:

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

- The longer the input, the less dominant the “majority” has to be.

Error reduction for BPP (Proof)

- Assume $L \in$ **BPP**, and
- Choose any $c > 0$.

Error reduction for BPP (Proof)

- Assume $L \in$ **BPP**, and
- Choose any $c > 0$.
- There exists certainly an n_0 s.t. for all $n \geq n_0$:

$$1/2 + n^{-c} \leq 3/4.$$

Error reduction for BPP (Proof)

- Assume $L \in \mathbf{BPP}$, and
- Choose any $c > 0$.
- There exists certainly an n_0 s.t. for all $n \geq n_0$:

$$1/2 + n^{-c} \leq 3/4.$$

- So: $L \cap \{0,1\}^{\geq n_0} \in \mathbf{BPP}(1/2 + n^{-c})$.

Error reduction for BPP (Proof)

- Assume $L \in \mathbf{BPP}$, and
- Choose any $c > 0$.
- There exists certainly an n_0 s.t. for all $n \geq n_0$:

$$1/2 + n^{-c} \leq 3/4.$$

- So: $L \cap \{0,1\}^{\geq n_0} \in \mathbf{BPP}(1/2 + n^{-c})$.
- Thus, $\mathbf{BPP}(1/2 + n^{-c})$-algorithm for L:
  - If $|x| < n_0$, decide $x \in L$ in $\mathbf{P}$ (error prob. $= 0$)
  - Else run $\mathbf{BPP}$-algorithm (error prob. $\leq 1/4$)

Error reduction for BPP (Proof)

- Let $L \in \textbf{BPP}(1/2 + n^{-c})$ for some $c > 0$.

Error reduction for BPP (Proof)

- Let $L \in \mathbf{BPP}(1/2 + n^{-c})$ for some $c > 0$.
- Run $1/2 + n^{-c}$-algorithm r-times on input x:
  - Outputs: $y = y_1 y_2 y_3 \ldots y_r$
  - with $y_i \in \{0, 1\}$ and $y_i = 1$ if output probably, $x \in L$
  - $Y_1 = \sum_{i=1}^{r} y_i$ number of 1s
  - $Y_0 = r - Y_1$ number of 0s

Error reduction for BPP (Proof)

- Let $L \in \mathbf{BPP}(1/2 + n^{-c})$ for some $c > 0$.
- Run $1/2 + n^{-c}$-algorithm r-times on input x:
  - Outputs: $y = y_1 y_2 y_3 \ldots y_r$
  - with $y_i \in \{0, 1\}$ and $y_i = 1$ if output probably, $x \in L$
  - $Y_1 = \sum_{i=1}^{r} y_i$ number of 1s
  - $Y_0 = r - Y_1$ number of 0s
- Probability of $y_i = 1$ if $x \in L$, resp. $y_i = 0$ if $x \notin L$:

$x \in L \ : \ \Pr[y_i = 1] \geq 1/2 + |x|^{-c}$ resp. $x \notin L \ : \ \Pr[y_i = 0] \geq 1/2 + |x|^{-c}$

(y_i indepedent, Bernoulli distributed RVs)

Error reduction for BPP (Proof)

- Let $L \in \mathbf{BPP}(1/2 + n^{-c})$ for some $c > 0$.
- Run $1/2 + n^{-c}$-algorithm r-times on input x:
  - Outputs: $y = y_1 y_2 y_3 \dots y_r$
  - with $y_i \in \{0, 1\}$ and $y_i = 1$ if output probably, $x \in L$
  - $Y_1 = \sum_{i=1}^{r} y_i$ number of 1s
  - $Y_0 = r - Y_1$ number of 0s
- Probability of $y_i = 1$ if $x \in L$, resp. $y_i = 0$ if $x \notin L$:

  $$x \in L \ : \ \Pr[y_i = 1] \geq 1/2 + |x|^{-c} \text{ resp. } x \notin L \ : \ \Pr[y_i = 0] \geq 1/2 + |x|^{-c}$$

  (y_i indepedent, Bernoulli distributed RVs)
- Expected number of 1s for $x \in L$, resp. of 0s if $x \notin L$:

  $$x \in L \ : \ \mathbb{E}[Y_1] \geq r/2 + r|x|^{-c} \text{ resp. } x \notin L \ : \ \mathbb{E}[Y_0] \geq r/2 + r|x|^{-c}$$

Error reduction for BPP (Proof)

- Let $L \in \mathbf{BPP}(1/2 + n^{-c})$ for some $c > 0$.
- Run $1/2 + n^{-c}$-algorithm r-times on input x:
  - Outputs: $y = y_1 y_2 y_3 \ldots y_r$
  - with $y_i \in \{0, 1\}$ and $y_i = 1$ if output probably, $x \in L$
  - $Y_1 = \sum_{i=1}^{r} y_i$ number of 1s
  - $Y_0 = r - Y_1$ number of 0s
- Probability of $y_i = 1$ if $x \in L$, resp. $y_i = 0$ if $x \notin L$:

  $$x \in L \ : \ \Pr[y_i = 1] \geq 1/2 + |x|^{-c} \text{ resp. } x \notin L \ : \ \Pr[y_i = 0] \geq 1/2 + |x|^{-c}$$

  (y_i indepedent, Bernoulli distributed RVs)
- Expected number of 1s for $x \in L$, resp. of 0s if $x \notin L$:

  $$x \in L \ : \ \mathbb{E}[Y_1] \geq r/2 + r|x|^{-c} \text{ resp. } x \notin L \ : \ \mathbb{E}[Y_0] \geq r/2 + r|x|^{-c}$$

- Assume $r = |x|^{c+d}$ for some $d \in \mathbb{N}$:

  $$x \in L \ : \ \mathbb{E}[Y_1 - Y_0] \geq 2|x|^{d} \text{ resp. } x \notin L \ : \ \mathbb{E}[Y_0 - Y_1] \geq 2|x|^{d}$$

  i.e., expect significant majority in favor of correct answer.

Error reduction for BPP (Proof)

- Idea: let majority decide, i.e., output $x \in L$ iff $Y_1 > Y_0$.

Error reduction for BPP (Proof)

- Idea: let majority decide, i.e., output $x \in L$ iff $Y_1 > Y_0$.
- Assume $x \in L$ in the following
  - Case $x \notin L$ symmetric:
  - set $z_i := 1 - y_i$ and consider Y_0 instead of Y_1

Error reduction for BPP (Proof)

- Idea: let majority decide, i.e., output $x \in L$ iff $Y_1 > Y_0$.
- Assume $x \in L$ in the following
  - Case $x \notin L$ symmetric:
  - set $z_i := 1 - y_i$ and consider Y_0 instead of Y_1
- Probability that "majority" wrongly says $x \notin L$:

$$\Pr\left[Y_1 \leq Y_0\right] = \Pr\left[Y_1 \leq r/2\right]$$

Error reduction for BPP (Proof)

- Idea: let majority decide, i.e., output $x \in L$ iff $Y_1 > Y_0$.
- Assume $x \in L$ in the following
  - Case $x \notin L$ symmetric:
  - set $z_i := 1 - y_i$ and consider Y_0 instead of Y_1
- Probability that "majority" wrongly says $x \notin L$:

$$\Pr[Y_1 \leq Y_0] = \Pr[Y_1 \leq r/2]$$

- Chernoff bound: for $X \sim \mathrm{Bin}(n; p)$ with $\mu := \mathbb{E}[X]$ and $\delta \in (0, 1)$

$$\Pr[X \leq (1 - \delta)\mu] \leq e^{-\mu\delta^2/2}$$

Error reduction for BPP (Proof)

- Idea: let majority decide, i.e., output $x \in L$ iff $Y_1 > Y_0$.
- Assume $x \in L$ in the following
  - Case $x \notin L$ symmetric:
  - set $z_i := 1 - y_i$ and consider Y_0 instead of Y_1
- Probability that "majority" wrongly says $x \notin L$:

$$\Pr[Y_1 \le Y_0] = \Pr[Y_1 \le r/2]$$

- Chernoff bound: for $X \sim \mathrm{Bin}(n; p)$ with $\mu := \mathbb{E}[X]$ and $\delta \in (0,1)$

$$\Pr[X \le (1-\delta)\mu] \le e^{-\mu\delta^2/2}$$

- Thus:

$$\Pr[Y_1 \le r/2] = \Pr[Y_1 \le (1-(1-r/(2\mu)))\mu] \le e^{-\mu\delta^2/2}$$

as long as $\delta := 1 - r/(2\mu) \in (0,1)$.

Error reduction for BPP (Proof)

- Bounds on $\delta = 1 - r/(2\mu)$:

$$0 < \delta < 1 \Leftrightarrow 0 < r/2 < \mu \Leftarrow r/2 + r|x|^{-c} \leq \mu$$

- Thus, choose r s.t.

$$\Pr\left[Y_1 \leq r/2\right] \leq e^{-\mu\delta^2/2} \leq 1/4.$$

i.e.,

$$\mu\delta^2 \geq 2 \log_e 4.$$

- With

$$\mu \geq r/2 + r|x|^{-c}$$

we obtain:

$$\mu\delta^2 = (\mu - r/2)(1-(r/2)/\mu) \geq r|x|^{-c}\left(1 - \frac{r/2}{r/2 + r|x|^{-c}}\right) = r \cdot \frac{|x|^{-2c}}{1/2 + |x|^{-c}}$$

- So, choose $r \geq (\log_e 4) \cdot (|x|^{2c} + 2|x|^c)$.

Error reduction for BPP (Proof)

- For $x \notin L$ we obtain analogously:

$$\Pr[Y_0 \leq Y_1] \leq 1/4 \text{ if } r \geq (\log_e 4) \cdot (|x|^{2c} + 2|x|^c).$$

- So, a polynomial number of rounds suffices to reduce error probability to at most $1/4$.
- Proof also yields:

Theorem (Error reduction for BPP)

For any $d > 0$:

$$\mathbf{BPP} = \mathbf{BPP}(1 - 2^{-n^d})$$

Error reduction for BPP (Proof)

- For $x \notin L$ we obtain analogously:

$$\Pr[Y_0 \leq Y_1] \leq 1/4 \text{ if } r \geq (\log_e 4) \cdot (|x|^{2c} + 2|x|^c).$$

- So, a polynomial number of rounds suffices to reduce error probability to at most $1/4$.
- Proof also yields:

Theorem (Error reduction for BPP)

For any $d > 0$:

$$\mathbf{BPP} = \mathbf{BPP}(1 - 2^{-n^d})$$

- Ex.: Show the theorem.

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP** ✓
- Power of randomization with two-sided error: **PP**, **BPP**
  - Enlarging **RP** by false negatives and false positives ✓
  - Comparison: **NP**, **RP**, **coRP**, **ZPP**, **BPP**, **PP**✓
  - Probabilistic Turing machines✓
  - Expected running time✓
  - Error reduction for **BPP**✓
  - Some kind of derandomization for **BPP**
  - **BPP** in the polynomial hierarchy

Some Kind of Derandomization

Theorem

*Let $L \in$ **BPP** be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.*
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \textit{ iff } M(x, u_n) = 1.$$

Some Kind of Derandomization

Theorem

*Let $L \in$ **BPP** be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.*
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: **BPP** $=$ **BPP**$(1 - 4^{-n})$
- For a given n let choose $u \in \{0, 1\}^{p(n)}$ uniformly at random.

Some Kind of Derandomization

Theorem

Let $L \in \mathbf{BPP}$ be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: $\mathbf{BPP} = \mathbf{BPP}(1 - 4^{-n})$
- For a given n let choose $u \in \{0,1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0,1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

Some Kind of Derandomization

Theorem

Let $L \in \mathbf{BPP}$ be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: $\mathbf{BPP} = \mathbf{BPP}(1 - 4^{-n})$
- For a given n let choose $u \in \{0,1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0,1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

- $\Pr[B_x] \leq$

Some Kind of Derandomization

Theorem

*Let $L \in$ **BPP** be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.*
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: **BPP** $=$ **BPP**$(1 - 4^{-n})$
- For a given n let choose $u \in \{0,1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0,1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

- $\Pr[B_x] \leq 4^{-n}$

Some Kind of Derandomization

Theorem

*Let $L \in$ **BPP** be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.*
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: **BPP** $=$ **BPP**$(1 - 4^{-n})$
- For a given n let choose $u \in \{0, 1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0, 1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

- $\Pr[B_x] \leq 4^{-n}$
- $\Pr\left[\bigcup_{|x|=n} B_x\right] \leq$

Some Kind of Derandomization

Theorem

*Let $L \in$ **BPP** be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.*
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: **BPP** = **BPP**$(1 - 4^{-n})$
- For a given n let choose $u \in \{0, 1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0, 1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

- $\Pr[B_x] \leq 4^{-n}$
- $\Pr\left[\bigcup_{|x|=n} B_x\right] \leq \sum_{|x|=n} \Pr[B_x] \leq 2^n \cdot 4^{-n} = 2^{-n}$

Some Kind of Derandomization

Theorem

*Let $L \in$ **BPP** be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.*
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: **BPP** $=$ **BPP**$(1 - 4^{-n})$
- For a given n let choose $u \in \{0,1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0,1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

- $\Pr[B_x] \leq 4^{-n}$
- $\Pr\left[\bigcup_{|x|=n} B_x\right] \leq \sum_{|x|=n} \Pr[B_x] \leq 2^n \cdot 4^{-n} = 2^{-n}$
- $\Pr\left[\bigcap_{|x|=n} \overline{B}_x\right] \geq$

Some Kind of Derandomization

Theorem

*Let $L \in$ **BPP** be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.*
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: **BPP** $=$ **BPP**$(1 - 4^{-n})$
- For a given n let choose $u \in \{0, 1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0, 1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

- $\Pr[B_x] \leq 4^{-n}$
- $\Pr\left[\bigcup_{|x|=n} B_x\right] \leq \sum_{|x|=n} \Pr[B_x] \leq 2^n \cdot 4^{-n} = 2^{-n}$
- $\Pr\left[\bigcap_{|x|=n} \overline{B}_x\right] \geq 1 - 2^{-n} > 0$

Some Kind of Derandomization

Theorem

Let $L \in \mathbf{BPP}$ be decided by a poly-time TM $M(x, u)$ using certificates of poly-length $p(n)$.
Then for every $n \in \mathbb{N}$ there exists a certificate u_n s.t. for all x with $|x| = n$:

$$x \in L \text{ iff } M(x, u_n) = 1.$$

- Error reduction: $\mathbf{BPP} = \mathbf{BPP}(1 - 4^{-n})$
- For a given n let choose $u \in \{0, 1\}^{p(n)}$ uniformly at random.
- Let B_x be the event of bad certificates for x:

$$B_x := \{u \in \{0, 1\}^{p(|x|)} \mid x \in L \Leftrightarrow M(x, u) = 0\}.$$

- $\Pr[B_x] \leq 4^{-n}$
- $\Pr\left[\bigcup_{|x|=n} B_x\right] \leq \sum_{|x|=n} \Pr[B_x] \leq 2^n \cdot 4^{-n} = 2^{-n}$
- $\Pr\left[\bigcap_{|x|=n} \overline{B}_x\right] \geq 1 - 2^{-n} > 0$
- Seems unlikely for **NP**.

Agenda

- Motivation: From **NP** to a more realistic class by randomization ✓
- Randomized poly-time with one-sided error: **RP**, **coRP**, **ZPP** ✓
- Power of randomization with two-sided error: **PP**, **BPP**
  - Enlarging **RP** by false negatives and false positives ✓
  - Comparison: **NP**, **RP**, **coRP**, **ZPP**, **BPP**, **PP**✓
  - Probabilistic Turing machines✓
  - Expected running time✓
  - Error reduction for **BPP**✓
  - Some kind of derandomization for **BPP**✓
  - **BPP** in the polynomial hierarchy

BPP in the Polynomial Hierarchy PH

Theorem

$$\mathbf{BPP} \subseteq \mathbf{\Sigma_2^p} \cap \mathbf{\Pi_2^p}$$

- Reminder:
  - Definition of $L \in \mathbf{\Sigma_2^p}$:

$$x \in L \text{ iff } \exists u \in \{0,1\}^{p(|x|)} \forall v \in \{0,1\}^{p(|x|)} : M(x,u,v) = 1.$$

  - Definition of $L \in \mathbf{\Pi_2^p}$:

$$x \in L \text{ iff } \forall u \in \{0,1\}^{p(|x|)} \exists v \in \{0,1\}^{p(|x|)} : M(x,u,v) = 1.$$

BPP in the Polynomial Hierarchy PH

Theorem

$$\mathbf{BPP} \subseteq \mathbf{\Sigma_2^p} \cap \mathbf{\Pi_2^p}$$

- Reminder:
  - Definition of $L \in \mathbf{\Sigma_2^p}$:

$$x \in L \text{ iff } \exists u \in \{0,1\}^{p(|x|)} \forall v \in \{0,1\}^{p(|x|)} : M(x,u,v) = 1.$$

  - Definition of $L \in \mathbf{\Pi_2^p}$:

$$x \in L \text{ iff } \forall u \in \{0,1\}^{p(|x|)} \exists v \in \{0,1\}^{p(|x|)} : M(x,u,v) = 1.$$

- As $\mathbf{BPP} = \mathbf{coBPP}$ it suffices to show $\mathbf{BPP} \subseteq \mathbf{\Sigma_2^p}$:

$$L \in \mathbf{BPP} \Rightarrow \overline{L} \in \mathbf{BPP} \Rightarrow \overline{L} \in \mathbf{\Sigma_2^p} \Rightarrow L \in \mathbf{\Pi_2^p}$$

BPP in the Polynomial Hierarchy PH

- We use again that $\mathbf{BPP} = \mathbf{BPP}(1 - 4^{-n})$.
- Let $p(\cdot)$ be the polynomial bounding the certificate length.
- Recall $A_{M,x}$: "accept-certificates"

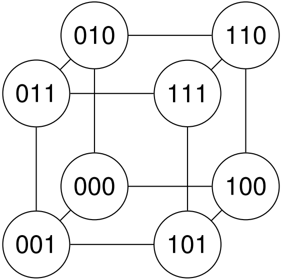
$$A_{M,x} := \{u \in \{0,1\}^{p(|x|)} \mid M(x,u) = 1\}$$

- Then

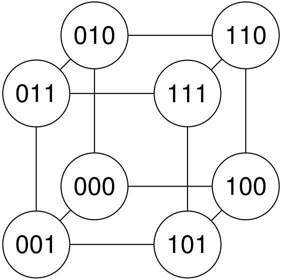
$$x \in L \Rightarrow |A_{M,x}| \geq (1 - 4^{-|x|})2^{p(|x|)} \text{ and } x \notin L \Rightarrow |A_{M,x}| \leq 4^{-n} \cdot 2^{p(|x|)}$$

- Need a formula to distinguish the two cases.

BPP in the Polynomial Hierarchy PH

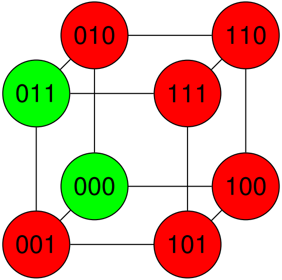


BPP in the Polynomial Hierarchy PH



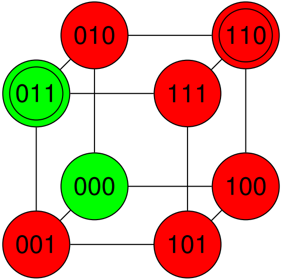
- Assume $|x| = 1$ and $p(|x|) = 3$,
- i.e., possible certificates in $\{0, 1\}^3$.
- If $x \in L$, then $|A_{M,x}| \geq 3/4 \cdot 2^3 = 6$.
- If $x \notin L$, then $|A_{M,x}| \leq 1/4 \cdot 2^3 = 2$.

BPP in the Polynomial Hierarchy PH



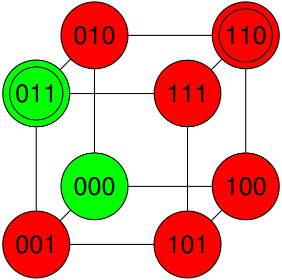
- Assume $x \notin L$, i.e., $|A_{M,x}| \leq 1/4 \cdot 8 = 2$

BPP in the Polynomial Hierarchy PH



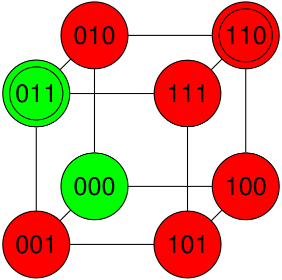
- Assume $x \notin L$, i.e., $|A_{M,x}| \leq 1/4 \cdot 8 = 2$
- Choose any $u_1, u_2 \in \{0,1\}^3$.

BPP in the Polynomial Hierarchy PH



- Assume $x \notin L$, i.e., $|A_{M,x}| \leq 1/4 \cdot 8 = 2$
- Choose any $u_1, u_2 \in \{0,1\}^3$.
- By chance, we might hit $A_{M,x}$.

BPP in the Polynomial Hierarchy PH

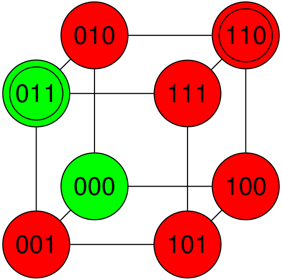


- Assume $x \notin L$, i.e., $|A_{M,x}| \leq 1/4 \cdot 8 = 2$
- Choose any $u_1, u_2 \in \{0,1\}^3$.
- By chance, we might hit $A_{M,x}$.
- Claim: But there is some $r \in \{0,1\}^3$ s.t.

$$\{u_1 \oplus r, u_2 \oplus r\} \cap A_{M,x} = \emptyset.$$

($\oplus$: bitwise xor)

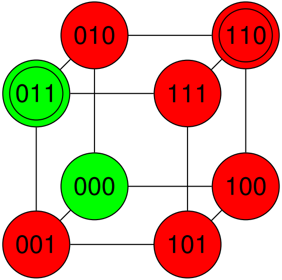
BPP in the Polynomial Hierarchy PH



- Note:

$$u_i \oplus r \in A_{M,x} \text{ iff } r \in A_{M,x} \oplus u_i.$$

BPP in the Polynomial Hierarchy PH



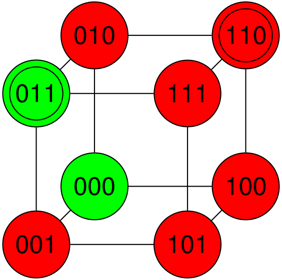
- Note:

$$u_i \oplus r \in A_{M,x} \text{ iff } r \in A_{M,x} \oplus u_i.$$

- So, choose

$$r \in \overline{A_{M,x} \oplus u_1 \cup A_{M,x} \oplus u_2} = \overline{\{000, 011\} \cup \{101, 110\}}.$$

BPP in the Polynomial Hierarchy PH



- Note:

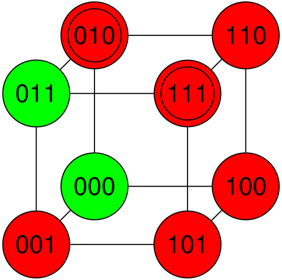
$$u_i \oplus r \in A_{M,x} \text{ iff } r \in A_{M,x} \oplus u_i.$$

- So, choose

$$r \in \overline{A_{M,x} \oplus u_1 \cup A_{M,x} \oplus u_2} = \overline{\{000, 011\} \cup \{101, 110\}}.$$

- E.g. $r = 001$.

BPP in the Polynomial Hierarchy PH



- Note:

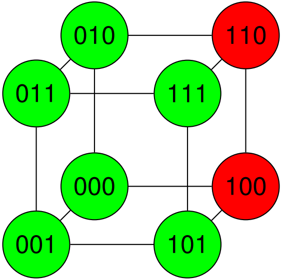
$$u_i \oplus r \in A_{M,x} \text{ iff } r \in A_{M,x} \oplus u_i.$$

- So, choose

$$r \in \overline{A_{M,x} \oplus u_1 \cup A_{M,x} \oplus u_2} = \overline{\{000, 011\} \cup \{101, 110\}}.$$

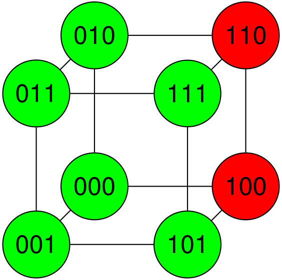
- E.g. $r = 001$.

BPP in the Polynomial Hierarchy PH



- Assume $x \in L$, i.e., $|A_{M,x}| \geq 6$.

BPP in the Polynomial Hierarchy PH

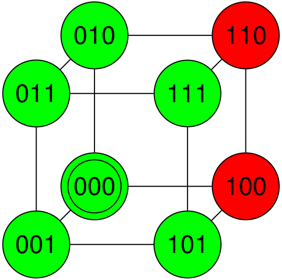


- Assume $x \in L$, i.e., $|A_{M,x}| \geq 6$.
- Claim: We can choose u_1, u_2 s.t. for any $r \in \{0,1\}^3$

$$\{u_1 \oplus r, u_2 \oplus r\} \cap A_{M,x} \neq \emptyset.$$

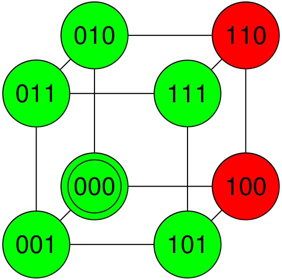
- Note: this is exactly the negation of the previous claim.

BPP in the Polynomial Hierarchy PH



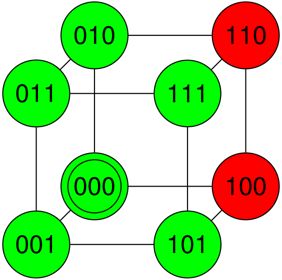
- E.g., take $u_1 = 000$.

BPP in the Polynomial Hierarchy PH



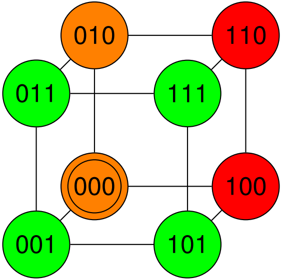
- E.g., take $u_1 = 000$.
- Then $u_1 \oplus r \in R_{M,x}$ iff $r \in u_1 \oplus R_{M,x} = \{100, 110\}$.

BPP in the Polynomial Hierarchy PH



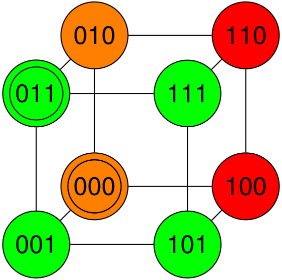
- E.g., take $u_1 = 000$.
- Then $u_1 \oplus r \in R_{M,x}$ iff $r \in u_1 \oplus R_{M,x} = \{100, 110\}$.
- So, take $u_2 \notin 100 \oplus R_{M,x} \cup 110 \oplus R_{M,x}$.

BPP in the Polynomial Hierarchy PH



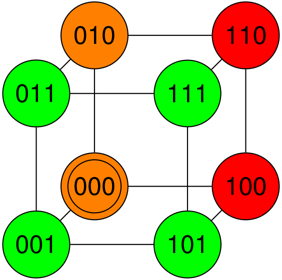
- E.g., take $u_1 = 000$.
- Then $u_1 \oplus r \in R_{M,x}$ iff $r \in u_1 \oplus R_{M,x} = \{100, 110\}$.
- So, take $u_2 \notin 100 \oplus R_{M,x} \cup 110 \oplus R_{M,x}$.

BPP in the Polynomial Hierarchy PH



- E.g., take $u_1 = 000$.
- Then $u_1 \oplus r \in R_{M,x}$ iff $r \in u_1 \oplus R_{M,x} = \{100, 110\}$.
- So, take $u_2 \notin 100 \oplus R_{M,x} \cup 110 \oplus R_{M,x}$.
- E.g., $u_2 = 011$.

BPP in the Polynomial Hierarchy PH

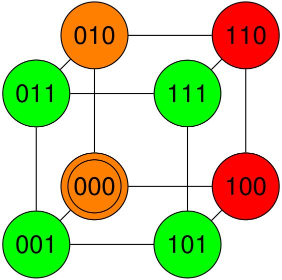


- Summary:

$$x \in L \cap \{0,1\}^1 \text{ iff } \exists u_1, u_2 \in \{0,1\}^3 \forall r \in \{0,1\}^3 : \bigvee_{i=1,2} u_i \oplus r \in A_{M,x}.$$

Reminder: $u_i \oplus r \in A_{M,x}$ iff $M(x, u_i \oplus r) = 1$.

BPP in the Polynomial Hierarchy PH



- Summary:

$$x \in L \cap \{0,1\}^1 \text{ iff } \exists u_1, u_2 \in \{0,1\}^3 \forall r \in \{0,1\}^3 \; : \; \bigvee_{i=1,2} u_i \oplus r \in A_{M,x}.$$

  Reminder: $u_i \oplus r \in A_{M,x}$ iff $M(x, u_i \oplus r) = 1$.

- So, this is in $\mathbf{\Sigma_2^p}$.
- And works also for $|x| > 1$ and arbitrary $p(|x|)$.

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Note, the certificate $u_1 u_2 \ldots u_k$ has length polynomial in $|x|$.
- So, this formula represents a computation in $\mathbf{\Sigma_2^p}$.

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} \ : \ \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \notin L$: To show there is always an r s.t.

$$\bigwedge_{i=1}^{k} r \oplus u_i \notin A_{M,x} \equiv r \notin \bigcup_{i=1}^{k} u_i \oplus A_{M,x}.$$

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \notin L$: To show there is always an r s.t.

$$\bigwedge_{i=1}^{k} r \oplus u_i \notin A_{M,x} \equiv r \notin \bigcup_{i=1}^{k} u_i \oplus A_{M,x}.$$

- Size of the complement of this set:

$$\left| \bigcup_{i=1}^{k} u_i \oplus A_{M,x} \right| \leq \sum_{i=1}^{k} \left| u_i \oplus A_{M,x} \right| = k \left| A_{M,x} \right| \leq k 4^{-|x|} 2^{p(|x|)} < 2^{p(|x|)}.$$

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \notin L$: To show there is always an r s.t.

$$\bigwedge_{i=1}^{k} r \oplus u_i \notin A_{M,x} \equiv r \notin \bigcup_{i=1}^{k} u_i \oplus A_{M,x}.$$

- Size of the complement of this set:

$$\left| \bigcup_{i=1}^{k} u_i \oplus A_{M,x} \right| \leq \sum_{i=1}^{k} \left| u_i \oplus A_{M,x} \right| = k \left| A_{M,x} \right| \leq k 4^{-|x|} 2^{p(|x|)} < 2^{p(|x|)}.$$

- So, this set cannot be empty no matter how we choose $u_1, \ldots, u_k$.

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \in L$: To show there are $u_1, \ldots, u_k$ s.t.

$$\forall r : \bigvee_{i=1}^{k} u_i \oplus r \in A_{M,x} \equiv \neg \exists r : \bigwedge_{i=1}^{k} u_i \in r \oplus R_{M,x}.$$

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \in L$: To show there are $u_1, \ldots, u_k$ s.t.

$$\forall r : \bigvee_{i=1}^{k} u_i \oplus r \in A_{M,x} \equiv \neg \exists r : \bigwedge_{i=1}^{k} u_i \in r \oplus R_{M,x}.$$

- Let $U_1, \ldots, U_k$ be independent random variables, uniformly distributed on $\{0,1\}^{p(|x|)}$. (Doesn't matter if some coincide!)

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \in L$: To show there are $u_1, \ldots, u_k$ s.t.

$$\forall r : \bigvee_{i=1}^{k} u_i \oplus r \in A_{M,x} \equiv \neg \exists r : \bigwedge_{i=1}^{k} u_i \in r \oplus R_{M,x}.$$

- Let $U_1, \ldots, U_k$ be independent random variables, uniformly distributed on $\{0,1\}^{p(|x|)}$. (Doesn't matter if some coincide!)
- For any given r: $\Pr\left[U_i \in r \oplus R_{M,x}\right] = \frac{|r \oplus R_{M,x}|}{2^{p(|x|)}} = \frac{|R_{M,x}|}{2^{p(|x|)}} \leq 4^{-n}$.

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} \; : \; \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \in L$: To show there are $u_1, \ldots, u_k$ s.t.

$$\forall r \; : \; \bigvee_{i=1}^{k} u_i \oplus r \in A_{M,x} \equiv \neg \exists r \; : \; \bigwedge_{i=1}^{k} u_i \in r \oplus R_{M,x}.$$

- Let $U_1, \ldots, U_k$ be independent random variables, uniformly distributed on $\{0,1\}^{p(|x|)}$. (Doesn't matter if some coincide!)
- For any given r: $\Pr[U_i \in r \oplus R_{M,x}] = \frac{|r \oplus R_{M,x}|}{2^{p(|x|)}} = \frac{|R_{M,x}|}{2^{p(|x|)}} \leq 4^{-n}$.
- $\Pr\left[\bigwedge_{i=1}^{k} U_i \in r \oplus R_{M,x}\right] = \Pr[U_1 \in r \oplus R_{M,x}]^k \leq 4^{-kn}$.

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- Assume $x \in L$: To show there are $u_1, \ldots, u_k$ s.t.

$$\forall r : \bigvee_{i=1}^{k} u_i \oplus r \in A_{M,x} \equiv \neg \exists r : \bigwedge_{i=1}^{k} u_i \in r \oplus R_{M,x}.$$

- Let $U_1, \ldots, U_k$ be independent random variables, uniformly distributed on $\{0,1\}^{p(|x|)}$. (Doesn't matter if some coincide!)
- For any given r: $\Pr\left[U_i \in r \oplus R_{M,x}\right] = \frac{|r \oplus R_{M,x}|}{2^{p(|x|)}} = \frac{|R_{M,x}|}{2^{p(|x|)}} \leq 4^{-n}$.
- $\Pr\left[\bigwedge_{i=1}^{k} U_i \in r \oplus R_{M,x}\right] = \Pr\left[U_1 \in r \oplus R_{M,x}\right]^k \leq 4^{-kn}$.
- $\Pr\left[\exists r : \bigwedge_{i=1}^{k} U_i \in r \oplus R_{M,x}\right] \leq \sum_{r \in \{0,1\}^*} 4^{-kn} = 2^{p(|x|)-4kn} < 1$.

BPP in the Polynomial Hierarchy PH

Claim:

Given x set $k := \lceil p(|x|)/|x| \rceil + 1$. Then:

$$x \in L \text{ iff } \exists u_1, \ldots, u_k \in \{0,1\}^{p(|x|)} \forall r \in \{0,1\}^{p(|x|)} : \bigvee_{i=1}^{k} M(x, u_i \oplus r) = 1.$$

- For both cases there is an n_0 s.t. the bounds hold for all x with $|x| > n_0$.
- $L \cap \{0,1\}^{n_0}$ can be decided trivially in **P**.

Summary

- Obtain **RP** from **NP** by
  - choosing the certificate (transition function) uniformaly at random
  - requiring a bound on $\Pr[A_{M,x}]$ if $x \in L$ s.t.
  - error prob. can be reduced within a polynomial number of reruns.

Summary

- Obtain **RP** from **NP** by
  - choosing the certificate (transition function) uniformaly at random
  - requiring a bound on $\Pr[A_{M,x}]$ if $x \in L$ s.t.
  - error prob. can be reduced within a polynomial number of reruns.
- One-sided probability of error:
  - **RP**: false negatives
  - **coRP**: false postives
  - Monte Carlo algorithms: ZEROP $\in$ **coRP**, perfect matchings $\in$ **RP**

Summary

- Obtain **RP** from **NP** by
  - choosing the certificate (transition function) uniformaly at random
  - requiring a bound on $\Pr[A_{M,x}]$ if $x \in L$ s.t.
  - error prob. can be reduced within a polynomial number of reruns.
- One-sided probability of error:
  - **RP**: false negatives
  - **coRP**: false postives
  - Monte Carlo algorithms: ZEROP $\in$ **coRP**, perfect matchings $\in$ **RP**
- **ZPP** $:=$ **RP** $\cap$ **coRP** can be decided in expected polynomial time
  - Zero probability of error (if we wait for the definitiv answer)
  - Las Vegas algorithms

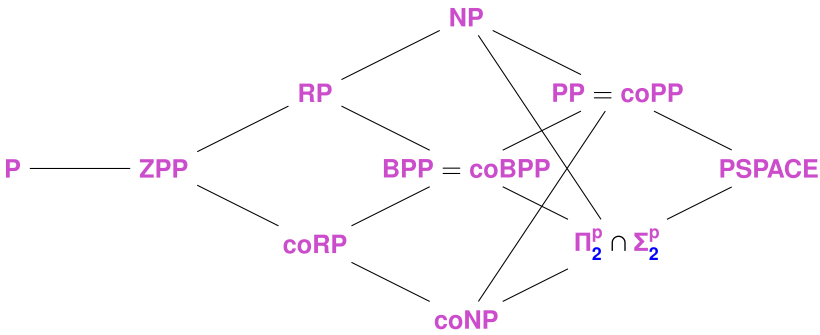
Summary

- Obtained **PP** from **RP** by
  - allowing also for false positives
  - Error probabilities depend on each other: $\leq 1/4$ and $< 1 - 1/4$
  - **NP** $\subseteq$ **PP**: "**PP** allows for trading one error prob. for the other"

Summary

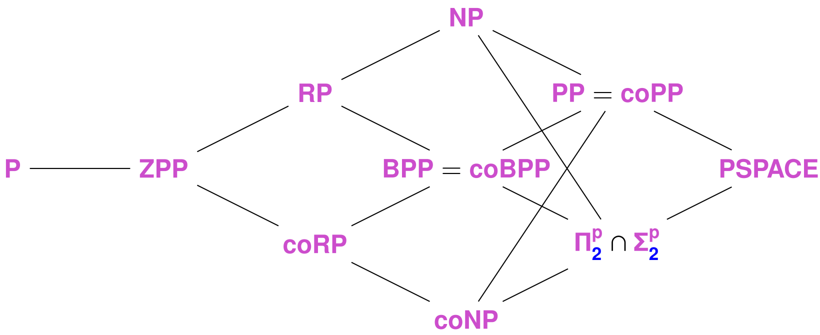
- Obtained **PP** from **RP** by
  - allowing also for false positives
  - Error probabilities depend on each other: $\leq 1/4$ and $< 1 - 1/4$
  - **NP** $\subseteq$ **PP**: "**PP** allows for trading one error prob. for the other"
- Obtained **BPP** from **PP** by
  - bounding both error prob. independently of each other.
  - Papadimitriou: "most comprehensive, yet plausible notion of realistic computation"
  - Conjecture: **BPP** $=$ **P**
  - Expected running time as powerful as exact running time.
  - One certificate u_n for all x with $|x| = n$.
  - Error reduction to 2^{-n^k} within a polynomial number of reruns.

Summary



- $\Pi_2^p \cap \Sigma_2^p \subseteq$ **PP** unknown.
- **NP** $\cup$ **coNP** $\subseteq$ **PP** known.

Summary



- Gödel Price (1998) for Toda's theorem (1989): $\mathbf{PH} \subseteq \mathbf{P}^{\mathbf{PP}}$
  - $\mathbf{P}^{\mathbf{PP}}$: poly-time TMs having access to a **PP**-oracle.
  - If $\mathbf{PP} \subseteq \Sigma_k^p$ for some k, then $\mathbf{PH} = \Sigma_k^p$.
  - If $\mathbf{PP} \subseteq \mathbf{PH}$, then **PH** collapses at some finite level as **PP** has complete problems (see exercises).

Syntactic and Semantic Complexity Classes

- Just mentioned: **PP** has complete probems
  - $\phi \in$ MAJSAT iff at least $2^{n-1} + 1$ satisfying assignments of 2^n possible (see exercises).
- Unknown if there are complete problems for **ZPP**, **RP**, **BPP**.

Syntactic and Semantic Complexity Classes

- Just mentioned: **PP** has complete probems
  - $\phi \in$ MAJSAT iff at least $2^{n-1} + 1$ satisfying assignments of 2^n possible (see exercises).
- Unknown if there are complete problems for **ZPP**, **RP**, **BPP**.
- Reason to believe that there are none:
  - **P**, **NP**, **coNP** are syntatic complexity classes (complete problems).
  - **ZPP**, **RP**, **coRP**, **BPP** are semantic complexity classes.
- Example:
  - **NP**:
    $$x \in L \Leftrightarrow \Pr\left[A_{M,x}\right] > 0.$$
    Every poly-time TM $M(x, u)$ defines a language in **NP**.
  - **BPP**:
    $$x \in L \Rightarrow \Pr\left[A_{M,x}\right] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr\left[R_{M,x}\right] \geq 3/4.$$
    Not every poly-time TM $M(x, u)$ defines a language in **BPP**.

Syntactic and Semantic Complexity Classes

- Just mentioned: **PP** has complete probems
  - $\phi \in$ MAJSAT iff at least $2^{n-1}+1$ satisfying assignments of 2^n possible (see exercises).
- Unknown if there are complete problems for **ZPP**, **RP**, **BPP**.
- Reason to believe that there are none:
  - **P**, **NP**, **coNP** are syntatic complexity classes (complete problems).
  - **ZPP**, **RP**, **coRP**, **BPP** are semantic complexity classes.
- Example:
  - **NP**:

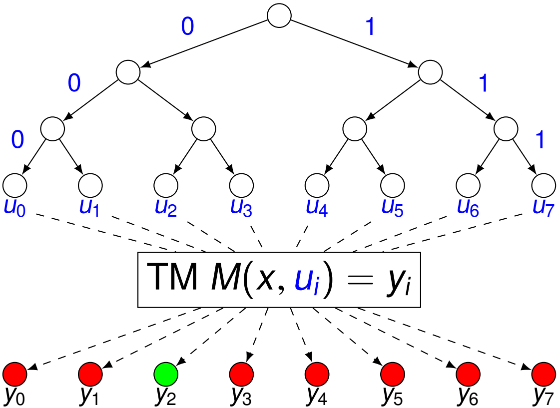
    $$x \in L \Leftrightarrow \Pr[A_{M,x}] > 0.$$

    Every poly-time TM $M(x,u)$ defines a language in **NP**.
  - **BPP**:

    $$x \in L \Rightarrow \Pr[A_{M,x}] \geq 3/4 \text{ and } x \notin L \Rightarrow \Pr[R_{M,x}] \geq 3/4.$$

    Not every poly-time TM $M(x,u)$ defines a language in **BPP**.
- Ex.: What about **PP**?

Leaf languages



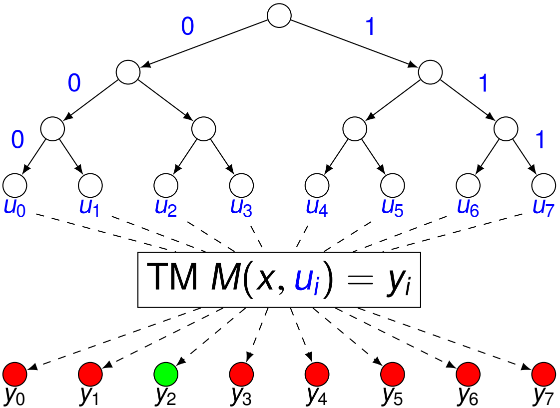
Definition

For a poly-time $M(x, u)$ using certificates $u \in \{0, 1\}^{p(|x|)}$ set

$$L_M(x) := y_0 y_1 \dots y_{2^{p(|x|)}-1} \text{ with } y_i = M(x, u_i) \text{ and } (u_i)_2 = i$$

The leaf-language of M is then $L_M := \{L_M(x) \mid x \in \{0, 1\}^*\}$.

Leaf languages

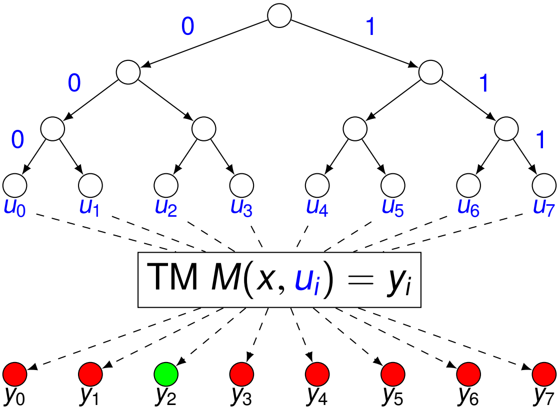


Definition (cont'd)

For $A, R \subseteq \{0,1\}^*$ with $A \cap R = \emptyset$ the class $\mathbf{C}[A, R]$ consists of all language L for which there is a TM $M(x, u)$ s.t. $\forall x \in \{0,1\}^*$:

$$x \in L \Rightarrow L_M(x) \in A \text{ and } x \notin L \Rightarrow L_M(x) \in R.$$

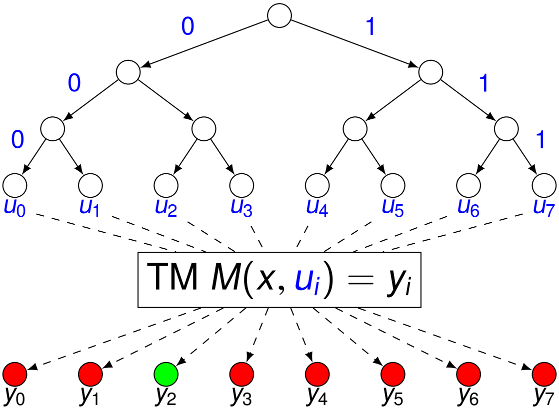
Leaf languages



Definition (cont'd)

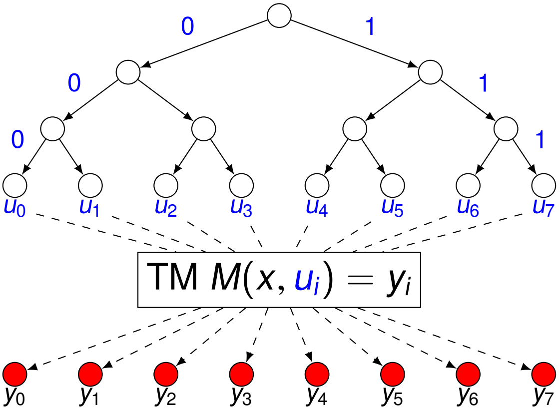
C$[A, R]$ is called syntactic if $A \cup R = \{0, 1\}^*$, otherwise it is called semantic.

Leaf languages



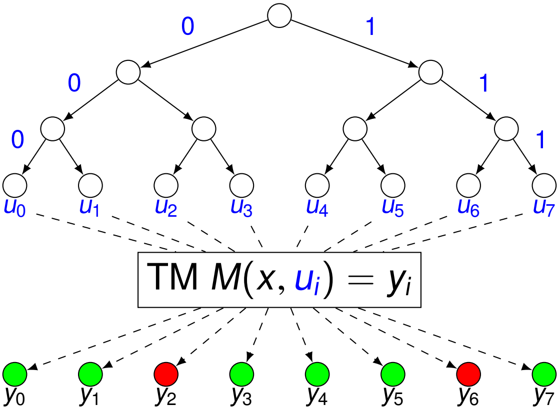
- **NP** $=$ **C**$[(0+1)^*1(0+1)^*, 0^*]$

Leaf languages



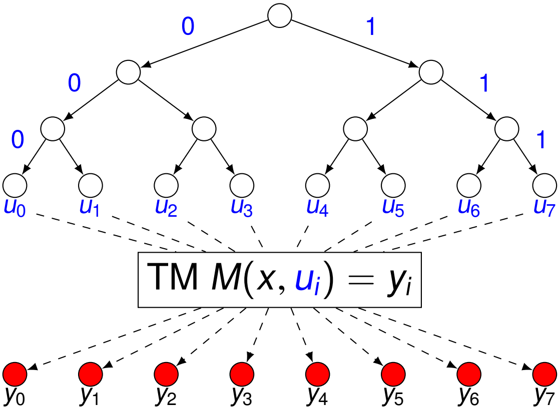
- **NP** $=$ **C**$[(0+1)^*1(0+1)^*, 0^*]$

Leaf languages



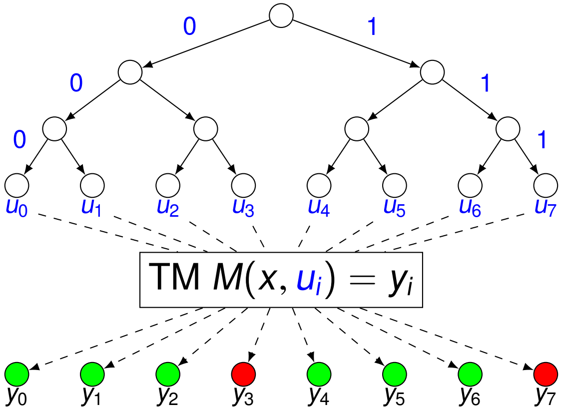
- **NP** $=$ **C**$[(0+1)^*1(0+1)^*, 0^*]$
- **RP** $=$ **C**$[\{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} \geq 3\}, 0^*]$

Leaf languages



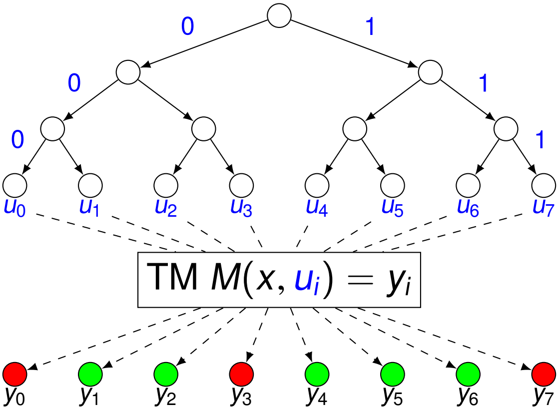
- **NP** $= \mathbf{C}[(0+1)^* 1 (0+1)^*, 0^*]$
- **RP** $= \mathbf{C}[\{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} \geq 3\}, 0^*]$

Leaf languages



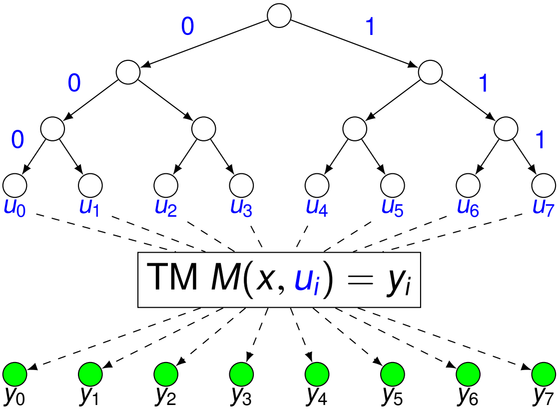
- **NP** $= \mathbf{C}[(0+1)^*1(0+1)^*, 0^*]$
- **RP** $= \mathbf{C}[\{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} \geq 3\}, 0^*]$
- **PP** $= \mathbf{C}[\{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} \geq 3\}, \{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} < 3\}]$

Leaf languages



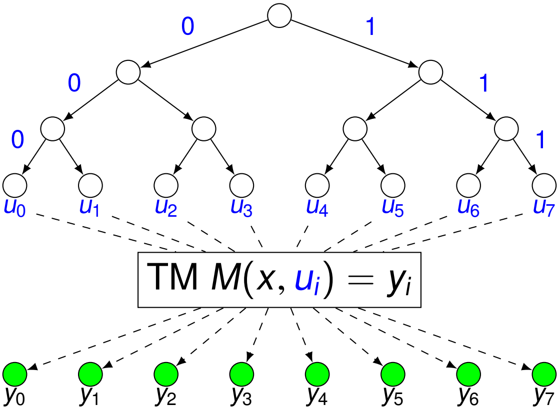
- **NP** $= \mathbf{C}[(0+1)^*1(0+1)^*, 0^*]$
- **RP** $= \mathbf{C}[\{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} \geq 3\}, 0^*]$
- **PP** $= \mathbf{C}[\{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} \geq 3\}, \{w \in \{0,1\}^* \mid \frac{\#_1 w}{\#_0 w} < 3\}]$

Leaf languages



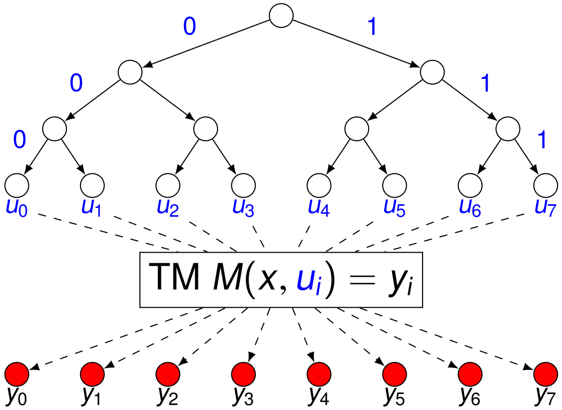
- What about **P**?

Leaf languages



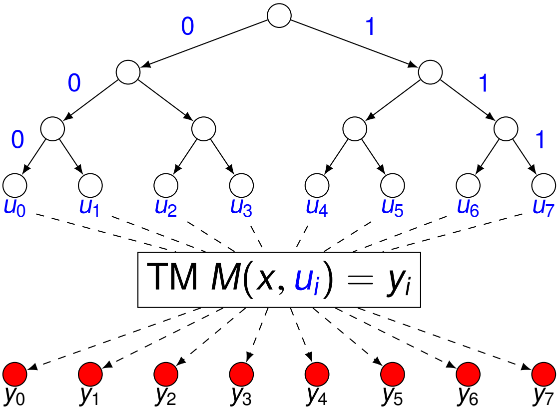
- What about **P**?
- $\mathbf{P} = \mathbf{P}[1(0+1)^*, 0(0+1)^*]$.

Leaf languages



- What about **P**?
- **P** $=$ **P**$[1(0+1)^*, 0(0+1)^*]$.

Leaf languages



- What about **P**?
- **P** $=$ **P**$[1(0+1)^*, 0(0+1)^*]$.
- Certificate $0 \dots 0$ can always be used (compare this to **BPP**)