CS Theory (Spring '25) Fall, 2025

Lecture Notes: Communication Complexity

Instructor: *Toniann Pitassi*

0 Some one-sided lower bound.

Let MAJ $= \{w \in \{0,1\}^* \mid w$ has at least as many 1s as 0s$\}$. For instance $w = 1010 \in$ MAJ but $w = 11000 \notin$ MAJ.

We want to show the following theorem:

Theorem 1. MAJ *isn't a regular language.*

To prove this, we can either use a direct argument based on DFAs, or we can show that no one-way communication protocol for MAJ uses $O(1)$ communication. The idea is that given $w = w^A w^B \in \{0,1\}^*$, Alice gets w^A and Bob gets w^B. If Alice doesn't send Bob the exact number of 1's in w^A, Bob can't be certain $w \in$ MAJ.

Hence, we will show the following:

Lemma 2. *Any one-way communication protocol for* MAJ *on inputs of length* $2n$ *requires* $\Omega(\log(n))$ *communication.*

Before diving into the proof, recall that if Alice sends only messages of length $\leq \ell$, then she can send a message of length $0, 1, 2 \ldots, \ell$. And for each length i, there's 2^i many binary strings of length i. So the number of messages is at most:

$$\sum_{i=1}^{\ell} 2^i = 2^{\ell+1} - 1.$$

Proof. Assume for contradiction that there exists a protocol where on inputs $w = w^A w^B$ of length $2n$, Alice only sends messages to Bob of length $\ell \leq \log_2(n) - 1$. [1]

Observe that the total number of different messages Alice can send is at most:

$$\sum_{i=0}^{\ell} 2^i \leq 2^{\ell+1} - 1 \leq 2^{\log_2(n)} - 1 < n.$$

Here the input w has length $2n$ so Alice's w^A has length n. Consider the following set X:

$$X = \{x \mid x = 1^k 0^{n-k}, k \geq 0\}.$$

[1] Here we use $\log_2(n) - 1$ since with this many bits, you can't encode n different messages. There's many other choices that would work.

Observe that every string in X has length n, hence each $x \in X$ is a possible input for Alice (we could have $w^A = x$). But every string in X has a different number of 1.

Why are we doing this ? Intuitively, if Alice doesn't send a unique message for each of $x \in X$, Bob wouldn't be able to know how many 1's Alice's string had. So something must go wrong.

Note that $|X| = n+1$ but as we already saw, there are only $< n$ many different messages Alice can send. Hence, by the pigeonhole principle, there exists $x, y \in X$ ($x \neq y$) such that $m(x) = m(y) = \alpha$. That is, there are two different inputs x, y for Alice such that Alice sends the same message on both x and y.

Since $x \neq y$ and they come from the set X, we can write:

$$x = 1^a 0^{n-a} \quad \text{and} \quad y = 1^b 0^{n-b} \quad \text{with } a \neq b.$$

Without loss of generality, we can assume $a < b$.[2]

Now consider the string $z = 1^{n-b}0^b$ which has length n.

If the input is $w = xz$, then Alice gets x and Bob gets z. We have $xz = 1^a 0^{n-a} 1^{n-b} 0^b$. This string has $t = n + a - b$ many 1's. Since $a < b$, we have $t < n$ so $xz \notin$ MAJ. So Bob must output 0 meaning:

$$0 = g(m(x), z) = g(\alpha, z). \tag{1}$$

However, if the input is $w = yz$, then Alice gets y and Bob gets z. We have $yz = 1^b 0^{n-b} 1^{n-b} 0^b$. This string has $t' = n + b - b = n$ many 1's. So $yz \in$ MAJ. So Bob must output 1 meaning:

$$1 = g(m(y), z) = g(\alpha, 1). \tag{2}$$

Now we have that Equations (5) and (6) give us a contradiction. Hence, there can't be a protocol where on inputs on length n, Alice sends a message of length $< \log_2(n) - 1$. Hence, any protocol for MAJ must use at least $\log_2(n) = \Omega(\log(n))$ communication on inputs of length $2n$. ■

Now, we know that no communication protocol for MAJ has $O(1)$ communication [3]. Hence, using the theorem from last lecture, we can conclude that MAJ isn't regular.

Here is an alternative proof, which is shorter. It fits with our intuition and with the upper bound, that Alice has to send the number of 1's in her input which requires $\log n$ message length.

Proof. Assume for contradiction that there exists a protocol where on inputs $w = w^A w^B$ of length $2n$, such that Alice sends messages to Bob of length at most $\log_2(n) - 1$. This means the total number of messages Alice can send is at most $\sum_{i=0}^{\ell} 2^i \leq 2^{\ell+1} - 1 \leq 2^{\log_2(n)} - 1 < n$.

Here the input w has length $2n$ so Alice's w^A has length n. Note that the number of 1's in w^A can range from 0 to n, so there's $n + 1$ options for the number of 1s.

[2] We write "Without loss of generality" so that we don't have to write the "proof" twice for $a < b$ and $b > a$. Indeed, if $b > a$, the proof would be the same, but we would switch the role of x and y. This would be a lot of unnecessary work, so we only do the case $a < b$. You can see the Wikipedia explanation here.

[3] The communication can't be constant, as we saw if the input has length $2n$, Alice must send a message of length $\Omega(\log(n))$

Since, there are only $< n$ many different messages Alice can send, by the pigeonhole principle, there exists $x, y \in \{0,1\}^n$ where x, y have a different number of 1's and such that Alice sends the same message, α on both x and y. That is, $m(x) = m(y) = \alpha$.

Let a be the number of 1's in x and let b be the number of 1's in y, where without loss of generality $a < b$. Now let $z \in \{0,1\}^n$ with exactly $n - b$ many 1s and b many 0s.

If the input is $w = xz$, then Alice gets x and Bob gets z. Now the number of 1's in xz is $a+n-b < n$ since $a < b$. Therefore, $xz \notin$ MAJ (since $|xz| = 2n$ and there are less than n 1's). So Bob must output 0 on xz, meaning:

$$g(m(x), z) = g(\alpha, z) = 0. \tag{3}$$

On the other hand, if the input is $w' = yz$, then the number of 1's in $w' = yz$ is $b + n - b = n$, so $w' \in$ MAJ. So Bob must output 1 meaning:

$$g(m(y), z) = g(\alpha, z) = 1. \tag{4}$$

Now we have that equations (3) and (4) give a contradiction. Hence, there can't be a protocol where on inputs on length $2n$, Alice sends a message of length $< \log_2(n) - 1$. Hence, any protocol for MAJ must use at least $\log_2(n) = \Omega(\log(n))$ communication on inputs of length $2n$. ■

A "fun" exercise: You should try to to prove MAJ isn't regular by a direct argument. (Assume there exists a DFA for MAJ and get a contradiction using that DFA, like we did for EQ last lecture.)[4]

Another example.

Let First $= \{w \in \{0,1\}^* \mid w = xy$ and the first 1 in x doesn't appear after the first 1 in $y\}$.

For instance $w = 110\,011$ and $w = 010\,011$ are both in First. But $w = 011\,100 \notin$ First.

Theorem 3. *Any one-way communication protocol for* First *on inputs of length* $2n$ *requires* $\Omega(log(n))$ *communication.*

Proof. Assume for contradiction that there exists a protocol where on inputs $w = w^A w^B$ of length $2n$, Alice only sends messages to Bob of length $\ell \leq \log_2(n) - 1$. This means the total number of messages Alice can send is at most $\sum_{i=0}^{\ell} 2^i \leq 2^{\ell+1} - 1 \leq 2^{\log_2(n)} - 1 < n$.

Here the input w has length $2n$ so Alice's w^A has length n. Note that the first 1 in w^A can be at an index between 1 and n, so there's n options for the index of the first 1.

Since, there are only $< n$ many different messages Alice can send. By the pigeonhole principle, there exists x, y where x, y where the first one in x is at a different position then in y but: $m(x) = m(y) = \alpha$ (Alice sends the message on x and y).

We have that the first 1 in x appears at index as but the first 1 in y appears at index b. Without loss of generality we can assume $a > b$. So, consider a string $z \in \{0,1\}^n$ with exactly one 1 at index b.

If the input is $w = xz$, then Alice gets x and Bob gets z. The first one in x is at index a, the first one is z is at index $b < a$. So, $xz \notin$ First. So Bob must output 0 meaning:

$$0 = g(m(x), z) = g(\alpha, z). \tag{5}$$

[4]The proof should reuse ideas from the lower bound for one-way communication. What happens if two strings x, y with a different number of 1 end in the same state q ?

However, if the input is $w = yz$, then Alice gets y and Bob gets z. In both x and y the first one is at index a. So $yz \in$ First. So Bob must output 1 meaning:

$$1 = g(m(y), z) = g(\alpha, 1). \tag{6}$$

Now we have that Equations (5) and (6) give us a contradiction. Hence, there can't be a protocol where on inputs on length n, Alice sends a message of length $< \log_2(n) - 1$. Hence, any protocol for First must use at least $\log_2(n) = \Omega(\log(n))$ communication on inputs of length $2n$. ■

1 Communication protocols

Last class we discussed one-way communication protocols, and showed that lower bounds for one-way communication for a decision problem imply that the associated language is not regular. Recall that in that setup, to compute a decision problem f_L on an input w, the input w was split into two halves, where Alice sees the first half of the input, w^A, and Bob receives the second half, w^B. In a one-way protocol, Alice sends a single message to Bob (based on her input), and then Bob should output either accept or reject (where he should output accept whenever $f_L(w) = 1$ and reject whenever $f_L(w) = 0$. We then proved strong lower bounds on the one-way communication complexity of EQ, which implied that the associated language is not regular.

Today we will discuss a vast generalization of the one-way communication model, which is known as the 2-party communication complexity model. This model is very well studied and has an enormous variety of applications across many subareas of computer science and mathematics, including applications in game theory, distributed computing, AI, blockchains, complexity theory, and combinatorics. In a 2-party protocol, both Alice and Bob send messages to another one another in a series of rounds, and at the end the last player should announce the final answer. Thus 2-party protocols generalize the one-round case in that they can have any number of rounds. We now describe the model a bit more formally.

Once upon a time, there were two persons, Alice and Bob. Suppose that Alice and Bob have $w^A \in \{0,1\}^*$ and $w^B \in \{0,1\}^*$, as inputs respectively, and their goal is to compute $f(w^A, w^B)$ for some function $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ and for both party to learn $f(w^A, w^B)$. To achieve this Goal, Alice and Bob communicates. Their communication might be something like the following:[5]

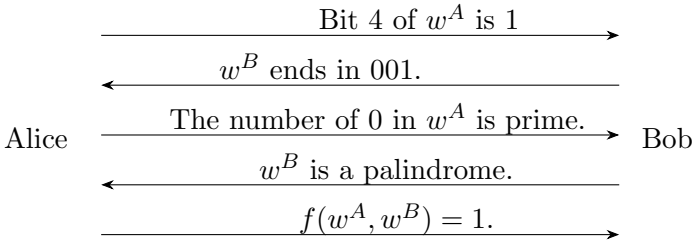

Figure 1: An example of communication between Alice and Bob.

We will assume that the last message of the protocol always has to be the value of $f(w^A, w^B)$ (which is 0 or 1). Below are a few examples of communication problems, that is, functions for which we study communication protocols. The convention is to use all capital letters for communication problems.

[5]This is just an arbitrary example.

Example 4. $\mathrm{EQ}(x,y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$.

Example 5. $\mathrm{MAJ}(x,y) = \begin{cases} 1 & \textit{if } xy \textit{ contains more 1's than 0's} \\ 0 & \textit{otherwise} \end{cases}$.

Example 6. $\mathrm{PARITY}(x,y) = \begin{cases} 1 & \textit{if } xy \textit{ contains an odd number of } 1\textit{'s} \\ 0 & \textit{otherwise} \end{cases}$.

The next example shows that for any communication problem, there is always a trivial protocol of cost $|w^A| + 1$; in fact, it is even a one-way protocol.

Example 7. *For any communication problem f, on input $w = w^A w^B$, we have the following trivial protocol: Alice first sends her whole input w^A to Bob. Bob then computes $f(w^A, w^A)$ and sends it to Alice. This protocol takes $O(|w^A| + 1)$ bits of communication, which is $O(|w|)$.*

However, for many communication problems f, there are much better protocols than the naïve one, in terms of total bits of communication. For example, we will see soon that MAJ has a protocol with $O(\log n)$ communication, and PARITY has a protocol with $O(1)$ communication.

In this and the next lecture, we will focus on the following question:

> For a given communication problem, what is the least amount of communication any protocol for the problem have to use?

Definition 8. *The number of bits of communication used by a protocol Π is called the* communication complexity *of the protocol, which is some function $cc^\Pi : \mathbb{N} \to \mathbb{N}$. $cc^\Pi(n)$ is the maximum number of bits communicated by Alice and Bob on input $w = w^A w^B$, where $|w| \leq n$.*

We also define the communication complexity of a communication problem P to be the minimum communication complexity among all protocols Π for the problem P.

The rationale that we focus on communication and not the local computation by Alice or Bob themselves is that communication usually takes much longer than local computation. For example, Alice might be in Manaus (Amazon) while Bob is in Yakutsk (Siberia). As another example, when designing chips, it is common for the bottleneck to be the communication between different parts of the chip, and this is also one of the reasons why people started to study communication complexity.

Now we discuss better-than-naïve protocols for a couple of well-studied problems. First, we review the simple and efficient protocol for PARITY that we discussed last class.

Example 9 (Protocol for PARITY)**.** *Let the input be $w = w^A w^B$, $|w^A| = \lceil |w|/2 \rceil$, $|w^B| = \lceil |w|/2 \rceil$. Alice computes the number of 1's in w^A mod 2; that is, she sends Bob '1' if the number of 1's is odd, and '0' otherwise. Then Bob computes the number of 1's in y mod 2. From this he can determine the number of 1's in the whole string w mod 2 and outputs this value. (If the the number of 1's in w^A and the number of 1's in w^B are both odd or both even, then he outputs '0' and otherwise he outputs '1'.)*

There are 2 bits of communication in this protocol regardless of $|w|$ and therefore PARITY has an $O(1)$-cost communication protocol. We can prove that this is optimal, as Bob needs 1 bit from Alice to learn about the parity of the number of 1's in x, and similarly Alice needs 1 bit of message from Bob.

Remark 10. *Bob already knows the answer before sending it to Alice. However, our requirement is that the last message is the value of $f(w^A, w^B)$.*

1.1 Is two-party communication "stronger" than 1-way communication ?

The above protocol for PARITY is the same as the one we did for one-way CC. *In particular, any one-way CC is a special case of two-party CC.*

Observation 11. *Given a one way protocol for a problem P where Alice has function m and Bob has function g, we can turn this protocol into a two-party protocol with (almost) the same cost as follows:*

1. *Given w^A, Alice sends $m(w^A)$ to Bob.*
2. *Bob then sends $g(m(w^A), w^B)$ to Alice.*

The only difference is that Bob now sends to Alice one bit at the end to tell her the final answer. So the communication is the same $+1$ bit.

So are there problems that can be solved more efficiently with two-party communication ? The answer is yes !

Let's consider the following problem.

Example 12. INDEX$(x, y) = x_i$ *where i is the first index with $y_i = 1$.*

Here is an example: If $x = 00101$ and $y = 01011$, then INDEX$(x, y) = 0$ because the first index where $y_i = 1$ is for $i = 2$ but $x_2 = 0$.

To solve INDEX, there's an easy two-party protocol:

Example 13 (Protocol for INDEX)**.** *Let the input be $w = w^A w^B$ with $|w| = n$. Bob sends to Alice the index i (in binary) where i is the first index with $y_i = 1$. Alice then simply says to Bob the value of x_i.*

How much communication ? Bob sends an index between 1 and $|w^B|$. To send such an index, he needs $O(\log(|w^B|))$ bits. Recall that $|w^B| = \lfloor |w|/2 \rfloor \leq n/2$. So the communication is $O(\log(n))$ bits. Then Alice sends 1 bit to Bob. So the total communication is $O(\log(n) + 1) = O(\log(n))$ bits.

In the above, we relied on the fact that Bob can send a message to Alice in a two-party protocol. So Bob can tell Alice what "index to look at". But in a one-way protocol only Alice gets to send a message to Bob, and she has no idea what the right index is.

It turns out that Alice basically needs to send all her input to Bob.

Theorem 14. *Any one-way protocol for* INDEX *on inputs of length $2n$ must use $\Omega(n)$ communication.*

We will not prove the above. But, to get some intuition, imagine she sends the same message α on $x = 1111$ and $x' = 0111$. Then, if Bob has $y = 1000$ he is in trouble ! INDEX$(x, y) = 1$ but INDEX$(x', y) = 0$. So he would need to have $1 = g(m(x), y) = g(\alpha, y)$ and $0 = g(m(x'), y) = g(\alpha, y)$, which is impossible.

Hence, index as two-party communication complexity $O(\log(n))$ but one-way communication complexity $\Omega(n)$. So the punchline is two-party protocols are strictly more powerful than one-way protocols.