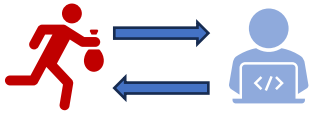
Robust Data Structures for Searching under Adaptive Queries

Alexandr Andoni (Columbia), Themistoklis Haris (BU), Esty Kelman (BU, MIT) , Krzysztof Onak (BU)

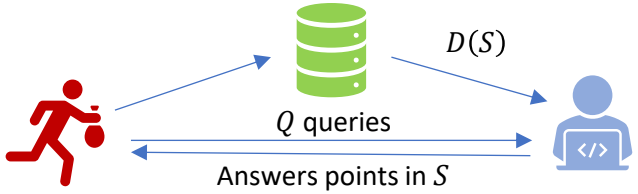
Author Info

How do we robustly and efficiently answer "search" queries, where the output should be a point from a public dataset?



Adversarial Robustness

- Randomized Algorithms can fail to correctly solve a problem, albeit with small probability over their internal randomness.
- An *adaptive* adversary can probe such an algorithm many times and break it.

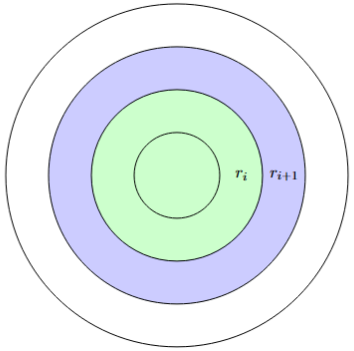


Search Problems

Given a dataset S produced by an adversary, create a data structure $D(S)$ which can efficiently answer adaptive queries about S by returning points from it.

Why is this hard?

- Existing approaches focus on estimation problems.
- Outputting a point from the dataset each time is a severe restriction. Obfuscation techniques such as differential privacy don't work!



Adversarially Robust ANN algorithm using $\tilde{O}(\sqrt{Q} \cdot n^{1.01+\rho})$ bits of space and $\tilde{O}(n^{\rho+0.01})$.

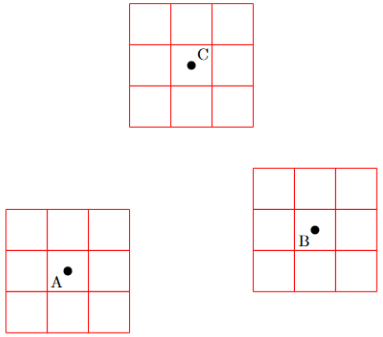
Theorem: Let A be an oblivious search algorithm that uses $O(n^s)$ bits of space and answers each query in $t(n)$ time. Then, there exists an adversarially robust algorithm A' for Q queries that uses $\tilde{O}(\sqrt{Q} \cdot n^s)$ bits of space and has a query time of $\tilde{O}(t(n))$.

Our Approach

1. *A robust decider algorithm A_{dec}:* This algorithm decides whether the query *can* be answered with some point from the dataset of not. We can create this algorithm by using multiple copies of A and combining their outputs using Differential Privacy.
2. Create a *binary tree of robust deciders:* This tree detects and outputs a valid query response.

Applications

1. Nearest Neighbor Search: we give a concentric circle LSH construction which also gives runtime improvements for the *fair ANN* problem.
2. Bloom Filters via *Online Set Intersection:* We improve the space complexity of this problem to $O(\sqrt{Q} \cdot n \cdot \log(Qt))$ from $O(n \log(Qt) + Q)$.



Data-based ℓ_p metric covering

Bonus!

Improved *for-all* ℓ_p ANN algorithms that are correct on all queries with high probability via the use of *metric coverings.*