Today we are discussing Communication Complexity and its applications. In general lower bounds in communication complexity are considered easier to get than in many other areas (like streaming algorithms, single tape TM lower bounds and circuit lower bounds). In this class we will show a couple examples of how to use CC lower bounds to get lower bounds in other areas.

1 Review

Communication Complexity is defined as the minimum number of bits needed to communicate between A and B to solve for \( f(x, y) \) when Alice is given \( x \) and Bob \( y \). Where \( x \) and \( y \) are \( n \) bit boolean strings.

\[
D(EQUALITY) = n \\
D(IP) \geq R(IP) = \Omega(n) \\
D(DIST) \geq R(DIST) = \Omega(n)
\]

2 Single Tape Turing Machine Lower Bounds

Definition \( PAL = \{xx^R : x \in 0, 1^*\} \) where \( x^R \) is the string \( x \) reversed.

Theorem 2.1. On a Single Tape Turing Machine (STTM), \( PAL \) requires \( \Omega(n^2) \) steps.

At a high level we will show that if there exists an \( STTM \) solving \( PAL \) in \( o(n^2) \) steps, then, this implies that \( \exists \) deterministic communication protocol to solve \( EQUALITY \) in \( o(n) \) communication.
Proof. Let \( x \in \{0, 1\}^n \). Consider the input \( g(x) = x0^a \). Let
\[
f_x(i) = \text{the number of times the head passes the } i^{th} 0 \text{ on the input } x0^a.
\]
Note that \( \sum f_x(i) \leq o(n^2) \). This implies that there exists an \( i \) s.t. \( f_x(i) = o(n) \). Now let \( i(x) = \) one of these values, determined deterministically.

Now we can construct the procedure to get equality from a \( \text{PAL} \) procedure.

1. \( A \) sends \( i(x) \) to \( B \) and \( B \) sends \( i(y) \) to \( A \). If \( i(x) \neq i(y) \) then reject.
   If \( x = y \) then \( i(x) = i(y) \). So if \( i(x) \neq i(y) \) then \( x \neq y \).

2. \( A \) simulates the \( \text{PAL} \) procedure on \( g(x) \) when the head of the TM is at locations \([0, i(x)]\). When the head passes from \( i(x) \) to \( i(x) + 1 \) then \( A \) sends the state of the TM to \( B \). \( B \) simulates the TM for any locations after or including \( i(x) + 1 \) where the input simulated by \( B \) is \( g(y) \). Any time the head passes left of \( i(x) + 1 \) \( B \) sends the state to \( A \). If the simulated machine accepts then \( A \) and \( B \) accept. If the simulated machine rejects then \( A \) and \( B \) reject.
   If \( x = y \) the TM should accept. If \( x \neq y \) then the ‘effective input’ to the TM starts with \( g(x) \) and ends with \( g(y) \) and won’t be a reflection, thus the TM will reject. So, iff the TM accepts then \( x = y \).

This procedure will take \( O(\lg(n)) \) bits for step 1 and \( O(1)O(o(n)) \) for step 2 (because we only cross the location \( i(x) o(n) \) times and we only send messages when we pass \( i(x) \)). Thus, this procedure takes \( o(n) \), which is impossible! Contradiction: \( \text{PAL} \) takes \( \Omega(n^2) \) steps on a TM.

\( \square \)

3 Streaming Algorithm Lower Bounds

Streaming algorithms take in a sequence of inputs and see each input once \[1\]. Unlike a Turing Machine they don’t get to view their input whenever they want. The vector of inputs is called \( y \). In general the question asked about streaming algorithms is the space requirement of the problem. The high level approach for proving lower bounds with communication complexity is have \( x \) be the first half of streaming inputs and \( y \) be the second half of streaming inputs. Then, certainly we could take the streaming algorithm run it on \( x \) and then send the current state of the memory from \( A \) to \( B \) and then \( B \) can run the streaming algorithm on the rest of the input. This means that the communication complexity serves as a lower bound.

Let \( n \) be the number of elements in \( y \), and let \( m \) denote the universe size that entries of the input stream come from. For every input stream \( y \), and
For a frequency vector \( F_y \), we can define the \( p \)-th moment of \( F_y \) as

\[
F_p = \left( \sum_a F_y(a)^p \right)^{1/p}
\]

This is the \( \ell_p \) norm of the vector \( F_p \).

We can consider the streaming problem of computing the \( p \)-th frequency moment of an input stream \( y \). This has a natural interpretation for different \( p \):

1. \( F_0 \) - number of distinct values in the input stream. There is an approach to only uses polylog\((n, m)\) space.
2. \( F_1 \) - number of elements in \( y \). Computing this uses \( O(lg(n)) \) space.
3. \( F_\infty = \max_a F_y(a) \) - the element \( a \) in the universe that occurs the most often in the input stream \( y \).

**Theorem 3.1.** \( F_\infty \) requires \( \Omega(m) \) space

**Proof.** Suppose we had a streaming algorithm \( A \) computing \( F_\infty \) using \( S \) space. We now show how to use this to get a protocol \( P \) for the DISJOINTNESS problem on input size \( m \) that uses \( S \) communication. Since \( R(DISJ) = \Omega(m) \), this implies that \( S = \Omega(n) \).

1. Alice simulates algorithm on \( a_1, ..., a_k \)
2. Alice sends state of algorithm to Bob.
3. Bob finishes simulation on on \( b_1, ..., b_j \)
4. If algorithm outputs 1 then output \( DISJ \).

\( \square \)

### 3.1 Puzzle Aside

With two words of space \((\lg(N) + \lg(M) \) space) can you calculate the majority in a streaming problem? Majority is the problem of returning the strict majority (not plurality) from a streaming input.
4 Circuit lower bound from Communication complexity

(Karchmer-Wogderson games) \( f : \{0,1\}^n \to \{0,1\} \) Question: what is the min formula size to compute \( f \)? Or, equivalently, what is the min depth boolean formula to compute \( f \)?

Claim: If \( F \) has size \( s \), \( \exists \) equivalent \( F' \) with depth \( O(\log(s)) \).

**Theorem 4.1.** \( \text{depth}(f) = CC(KW_f) \)

*Proof.* The game is given \( f(x) \neq f(y) = 0 \) find an \( i \) such that \( x_i \neq y_i \).

\[ D(KW_f) = \text{depth}(f) \]

Let \( f \) be depth-optimal formula of depth \( d \).

They will create a communication protocol based on the depth-optimal circuit. They will maintain the invariant that \( F'(x) \neq F'(y) = 0 \) where \( F' \) is a sub formula.

Alice will “own” all the or gates and Bob will “own” all the and gates. Bob tells Alice which child sub formula evaluates to 0. When at an or gate Alice is outputting 1 and Bob is outputting 0, so Alice gives some input on which she has a 1. On and gates Bob tells Alice an input at which he has a 0. Note that we spend \( O(1) \) bits at each layer of height. This means that the protocol takes \( O(d) \) communication total. Thus, a lower bound on depth is the communication complexity of the corresponding Karchmer-Wogderson game.

**Theorem 4.2.** \( D(KW_f) \geq \text{depth}(f) \)

*Proof.* Run other proof in reverse.

**Theorem 4.3.** \( \text{depth}(\text{PARITY}) \geq 2 \log(n) \)

*Proof.* \( KW_{\text{parity}} = \text{Alice gets odd parity string, Bob gets even parity string. Find place they differ.} \)

By pigeon hole principal Alice needs at least \( \log(n) \) bits. If she had fewer she wouldn’t be able to index all the locations in the string. Bob must also receive at least \( \log(n) \) bits for the same reason. Thus, the communication must have greater than \( 2 \log(n) \) bits.

A current problem is \( P \) vs \( NC \) which is basically the question of “can \( P \) be parallelized”.

4
References