COMS 4232: Advanced Algorithms Jan 18, 2022

Self-Evaluation Test

Instructor: *Alex Andoni*

This is a self-evaluation test for you to confirm that you have the sufficient background for the class, and identify potential parts to brush up before the class. You are expected to understand and solve all problems on this test. You do not have to turn in solutions.

The main prerequisite for the class is mathematical maturity, i.e., being able to follow and write rigorous mathematical proofs, of both combinatorial and analytical flavors. In terms of concrete topics, you should be comfortable with linear algebra, minimal probability theory, some basic algorithmic notions. As a result, the problems below are partitioned by category.

Questions marked with $\star$ are a bit harder — they will be covered in the class (or at the level) but briefly and hence prior understanding would be very much helpful.

1 Math

1.1 Norms

For a vector $x \in \mathbb{R}^n$, and an integer $p \in (0, \infty]$, the p-norm of x is defined as $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$.[1] Prove the following:

- $\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\|x\|_2$, using Cauchy-Schwartz if needed. For which vectors x is each of the inequalities tight?
- when $p = \log_2 n$, $\|x\|_p$ is a 2 approximation to $\|x\|_\infty$ (i.e., $\|x\|_\infty \leq \|x\|_p \leq 2\|x\|_\infty$).
- $\star$ In the spirit of above, how can you relate the 10-norm of x to the 5-norm of x?

1.2 Runtime of Merge-Sort

Merge-Sort is an algorithm for sorting n numbers, which proceeds as follows. Suppose the input is an array $A[1 \dots n]$ of length n (which, for simplicity, assume is a power of 2). We partition the array into two, $A[1 \dots n/2]$ and $A[n/2+1 \dots n]$. We sort each part recursively and then merge the 2 (sorted) parts. It is known that the latter merge operation can be accomplished in time $O(n)$.[2] The base case of the Merge-Sort is when $n = 1$, in which case we do nothing (the array is already sorted).

Here, you are to determine whether the following statement/proof is correct or not (why or why not).

Claim 1. *The runtime of the above MergeSort algorithm is $O(n)$.*

[1] For $p = \infty$, $\|x\|_\infty$ is formally defined to be $\max_{i \in [n]} |x_i|$, which can be seen as the limit as $p \to \infty$.

[2] Remember that $O(f(n))$ is the set of functions $g : \mathbb{N} \to \mathbb{N}$ such that there exist $c > 0$ and $n_0 \in \mathbb{N}$ such that, for $n \geq n_0$, $g(n) \leq c \cdot f(n)$.

Proof. Let $T(n)$ be the runtime of Merge-Sort on an array of length n. Then, by the above description we have that:

$$T(n) = 2 \cdot T(n/2) + O(n),$$

corresponding to the two recursive calls and the merge operation respectively.

We prove the claim that $T(n) = O(n)$ by induction. Indeed, we have that:

$$T(n) = 2 \cdot T(n/2) + O(n) = 2 \cdot O(n/2) + O(n) = O(n).$$

This completes the proof of the claim. □

2 Linear algebra

2.1 Quadratic forms

Let A be a n-by-n symmetric matrix whose maximum eigenvalue is 2022. What is the minimum and maximum possible value for the quadratic form $x^T Ax$, where $x \in \mathbb{R}^n$ are unit-norm vectors? What about $x^T A^2 x$?

2.2 Linear systems

Suppose A is a n-by-n real matrix and $b \in \mathbb{R}^n$. For the unknown vector $x \in \mathbb{R}^n$, describe what are the possible sets of solutions for the following:

- $Ax = b$;
- $\star$ $\operatorname{argmin}_{x \in \mathbb{R}^n} \|Ax - b\|_2$.

To give a better idea of what's asked, consider the "$Ax = b$" case when $n = 1$ (i.e., just a simple equation). Here's the answer for this case:

The possible sets of solutions are: 1) there may not be a solution (empty set), 2) may be exactly one solution, or 3) any $x \in \mathbb{R}$ is a solution.

The above 3 cases correspond to when 1) $A = 0$, $b \neq 0$, 2) $A \neq 0$, 3) $A = b = 0$.

3 Probability

3.1 Hashing

A function $h : U \to [m]$, where U is a discrete set (e.g., integers of $O(\log n)$ bits), and $m \in \mathbb{N}$, is called a hash function with m buckets. (One can see this as assigning one of the m buckets to each possible item from U.) Most common hash function is a random hash function: each $h(i)$, $i \in U$, is chosen iid at random from $[m]$.

Below, consider a set $S \subset U$ of size n.

- Let h be a random hash function with $m = n$. What is the expected size of a bucket, i.e., $|h^{-1}(i) \cap S|$ for any $i \in [m]$?

- A collision is an event where distinct $x, y \in S$ satisfy $h(x) = h(y)$. What is the expected number of collisions among elements of S under h? (To give an example, if one has precisely three distinct items $x, y, z \in S$ falling into the same bucket, they generate 3 collisions: (x, y), (y, z), and (z, x)).

A family $\mathcal{H}$ of hash functions h is k-wise indepedent if for any distinct $a_1, \ldots a_k \in U$ and (not necessarily distinct) $q_1, \ldots q_k \in [m]$, we have that

$$\Pr_{h \in \mathcal{H}}[h(a_1) = q_1 \wedge \ldots \wedge h(a_k) = q_k] = 1/m^k.$$

Note that the aforementioned random hash function is $|U|$-wise independent (or, to be more precise, the family $\mathcal{H}$ of *all* functions $h : U \to [m]$ is $|U|$-wise independent).

- $\star$ How do the estimates from above change if our hash function h is chosen from some 2-wise independent hash family $\mathcal{H}_2$?

4 Algorithms

4.1 Graphs

A graph G has n nodes and m edges. Suppose that $m = n/2$ (assuming n is even). What is the smallest and the largest number of connected components that G can have?

4.2 $\star$ Alice tells Bob

Consider Alice holds a number n and wants to communite it to Bob *approximately* using minimal possible communication. In particular, Bob should be able to output some n' satisfying $n \leq n' \leq 2n$. How many bits does Alice need to send to Bob to achieve this goal? (Answer up to $O(\cdot)$ is enough.)

Does the answer change if it's a "dialogue" — Alice and Bob sequentially communicate to each other a number of bits (think of it as if Bob can "ask questions") — and the "total communication" is the total number of bits exchanged ?