Discrete Structures (CSCI 246)

Homework 10

Purpose & Goals

The following problems provide practice relating to:

- probability definitions,
-
- the problem solving process.

Submission Requirements

- **Type or clearly hand-write your solutions** into a pdf format so that they are legible and professional. Submit your pdf to Gradescope. **Illegible, non-pdf, or emailed solutions will not be graded.**
- Each problem should start on a new page of the document. When you submit to Gradescope, associate each page of your submission with the correct problem number. Please post in Discord if you are having any trouble using Gradescope.
- Try to model your formatting off of the proofs from lecture and/or the textbook.
- Submit to Gradescope early and often so that last-minute technical problems don't cause you any issues. Only the latest submission is kept. Per the syllabus, assignments submitted within 24 hours of the due date will receive a 25% penalty and assignments submitted within 48 hours will receive a 50% penalty. After that, no points are possible.

Academic Integrity

- You may work with your peers, but **you must construct your solutions in your own words on your own.**
- Do not search the web for solutions or hints, post the problem set, or otherwise violate the course collaboration policy or the MSU student code of conduct.
- Violations (regardless of intent) will be reported to the Dean of Students, per the MSU student code of conduct, and you will receive a 0 on the assignment.

Tips

- Answer each problem to the best of your ability. Partial credit is your friend!
- Read the hints for where to find relevant examples for each problem.
- Refer to the problem solving and homework tips guide.
- It is not a badge of honor to say that you spent 5 hours on a single problem or 15 hours on a single assignment. Use your time wisely and get help (see "How to Get Help" below).

How to Get Help

When you are stuck and need a little or big push, **please ask for help!**

- Timebox your effort for each problem so that you don't spend your life on the problem sets. (See the problem solving tips guide for how to do this effectively.)
- Post in Discord. If you're following the timebox guide, you can post the exact statement that you produced after spending 20 minutes being stuck.
- Come to office hours or visit the CS Student Success Center.

Problem 1 (13 points)

One of the most important data structures in computer science is a *hash table*. In this problem, you will learn about hash tables and prove some of their properties. You will learn everything you need to know about hashing and hash tables in this problem, so it's okay if you haven't learned about them before. However, you may need to review functions and The Pigeonhole Principle for this problem. Make sure that you do both part (a) and part (b).

(a) (8 points) Given a set U of possible values and a set $S \subseteq U$ of values from U, a hash table allows us to quickly answer the question "is x in S?" without needing to examine every element of S. A hash table is defined as follows:

- Let $T[1 \ldots n]$ be a table with n cells.
- Let $h : U \to \{1, 2, \ldots, n\}$ be a function (called a *hash function*).
- Each element $x \in S$ is stored in the cell $T[h(x)]$.

To check that a new value $y \in U$ is in S, we can compute $h(y)$, look at the cell $T[h(y)]$, and see if y is there.

Consider the set $U = \{x \in \mathbb{Z} : -100 \leq x < 100\}$ and a table $T[1 \ldots 100]$.

Use the PHP to prove that *any* well-defined function $h : U \to \{1, 2, \ldots, 100\}$ will have at least one collision (where two elements from U are stored in the same cell of T).

Hint: When using the Pigeonhole Principle, always

- clearly define your set A (of pigeons),
- clearly define your set B (of pigeonholes),
- clearly define the function $f : A \to B$ that maps each pigeon $a \in A$ to a single pigeonhole $f(a)$ so that $f(a) \in B$ (i.e., f has the three properties of a well-defined function), and
- explain how you're able to apply the Pigeonhole Principle to obtain the desired result.

Grading Notes. While a detailed rubric cannot be provided in advance as it gives away solution details, the following is a general idea of how points are distributed for this problem. We give partial credit where we can.

(7) **Correctness.** If your proof is not correct, this is where you'll get docked. You'll need to
  (1) define the pigeons (or set A),
  (1) define the pigeonholes (or set B),
  (1) define the function $f : A \to B$,
  (1) briefly explain why f is well-defined,
  (1) explain why $|A| > |B|$,
  (1) correctly apply the PHP,
  (1) explain how the resulty of the PHP achieves the desired result of the problem.

(1) **Communication.** We need to see a mix of notation and intuition. If you skip too many steps at once, or we cannot follow your proof, or if your proof is overly wordy or confusing, this is where you'll get docked.

(b) One way to resolve collisions in a hash table is *linear probing.* When we insert an element x into the table, we put x in the first unoccupied cell, moving left to right, starting at cell $h(x)$. For example, suppose T is initially empty. We map x to $h(x) = 1$ and store x in $T(1)$. Then, we map y, and it happens that $h(y) = 1$ as well. Since $T(1)$ is occupied, we check the next cell, and since it is unoccupied, we store y in $T(2)$. Note that if we reach the end of the table as we are looking for unoccupied slots, we circle back to the front again.

Suppose we hash 2 elements into a hash table with 5 slots using a hash function h that puts elements into slots of the table with equal probability.

(a) (3 points) Draw a tree diagram to represent all outcomes of the probabilistic process of putting 2 elements into the table.

(b) (1 point) What is the probability that the two elements end up in adjacent cells (including the first and last)?

(c) (1 point) What is the probability that the two elements end up in non-adjacent cells (counting the first and last as adjacent)?

Problem 2 (12 points)

You are dealt a 5-card hand from a standard 52-card deck. Define a *pair* as any two cards with the same rank—so a hand consisting of an ace of clubs, ace of hearts, ace of diamonds, two of diamonds, and three of diamonds has three pairs (ace of clubs and ace of hearts, ace of clubs and ace of diamonds, ace of hearts and ace of diamonds). Let P denote the number of pairs in your hand.

(a) (2 point) Let $m \in \mathbb{Z}^{\geq 0}$ denote the maximum possible number of pairs in your hand. What is the value of m?

(b) (7 points) For each of $i \in \{0, 1, \ldots, m\}$, compute $P[P = i]$—that is, the probability of having exactly i pairs in your hand.

(c) (3 points) Compute $E[P]$, the expected number of pairs in your hand.