Design and Analysis of Algorithms
Massachusetts Institute of Technology — 6.046J/18.410J
Debayan Gupta, Aleksander Madry and Bruce Tidor — Extra Practice Problems for Quiz 2

Extra Practice Problems for Quiz 2

Problem-1: True/False Questions

(a) A Nash Equilibrium is a state of a game such that a player can improve his or her expected utility by changing his/her strategy if and only if every other player can also improve his/her expected utility by changing his/her strategy.

(b) The matrix [[(1,-1), (-1, 1)], [(-1, 1), (1, -1)] has no pure-strategy or mixed-strategy Nash equilibria.

(c) Given a 2 player zero sum game, if player A declares a deterministic strategy, and then player B responds with a deterministic strategy, the payoffs will be the same as if player B declares a deterministic strategy, and then player A responds with a deterministic strategy.

(d) Consider the 2-player game where each player has 3 possible strategies and has the following payoff matrix:

Player B \ Player A	1	2	3
a	0 0	+1 -1	-2 +2
b	-1 +1	0 0	+1 -1
c	+2 -2	-1 +1	0 0

The mixed strategy where

i. A chooses one of the strategies 1, 2, 3 with probability 1/3 each.
ii. B chooses one of the strategies a, b, c with probability 1/3 each.

is a Nash equilibrium of the described game.

Problem-2: Monte Carlo Advantage

Alyssa P. Hacker is a space scientist at SpaceX where she works with various collections of space rocks. Her task is to figure out whether or not a given space rock contains any evidence of organic material. Unfortunately, the state of the art technology for detecting organic material in space rocks is not entirely reliable. This device is a bulky box where Alyssa can place a rock and get an output of whether or not the rock contains organic material.

The box has the following properties. If the rock contains organic material, it outputs YES with certainty. However if the rock does not contain organic material, it only outputs NO with probability $1/5$. Alyssa was discouraged at first, given that the box is not even reliable enough for a Monte-Carlo algorithm. But she quickly realized the box was actually much more capable than she thought.

(a) Using Alyssa's box, give a Monte-Carlo algorithm for detecting whether a rock contains organic material. Your algorithm should be correct with probability strictly greater than $\frac{1}{2}$. For full credit, your algorithm should make $O(1)$ queries to the box.

(b) Alyssa wants to feel confident about her findings before she presents a report to NASA. Using k queries to the box, give an algorithm which allows her to detect organic material with a probability which depends on k. i.e. the success probability increases with k.

Problem-3: Broadcast Channel

A set of up to n processors attempt to communicate over a network. The communication process is deemed successful if *any* of the processors manages to broadcast its information (since the successful processor can then lead the remainder of the communication process). However, the only means of communication is through a common broadcast channel. At any given time step (we assume the time is discrete), any subset of the processors can attempt to communicate through the channel by sending a message. The channel operates as follows:

- If *none* of the processors attempts to send a message, then all processors receive a special "none" message.
- If *only one* of the processors attempts to send a message, then all the processors receive that message, and the communication process is deemed successful.
- If *two or more* processors attempt to send a message, then all the processors receive a special "collision" message.

Suppose that the number of processors is at least $\frac{n}{2}$. Design a randomized protocol that, if followed by all processors, will result in successful communication. The expected number of time steps used by the protocol should be $O(1)$. You may assume all processors know the upper bound n and the lower bound $\frac{n}{2}$.

Problem-4: Testing Polynomial Products

An instance of the *Testing Polynomial Products (TPC)* decision problem is a quadruple of univariate polynomials (A, B, C, D). We say that (A, B, C, D) is a yes-instance if $AB = CD$, and a no-instance otherwise.

Consider the following probabilistic algorithm for solving the problem. On input polynomials (A, B, C, D): Choose uniformly at random an integer y from a set of integers $\{1, \ldots, m\}$. Compute the product $L = A(y)B(y)$ and the product $R = C(y)D(y)$ If $R = L$ outputs "PASS", otherwise output "FAIL".

(a) Analyze the probability that the algorithm is correct as a function of m and n (where the degree of any of the polynomials is at most n). Namely, it passes yes-instances and fails no-instances.

(b) How large should we make m to make the probability of correctness at least $1 - \frac{1}{\log n}$.

Problem-5: Streaming

(a) You are given an input stream $x_1, x_2, x_3, \ldots, x_n$ where each x_i is an n bit integer. You want to compute the sum of the pairwise products of the integers, but since space is limited, you are only allowed to use $O(n)$ bits of storage.

Design a deterministic streaming algorithm that uses $O(n)$ bits of storage and outputs the sum of pairwise products, i.e. $\sum_{1 \le i < j \le n} x_i x_j$.

Example: Given the input stream $[2, 3, 1, 4]$, the output should be:

$$(2 \times 3) + (2 \times 1) + (2 \times 4) + (3 \times 1) + (3 \times 4) + (1 \times 4) = 35$$

(b) Sarah has an $n \times n$ matrix A, and she wants to find the inverse A^{-1} (recall: $A^{-1}A = I$). She finds code on the Internet that claims it does the job. She runs the code on A and gets the output B. However, Sarah is not sure that the code is correct. Thus, she decides to test whether $B = A^{-1}$.

Provide a Monte Carlo algorithm that runs in $O(n^2)$ time for this job and outputs the correct answer with probability at least $3/4$. Prove the correctness of your algorithm.

Problem-6: Graph Completeness

Alice is given a very large undirected graph $G = (V, E)$ with $n = |V|$. We say that a graph is complete if it contains an edge between all $\binom{n}{2}$ pairs of vertices. We say that a graph is ϵ-far from complete if it is missing at least $\epsilon\binom{n}{2}$ edges in order to make it complete.

Alice is given query access to G. Namely, for any u and v vertices in V, Alice can issue queries (u, v) to find out whether $(u, v) \in E$ or $(u, v) \notin E$.

(a) **[14 points]** Help Alice design a Property Testing algorithm with parameter ϵ which makes $O(1/\epsilon)$ queries to G and satisfies the following:

- If G is complete, the algorithm outputs ACCEPT with probability 1.
- If G is ϵ-far from being complete, the algorithm outputs REJECT with probability at least 2/3.

2**Note:** The following inequality may be helpful: $(1 + x) \le e^x$ for any x.

(b) **[6 points]** Modify your algorithm from part (a) to reduce the error probability $1/3$ to less than $1/\sqrt{n}$ while keeping the number of queries sublinear.

Note: You can do this part without solving part (a).

Problem-7: [20 points] $\frac{3}{4}$-SAT

In the $\frac{3}{4}$-SAT problem, you are given n boolean variables x_i which can be set to TRUE or FALSE. You are also given m clauses C_j composed of a logical OR of three "literals," i.e. a variable x_i or its negation $\neg x_i$. A clause is "satisfied" if it evaluates to TRUE. A "$\frac{3}{4}$-satisfying assignment" is an assignment of values to variables such that at least $\frac{3m}{4}$ clauses are satisfied.

Below is an example of a clause and a table of assignments that satisfy the clause.

$$C_j = (x_a \vee \neg x_b \vee x_c)$$

x_a	x_b	x_c
1	0	0
1	0	1
1	1	0
1	1	1
0	0	0
0	0	1
0	1	1

(a) **[5 points]** Design an $O(n)$ time Monte-Carlo algorithm that satisfies $7m/8$ clauses in expectation. You may assume clauses do not have duplicate variables (a literal or its negation).

Hint: Try a random assignment.

(b) **[10 points]** Show that your algorithm satisfies at least $3m/4$ clauses with probability at least 1/2.

Hint: Use Markov Bound.

(c) **[5 points]** Use parts (a) and (b) to design a Monte Carlo randomized algorithm that runs in $O((m+n)\log n)$ time and outputs a correct $\frac{3}{4}$-satisfying assignment with probability at least $1 - \frac{1}{n^2}$.

Note: You may assume the results from parts (a) and (b).