

CS 473 (Fall 2026)

Homework 4 (due Oct 8 Thu 10am)

Instructions: Carefully read <http://courses.grainger.illinois.edu/cs473/fa2026/policies.html#hw> and <http://courses.grainger.illinois.edu/cs473/fa2026/integrity.html> (in particular, remember to include the 3 disclosure statements, on collaborators, outside sources, and AI use, for each problem).

Problem 4.1: Given a set S of n points and a set P of m points (the “pattern”) in 2D (with $m \leq n$), a basic question in pattern recognition is to determine whether the pattern “occurs” in the dataset S . Due to the possibility of noises in the data, an exact match is not required and we are satisfied if, say, 90% of the pattern points are matched.

Here is one precise formulation of the problem that you will study: determine whether there exists a translation of P so that at least $0.9m$ of the translated points in P are in S .

(You may assume that S is already stored in a data structure that can test whether any given point is in S in constant time, by hashing.)

- (a) (20 pts) Describe a deterministic algorithm that solves this problem in $O(mn^2)$ time. [Hint: try all pairs $p \in P$ and $s \in S$ and consider the translation that maps p to s ...]
- (b) (40 pts) Give a Monte-Carlo algorithm that runs in $O(n^2)$ time with error probability at most 0.001. [Hint: choose a point $p \in P$ at random...]
- (c) (40 pts) Give a Monte-Carlo algorithm that runs in $O(n)$ time, with error probability at most 0.001, under the assumption that no two pairs of points in S have the same exact distance. [Hint: choose two points $p, q \in P$ at random...]

Problem 4.2: Consider the following variant of the string matching problem: Given a string $a = a_1a_2 \cdots a_n \in \Sigma^*$ and a “pattern” string $b = b_1b_2 \cdots b_m \in \Sigma^*$ over an alphabet Σ (with $m \leq n$), we want to determine whether it is possible to insert one character to b so that the modified string b' appears as a substring in a .

For example: for $a = \text{“randomizedalgorithms”}$ and $b = \text{“zeal”}$ over the English alphabet, the answer is yes (by adding the single character ‘d’ to get “zedal”).

Assuming Σ has constant size, describe a randomized Las Vegas algorithm for this problem that runs in $O(n)$ expected time, by using the Karp–Rabin fingerprint technique.

[Hint: first generate the fingerprints of all strings b' that can be obtained from b by inserting one character... You may use constant-time hashing.]

Optional Problem 4.3 (not graded): In a popular form of logic puzzles, you land on an island that has three types of inhabitants: “knights”, who always tell the truth; “knaves”, who always lie; and “spies”, who sometimes lie and sometimes tell the truth.

Suppose there are n inhabitants, where 60% are known to be decent folks, i.e., knights. The remaining 40% are bad, i.e., knaves or spies. You want to know who are the good/bad guys, i.e., determine the types of all n inhabitants. You are allowed to ask only questions of the form, “is person A a knight/knave/spy?”, to another person B. (All the inhabitants know each other.) Obviously, if you can find a person who you know is a knight, the problem can be solved after asking n additional questions.

- (a) (10 pts) Give a (very) efficient Monte Carlo algorithm that finds a knight. State the probability of error. (Hint: this is supposed to be very easy!)
- (b) (90 pts) Give a Las Vegas algorithm that finds a knight by asking $O(n)$ expected number of questions. Analyze the constant factor in the big-Oh and make it smaller than 1.5. (Hint: use (a). How can you confirm whether a specific person is a knight by asking $O(n)$ questions?)
(Note: there is also a deterministic algorithm that requires $O(n)$ questions, but it is more complicated and has a larger constant.)