15-455: UCT **K. Sutner**

Assignment 9 Due: **April 25**, 2022.

1. Quadratic Residues (20)

Background

Quadratic residues play a key role in the Solovay-Strassen primality test. They are easy to explain when $\mathbb{Z}_n^*$ has a generator, and element g such that $\mathbb{Z}_n^* = \{ g^i \mid i \geq 0 \}$.

One can show that $\mathbb{Z}_n^*$ has a generator iff $n = 2, 4, p^e, 2p^e$ for some odd prime p. The general proofs are quite fumbly, we'll jus consider the case when the modulus is an odd prime p.

Task

A. Suppose $\mathbb{Z}_p^*$ has a generator. Describe quadratic residues and non-residues.

B. Determine and count the elements of order d in $\mathbb{Z}_p^*$.

C. Conclude that there must be a generator.

Comment

Euler's totient function will come in handy, make sure to recall its basic properties.

2. Wurzelbrunft SAT (20)

Background

Prof. Dr. Alois Wurzelbrunft is currently fascinated by randomized algorithms. He thinks that checking satisfiabilty of a CNF formula is best done by picking values for the variables at random. Say, $\varphi(x_1, x_2, \ldots, x_n)$ is the given formula. The algorithm is beautifully simple:

> Pick a random vector $\boldsymbol{b} \in \mathbf{2}^n$ and return $\varphi(\boldsymbol{b})$.

Wurzelbrunft is vaguely aware that there is a minor problem: the algorithm will produce false negatives: the formula is satisfiable, but we get No instead. He thinks that can be handled by repeating the method r times for small r.

Task

A. Show that the probability of a false negative may be as high as $1 - 2^{-n}$.

B. Suppse we repeat the algorithm r times, independently. Estimate the improved probability of false negatives.

C. What professional advice can you give Wurzelbrunft? Explain.

Comment

For part (B), use the fact that for small δ we have $(1 - \delta)^r \approx e^{-r\delta}$.

3. ZPP (40)

Background

Ordinarily Turing machines for decision problems are required to return yes or no. Let's generalize slightly to allow for an additional output ¿, meaning "don't know." We'll call these machines ambiguous, think of them as having three different halting states. We are interested in probabilistic ambiguous machines $\mathcal{M}$ that run in polynomial time and satisfy

$$\Pr[\mathcal{M}(x) \in \{L(x), ¿\}] = 1$$
$$\Pr[\mathcal{M}(x) = ¿] \leq 1/2$$

So they never give a wrong answer but may return ¿. As usual we identify a language L with its characteristic function.

Task

A. Show that ZPP is the set of languages accepted by a ¿-machine.

B. Show that $\mathsf{ZPP} = \mathsf{RP} \cap \text{co-}\mathsf{RP}$.

Comment

You can use part (A) for (B), but a direct proof is also possible.

4. Closure (40)

Background

As usual in the study of complexity classes, one tries to establish closure properties of probabilistic classes with respect to Boolean operations. Here are some more exotic cases.

Task

A. Show that BPP is closed under polynomial time reductions.

B. Show that BPP is closed under Kleene star.

Comment

For part (A), you can use the following Chernoff variant: let $p = 1/2 + \delta$ and $X = \sum_{i \leq r} X_i$, Then

$$\mathsf{Pr}[X \leq r/2] \leq e^{-\delta^2 r/2}$$

.

For part (B) it might help to first give a *simple* graph based proof that $\mathbb{P}$ is closed under Kleene star; it's just easier to write down than the usual dynamic programming argument.

5. What If? (40)

Background

It appears that $\mathbb{P} = \mathsf{BPP}$ is a reasonable conjecture, in which case we would expect $\mathbb{NP}$ to be strictly larger. Here is a corresponding "what if" szenario that explores a consequence of the assumption that $\mathbb{NP}$ is actually contained in BPP.

Task

A. Show that $\mathbb{NP} \subseteq \mathsf{BPP}$ implies that $\mathbb{NP} = \mathsf{RP}$.

B. How plausible is this consequence?

Comment You may safely assume that one can encode Boolean formulae in a way so that substituting truth values for variables does not change the size of the formula.