

CSE 203A: Randomized Algorithms

Spring 2026

Lecture 2: Randomized Complexity Classes

Date: April 1 2026

Instructor: Daniel M. Kane

Scribe: Tyler Lee

1 Introduction

This lecture covers chapter 1.5 of the textbook. A multitude of random complexity classes were introduced in this lecture, covering the connections between PP, BPP, RP, coRP, and ZPP as well as their strengths and limitations.

2 Main Definitions

Definition 2.1 (Randomized Algorithm). A randomized algorithm is an algorithm that takes as input x and a random string r .

Definition 2.2 (Efficiency). A randomized algorithm is efficient (polynomial time) if the run time of $A(x,r)$ is $\text{poly}(|x|)$ for all x,r .

Definition 2.3 (Correctness). A randomized algorithm is correct if the algorithm correctly identifies the solution to the problem more times than not by an acceptable threshold.

Definition 2.4 (PP: Probabilistic Polynomial Time). A language L is in PP if and only if there exists a polynomial time random algorithm A such that:

$$x \in L \rightarrow \Pr(A(x)\text{accepts}) > 0.5$$

$$x \notin L \rightarrow \Pr(A(x)\text{accepts}) < 0.5$$

Definition 2.5 (BPP: Bounded Error Probabilistic Polynomial Time). A language L is in BPP if and only if there exists a polynomial time random algorithm A such that:

$$x \in L \rightarrow \Pr(A(x)\text{accepts}) > \frac{2}{3}$$

$$x \notin L \rightarrow \Pr(A(x)\text{accepts}) < \frac{1}{3}$$

Note: these bounds can be anything (e.g):

$$x \in L \rightarrow \Pr(A(x)\text{accepts}) > c + \epsilon$$

$$x \notin L \rightarrow \Pr(A(x)\text{accepts}) < c - \epsilon$$

Any reasonably sized gap performs better than PP.

If ϵ can be exponentially small, $\text{BPP} \approx \text{PP}$.

Definition 2.6 (RP: Randomized Polynomial Time). A language L is in RP if and only if there exists a polynomial time random algorithm A such that:

$$x \in L \rightarrow \Pr(A(x)\text{accepts}) > \frac{1}{2}$$

$$x \notin L \rightarrow \Pr(A(x) \text{ accepts}) = 0$$

Note: the bound $x \in L$ can be any positive fraction, but performs better when it is not close to 0 or 1.

Definition 2.7 (coRP: Randomized Polynomial Time). A language L is in coRP if and only if there exists a polynomial time random algorithm A such that:

$$x \in L \rightarrow \Pr(A(x) \text{ accepts}) = 0$$

$$x \notin L \rightarrow \Pr(A(x) \text{ accepts}) < \frac{1}{2}$$

Note: the bound $x \notin L$ can be any positive fraction, but performs better when it is not close to 0 or 1.

Definition 2.8 (ZPP: Zero Error Probabilistic Polynomial Time). A language L is in ZPP if there exists a random algorithm A that always computes the right answer and $E[\text{runtime}] = \text{poly}(|x|)$. For all x , $A(x, r)$ accepts if and only if $x \in L$.

3 Main Results

Theorem 3.1. *We can amplify the probability of success by running a BPP algorithm many times with independent randomness.*

Proof. Run BPP algorithm $O\left(\frac{\log(\frac{1}{\delta})}{\epsilon^2}\right)$ times with independent randomness. If fraction of accepts is greater than c , then the output is correct with probability greater than $1 - \delta$. \square

Theorem 3.2 ($ZPP = RP \cap coRP$).

Proof. $ZPP \subset RP$

A ZPP algorithm has expected runtime R .

If algorithm takes longer than $10R$, terminate and reject (return 0).

if $x \notin L$:

- terminate early $\rightarrow 0$

- time out $\rightarrow 0$

if $x \in L$:

- terminate early $\rightarrow 1$

- time out $\rightarrow 0$

Want to show $\Pr(\text{run too long}) < \frac{1}{2}$

$$\Pr(\text{runtime} > 10R) < \frac{E(\text{runtime})}{10R} = \frac{R}{10R} = \frac{1}{10}$$

\square

Remark 3.3. The proof for coRP was left out but it is essentially the same steps as the proof for RP.

4 Proof Ideas and Intuition

PP doesn't work in real case scenarios as the threshold between accepting and rejecting can be so small that the algorithm essentially works no better than a 50-50 chance.

BPP works better because we induce a gap between the thresholds of accepting and rejecting, making it more likely to produce usable results as there is a larger probability gap between whether we accept or reject the input.

RP does not allow for false positives.

coRP does not allow for false negatives.

ZPP has zero error (no false positives or negatives). Combining the subset (algorithms) of RP and coRP together will result in a set (algorithm) that does not allow for false positives nor false negatives, hence creating a zero error set (ZPP algorithm).

5 Examples

Example 5.1 (Solovay-Strassen). If n is prime, then for all a , $a^{n-1} \equiv \left(\frac{a}{n}\right) \pmod{n}$

Solovay-Strassen is an example of coRP. This algorithm always identifies prime numbers but it can have false positives. This algorithm will never have false negatives.

Example 5.2 (Quick Sort). Quick sort is an example of ZPP. This algorithm acts randomly upon its input but always outputs the correct solution.

6 Summary

There are a multitude of different randomized complexity classes that have algorithms that are useful in different scenarios depending on what needs to be prioritized in both efficiency and correctness.