CS 1511 Homework 14

Mathew Varughese, Justin Kramer, Zach Smith

Wednesday, March 6

26. a) If a language is in ZPP, then the polynomial-time PTM M can prove that the language is in ZPP.

For this Turing Machine, it will either output the correct answer or "DON'T KNOW".

There is at most a 1/2 chance that it outputs "DON'T KNOW". If the machine is run once, then the chance that it outputs this is at most 1/2, with at least a 1/2 chance of outputting the correct answer.

If the machine is run twice, there is at most a 1/4 chance that it doesn't output the correct answer.

If you take this to a limit of infinity, there is no chance that you do not eventually output the correct answer when you run the machine an infinite amount of times. It will take poly-time to run, so it doesn't matter if we simulate this machine for many many times.

In the other direction, we can use Markov's inequality to show that with our PTM M, L is in ZPP.

With this inequality, we have that the P(X $\geq$ a) $\leq E(X)/a$.

In this case, we can make a = 3E(X) for an example. Therefore, the probability that our X, our runtime, is greater than expected poly-time is now 1/3.

If we take this to a limit of inifinity, the probability reduces to 0 that our expected runtime is greater than polynomial time.

We can take our PTM M and run it to build our language L. This language will be in ZPP because if the PTM outputs an answer that isn't "DON'T KNOW", then it must be a correct answer. This is necessary for ZPP. We can count these inputs as part of our language L.

With a runtime established and correct answers established to build a language L, then we have shown that if our PTM M exists, we can build a language L in ZPP.

26. b)

In order to show that $ZPP = RP \cap coRP$, we need to show two things.

We will begin by showing that $ZPP \subseteq RP \cap coRP$.

To start this, we must show that $ZPP \subseteq RP$ and $ZPP \subseteq coRP$.

We will begin by showing that $ZPP \subseteq coRP$.

Let L $\in$ ZPP. Then there exists a PTM M that for all x either correctly decides $x \in L$ or outputs "Don't know".

Let M' be a TM that on all input x, returns 1 if M(x) = "Don't know" and otherwise returns M(x).

So if $x \in L$, M(x) = 1 or M(x) = "Don't know" so M'(x) = 1 if $x \notin L$ M(x) = 0 correctly or M(x) = "Don't know", and here M'(x) returns incorrect with probability $\leq 1/2$.

Therefore, this clearly demonstrates that $ZPP \subseteq coRP$.

A very similar proof demonstrates that $ZPP \subseteq RP$ with the same methodology, except that M' returns 0 if M(x) = "Don't know".

Now we must demonstrate the second part, which is that $RP \cap coRP \subseteq ZPP$.

To show this, we begin with a language L $\in RP \cap coRP$. There exists two TM's from this, with

M_1(x) = 1 if $x \in$ L, and is incorrect for $x \notin$ L with probability $\leq 1/2$.

M_2(x) = 0 if $x \notin$ L, and is incorrect for $x \in$ L with probability $\leq 1/2$.

We then construct a PTM M' which works as follows:

Run M_1(x) and if M_1(x) = 0, return x $\notin$ L.

Run M_2(x) and if M_2(x) = 1, return x $\in$ L.

Else return "Don't know"

In this way, if M'(x) returns a 0 or a 1, it is correct.

If M'(x) returns "Don't know" with probability $\leq 1/2$ of being correct.

This is therefore ZPP, and we have shown that $RP \cap coRP \subseteq ZPP$.

With both of these proven, this shows that $ZPP = RP \cap coRP$.

27 Lemma 7.12 says that A coin with Pr[Heads] = p can be simulated by a PTM in expected time O(1) provided the ith bit of p is computable in poly(i) time.

We can construct this p such that a random coin that comes up with this probability will make a Turing Machine able to decide a language in undecidable time.

Assume that this is possible to make a PTM that simulates a coin flip with probability p. The ith bit of p can be computed in constant time.

The p is comprised of 1s and 0s. The turing machine in question (say TM T) generates a random "x". This x is a random variable. Then we compare x to that probability p. We do this by comparing bit by bit. If p is 0.0101010 and x is 0.0101011, we know that x has a greater probability than p.

We make each bit of this p represent whether a Turing Machine will halt on a particular language. So, order every Turing Machine and input w like (M,w) and correspond it with a integer. The ith integer in p will be a 1 if M halts on w and 0 if not.

Then, the Turing Machine T could solve the halting problem. To figure out if M halts on w, it would first correspond that M,w pair to an integer. Call this integer i. The ith digit of p would tell us what the answer to this question is. So we flip We flip a coin i times. We count the distribution of heads and tails. We can say with p confidence whether or not M halts w.

28 First we show NP $\in$ PP. Then we show coNP $\in$ PP.