1 Lecture 18, Scribe: Giorgos Zirdelis

In this lecture we study lower bounds on data structures. First, we define the setting. We have \( n \) bits of data, stored in \( s \) bits of memory (the data structure) and want to answer \( m \) queries about the data. Each query is answered with \( d \) probes. There are two types of probes:

- **bit-probe** which return one bit from the memory, and
- **cell-probe** in which the memory is divided into cells of \( \log n \) bits, and each probe returns one cell.

The queries can be adaptive or non-adaptive. In the adaptive case, the data structure probes locations which may depend on the answer to previous probes. For bit-probes it means that we answer a query with depth-\( d \) decision trees.

Finally, there are two types of data structure problems:

- The **static** case, in which we map the data to the memory arbitrarily and afterwards the memory remains unchanged.
- The **dynamic** case, in which we have update queries that change the memory and also run in bounded time.

In this lecture we focus on the non-adaptive, bit-probe, and static setting. Some trivial extremes for this setting are the following. Any problem (i.e., collection of queries) admits data structures with the following parameters:

- \( s = m \) and \( d = 1 \), i.e. you write down all the answers, and
- \( s = n \) and \( d = n \), i.e. you can always answer a query about the data if you read the entire data.

Next, we review the best current lower bound, a bound proved in the 80’s by Siegel [Sie04] and rediscovered later. We state and prove the lower bound in a different way. The lower bound is for the problem of \( k \)-wise independence.
**Problem 1.** The data is a seed of size \( n = k \log m \) for a \( k \)-wise independent distribution over \( \{0, 1\}^m \). A query \( i \) is defined to be the \( i \)-th bit of the sample.

The question is: if we allow a little more space than seed length, can we compute such distributions fast?

**Theorem 2.** For the above problem with \( k = m^{1/3} \) it holds that

\[
d \geq \Omega \left( \frac{\lg m}{\lg(s/n)} \right) .
\]

It follows, that if \( s = O(n) \) then \( d \) is \( \Omega(\lg m) \). But if \( s = n^{1+\Omega(1)} \) then nothing is known.

**Proof.** Let \( p = 1/m^{1/4d} \). We have the memory of \( s \) bits and we are going to subsample it. Specifically, we will select a bit of \( s \) with probability \( p \), independently.

The intuition is that we will shrink the memory but still answer a lot of queries, and derive a contradiction because of the seed length required to sample \( k \)-wise independence.

For the “shrinking” part we have the following. We expect to keep \( p \cdot s \) memory bits. By a Chernoff bound, it follows that we keep \( O(p \cdot s) \) bits except with probability \( 2^{-\Omega(p \cdot s)} \).

For the “answer a lot of queries” part, recall that each query probes \( d \) bits from the memory. We keep one of the \( m \) queries if it so happens that we keep all the \( d \) bits that it probed in the memory. For a fixed query, the probability that we keep all its \( d \) probes is \( p^d = 1/m^{1/4} \).

We claim that with probability at least \( 1/m^{O(1)} \), we keep \( \sqrt{m} \) queries. This follows by Markov’s inequality. We expect to not keep \( m - m^{3/4} \) queries on average. We now apply Markov’s inequality to get that the probability that we don’t keep at least \( m - \sqrt{m} \) queries is at most \( (m - m^{3/4})/(m - \sqrt{m}) \).

Thus, if \( 2^{-\Omega(p \cdot s)} \leq 1/m^{O(1)} \), then there exists a fixed choice of memory bits that we keep, to achieve both the “shrinking” part and the “answer a lot of queries” part as above. This inequality is true because \( s \geq n > m^{1/3} \) and so \( p \cdot s \geq m^{-1/4+1/3} = m^{\Omega(1)} \). But now we have \( O(p \cdot s) \) bits of memory while still answering as many as \( \sqrt{m} \) queries.

The minimum seed length to answer that many queries while maintaining \( k \)-wise independence is \( k \log \sqrt{m} = \Omega(k \lg m) = \Omega(n) \). Therefore the memory
has to be at least as big as the seed. This yields
\[ O(ps) \geq \Omega(n) \]
from which the result follows. □

This lower bound holds even if the \( s \) memory bits are filled arbitrarily (rather than having entropy at most \( n \)). It can also be extended to adaptive cell probes.

We will now show a conceptually simple data structure which nearly matches the lower bound. Pick a random bipartite graph with \( s \) nodes on the left and \( m \) nodes on the right. Every node on the right side has degree \( d \). We answer each probe with an XOR of its neighbor bits. By the Vazirani XOR lemma, it suffices to show that any subset \( S \subseteq [m] \) of at most \( k \) memory bits has an XOR which is unbiased. Hence it suffices that every subset \( S \subseteq [m] \) with \( |S| \leq k \) has a unique neighbor. For that, in turn, it suffices that \( S \) has a neighborhood of size greater than \( \frac{d|S|}{2} \) (because if every element in the neighborhood of \( S \) has two neighbors in \( S \) then \( S \) has a neighborhood of size \( < \frac{d|S|}{2} \). We pick the graph at random and show by standard calculations that it has this property with non-zero probability.

\[
\Pr \left[ \exists S \subseteq [m], |S| \leq k, \text{ s.t. } |\text{neighborhood}(S)| \leq \frac{d|S|}{2} \right] \\
= \Pr \left[ \exists S \subseteq [m], |S| \leq k, \text{ and } \exists T \subseteq [s], |T| \leq \frac{d|S|}{2} \text{ s.t. all neighbors of } S \text{ land in } T \right] \\
\leq \sum_{i=1}^{k} \binom{m}{i} \cdot \left( \frac{s}{d \cdot i/2} \right)^{d-i} \cdot \left( \frac{d \cdot i}{s} \right)^{d-i} \\
\leq \sum_{i=1}^{k} \left( \frac{e \cdot m}{i} \right)^{i} \cdot \left( \frac{e \cdot s}{d \cdot i/2} \right)^{d-i/2} \cdot \left( \frac{d \cdot i}{s} \right)^{d-i} \\
= \sum_{i=1}^{k} \left( \frac{e \cdot m}{i} \right)^{i} \cdot \left( \frac{e \cdot d \cdot i/2}{s} \right)^{d-i/2} \\
= \sum_{i=1}^{k} \left[ \frac{e \cdot m}{i} \cdot \left( \frac{e \cdot d \cdot i/2}{s} \right)^{d/2} \right]^i.
It suffices to have $C \leq 1/2$, so that the probability is strictly less than 1, because $\sum_{i=1}^{k} 1/2^i = 1 - 2^{-k}$. We can match the lower bound in two settings:

- if $s = m^{\epsilon}$ for some constant $\epsilon$, then $d = O(1)$ suffices,
- $s = O(k \cdot \log m)$ and $d = O(\log m)$ suffices.

**Remark 3.** It is enough if the memory is $(d \cdot k)$-wise independent as opposed to completely uniform, so one can have $n = d \cdot k \cdot \log s$. An open question is if you can improve the seed length to optimal.

As remarked earlier the lower bound does not give anything when $s$ is much larger than $n$. In particular it is not clear if it rules out $d = 2$. Next we show a lower bound which applies to this case.

**Problem 4.** Take $n$ bits to be a seed for 1/100-biased distribution over $\{0,1\}^m$. The queries, like before, are the bits of that distribution. Recall that $n = O(\log m)$.

**Theorem 5.** You need $s = \Omega(m)$.

*Proof.* Every query is answered by looking at $d = 2$ bits. But $t = \Omega(m)$ queries are answered by the same 2-bit function $f$ of probes (because there is a constant number of functions on 2-bits). There are two cases for $f$:

1. $f$ is linear (or affine). Suppose for the sake of contradiction that $t > s$. Then you have a linear dependence, because the space of linear functions on $s$ bits is $s$. This implies that if you XOR those bits, you always get 0. This in turn contradicts the assumption that the distributions has small bias.

2. $f$ is AND (up to negating the input variables or the output). In this case, we keep collecting queries as long as they probe at least one new memory bit. If $t > s$ when we stop we have a query left such that both their probes query bits that have already been queried. This means that there exist two queries $q_1$ and $q_2$ whose probes cover the probes of a third query $q_3$. This in turn implies that the queries are not close to uniform. That is because there exist answers to $q_1$ and $q_2$ that fix bits probed by them, and so also fix the bits probed by $q_3$. But this contradicts the small bias of the distribution.

\[ \square \]
References