CS5110: Complexity Theory

Shreyas Havaldar
CS18BTECH11042

October 29, 2020

1 Exercise Set 12

1.1 M to M'

$$x \in L, \implies Pr[M(x) = Acc.] \geq (1/p(x))$$
$$\implies x \in L, \implies Pr[M(x) \neq Acc.] \leq \left(1 - \frac{1}{p(x)}\right)$$

Now on running the input x on M, for n(x) times via our machine M'. As we know probability of any occurrence is less than equal to 1.

$$\implies x \in L, \implies Pr[M'(x) \neq Acc.] \leq \left(1 - \frac{1}{p(x)}\right)^n (x)$$

We know $\forall x, x \geq 0, 1 + x \leq e^x$. Thus we can substitute $\left(1 - \frac{1}{p(x)}\right)$ by $e^{\frac{-1}{p(x)}}$ and still the inequality would hold.

$$\implies x \in L, \implies Pr[M'(x) \neq Acc.] \leq \left(e^{\frac{-1}{p(x)}}\right)^n (x)$$
$$\implies x \in L, \implies Pr[M'(x) \neq Acc.] \leq \left(e^{\frac{-n(x)}{p(x)}}\right)$$
$$\implies x \in L, \implies Pr[M'(x) = Acc.] \geq \left(1 - e^{\frac{-n(x)}{p(x)}}\right)$$

Thus, for any polynomial function q(x) we choose n(x) such that $n(x) = q(x) \times p(x)$, so:

$$\implies x \in L, \implies Pr[M'(x) = Acc.] \geq \left(1 - e^{\frac{-q(x) \times p(x)}{p(x)}}\right)$$
$$\implies x \in L, \implies Pr[M'(x) = Acc.] \geq \left(1 - e^{-q(x)}\right)$$
$$\implies x \in L, \implies Pr[M'(x) = Acc.] \geq \left(1 - e^{-q(x)}\right)$$

And we know,

$$x \notin L, \implies Pr[M(x) = Acc.] = 0$$

And therefore for an input x that is not in language L, running the machine M on x, n(x) times would have no effect on the probability of it being accepted and it would still be 0, thus our machine M' would

$$\implies x \notin L, \implies Pr[M'(x) = Acc.] = 0$$

Thus for any q(x) we can construct a prob poly time machine M', such that it runs the prob poly time machine provided to us n(x) times and achieves our desired result.

1.2 To prove that $RP, coRP \subseteq BPP$

For a prob turing machine M, a language L is in RP if,

$$x \in L, \implies Pr[M(x) = Acc.] \geq \frac{1}{2}$$

$$x \notin L, \implies Pr[M(x) = Acc.] = 0$$

As we showed in 1.1, we can convert this is equivalent to a prob machine M' where

$$\implies x \in L, \implies Pr[M'(x) = Acc.] \geq \left(1 - e^{-q(x)}\right)$$

for any q(x) where our p(x) was $\frac{1}{2}$, now for $\left(1 - e^{-q(x)}\right) = \frac{1}{3}, -q(x) = \ln 2/3 \implies q(x) = \ln 3/2$ and thus $n(x) = \frac{ln3/2}{2}$, thus via our prob machine M', we run M the above calculated n(x) times to convert the RP instance into a BPP instance by satisfying the BPP requirement of giving the right answer with probability at least 2/3, when x belongs to the language L. Thus

$$x \in L, \implies Pr[M'(x) = Acc.] \geq \frac{2}{3}$$

When x is not in the language L, RP instance predicts correctly for all inputs and thus the converted instance by running M, n(x) times would also predict correctly for all inputs as shown in 1.1. Thus,

$$x \notin L, \implies Pr[M'(x) = Acc.] = 0 \leq \frac{1}{3}$$

Thus by the above algorithm we can convert every instance of a deciding via RP to an instance of deciding via BPP. Thus $RP \subseteq BPP$

Likewise for coRP we perform the same operation but this time, For a prob turing machine M, a language L is in coRP if,

$$x \in L, \implies Pr[M(x) = Acc.] = 1$$

$$x \notin L, \implies Pr[M(x) = Acc.] \leq \frac{1}{2}$$

and we would construct M' such that

$$x \in L, \implies Pr[M'(x) = Acc.] = 1 \geq \frac{2}{3}$$

$$x \notin L, \implies Pr[M'(x) = Acc.] \leq \frac{1}{3}$$

For x in the language L, the probability would stay 1 irrespective the number of runs of M on x, and thus it would always accept an input x in L, which would suffice the condition of x in L, being accepted with the probability of $\frac{2}{3}$ And likewise we would achieve a probability of atmost $\frac{1}{3}$ by simply running the machine M on an x not in L, for a c number of times to reduce the probability to $\frac{1}{3}$ from $\frac{1}{2}$ where $\frac{1}{3} = \left(\frac{1}{2}\right)^c \implies c = \ln 3/2$

Thus by the above algorithm we can convert every instance of a deciding via coRP to an instance of deciding via BPP. Thus $coRP \subseteq BPP$

1.3 Equivalence of the two ZPP definitions

Let X be a random variable denoting the running time of the algorithm. Given to us is $E[X] = O(t(n))$, where t is a polynomial function and n is the input size. And we know that for a language $L \in ZPP$ we have a prob turing machine M such that, $\forall x, Pr\left[M(x) = \text{Don't Know}\right] \leq \frac{1}{2}$

For the forward direction, we know that we keep running the machine until it either accepts or rejects the input, ie. until we don't get rid of the "Don't Know" output on x.

Let Y be the random variable denoting the run time for us to definitely receive a verdict on x.

$$E[X] = \sum_{i=1}^{\infty}(i \cdot (\text{Getting verdict "Don't Know" on } i^{th} \text{ run})) = \sum_{i=1}^{\infty}\left(i \cdot \left(\frac{1}{2}\right)^i\right) \tag{1}$$

As for each subsequent i^{th} re-run of M on x, all the previous i-1 times the machine should have also resulted in "Don't Know" answer for it to continue running further as we halt otherwise. Thus we can calculate the sum of this AGP.

$$\frac{E[X]}{2} = \sum_{i=1}^{\infty}\left(i \cdot \left(\frac{1}{2}\right)^{i+1}\right) \tag{2}$$

Subtracting (2) from (1) we get, a simple GP

$$\frac{E[X]}{2} = \sum_{i=1}^{\infty}\left(\frac{1}{2}\right)^i \tag{3}$$

$$= \frac{\frac{1}{2}}{1 - \frac{1}{2}} \tag{4}$$

$$= 1 \tag{5}$$

Therefore $E[X] = 2$, and we thus know that the expected number of runs of the machine to output a definitive "Yes" or "No" answer is 2 times the original run time of the machine, which we know is $O(t(n))$ and thus expected time of run of a prob machine is still $O(t(n))$ for getting rid of the "Don't Know" answer and halt, thus expected poly time.

Likewise for the other direction, we need to show that after running the machine on x for a certain amount of time, the probability of it resulting in "Don't Know" is atmost 1/2 to satisfy the definition of ZPP.

Using Markov's Inequality:

$$Pr[X \geq k \cdot \mathrm{E}[X]]) \leq \frac{1}{k}$$

where X is the random variable denoting the run time of the machine. Thus the probability of the machine not halting in time t, where $t = 2E[x]$, is atmost $\frac{1}{2}$. Thus we run the machine for $t = 2E[x]$ steps, and we know $E[x] = O(t(n))$ and thus we run the machine for poly time only.

If the machine halted within t steps, it would have given a definite answer "Yes" or "No" on having completed execution on x, which would have definitely been correct. Now if the machine did not finish running on x in t steps, we simply halt the machine and output "Don't Know", this will happen with atmost 1/2 probability as shown above, thus we output "Don't Know" with atmost 1/2 probability which satisfies our condition of ZPP.

And thus we have completed the proof showing the time calculations for both directions.