CMPUT 675: Computational Complexity Theory **Winter 2019**

Lecture 10 (Feb 7): Probabilistic Turing Machines

Lecturer: Zachary Friggstad *Scribe: Jacob Garber*

As we saw last class, it is often possible to achieve efficient solutions to problems using randomized algorithms. Augmenting an algorithm with a random source seems to increase its possible power, and it is an open question if randomness is inherently required to solve some problems efficiently. In this section we will analyze Turing machines that are able to make random decisions, and several corresponding computational complexity classes. As we will define them, these Turing machines are allowed to give incorrect outputs, or even loop forever. Due to their random nature, we will prove several theorems from probability theory to aid our analysis.

10.1 Probabilistic Turing Machines

A **probabilistic Turing machine** is a Turing machine M that has two transition functions δ_0 and δ_1. At each point of the computation on some input x, the Turing machine randomly chooses with probability $\frac{1}{2}$ to apply δ_0 or δ_1. This choice is made independently of all previous choices. Running $M(x)$ may lead to M accepting x on some runs and rejecting it on others, so we can treat $M(x)$ as a random variable into the set $\{\text{ACCEPT}, \text{REJECT}\}$.

Like a NDTM, using two transition functions leads to a tree of possible computational paths. If on an input x every computational branch of the tree eventually halts, we can define $\Pr[M(x) = \text{ACCEPT}]$ as the fraction of all computational branches that lead to an accepting state. Note in particular that since the computational tree has a finite size, all leaves in the tree have some non-zero probability of being taken. In this case, we say M has running time $T(n)$ if for all inputs $x \in \{0,1\}^*$, M halts on all possible computational paths within at most $T(|x|)$ steps.

Using probabilistic Turing Machines, we can define a variety of complexity classes. Though it is possible to consider classes of arbitrary computational complexity, we will focus most of our attention on ones that run in polynomial time.

10.2 One-Sided Error and RP

Definition 1 *For any function $T : \mathbb{N} \to \mathbb{N}$, we say a language L is in* $\mathbf{RTIME}(T(n))$ *if there is a PTM M that runs in time $O(T(n))$, and that for all $x \in \{0,1\}^*$,*

- $x \in L \implies \Pr[M(x) = \text{ACCEPT}] \geq \frac{2}{3}$
- $x \notin L \implies \Pr[M(x) = \text{ACCEPT}] = 0$

Using this, we define

$$\mathbf{RP} = \bigcup_{c>0} \mathbf{RTIME}(n^c)$$

Algorithms of this sort are called "Monte-Carlo" algorithms and have "one-sided error": inputs not in the language are always rejected, but if an input is in the language, there is some possibility the algorithm will incorrectly reject it.

As we saw last time, an example of this is the Schwartz-Zippel algorithm for nonzero polynomial circuit identification. Recall that given a polynomial circuit, this algorithm will randomly generate a testing value, and evaluate the polynomial at the value to see if it is zero. Any zero polynomial will always evaluate to zero, and thus be rejected, but given a nonzero polynomial, there is a small possibility that the value tested is a root, and then the polynomial is rejected incorrectly.

From the definition of **RP**, we immediately have the following.

Proposition 1 $\mathbf{P} \subseteq \mathbf{RP} \subseteq \mathbf{NP}$

Proof. The first inclusion is simple, since any polynomial time DTM can be interpreted as a PTM with identical transition functions that makes no errors. For the second inclusion, take any language $L \in \mathbf{RP}$ with a polynomial time PTM M. Dropping the probabilities from the transition functions, we can reinterpret M and its tree of computation as a NDTM. For any $x \in L$, we know $\Pr[M(x) = \text{ACCEPT}] \geq \frac{2}{3}$, so there must exist some accepting path within the tree. Conversely, for any $x \notin L$, we know $\Pr[M(x) = \text{ACCEPT}] = 0$, so no accepting path can exist. Thus $L \in NP$. ■

10.2.1 Robustness of RP

In the definition of **RP**, the constant $\frac{2}{3}$ seems a bit arbitrary. However, this is only for the sake of convenience: we can replace it with an arbitrarily low or high value, as we shall see below.

Definition 2 *We say a language L is in $\mathbf{RP}^1$ if the same conditions as **RP** hold, except that there is some fixed constant $c > 0$ such that for all $x \in L$,*

$$\Pr[M(x) = \text{ACCEPT}] \geq \frac{1}{|x|^c}$$

*Likewise, we say a language L is in $\mathbf{RP}^2$ if the same conditions hold as in **RP**, except that there is some fixed constant $d > 0$ such that for all $x \in L$,*

$$\Pr[M(x) = \text{ACCEPT}] \geq 1 - \frac{1}{2^{|x|^d}}$$

Claim 1 $\mathbf{RP}^1 = \mathbf{RP}^2 = \mathbf{RP}$

Proof. The inclusions $\mathbf{RP}^2 \subseteq \mathbf{RP} \subseteq \mathbf{RP}^1$ are immediate from the definitions, so it remains to show $\mathbf{RP}^1 \subseteq \mathbf{RP}^2$. The essential idea of the technique is that given a Turing machine with a high probability of failing, we can amplify the probability of success arbitrarily high by performing many independent trial runs and accepting iff at least one of the runs accepts.

Let $L \in \mathbf{RP}^1$ be decided by a polynomial time PTM M, such that there is a $c > 0$ so that for all $x \in L$,

$$\Pr[M(x) = \text{ACCEPT}] \geq \frac{1}{|x|^c}$$

Let $d > 0$ be arbitrary. Consider a new PTM N that performs the following: on input x, run $M(x)$ for $|x|^{c+d}$ runs, and accept iff at least one of the runs accepts. Note that if $x \notin L$, then M will always reject x, and so N will reject x as well. Otherwise, if $x \in L$, then

$$\Pr[M(x) = \text{REJECT}] \leq 1 - \frac{1}{|x|^c}$$

and then so over $|x|^{c+d}$ independent runs

$$\Pr[N(x) = \text{REJECT}] \leq \left(1 - \frac{1}{|x|^c}\right)^{|x|^{c+d}}$$

Using that $\left(1 - \frac{1}{t}\right)^t \leq \frac{1}{e}$ for arbitrary $t > 0$, this implies

$$\Pr[N(x) = \text{REJECT}] \leq \frac{1}{e^{|x|^d}} \leq \frac{1}{2^{|x|^d}}$$

and so then

$$\Pr[N(x) = \text{ACCEPT}] \geq 1 - \frac{1}{2^{|x|^d}}$$

The PTM N runs in polynomial time because it consists of a polynomial number of independent repetitions of the calculation $M(x)$, and so $L \in \mathbf{RP}^2$. ■

10.2.2 coRP

As usual, we define $\mathbf{coRP} = \{\overline{L} \mid L \in \mathbf{RP}\}$. More explicitly, a language L is in **coRP** iff there exists a PTM M that runs in time $p(n)$ for some polynomial p, and that for all $x \in \{0,1\}^*$,

- $x \in L \implies \Pr[M(x) = \text{ACCEPT}] = 1$
- $x \notin L \implies \Pr[M(x) = \text{ACCEPT}] \leq \frac{1}{3}$

Algorithms of this sort also have "one-sided error": in this case, inputs in the language are always accepted, but if an input is not in the language, there is some possibility the algorithm will incorrectly accept it.

A classic example of a **coRP** algorithm is the Miller-Rabin primality test. Any numbers this test rejects are certainly composite by providing a witness of non-primality, but this algorithm may also accept a composite number if no witnesses can be found (though of course, the odds of this happening can be driven down using many independent trials).

As expected, we have $\mathbf{P} \subseteq \mathbf{coRP} \subseteq \mathbf{coNP}$, and that $\mathbf{coRP}^1 = \mathbf{coRP}^2 = \mathbf{coRP}$.

10.3 Zero-Sided Error and ZPP

Let M be a PTM and x any input. When analyzing the running time of M on x in the previous section, we assumed that all branches of the computational tree eventually halted. Now we will relax this restriction: M is allowed to not halt on some paths, as long as the number of such paths is "essentially zero". More formally, for all inputs x, we require that $\Pr[M(x) \text{ terminates}] = 1$. An example of such a Turing machine is one that flips a coin until it reaches heads. Of course, it is possible that this Turing machine will always get tails and flip forever, but the odds of this happening are vanishingly small, in fact 0.

For a PTM Turing machine M and input x, let $T_{M,x}$ be the random variable of the running time of $M(x)$. Since it is possible for the running time to be unbounded, we do not analyze the worst-case running time, but instead the average running time. Formally, we say M has *expected running time* $T(n)$ if for all $x \in \{0,1\}^*$, the expectation $\mathrm{E}[T_{M,x}]$ is at most $T(|x|)$. For the coin-flipping Turing machine above, the odds of landing a heads are exactly $\frac{1}{2}$ each time, so the expected number of flips will be 2. We use this to define PTMs that never err.

Definition 3 *For any function $T : \mathbb{N} \to \mathbb{N}$, we say a language L is in* $\mathbf{ZTIME}(T(n))$ *if there is a PTM M that has expected running time $O(T(n))$, and that for all $x \in \{0,1\}^*$,*

- $x \in L \implies \Pr[M(x) = \text{ACCEPT}] = 1$
- $x \notin L \implies \Pr[M(x) = \text{ACCEPT}] = 0$

Using this, we define

$$\mathbf{ZPP} = \bigcup_{c>0} \mathbf{ZTIME}(n^c)$$

Algorithms of this sort are often called "Las Vegas" algorithms and have "zero-sided error": it is impossible for them to make mistakes. From the definition of **ZPP**, we can immediately deduce two simple propositions.

Lemma 1 $\mathbf{P} \subseteq \mathbf{ZPP}$

Proof. Any polynomial time DTM makes no errors and has guaranteed polynomial running time. ■

Lemma 2 $\mathbf{ZPP} = \mathbf{coZPP}$

Proof. Given any language $L \in \mathbf{ZPP}$, its complement $\overline{L}$ can be recognized by simply negating the output of the PTM that recognizes L. This implies $\overline{L} \in \mathbf{ZPP}$, and so $L \in \mathbf{coZPP}$. Thus $\mathbf{ZPP} \subseteq \mathbf{coZPP}$, with $\mathbf{coZPP} \subseteq \mathbf{ZPP}$ being similar. ■

Given our definition of **ZPP** as being PTMs that make no errors, it seems plausible that it is contained in both **RP** and **coRP**. This is indeed the case, and in fact **ZPP** is the intersection of those two classes. To prove this, we first need the following result from probability theory.

Theorem 1 (Markov's Inequality) *Let X be any random variable that satisfies $X \geq 0$. Then for any $t > 0$,*

$$\Pr[X \geq t] \leq \frac{\mathrm{E}[X]}{t}$$

Proof. We will show the claim when X is a discrete random variable. That is, X take values within a countable set $\Omega \subseteq \mathbb{R}$. Using the definition of expectation,

$$\begin{aligned}
\mathrm{E}[X] &= \sum_{k \in \Omega} \Pr[X = k] \cdot k \\
&= \sum_{k \geq t} \Pr[X = k] \cdot k + \sum_{k < t} \Pr[X = k] \cdot k \\
&\geq \sum_{k \geq t} \Pr[X = k] \cdot k && \text{the second sum is positive since all } k \geq 0 \\
&\geq \sum_{k \geq t} \Pr[X = k] \cdot t && \text{since all } k \geq t \text{ in this sum} \\
&= t \cdot \Pr[X \geq t]
\end{aligned}$$

The result then follows by rearrangement. For continuous random variables, the proof is similar, but uses integrals instead of summations. ■

Theorem 2 $\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$

Proof.

We first show $\mathbf{ZPP} \subseteq \mathbf{RP}$. From this it will follow that $\mathbf{coZPP} \subseteq \mathbf{coRP}$, and so then $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$ by Lemma 2.

Let $L \in \mathbf{ZPP}$ with a zero-error PTM M, with expected polynomial running time $p(n)$. Note that this inclusion is not immediately trivial, since while M makes no errors, some computational branches may run forever. To convert it to an algorithm with a guaranteed polynomial running time, consider the following PTM N. On input x, simulate $M(x)$ for $3p(|x|)$ steps. If $M(x)$ halts in that time, return its answer, and otherwise reject.

First, note that if $x \notin L$, then $N(x)$ will always reject. Indeed, if the computation of $M(x)$ halts in time, M will reject x correctly, and so N will reject it as well. Otherwise if the computation runs overtime x will be rejected by default. Thus in this case, $\Pr[N(x) = \text{ACCEPT}] = 0$.

Conversely, suppose $x \in L$, and let Y be the random variable for the running time of $M(x)$. Then using Markov's Inequality and that $\mathrm{E}[Y] \leq p(|x|)$,

$$\Pr[N(x) = \text{REJECT}] = \Pr[Y > 3p(|x|)] \leq \frac{\mathrm{E}[Y]}{3p(|x|)} \leq \frac{p(|x|)}{3p(|x|)} = \frac{1}{3}$$

And so $\Pr[N(x) = \text{ACCEPT}] \geq \frac{2}{3}$. By construction, N also runs in polynomial time, and so $L \in \mathbf{RP}$.

Next, we show $\mathbf{RP} \cap \mathbf{coRP} \subseteq \mathbf{ZPP}$. Suppose $L \in \mathbf{RP} \cap \mathbf{coRP}$, with corresponding polynomial time PTMs M_1 and M_2. Consider the following PTM N. On input x, simulate $M_1(x)$ and then $M_2(x)$. If $M_1(x) = \text{ACCEPT}$, then accept, and if $M_2(x) = \text{REJECT}$, then reject. If neither happen, then loop again until it does.

This algorithm is never incorrect, since if $M_1(x) = \text{ACCEPT}$, then it is correct that $x \in L$, and if $M_2(x) = \text{REJECT}$, then it is correct that $x \notin L$. We now check the running time.

On some input x, let Y be the random variable of the number of iterations of $N(x)$. Note that if $x \in L$, we have

$$\Pr[N(x) \text{ halts after one iteration}] = \Pr[M_1(x) = \text{ACCEPT}] \geq \frac{2}{3}$$

and likewise that if $x \notin L$, then

$$\Pr[N(x) \text{ halts after one iteration}] = \Pr[M_2(x) = \text{REJECT}] \geq \frac{2}{3}$$

so recursively,

$$\mathrm{E}[Y] = 1 + \Pr[N(x) \text{ didn't halt after one iteration}] \cdot \mathrm{E}[Y] \leq 1 + \frac{1}{3}\mathrm{E}[Y] \implies \mathrm{E}[Y] \leq \frac{3}{2}$$

Each iteration of the loop runs in guaranteed polynomial time, and so N runs in expected polynomial time overall. Thus $L \in \mathbf{ZPP}$. ■

10.4 Two-Sided Error and BPP

Having studied algorithms that can err in one or zero directions, we now look at algorithms that can err in both directions.

Definition 4 *For any function $T : \mathbb{N} \to \mathbb{N}$, we say a language L is in* $\mathbf{BTIME}(T(n))$ *if there is a PTM M that has running time $T(n)$, and that for all $x \in \{0,1\}^*$,*

- $x \in L \implies \Pr[M(x) = \text{ACCEPT}] \geq \frac{2}{3}$
- $x \notin L \implies \Pr[M(x) = \text{ACCEPT}] \leq \frac{1}{3}$

Using this, we define

$$\mathbf{BPP} = \bigcup_{c>0} \mathbf{BTIME}(n^c)$$

Algorithms of this sort have "two-sided error": if the algorithm accepts or rejects the input, we cannot be sure either way that it is correct. Note that similar to **RP**, we require the PTMs for this class to halt on all computational paths.

As with **RP** and **coRP**, the definition of **BPP** is quite robust, since we can replace the value of $\frac{2}{3}$ with any value $1/2 < c < 1$. Proving this will require the following theorem from probability theory.

Theorem 3 (Chernoff Bound) *Let $X_1, \ldots, X_n$ be mutually independent random variables over the set $\{0, 1\}$ (eg. coin flips). Define*

$$X = \sum_{i=1}^{n} X_i \qquad \mu = \mathrm{E}[X]$$

Then for all $0 \leq \delta \leq 1$ and $c \geq \mu$,

$$\Pr[X \geq (1+\delta)c] \leq e^{-\frac{\delta^2}{3}c}$$

Proof. We will prove in the case $c = \mu$. Let $t > 0$ be arbitrary. Then using Markov's Inequality,

$$\Pr\left[X \geq (1+\delta)\mu\right] = \Pr\left[e^{tX} \geq e^{t(1+\delta)\mu}\right] \leq \frac{\mathrm{E}\left[e^{tX}\right]}{e^{t(1+\delta)\mu}}$$

Then using the definition of X,

$$\mathrm{E}\left[e^{tX}\right] = \mathrm{E}\left[\prod_{i=1}^{n} e^{tX_i}\right] = \prod_{i=1}^{n} \mathrm{E}\left[e^{tX_i}\right]$$

where the last equality holds by the independence of the random variables. Now for each $1 \leq i \leq n$, let $\mu_i = \mathrm{E}[X_i]$. Then,

$$\mathrm{E}\left[e^{tX_i}\right] = \Pr[X_i = 0] \cdot e^{t\cdot 0} + \Pr[X_i = 1] \cdot e^{t\cdot 1} = (1-\mu_i) + \mu_i e^t = 1 + \mu_i\left(e^t - 1\right)$$

Then using $1 + x \leq e^x$ and linearity of expectation, we have

$$\mathrm{E}\left[e^{tX}\right] = \prod_{i=1}^{n}\left(1 + \mu_i\left(e^t - 1\right)\right) \leq \prod_{i=1}^{n}\left(e^{\mu_i\left(e^t-1\right)}\right) = e^{\mu\left(e^t-1\right)}$$

And so

$$\Pr\left[X \geq (1+\delta)\mu\right] \leq \frac{e^{\mu\left(e^t-1\right)}}{e^{t(1+\delta)\mu}} = \left(\frac{e^{e^t-1}}{e^{t(1+\delta)}}\right)^{\mu}$$

This equation is true for any $t > 0$, and so in particular we can choose a t that minimizes the right hand side. Using simple calculus, we see this is the case for $t = \ln(1+\delta)$, so we conclude

$$\Pr\left[X \geq (1+\delta)\mu\right] \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu} \leq e^{-\frac{\delta^2}{3}\mu}$$

where the last inequality follows by the next lemma. ■

Lemma 3 *For all* $0 \leq \delta \leq 1$,

$$\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}} \leq e^{-\frac{\delta^2}{3}}$$

Proof. Taking logarithms of both sides, it suffices to show that

$$\delta - (1+\delta)\ln(1+\delta) \leq -\frac{\delta^2}{3}$$

which is equivalent to showing that $f(\delta) \leq 0$, where

$$f(\delta) = \frac{\delta^2}{3} + \delta - (1+\delta)\ln(1+\delta)$$

First, note that $f(0) = 0$, so it suffices to show that $f'(\delta) \leq 0$ on $[0,1]$. Differentiating, we have

$$f'(\delta) = \frac{2\delta}{3} - \ln(1+\delta) \qquad f''(\delta) = \frac{2}{3} - \frac{1}{1+\delta}$$

Next, $f'(0) = 0$, and a simple calculation shows that $f'(1) < 0$. Also, f'' only has a single root in $[0,1]$, and so the claim follows by Rolle's Theorem. ■

Remark 1 Intuitively, the Chernoff Bound states that the odds of achieving a value "far away" from the expected value decrease exponentially as the number of trials increases. For example, consider a random variable from flipping a coin, where heads is 1 and tails is 0. For 4 flips, the expected number of heads is 2, but the odds of flipping 3 heads is 1/4. Not too bad. However, if we have 400 coins, the odds of flipping 300 heads is now less than 10^{-24}! Visually, the probability distribution of flipping coins (called the *binomial distribution*) approximates a Bell curve, and as the number of trials increases, the distribution will "shronk" to the expected value.

Definition 5 *We say a language L is in* $\mathbf{BPP}^1$ *if the same conditions as* **BPP** *hold, except that there is some fixed constant $c > 0$ such that for all $x \in \{0,1\}^*$,*

$$\Pr[M(x) \text{ is correct}] \geq \frac{1}{2} + \frac{1}{|x|^c}$$

Likewise, we say a language L is in $\mathbf{BPP}^2$ *if the same conditions hold as in* **BPP**, *except that there is some fixed constant $d > 0$ such that for all $x \in \{0,1\}^*$,*

$$\Pr[M(x) \text{ is correct}] \geq 1 - \frac{1}{2^{|x|^d}}$$

Claim 2 $\mathbf{BPP}^1 = \mathbf{BPP}^2 = \mathbf{BPP}$

Proof. The inclusions $\mathbf{BPP}^2 \subseteq \mathbf{BPP} \subseteq \mathbf{BPP}^1$ are immediate from the definitions, so it remains to show $\mathbf{BPP}^1 \subseteq \mathbf{BPP}^2$. Suppose $L \in \mathbf{BPP}^1$ has a polynomial time PTM M, such that for some constant $c > 0$ and all $x \in \{0,1\}^*$,

$$\Pr[M(x) \text{ is correct}] \geq \frac{1}{2} + \frac{1}{|x|^c}$$

For an arbitrary $d > 0$, define the following PTM N: on input x, run $M(x)$ for $t = 6|x|^{3c+d}$ trials and output the majority answer.

For each $1 \leq i \leq t$, define the random variable X_i to be 0 if the calculation $M(x)$ in the ith trial is correct, and 1 if it is incorrect. Let X be their sum. The definition of the X_i may seem opposite of normal, but it ensures $N(x)$ outputs the correct answer iff $X < t/2$ (that is, iff less than half of all answers are incorrect). So then,

$$\mathrm{E}[X] = \sum_{i=1}^{t} \mathrm{E}[X_i] = \sum_{i=1}^{t} \Pr[X_i = 1] \leq \sum_{i=1}^{t} \left(\frac{1}{2} - \frac{1}{|x|^c}\right) = \frac{t}{2} - \frac{t}{|x|^c}$$

The X_i are all mutually independent random variables, so choosing $\delta = 1/|x|^c$, we have by the Chernoff Bound that

$$\Pr\left[X \geq (1+\delta)\left(\frac{t}{2} - \frac{t}{|x|^c}\right)\right] \leq \exp\left(-\frac{\delta^2}{3} t \left(\frac{1}{2} - \frac{1}{|x|^c}\right)\right) = \exp\left(-\left(|x|^c - 2\right)|x|^d\right) \leq e^{-|x|^d} \leq \frac{1}{2^{|x|^d}}$$

Furthermore, a simple calculation shows

$$(1+\delta)\left(\frac{1}{2} - \frac{1}{|x|^c}\right) = \frac{1}{2} - \frac{1}{2|x|^c} - \frac{1}{|x|^{2c}} < \frac{1}{2} \implies (1+\delta)\left(\frac{t}{2} - \frac{t}{|x|^c}\right) < \frac{t}{2}$$

And so thus,

$$\Pr[N(x) \text{ is incorrect}] = \Pr[X \geq t/2] \leq \Pr\left[X \geq (1+\delta)\left(\frac{t}{2} - \frac{t}{|x|^c}\right)\right] \leq \frac{1}{2^{|x|^d}}$$

which implies

$$\Pr[N(x) \text{ is correct}] \geq 1 - \frac{1}{2^{|x|^d}}$$

By construction, N will have polynomial overhead in the length of $|x|$, and thus $L \in \mathbf{BPP}^2$. ■

10.4.1 BPP and Other Classes

From the definition of **BPP**, it is immediate that $\mathbf{RP} \subseteq \mathbf{BPP}$ and $\mathbf{coRP} \subseteq \mathbf{BPP}$. For other complexity classes, the case is not so clear. For example, it is unlikely that $\mathbf{NP} \subseteq \mathbf{BPP}$, since then Claim 2 would imply all **NP**-complete problems have efficient randomized algorithms. However, it is conjectured that $\mathbf{BPP} \subseteq \mathbf{NP}$, and in fact we have even the following:

Conjecture 1 $\mathbf{BPP} = \mathbf{P}$

In this case, all the probabilistic complexity classes with dashed lines between them in Figure 10.4.1 would collapse to **P**. This seems quite surprising, since it would mean all probabilistic algorithms in **BPP** can be derandomized with only a polynomial loss of efficiency. However, based on several minimal assumptions about circuit complexity, most researchers today believe that this is possible. (See [AB09] chapter 20 for more details.)

References

AB09 S. Arora and B. Barak, Computational Complexity: A Modern Approach, *Cambridge University Press, New York, NY, USA*, 2009.

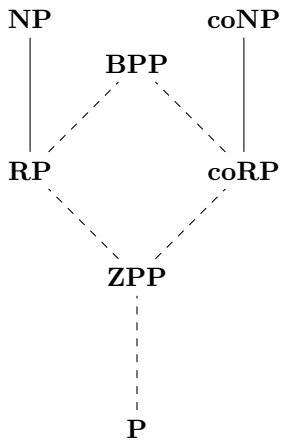

Figure 10.1: Known Inclusions Between Probabilistic Complexity Classes