Q1 Read me first

0 Points

- Tests show that the people who get the most out of this assignment are those who read the "read me first" like this one.
- Collaboration and use of external sources are permitted, but must be fully acknowledged and cited. For your own learning, you are advised to work individually. Collaboration may involve only discussion; all the writing must be done individually.
- Please acknowledge, **individually for every problem** at the beginning of each solution, a list of all collaborators and sources consulted other than the course notes. Examples include: names of people you discussed homework with, books, other notes, Wikipedia and other websites. If no additional sources are consulted, you must write "sources consulted: none" or equivalent. **Failure to acknowledge sources will lead to an automatic 1pt penalty.**
- Late policy: In general **no late homework** will be accepted unless there is a genuine emergency backed up by official documents.
- All steps should be justified.
- You are encouraged to be **type in LaTeX**. To learn how to use LaTeX, I recommend the tutorials on Overleaf. It is ok to draw diagrams by hand and insert them as pictures in your TeX files.
- For each question below, upload a PDF file and/or type in the box (see Gradescope x LaTeX tutorial). Each submission should contain (1) the acknowledgement of all collaborators and sources consulted and (2) your solution.

Q2 Simulating a biased coin

6 Points

In class, we obtained a probabilistic computation model by taking a standard model of deterministic computation (say your favorite programming language) and add a new "coin flip" operation, which returns 0 or 1 equally likely.

(a) Prove that there is a simple subroutine (written in pseudocode using only deterministic computation and "coin flip" operations) that simulates a biased coin, which returns 0 with probability 1/4 and 1 with probability 3/4.

(b) Prove that for any $\varepsilon > 0$ there is a subroutine that almost simulates a biased coin, which returns a value

$$r \in \{0, 1, \mathrm{FAILURE}\}$$

such that $\Pr(r = \mathrm{FAILURE}) \leq \varepsilon$, and the conditional probability $\Pr(r = 0 \mid r \neq \mathrm{FAILURE}) = 1/3$.

Remark: A subroutine is a program that always halts in $O(1)$ steps. In (b) this $O(1)$ could depend on ε.

Sources consulted:

None

Solution:

▾ Q2.pdf Download

1 / 2

Collaborators : None

Sources : None

Q2 Simulating a biased

2-a) Prove that there is a simple su
computation and "coin flip" operation
probability 1/4 and 1 with probability 3

To simulate a biased coin that return
use the following subroutine:

BiasedCoin():

Q3 Gates for universal classical computation

8 Points

(a) Show that any Boolean function $f : \{0,1\}^n \to \{0,1\}$ can be computed by a classical Boolean circuit using the following set of logic gates: 2-bit AND, 2-bit OR, and NOT. (Hint: look up DNF formula.)

(b) Show that any Boolean function $f : \{0,1\}^n \to \{0,1\}$ can be computed by a classical Boolean circuit using the following single logic gate: 2-bit NAND. Also, show this for the following single logic gate: 2-bit NOR.

(c) Show that there are infinitely many Boolean functions $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by a classical Boolean circuit using the following set of logic gates: 2-bit XOR, and NOT.

Sources consulted:

1) https://en.wikipedia.org/wiki/Disjunctive_normal_form ; 2) https://mathworld.wolfram.com/DisjunctiveNormalForm.html

Solution:

Q3.pdf Download

1 / 2

Collaborators : None

Sources :

1) https://en.wikipedia.org/wiki/Disjunctive_nor
2) https://mathworld.wolfram.com/DisjunctiveN

Q3 Gates for universal classical

3-a)

Disjunctive Normal Form (DNF

A DNF formula is a standardized way to rep
(OR) of conjunctions (AND) of literals (varia
function can be expressed in DNF

Q4 Dealing with error in randomized computation

8 Points

Suppose you are trying to write a computer program C to compute a certain Boolean function $f : \{0,1\}^n \to \{0,1\}$, mapping n bits to 1 bit. (For example, perhaps f specifies that $f(x) = 1$ if and only if x represents a prime number written in base 2.) If C is a deterministic algorithm, then there is an obvious definition for "C successfully computes f"; namely, it should be that $C(x) = f(x)$ for all inputs $x \in \{0,1\}^n$. But what if C is a probabilistic algorithm?

The best thing is if C is a zero-error algorithm for f, with failure probability p. This means:

- on every input x, the output of $C(x)$ is either $f(x)$ or is "?"
- on every input x we have $\Pr[C(x) = ?] \leq p$

Important note: The second condition is not about what happens for a *random input* x. Instead, it demands that for *every* input x the probability of failure is at most p, where the probability is only over the internal "coin flips" of C.

(a) If you have a zero-error algorithm C for f with failure probability 90% (quite high!), show how to convert it to a zero-error algorithm C' for f with failure probability at most 2^{-500}. The "slowdown" should only be a factor of a few thousand.

(b) Alternatively, show how to convert C to an algorithm C'' for f which: (i) always outputs the correct answer, meaning $C''(x) = f(x)$; (ii) has expected running time only a few powers of 2 worse than that of C. (Hint: look up the mean of a *geometric random variable*.)

The second best thing is if C is a one-sided error algorithm for f, with failure probability p. There are two kinds of such algorithms, "no-false-positives" and "no-false-negatives". For simplicity, let's just consider "no false-negatives" (the other case is symmetric); this means:

- on every input x, the output $C(x)$ is either 0 or 1

- on every input x such that $f(x) = 1$, the output $C(x)$ is also 1
- on every input x such that $f(x) = 0$, we have $\Pr[C(x) = 1] \le p$

(c) If you have a no-false-negatives algorithm C for f with failure probability 90% (quite high!), show how to convert it to a no-false-negatives algorithm C' for f with failure probability at most 2^{-500}. The "slowdown" should only be a factor of a few thousand.

The third best thing (in fact, the worst thing, but it's still not so bad) is if C is a two-sided error algorithm for f, with failure probability p. This means:

- on every input x, the output $C(x)$ is either 0 or 1
- on every input x we have $\Pr[C(x) \neq f(x)] \le p$

Remark: It is actually very very rare in practice for a probabilistic algorithm to have two-sided error; in almost every natural case, an algorithm you design will have one-sided error at worst.

(d) If you have a two-sided error algorithm C for f with failure probability 40%, show how to convert it to a two-sided error algorithm C' for f with failure probability at most 2^{-500}. The "slowdown" should only be a factor of a few dozen thousand. (Hint: look up the Chernoff bound.)

Sources consulted:

https://www.perplexity.ai/

Solution:

▾ Q4.pdf Download

1 / 3

Collaborators : None

Sources :

1) https://www.perplexity.ai/

Q4 Dealing with error in randomi

4-a) If you have a zero-error algorithm C for f witl
how to convert it to a zero-error algorithm C' for f w
"slowdown" should only be a factor of a few thousa

Algorithm for C':

1. Run C repeatedly for k times.

Q5 Dirac notation and measurement exercises

10 Points

(a) Let $|\phi\rangle = 3|0\rangle - 5i|1\rangle$. What is $\langle\phi|\phi\rangle$?

(b) What number, C, should $|\phi\rangle$ be divided by to make it a “normalized” state; i.e., a unit vector? For future reference, define $|\psi\rangle = C^{-1}|\phi\rangle$ to be this state vector.

(c) What are the possible outcomes and associated probabilities if $|\psi\rangle$ is measured in the standard $\{|0\rangle, |1\rangle\}$ basis?

(d) Same question as above for measuring in the $\{|+\rangle, |-\rangle\}$ basis.

(e) Verify that $\frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle$ and $\frac{1}{\sqrt{2}}|0\rangle - \frac{i}{\sqrt{2}}|1\rangle$ form an orthonormal basis for $\mathbb{C}^2$. (These two vectors are sometimes called $|i\rangle$ and $|-i\rangle$.) Then do the prior question for measuring in the $\{|i\rangle, |-i\rangle\}$ basis.

Sources consulted:

Lecture Notes

Solution:

Q5.pdf Download

1 / 2

Collaborators : None

Sources : Lecture Notes

Q5 Dirac notation and measurem

5-a)

Let $|\phi\rangle = 3|0\rangle - 5i|1\rangle$

Then, $\langle\phi|\phi\rangle = 3*3 + (-5i)*(-5i) = 9 - 25$

5-b)

What number, C, should $|\phi\rangle$ be divided by to make

Assignment 1

Graded

Select each question to review feedback and grading details.

Student

Sujith Potineni

Total Points

29 / 32 pts

Question 1

Read me first **0** / 0 pts

Question 2

Simulating a biased coin **6** / 6 pts

Question 3

Gates for universal classical computation **5** / 8 pts

Question 4

Dealing with error in randomized computation **8** / 8 pts

Question 5

Dirac notation and measurement exercises **10** / 10 pts