In the last lecture we defined 2-party communication complexity:

\[ f : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\} \]

Communication Complexity \( CC(f) \) denotes the minimum number of bits that Alice and Bob need to exchange to compute \( f(x,y) \) in the worst case.

The general protocol is simply having Alice send her entire bit string to Bob, letting Bob compute \( f(x,y) \) and reply the result bit back to Alice, leading to the upper bound of \( CC(f) \leq n + 1 \).

In this lecture we will examine four functions and their communication complexity:

1. Parity equality: \( f_1(x,y) = 1 \iff \sum_i x_i \neq \sum_i y_i \) (over \( F_2 \))
2. Weight equality: \( f_2(x,y) = 1 \iff wt(x) + wt(y) \geq t \)
3. Set equality: \( f_3(x,y) = 1 \iff x = y \)
4. Set disjointness: \( f_4(x,y) = 1 \iff \sum_i x_i y_i = 0 \)

# Communication Complexity

Let \( f : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\} \) be a binary function. Further let Alice and Bob have \( x \in \{0,1\}^n \) and \( y \in \{0,1\}^n \) respectively. Then \( CC(f) \) is the communication complexity.

## 1.1 "Parity Equality"

\( f_1(x,y) = 1 \iff \sum_i x_i \neq \sum_i y_i \) (over \( F_2 \))

Note that \( f_1(x,y) = 1 \) if and only if the parity of \( x \) is different than the parity of \( y \).

The communication between Alice and Bob can be illustrated as:
We now show that:

\[ CC(f_1) \leq 2 \]

That is, we will present a communication protocol that computes \( f_1 \) with two bits of communication. The protocol is simple: Alice computes the parity of her inputs and sends it to Bob. Then Bob knows the value of \( f_1(x, y) \) which he can send to Alice (as Bob can his own parity value and check if it matches the one sent by Alice).

We also have the lower bound \( CC(f_1) \geq 1 \) because there must be a minimum of communication, i.e. sending a true/false to the other party, to determine a non-constant function.

### 1.2 “Weight Equality”

\[ f_2(x, y) = 1 \text{ iff } wt(x) + wt(y) \geq t \]

Note that \( f_2(x, y) = 1 \) if and only if the Hamming weights for \( x \) and \( y \) sums to a value at least \( t \).

Next, we consider the following natural protocols for \( f_2 \):

→ Send \( wt(x) \) to Bob
  Alice computes the weight of \( x \) and sends it to Bob. Since \( x \) contains \( n \) bits, Alice might need to send \( O(\log(n)) \) bits in the worst case.

→ Send \( wt(x) \) to Bob if \( wt(x) < t \)
  else send \( t \) to Bob
  If the weight is smaller than \( t \), Alice sends the weight, but if the weight is larger than \( t \) Alice sends \( t \). The function only needs to tell if the sum of the weights is at least \( t \), so sending \( t \) even though the weight is larger will not change the functions resulting value. This protocol sends a number at most \( t \) (and not at most \( n \) as before), so the amount of communication is \( O(\log(t)) \).

### 1.3 “Set Equality”

\[ f_3(x, y) = 1 \text{ iff } x = y \]

Let \( f_3(x, y) = 1 \) if and only if two inputs are the same. Let us look at a typical exchange of messages between Alice and Bob:
At the end of the protocol, Alice knows the value of $f_3(x, y)$. Let the transcript $(m_1, \ldots, m_t)$ be denoted $\tau(x, y)$.

We will prove a lower bound on $CC(f_3)$, where the main idea is to show that for any protocol with low communication complexity, there exist $(x, y)$ and $(x, y')$ such that $\tau(x, y) = \tau(x, y')$ (where $y \neq y'$). Note that Alice will output the same answer for both $(x, y)$ and $(x, y')$. This is incorrect since $f_3(x, y) \neq f_3(x, y')$.

**Proposition 1.** $CC(f_3) \geq n$

**Proof.** For the sake of contradiction, assume there exists a protocol that decides $f_3$ and exchanges at most $n - 1$ bits over all inputs.

$$J = \{(x, x) \mid x \in \{0, 1\}^n\}$$

We claim that there exist $x \neq y$ such that $\tau(x, x) = \tau(y, y)$. Number of bits to represent a transcript is at most $n - 1$ which means that there exist at most $2^{n-1}$ distinct transcripts. On the other hand $|J| = 2^n$. In other words, there are more distinct inputs in $J$ than there are distinct transcripts, so there must exist $(x, x) \neq (y, y) \in J$ that lead to the same transcript under the assumed protocol. This can be illustrated as follows:

We see that the protocol exchanges the same messages for $(x, x)$ and $(y, y)$. Now if we assume that Alice holds the codeword $x$ and Bob holds $y$, then we still get the same exchange of messages as before:

$$\text{Alice } \xrightarrow{m_1} \text{ Bob}$$

$$\text{Alice } \xleftarrow{m_2} \text{ Bob}$$

$$\text{Alice } \xrightarrow{m_3} \text{ Bob}$$

$$\vdots$$

$$\text{Alice } \xleftarrow{m_t} \text{ Bob}$$

$\text{ same protocol }$
In particular, the protocol accepts \((x, y)\) yet \(f_3(x, y) = 0\). Thus, the protocol is incorrect, which proves the desired result.

\[f_4(x, y) = 1 \text{ iff } \sum_i x_i y_i = 0\]

\(f_4(x, y) = 0\) if and only if \(x\) and \(y\) do not have 1s in the same position. Alternatively, if we think of \(x\) and \(y\) as subsets of \(\{1, ..., n\}\), \(f_4(x, y) = 0\) if and only if \(x\) and \(y\) are disjoint sets. We next show that:

\textbf{Proposition 2.} \(\text{CC}(f_4) \geq \frac{n}{2}\)

\textit{Proof.} We will reduce from the set equality function. As a notational convenience, define \(\neg y\) to be \(y\) with all its bits flipped.

We reduce an arbitrary input \((x, y)\) for \(f_3\) to two inputs \((x, \neg y)\) and \((\neg x, y)\) for \(f_4\) with the following properties:

1. If \(f_3(x, y) = 1\), then both \(f_4(x, \neg y) = f_4(\neg x, y) = 0\).

2. If \(f_3(x, y) = 0\), then either \(f_4(x, \neg y) = 1\) or \(f_4(\neg x, y) = 1\).

1. is realized since \(f_3(x, y) = 1\) if and only if the sets \(x\) and \(y\) are elementwise equal. Therefore flipping every element in one of the sets will result in two disjoint sets.

2. is realized since \(f_3(x, y) = 0\) implies that there exists a \(j\) such that \(x_j \neq y_j\). Now if \(x_j = 1\), then \(x_j = \neg y_j = 1\), and thus \(f_4(x, \neg y) = 1\). Similarly, if \(x_j = 0\), then \(\neg x_j = y_j = 1\), and thus \(f_4(\neg x, y) = 1\).

Note that given the above, given a protocol for \(f_4\), one has a protocol for \(f_3\) (run on both \((x, \neg y)\) and \((\neg x, y)\)). Now if this protocol uses \(< \frac{n}{2}\) bits, then we get a protocol for \(f_3\) that uses \(< n\) bits. This would, however, contradict the result we just proved \(\text{CC}(f_3) \geq n\). The lower bound for \(\text{CC}(f_4)\) is thus a loose one.

To conclude, we state the following theorem without proof:

\textbf{Theorem 1.} \(\text{CC}(f_4), \text{CC}(f_3) \geq n + 1\)