Name: ____

CS 620 Fall 2010 at ISU, Exam 2 SAMPLE

Prepared by assistant professor Jeff Kinne on October 25, 2010. You have until 4:00pm to finish the exam. The exam itself and blank paper I provide are all you will have to use (no computer, textbook, notes, cellphone, calculator, etc.). I have put point values on the problems so the total adds up to 24.

Problem 1 (5 points) Recall that PP consists of problems that can be solved by a poly-time randomized machine M such that for every input x, $\Pr_r[M(x,r) \text{ is correct}] > \frac{1}{2}$. Show that SAT is contained in PP. (By the NP-completeness of SAT, this shows that NP is contained in PP.)

Answer: Let ϕ be a formula on n variables, and let ϕ' be a formula on $n+1$ variables such that $\phi' = (x_{n+1} \wedge \phi) \vee (\neg x_{n+1})$. Note that ϕ' is satisfied by any of the 2^n assignments where x_{n+1} is false. Then ϕ' has more than half satisfying assignments iff ϕ is satisfiable. Let M be the randomized machine that takes an input formula ϕ' on $n+1$ variables, uses $n+1$ random bits, and outputs 1 iff the values of the random bits satisfy ϕ'. Then ϕ is in SAT iff ϕ' is a "yes" instance for the PP machine M (i.e., if $\Pr_r[M(\phi', r) = 1] > \frac{1}{2}$.

Problem 2 (3 points) Recall that RP consists of problems that can be solved by a poly-time randomized machine M such that: (i) for "yes" instances x, for every input x, $\Pr_r[M(x,r) = 1] \geq \frac{1}{2}$, and (ii) for "no" instances x, $\Pr_r[M(x,r) = 1] = 0$. Show that RP is in NP.

Answer: Let M be a randomized machine for an RP problem. If we interpret M as an NP verifier, we see that it works for this purpose too, where the random string is used for the candidate witness. If x is a "yes" instance, then there is a random string causing $M(x,r) = 1$. If x is a "no" instance, then there is no random string causing $M(x,r) = 1$.

Problem 3 (5 points) Recall that ZPP consists of problems that can be solved by a poly-time randomized machine M that can output 0, 1, or ? such that:
(i) for all x, $\Pr_r[M(x,r)$ outputs correct answer (0 or 1)$] \geq \frac{1}{2}$, and
(ii), for all x, $\Pr_r[M(x,r)$ outputs wrong 0/1 value$] = 0$.

Let M' be a randomized machine that solves a problem Π in the following way. When it outputs a 0/1 value, it is always correct. The running time could be larger than polynomial, but the expected running time is small. That is, if $t_{M(x,\cdot)}$ is a random variable for the running time of M on input x, then $E[t_{M(x,\cdot)}] \leq |x|^c$ for some constant c.

Show that Π is in ZPP. That is, convert M' into a randomized machine M that satisfies the definition of ZPP.

Hint: use Markov's inequality.

Answer: We will run M' for $2n^c$ steps. If it outputs a value by this point, we output that value; otherwise we output ?. Notice that if we output a value we are surely correct (because this is true for M'), this satisfies property (ii). Then what is the probability we output ?? Using Markov's inequality, $\Pr[t_{M(x,\cdot)} > 2n^c] \leq \frac{E[t_{M(x,\cdot)}]}{2n^c} \leq \frac{n^c}{2n^c} = \frac{1}{2}$. And we have satisfied property (i) above as well.

Problem 4 (3 points) Show that given a randomized machine M satisfying the definition of ZPP for solving a problem, we can reduce the error to less than $\frac{1}{2^n}$ while maintaining polynomial running time.

Answer: On input x, we run $M(x,\cdot)$ n times with independent random bits for each trial. If any trial outputs a value, we output a value. If all of the trials output ?, then we output ?. We know we never output an incorrect 0/1 value because M does not. What is the probability we output ?? That is the probability that n independent events all happen that each have probability at most 1/2, so the probability is at most $\frac{1}{2^n}$.

Problem 5 (3 points) Let M be a poly-time randomized machine that satisfies the definition of BPP for solving a problem. We want to replace the random bits of this algorithm by the output of a pseudorandom generator G such that the majority vote of running $M(x, G(s))$ for all possible seeds s is correct. How many bits does G need to output? What types of tests does G need to be pseudorandom against?

Answer: Suppose M runs in time n^c. Then it needs at most n^c random bits, and we can let G output n^c bits. G needs to be pseudorandom against tests where we fix some input x for M, and then run M using a true random string or using the output of G where its seed is taken at random.

Problem 6 (5 points) Suppose we roll a fair 6-sided dice 5 times, and assume all rolls are independent of each other. What is the expected value of the sum of the 5 rolls. What is the probability the sum is greater than 27?

Answer: The expected value of each roll is (1+2+3+4+5+6)/6 = 3.5. Using linearity of expectation, the expected value of the sum is 3.5*5 = 17.5.

For the probability the sum is greater than 27, let's count the number of ways this can happen. The total can be 30 in 1 way. The total can be 29 if 1 die is 5 and the rest 6, this happens in 5 ways. The total can be 28 if 1 die is 4 and the rest 6, which can happen in 5 ways, or if 2 dice are 5 and the rest 6, which can happen in (5 choose 2)=10 ways. So the total number of ways the sum can be greater than 27 is 1+5+5+10 =21. The total possible number of outcomes is 6^5, so the probability of rolling greater than 27 is $\frac{21}{6^5}$.