CS 860: Space Bounded Computation **University of Waterloo: Winter 2026**

Lecture 7: Time-Space Tradeoffs on Branching Programs II

March 3, 2026

Instructor: Sepehr Assadi *Scribe: Peter Matsakis*

Topics of this Lecture

We continue our study of branching program (BP) lower bounds for establishing time-space tradeoffs. In the last lecture, we saw simple problems such as sorting n elements require tradeoffs of $S \cdot T = \Omega(n^2)$ for S-space T-time algorithms (or even BPs). The lower bound crucially relied on the fact that the output has to be large.

In this lecture, we will see BP lower bounds for *decision* problems, namely, when the answer is only *Yes* or *No*. For this, we will follow the seminal work of [BSSV03] with some simplification in their arguments (by focusing on proving weaker guarantees than what [BSSV03] proved). For simplicity, throughout this lecture, we focus on *deterministic* BPs that always output the correct answer even though [BSSV03] can still prove lower bounds for randomized BPs (with more involved techniques).

1 Element Distinctness in Oblivious Branching Programs

To build intuition, we first consider a simpler task: proving lower bounds for *oblivious* branching programs, namely, BPs wherein the location of which input to read does not depend on the input itself (recall that we saw a similar condition for oblivious multi-tape Turing machines in Lecture 2). Recall the definition of BPs from the first lecture as well as the previous one.

Definition 1. An **oblivious** branching program is a branching program such that in each layer, every node reads the same index of the input.

The problem we study for this result is the decision version of the unique elements problem we saw in the previous lecture.

Problem 1. The **element distinctness** problem with parameters $n, N \geqslant 1$ is defined as computing the function $\mathsf{ED} : [N]^n \to \{0,1\}$ where $\mathsf{ED}(x_1, \ldots, x_n) = 1$ iff all x_i's are distinct.

As a warm-up, we first see a proof of the following theorem which is a weaker version of a result in [BSSV03] (that proved this result for general (non-oblivious) BPs, even randomized ones).

Theorem 2. *Any S-space T-time oblivious branching program for the element distinctness problem with $N \geqslant n^3$ satisfies $T = \Omega(n \log\left(\frac{n}{S}\right))$. Specifically, as long as $S = O(n^{0.99})$, $T = \Omega(n \log n)$.*

To prove this theorem, we will use a reduction from *communication complexity*, which we introduce next.

1.1 A (Super) Quick Introduction to Communication Complexity

Suppose we want to compute a function $f : [N]^n \to \{0,1\}$, but the input is split between two parties, Alice and Bob. More specifically, Alice has the input corresponding to indices $A \subseteq [n]$ and Bob has $B := [n] \setminus A$. We further assume that $|A| = |B| = n/2$. Since the function may depend on both halves of the input, we will allow Alice and Bob to communicate with each other but charge them for every bit sent. Specifically, Alice will communicate a message to Bob based on her input, then, Bob communicates back a message based on his input and the received message, and so on. The communication cost of the process is the number of bits Alice and Bob communicated to each other to compute the answer.

We can formalize algorithms, also called *protocols*, in this communication model as follows.

Definition 3. Fix any function $f : [N]^n \to \{0,1\}$ and an equipartition $A \sqcup B = [n]$ of indices of the inputs given to Alice and Bob, respectively. A protocol π for this problem is a rooted binary tree where nodes at even depth, including the root, are considered A-nodes and nodes at odd depth are B-nodes. Each A-node v has a function $f_v : [N]^A \to \{0,1\}$ that maps Alice's input to a bit 0 or 1. Similarly, each B-node v has a function $g_v : [N]^B \to \{0,1\}$ on Bob's input. The leaf-nodes are simply labeled with either 0 or 1. Finally, every edge of the tree is further labeled with 0 or 1.

The computation of the protocol is as follows. Suppose the input to the players is x_A and x_B and let v_0 be the root of the tree. Alice evaluates $b_1 = f_{v_0}(x_A)$ using her input and communicates this bit to Bob. Both players now move to the node v_1 from the root reachable using the edge labeled b_1. Then, Bob evaluates $b_2 = g_{v_1}(x_B)$ using his input and communicates this bit to Alice and both players go the child-node v_2 of v_1 using the edge labeled b_2. The players follow this until they reach a leaf-node – the value of the leaf-node is now considered the output of the protocol (which should be $f(x_A x_B)$).

The depth of this tree is the **communication cost** of the protocol π, denoted by $\|\pi\|$, which corresponds to the largest number of bits communicated by π on any input.

Remark. We can draw a protocol as a communication tree, as shown in Figure 1. Each node has a certain subset of inputs which will result in sending a 1, and the rest will send a 0. Every subsequent node filters these inputs down further, an idea which will become important in just a moment.

Equipped with a "computational device", namely a communication protocol, we can now define communication complexity of a problem.

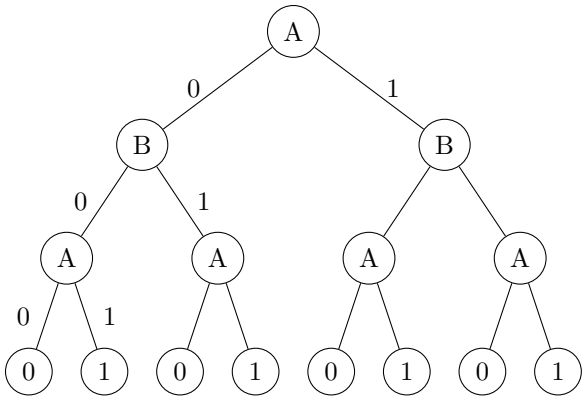

Figure 1: A communication tree of depth 3 (so communication cost 3). For simplicity, we include the output in the communication cost since one extra added bit is inconsequential for the bounds we establish.

Definition 4. The communication complexity of a function $f : [N]^n \to \{0,1\}$, denoted by $CC(f)$, is

$$CC(f) := \min_{\substack{A \sqcup B = [n] \\ |A| = |B| = n/2}} \min_{\pi} \|\pi\| \quad \text{such that } \pi \text{ solves } f \text{ under the partition } A, B \text{ of indices.}$$

In words, this is the minimum communication cost we need to solve the problem among all possible partitioning of input indices.

Remark. We note our definition above is often called the **best-partition** communication complexity to contrast it with the more standard case where A, B are also chosen adversarially. In this definition, the first term above is a max instead of a min. Since we will not use the second notion in this lecture, we omit the term "best-partition" and call our definition "communication complexity."

We could make these definitions more general by allowing protocols to send more bits at a time, which is necessary in certain scenarios. However, in our application, the party who is supposed to receive multiple bits can simply communicate 0 to wait for more communication from the other side side, which increases the communication cost by at most a factor of 2. As we will focus on asymptotics of communication complexity for our problems, this factor of 2 is negligible and so we can stick to our less general definition.

We now can return to our remark from earlier that each node filters inputs. Because Alice and Bob only have access to their own inputs, we actually get a stronger result: at every node, the possible inputs form a **combinatorial rectangle**, namely, a set of the form $X \times Y$. Formally,

Proposition 5 ("Rectangle property of protocols")**.** *Let π be any protocol and assume the inputs (x_A, x_B) of players can be any pair in $X_A \times X_B$. At every node v of the protocol's communication tree, the set of possible inputs that reach v is $Y_A^v \times Y_B^v$, for $Y_A^v \subseteq X_A$ and $Y_B^v \subseteq X_B$. We call the set $Y_A^v \times Y_B^v$ **the inputs consistent with** v.*

Proof. We prove the result by induction. The result holds at the root as all inputs consistent with the root belong to $X_A \times X_B$, which is a combinatorial rectangle.

Consider a node v for which the proposition holds. We will show the result holds for v's child-nodes. Suppose v is an A-node and the set of inputs consistent with v is $Y_A^v \times Y_B^v$. Let $X(b)_A$ for $b \in \{0,1\}$ be the elements x_A of X_A such that $f_v(x_A) = b$. Then, the inputs at 0-child-node of v, denoted v_0, will be

$X(0)_A \times Y_B^v$ and on the 1-child-node, denoted by v_1, will be $X(1)_A \times Y_B^v$. This is because the branching to the child-nodes is done exclusively based on the input of Alice. As such, setting $Y_A^{v_b} = X(b)_A$ and $Y_B^{v_b} = Y_B^v$ for $b \in \{0,1\}$ proves the induction step (the case for B-nodes holds by symmetry). This concludes the proof. □

This proposition leads us to the notion of a **communication matrix** for a function $f : X_A \times X_B \to \{0,1\}$, another useful tool for analyzing communication complexity. An example is shown in Figure 2. Each possible input Alice can receive corresponds to a row and each possible input Bob can receive corresponds to a column. The entry in row a and column b is $f(a,b)$. As we go down the communication tree, Alice and Bob eliminate rows and columns of this matrix until they land in a *monochromatic* combinatorial rectangle, namely, either an all-0 or all-1 (combinatorial) rectangle. If the players stop with both 0s and 1s remaining, then, by Proposition 5, f can be either 0 or 1 on inputs consistent with this node. Thus, whichever output the players choose, the protocol will be wrong on some input."

		Bob			
		00	01	10	11
Alice	00	1	0	0	0
	01	0	1	0	0
	10	0	0	1	0
	11	0	0	0	1

Figure 2: An example communication matrix for the equality problem.

1.2 Communication Complexity of Element Distinctness

We now prove our first communication complexity lower bound in this course (for a communication version of Problem 1). Our goal will be to show a communication complexity lower bound for trying to solve ED after giving each of Alice and Bob exactly half of the input indices. Namely, fix an equipartition (A,B) of indices $[n]$; we consider the problem of $\mathsf{ED}(x)$ when Alice received x_A and Bob receives x_B. We denote this communication problem by $\mathsf{ED}_{A,B}$ for this fixed choice of A, B.

We have a 3-step strategy as follows:

1. We will show the communication matrix M of $\mathsf{ED}_{A,B}$ has many 1s;
2. We will also show that any (combinatorial) 1-rectangle (namely, a rectangle consisting of only 1s) in M has to be small;
3. We will use these, alongside Proposition 5 and our earlier observations, to argue that Alice and Bob will require a lot of communication to narrow down all 1-inputs to 1-rectangles.

Using this, we can prove the following theorem.

Theorem 6. *For $N \geqslant n^2$ and $\mathsf{ED}_{A,B} : [N]^A \times [N]^B \to \{0,1\}$ for any equipartition (A,B) of $[n]$,*

$$CC(\mathsf{ED}_{A,B}) = \Omega(n).$$

Proof. We first lower bound the number of 1-entries in the communication matrix M of $\mathsf{ED}_{A,B}$. Any 1-entry of the matrix corresponds to an $x \in [N]^n$ with all distinct entries; so, their total number is

$$\binom{N}{n} \cdot n! = \frac{N!}{(N-n)!} \geqslant (N-n)^n \geqslant N^n \cdot \left(1 - \frac{1}{n}\right)^n \geqslant \frac{N^n}{2e},$$

where we used the bound $N \geqslant n^2$ for the second inequality, and $\lim_{n\to\infty}(1-1/n)^n = 1/e$ for the third inequality. This constitutes our first step of our plan.

For step two, fix any protocol π for $\mathsf{ED}_{A,B}$ and let $Y_A \times Y_B$ be the 1-rectangle of consistent inputs for a *leaf*-node in π (using Proposition 5). Define

$$S_A := \{s \in [N] \mid \exists x_A \in Y_A \text{ such that } s \in x_A\} \quad \text{and} \quad S_B := \{s \in [N] \mid \exists x_B \in Y_B \text{ such that } s \in x_B\};$$

to be all elements that appear in some consistent input for Alice and Bob, respectively. We argue that $S_A \cap S_B = \emptyset$: if this were not the case, there would be inputs $x_A \in Y_A$ and $x_B \in Y_B$ both containing s, and therefore an input $x = (x_A, x_B)$ in the 1-rectangle $Y_A \times Y_B$ where s appears twice inside x, making $\mathsf{ED}(x) = 0$ instead of 1 which is a contradiction.

Since $S_A \cap S_B = \emptyset$, and both are subsets of $[N]$, at least one of S_A and S_B has size $\leqslant \frac{N}{2}$. Without loss of generality, we will assume it is S_A. This means there are at most $\left(\frac{N}{2}\right)^{n/2}$ ways to choose the $\frac{n}{2}$ elements in an input for Alice. So, the 1-rectangle $Y_A \times Y_B$ has size

$$|Y_A \times Y_B| \leqslant \left(\frac{N}{2}\right)^{n/2} \cdot N^{n/2} = \frac{N^n}{2^{n/2}}.$$

Putting this all together, we have that each 1-entry in the communication matrix must be covered by a leaf-node, and that the protocol π can have at most $2^{\|\pi\|}$ many leaf-nodes. Thus, we have,

$$2^{\|\pi\|} \cdot \frac{N^n}{2^{n/2}} \geqslant \frac{N^n}{2e},$$

which implies that $\|\pi\| = \Omega(n)$ as desired. □

1.3 An Oblivious BP Lower Bound for Element Distinctness

We will now prove a space-time bound for solving ED with oblivious branching programs, proving Theorem 2.

Because our bound will show a time bound of $\omega(n)$, we can assume all branching programs read the entire input by simply appending n useless reads to the beginning. We assume this for the rest of this lecture. As we stated earlier, the proof is a via a reduction to Theorem 6.

We fix a branching program $\mathcal{B}$ for ED with parameters N, n with $N \geqslant n^3$. Let k be such that depth (time) of $\mathcal{B}$ is $T = k \cdot n$. Our goal is to eventually prove that $k = \Omega(\log(n/S))$. We partition the T layers of $\mathcal{B}$ into b blocks of size $\frac{T}{b}$, say $B_1, \ldots B_b$, for some b to be chosen later (we will set $b \approx 2^k$). We will then partition these blocks into two sets $F_A, F_B \subseteq [b]$, not necessarily of equal size; the choice of this partition is determined later in the proof. For now, think of F_A, F_B as being any partition of $[b]$, namely, the blocks defined above.

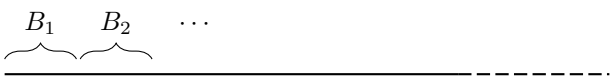

Figure 3: The layers of the branching program $\mathcal{B}$, partitioned into b blocks.

Definition 7. Fix any $x \in [N]^n$ and a partition F_A, F_B of the blocks. Let $\texttt{read}(F_A, x)$ (resp. $\texttt{read}(F_B, x)$ be the set of indices read by blocks in F_A (resp. F_B) when $\mathcal{B}$ is given the input x. For oblivious branching programs, since the indices read are independent of the input x, we can abbreviate these sets as $\texttt{read}(F_A)$ and $\texttt{read}(F_B)$, respectively, which are the same for all $x \in [N]^n$.

We can further narrow down the definitions of these sets as follows.

Definition 8. Fix any $x \in [N]^n$ and a partition F_A, F_B of the blocks. Let

$$\texttt{core}(F_A, x) = \texttt{read}(F_A, x) \setminus \texttt{read}(F_B, x) = [n] \setminus \texttt{read}(F_B, x),$$

that is to say the indices *only* read in the blocks in F_A (the second equality is by our assumption that all indices are being read by the BP). We define $\texttt{core}(F_B, x)$ analogously. Finally, as before, for oblivious branching programs, we can abbreviate these as $\texttt{core}(F_A)$ and $\texttt{core}(F_B)$.

Before moving on, let us see a motivation behind these definitions. Suppose we have an instance of the communication problem $\mathsf{ED}_{A,B}$ for some equipartition A, B of $[n]$. One way for Alice and Bob to find the value of $\mathsf{ED}_{A,B}(x)$ is to simulate the BP $\mathcal{B}$ on x and output the same answer. This is clearly going to output a correct answer. The simulation works as follows: suppose the players have successfully simulated the first t layers of $\mathcal{B}$ and both know which state in the layer $t+1$ the computation path lies on; to simulate the next step, they should both figure out the next transition step of $\mathcal{B}$. Since each layer of $\mathcal{B}$ needs to read one fixed index $i \in [n]$ of the input (by obliviousness), the player that has that index (say, Alice if $i \in A$), can know the transition of $\mathcal{B}$ in this layer; so, this player only needs to send the name of the next layer vertex the computation path reaches to the next player. This can be done with S bits of communication.

Now, if the players have to keep alternating in speaking with each other, then the total communication cost of this protocol will become $O(T \cdot S)$ bits. This way, the lower bound in Theorem 6 would imply $T \cdot S = \Omega(n)$ – unfortunately, this is entirely useless! But, what if we can ensure that each player can simulate $\mathcal{B}$ for "quite a while" on their own, before having to send the new state to the other player? In particular, what if we could ensure that either Alice or Bob can simulate an entire block of layers on their own before having to communicate the state and rely on the other player to continue the computation? This way, the total communication we have will instead be $O(b \cdot S)$ which, as long as $b \ll T$, can potentially lead us to non-trivial lower bounds. This is precisely what we are aiming to achieve using $\texttt{core}$ sets.

In the following lemma, we show that there are choices of F_A, F_B that will lead to non-trivially large $\texttt{core}$ sets; we then use this to show that Alice and Bob can simulate the branching program without having to pass off its state very often. We note that the use of x in the following lemma is redundant for oblivious BPs but is provided to avoid repeating exactly the same proof later for non-oblivious BPs.

Lemma 9. *For any BP with runtime $T = kn$ (even a non-oblivious one) and $b \geqslant 16k^2 \cdot 2^k$ time blocks, and for every choice of $x \in [N]^n$,*

$$\Pr_{F_A, F_B}\left(\textit{both } |\texttt{core}(F_A, x)| \textit{ and } |\texttt{core}(F_B, x)| \geqslant \frac{n}{2^{k+1}}\right) \geqslant \frac{1}{2},$$

over a random partitioning F_A, F_B of $[b]$.

Proof. Fix any $x \in [N]^n$ and for a choice of F_A, F_B, define $core_A = |\texttt{core}(F_A, x)|$ and $core_B = |\texttt{core}(F_B, x)|$ be random variables. For any index $i \in [n]$, let $r(i)$ denote the number of blocks in $B_1, \cdots, B_b$ wherein the index i is read by the BP on x. We will bound the expected value and variance of $core_A$ and use Chebyshev's inequality to conclude the proof. We have,

$$\begin{aligned}
\mathbb{E}\left[core_A\right] &= \sum_{i=1}^{n} \Pr\left(\text{index i is } \underline{\text{not}} \text{ read by } F_B\right) = \sum_{i=1}^{n} 2^{-r(i)} \\
&\qquad\qquad \text{(each block skips being in } F_B \text{ with probability } 1/2) \\
&\geqslant n \cdot \left(\prod_{i=1}^{n} 2^{-r(i)}\right)^{1/n} \qquad\qquad \text{(by the AM-GM inequality)} \\
&= n \cdot 2^{-(1/n)\cdot\sum_{i=1}^{n} r(i)} \geqslant n \cdot 2^{-T/n} = n \cdot 2^{-k}.
\end{aligned}$$

(as $\sum_{i=1}^{n} r(i) \leqslant T$; the equality holds if each index is read exactly once inside the blocks counted by $r(i)$)

Thus, in expectation, $core_A$ is "around" the bounds we would like it to be. We now bound its variance by setting the value of b large enough. Let R_i be the event that i is not read by any block in F_B. We write $i \sim j$ to mean both indices are read together in at least one block for the input x. This means if $i \not\sim j$, the events R_i and R_j are independent (since the randomness of the blocks determining their fates are disjoint). With this in hand, we are now ready to compute the variance:

$$\begin{aligned}
\mathrm{Var}\left[core_A\right] &= \sum_{i,j} \Pr\left(R_i \wedge R_j\right) - \Pr\left(R_i\right) \cdot \Pr\left(R_j\right) \\
&\qquad \text{(as } core_A = \textstyle\sum_i R_i \text{ and variance of a sum is equal to sum of pairwise covariances)} \\
&= \sum_{i \sim j} \Pr\left(R_i \wedge R_j\right) - \Pr\left(R_i\right) \cdot \Pr\left(R_j\right) \leqslant \sum_{i \sim j} \Pr\left(R_i \wedge R_j\right) = \sum_i \Pr\left(R_i\right) \sum_{j \sim i} \Pr\left(R_j \mid R_i\right) \\
&\qquad (R_i \perp R_j \text{ whenever } i \not\sim j \text{ as argued earlier}) \\
&\leqslant \sum_i \Pr\left(R_i\right) \cdot \left(r(i) \cdot \frac{kn}{b}\right) \\
&\qquad \text{(since each block that reads } i \text{ can read at most } kn/b \text{ other indices)} \\
&= \sum_i 2^{-r(i)} \cdot r(i) \cdot \frac{kn}{b} \qquad \text{(since } \Pr\left(R_i\right) = 2^{-r(i)} \text{ as calculated earlier)} \\
&\leqslant \frac{kn}{b} \cdot \frac{1}{n} \cdot \sum_i 2^{-r(i)} \cdot \sum_j r(j) \qquad \text{(by Fact 10 stated below)} \\
&\leqslant \frac{k}{b} \cdot \mathbb{E}\left[core_A\right] \cdot kn = \frac{k^2 \cdot n}{b} \cdot \mathbb{E}\left[core_A\right]. \\
&\qquad \text{(as } \mathbb{E}\left[core_A\right] = \textstyle\sum_i 2^{-r(i)} \text{ as calculated earlier and } \sum_j r(j) \leqslant T = k \cdot n)
\end{aligned}$$

In the above calculations, we use the following fact.

Fact 10. *For any two lists of numbers $a_1 \leqslant a_2 \leqslant \ldots \leqslant a_n$ and $b_1 \geqslant b_2 \geqslant \ldots \geqslant b_n$, we have*

$$\sum_{i=1}^{n} a_i b_i \leqslant \frac{1}{n} \sum_{i=1}^{n} a_i \cdot \sum_{i=1}^{n} b_i.$$

Proof. The rearrangement inequality states that for any list of numbers $x_1 \leqslant \ldots \leqslant x_n$ and $y_1 \leqslant \ldots \leqslant y_n$ and any permutation σ of $[n]$,

$$x_1 \cdot y_n + \cdots + x_n \cdot y_1 \leqslant x_1 \cdot y_{\sigma(1)} + \cdots + x_n \cdot y_{\sigma(n)} \leqslant x_1 \cdot y_1 + \cdots + x_n \cdot y_n.$$

By rearrangement inequality, for any $0 \leqslant j < n$,

$$\sum_{i=1}^{n} a_i b_i \leqslant \sum_{i=1}^{n} a_i b_{i+j},$$

where, with a slight abuse of notation, we use b_{i+j} for $i + j > n$ to denote b_{i+j-n}. As such,

$$\sum_{i=1}^{n} a_i b_i \leq \frac{1}{n} \sum_{j=0}^{n-1} \sum_{i=1}^{n} a_i b_{i+j} = \frac{1}{n} \sum_{i=1}^{n} a_i \cdot \left(\sum_{j=0}^{n-1} b_{i+j}\right) = \frac{1}{n} \sum_{i=1}^{n} a_i \cdot \sum_{i=1}^{n} b_i,$$

concluding the proof. □

We can now conclude the proof of Lemma 9. By Chebyshev's inequality, we have,

$$\Pr\left(core_A \leqslant \frac{n}{2^{k+1}}\right) \leqslant \Pr\left(\left|core_A - \mathbb{E}\left[core_A\right]\right| \geqslant \frac{1}{2} \cdot \mathbb{E}\left[core_A\right]\right) \leqslant \frac{4 \cdot \mathrm{Var}\left[core_A\right]}{\mathbb{E}\left[core_A\right]^2} \leqslant \frac{4k^2 \cdot n \cdot 2^k}{b \cdot n} = \frac{1}{4},$$

by the choice of b. Since the distributions of $core_A$ and $core_B$ are the same, we can conclude the proof using a union bound. □

An immediate corollary of Lemma 9 is the following.

Corollary 11. *For any oblivious BP $\mathcal{B}$ with $T = kn$ and $b \geqslant 16k^2 \cdot 2^k$ time blocks, there exist a partition F_A, F_B of the blocks in $[b]$ such that*

$$|\mathtt{core}(F_A)| \geqslant \frac{n}{2^{k+1}} \quad \textit{and} \quad |\mathtt{core}(F_B)| \geqslant \frac{n}{2^{k+1}}.$$

We now have all the tools needed to prove Theorem 2.

Proof of Theorem 2. Let $\mathcal{B}$ be an oblivious BP for $\mathsf{ED}_{N,n}$ for $N \geqslant n^3$. Define the blocks $B_1, \ldots, B_b$ for $b = 16k^2 \cdot 2^k$ as before and let F_A, F_B be a partition of the blocks specified by Corollary 11 and let $m := n/2^{k+1}$. Consider this protocol π for $\mathsf{ED}_{A,B} : [N-n]^{2m} \to \{0,1\}$ for any equipartition A, B of $[2m]$.

Algorithm 1.

To get the protocol π on input $y \in [N-n]^{2m}$, we select m indices from $\mathtt{core}(F_A)$ for Alice, denoted by S_A and m indices from $\mathtt{core}(F_B)$ for Bob, denoted by S_B. We define $x \in [N]^n$ where $x_{S_A} = y_A$, $x_{S_B} = y_B$ and $x_{[n]\setminus(S_A \cup S_B)}$ is fixed to n distinct values $> N-n$. Note that x is well-defined although it is not known entirely by either Alice or Bob (the same way y is not known entirely by either of them). All of these can be done with no communication. The protocol now works as follows:

1. The players simulate $\mathcal{B}$ on x. Whenever a time block is in F_A, Alice simulates the execution of $\mathcal{B}(x)$ for that block. She can do this since she only reads indices from $\mathtt{read}(F_A)$ all of which are known to her, either as part of $S_A \subseteq \mathtt{core}(F_A)$ or in $[n] \setminus (S_A \cup S_B)$ which is fully fixed. Once she finishes the time block, Alice communicates the name of the vertex $\mathcal{B}$ that the computation lies on to Bob (assuming the next time block is in F_B, otherwise, she simply continues as above).
2. Bob can continue the simulation on time blocks in F_B exactly as above.
3. The players continue the simulation this way until they compute $\mathcal{B}(x)$. They will then return the same answer to $\mathsf{ED}_{A,B}(y)$ as what $\mathcal{B}(x)$ outputs.

The protocol π is deterministically correct since $\mathsf{ED}_{A,B}(y) = 1$ iff y_A and y_B contain distinct elements which happens iff x contains all distinct elements (since the fixed part of x is always distinct from the parts set to be y_A, y_B and thus cannot interfere). Thus, since $\mathcal{B}(x)$ is correct by our assumption, we get that π is also correct on all inputs of $\mathsf{ED}_{A,B}$.

Furthermore, this protocol has communication cost $\|\pi\| \leqslant b \cdot S$ bits to communicate the states of $\mathcal{B}$ between at most b transition between blocks, each requiring S bits to specify the vertex in the corresponding layer of the BP. Thus, π is a protocol for $\mathsf{ED}_{A,B}$ on $[n^3-n]^{2m}$ and $n^3 - n \geqslant 4n^2 \geqslant (2m)^2$; hence, by Theorem 6, we have $\|\pi\| = \Omega(m)$ which implies that

$$b \cdot S = \Omega(m).$$

Plugging in the choice of $m = n/2^{k+1}$, $b = 16k^2 2^k$, and $k = T/n$, we have

$$2^{\Theta(T/n)} \cdot S \geqslant n,$$

which implies that (by taking the log on both sides and using the fact $\log n - \log S = \log(n/S)$),

$$T = \Omega(n \cdot \log(n/S),$$

concluding the proof. □

2 Hamming Distance in General Branching Programs

Theorem 2 proved our first (in this course) lower bound for BPs for a decision problem, albeit by focusing on oblivious BPs. We now show how to prove a lower bound for general (non-oblivious) BPs, but for a harder problem defined as follows.

Problem 2. The **Hamming distance** problem with parameters $n, N \geqslant 1$ is defined as computing the function $\mathsf{HD} : [N]^n \to \{0,1\}$ where $\mathsf{HD}(x_1, \ldots, x_n) = 1$ iff for all pairs $i \neq j \in [n]$, $\|x_i - x_j\|_0 \geqslant \frac{1}{4} \cdot \log N$. Here, $\|x - y\|_0$ for two $x, y \in [N]$ represented as $\log N$-bits strings counts the number of bits by which x and y differ, namely, the Hamming distance between x, y.

Notice that HD in Problem 2 is quite related to the function ED in the element distinctness problem in Problem 1 – the difference is that the function ED outputs one iff for all pairs of input strings, the Hamming distance between them is non-zero, whereas HD outputs one iff for all all pairs of input strings, the Hamming distance is "non-trivially large".

We will prove the following theorem due to [BSSV03] for the Hamming distance problem.

Theorem 12. *Any S-space T-time branching program (not necessarily oblivious) for the Hamming distance problem for $N \geqslant n^{400}$ satisfies $T = \Omega\left(n \log\left(\frac{n \log n}{S}\right)\right)$. Specifically, as long as $S = O(n^{0.99})$, $T = \Omega(n \log n)$.*

We will follow our communication complexity approach for proving this lower bound also – the main step will be to account for the non-obliviousness of the BP. However, we first have to establish a communication complexity lower bound for HD itself.

2.1 Communication Complexity of Hamming Distance

We can define a communication problem out of HD exactly as before. Let A, B be any equipartition of $[n]$. The communication problem $\mathsf{HD}_{A,B}$ is defined as Alice having $x_A \in [N]^{n/2}$, Bob having $x_B \in [N]^{n/2}$ and the goal of the players is to compute $\mathsf{HD}(x)$ for $x = (x_A, x_B)$ where this notation means indices marked by A (resp. B) in x are filled with x_A (resp. x_B).

Theorem 13. *For $N \geqslant n^{200}$ and $\mathsf{HD}_{A,B} : [N]^A \times [N]^B \to \{0,1\}$ for any equipartition (A, B) of $[n]$,*

$$CC(\mathsf{HD}_{A,B}) = \Omega(n \log N).$$

We follow the same 3-step strategy as before: (1) lower bounding the number of 1s in the communication matrix of $\mathsf{HD}_{A,B}$; then, (2) bounding the size of 1-rectangles in the matrix, and finally, (3) using these along Proposition 5 to argue a lower bound on the communication cost of any protocol for HD.

As an aside, we note that the only reason we can use HD and not ED for the next steps of our arguments for (non-oblivious) BPs is the extra factor of $\log N$ in the lower bound of Theorem 13 for HD compared to that of Theorem 6 for ED. But to achieve this extra $\log N$ factor, the proof of Theorem 13 is inevitably more involved than Theorem 6.

Finally, before moving on, we note that it is going to be quite helpful to think of entries in $[N]$ as strings $\{0,1\}^t$ for $t = \log N = 200 \log n$ (by considering their bit-representation) directly. We will use this view throughout the proof.

The following lemma formalizes the first step of our approach.

Lemma 14. *The communication matrix M for $\mathsf{HD}_{A,B}$ has at least $2^{tn}/2$ many 1-entries.*

Proof. We use the probabilistic method. Observe that when a, b are sampled uniformly and independently from $\{0,1\}^t$, we have,

$$\mathbb{E}\|a-b\|_0 = \sum_{i=1}^{t} \Pr\left(a_i \neq b_i\right) = t \cdot \frac{1}{2},$$

by linearity of expectation and since each coordinate of a, b takes a value in $\{0,1\}$ uniformly and independently.

Thus, by an application of (additive) Chernoff bound, for $x \in \left(\{0,1\}^t\right)^n$ chosen uniformly at random,

$$\begin{aligned}
\Pr\left(\mathsf{HD}_{A,B}(x)=0\right) &= \Pr\left(\exists\ x_i, x_j \text{ such that } \|x_i - x_j\|_0 \leqslant \frac{t}{4}\right) \\
&\leqslant \binom{n}{2} \cdot \Pr_{a,b \in_R \{0,1\}^t}\left(\|a-b\|_0 \leqslant \frac{t}{4}\right) && \text{(by a union bound)} \\
&\leqslant \binom{n}{2} \cdot \Pr_{a,b \in_R \{0,1\}^t}\left(\left|\|a-b\|_0 - \mathbb{E}\left[\|a-b\|_0\right]\right| \geqslant \frac{t}{4}\right) && \text{(as } \mathbb{E}\|a-b\|_0 = t/2\text{)} \\
&\leqslant \binom{n}{2} \cdot \exp\left(-\frac{t^2}{32t}\right) && \text{(by (additive) Chernoff bound)} \\
&\leqslant \binom{n}{2} \cdot \exp\left(-6.25 \log n\right) \ll \frac{1}{2}.
\end{aligned}$$

Thus, at most (much less than) half of the entries x_A, x_B in the communication matrix can be 0, which proves the lemma. □

The next lemma allows us to show that despite M having many 1-entries, there are no large 1-rectangles in M. We first prove this for subsets of $\{0,1\}^t$ (and not $\left(\{0,1\}^t\right)^n$ that are inputs to HD).

Lemma 15. *Let $U, V \subseteq \{0,1\}^t$ with $|U \times V| \geqslant 2^{1.99t}$. Then is some $a \in U$ and $b \in V$ such that $\|a-b\|_0 \leqslant t/4$.*

Proof. Given the bound on the size of $U \times V$, we have that $\min\{|U|, |V|\} \geqslant 2^{0.99t}$. Define $N_{t/4}(U)$ as the following set:

$$N_{t/4}(U) := \left\{x \in \{0,1\}^t \mid \exists y \in U \text{ such that } \|x-y\|_0 \leqslant t/4\right\};$$

that is, all strings in $\{0,1\}^t$ obtained by changing at most $t/4$ coordinates of some string in U. Our goal is to prove that $N_{t/4}(U) \cap V \neq \emptyset$. We do this by proving that $|N_{t/4}(U)| > 2^t - 2^{0.99t}$, which immediately implies the lemma since $|V| \geqslant 2^{0.99t}$ and both sets are subsets of $\{0,1\}^t$ which has 2^t elements.

For any integer $d \in [t]$, define $B_t(d)$ as the Hamming ball of radius d (around 0) in $\{0,1\}^t$, i.e.,

$$B_t(d) := \left\{x \in \{0,1\}^t \mid \|x\|_0 \leqslant d\right\}.$$

Harper's *isoperimetric inequality* for hypercubes states that Hamming balls have the smallest vertex boundary among all sets of a given size. As a corollary,

$$|N_{t/4}(U)| \geqslant \max_d \ |B_t(d+t/4)| \quad \text{such that} \quad |B_t(d)| \leqslant 2^{0.99t} (\leqslant |U|). \tag{1}$$

At the same time, we have

$$|B_t(d)| = \sum_{i=0}^{d} \binom{t}{i}.$$

The standard approximation of this term, based on *binary entropy* $H_2(\cdot)$ then implies that as long as $d \leqslant t/2$,

$$\frac{1}{\sqrt{t}} \cdot 2^{t \cdot H_2(d/t)} \leqslant |B_t(d)| \leqslant 2^{t \cdot H_2(d/t)}.$$

Since $H_2(3/8) \approx 0.95$, we have that $|B_t(3t/8)| \leqslant 2^{0.99t}$ still. Plugging in $d = 3t/8$ as a feasible choice in Eq (1), we have,

$$|N_{t/4}(U)| \geqslant |B_t(5t/8)| = 2^t - |B_t(3t/8)| \geqslant 2^t - 2^{0.99t},$$

concluding the proof. □

We now use Lemma 15 to argue that the communication matrix M of $\mathsf{HD}_{A,B}$ does not have large 1-rectangles.

Lemma 16. *Any 1-rectangle $Y_A \times Y_B$ in the communication matrix M for $\mathsf{HD}_{A,B}$ has $\leqslant 2^{0.9975tn}$ entries.*

Proof. Define

$$S_A := \left\{ s \in \{0,1\}^t \mid \exists x_A \in Y_A \text{ such that } s \in x_A \right\} \quad \text{and} \quad S_B := \left\{ s \in \{0,1\}^t \mid \exists x_B \in Y_B \text{ such that } s \in x_B \right\}.$$

By Lemma 15, we know that $|S_A \times S_B| \leqslant 2^{1.99t}$ as otherwise, there will be some $a \in S_A$ and $b \in S_B$, with $\|a - b\|_0 \leqslant t/4$, which in turn implies there exists some $x_A \in Y_A$ and some $x_B \in Y_B$ causing $\mathsf{HD}(x) = 0$, contradicting $Y_A \times Y_B$ being a 1-rectangle.

This implies that $\min\{|S_A|, |S_B|\} \leqslant 2^{0.995t}$. Thus,

$$|Y_A \times Y_B| \leqslant |S_A|^{|A|} \cdot |S_B|^{|B|} = (2^{0.995t})^{n/2} \cdot (2^t)^{n/2} = 2^{0.9975tn},$$

concluding the proof. □

Theorem 13 can now be proven exactly as in Theorem 6.

Proof of Theorem 13. Fix any protocol π for $\mathsf{HD}_{A,B}$. By Proposition 5, the inputs consistent with leaf-nodes of the communication tree of π form monochromatic rectangle. Since these rectangles, that are $2^{\|\pi\|}$ many at most, should cover all 1s of the communication matrix M of $\mathsf{HD}_{A,B}$, we have,

$$2^{\|\pi\|} \cdot 2^{0.9975tn} \geqslant 2^{tn}/2,$$

by Lemma 16 in the LHS and Lemma 14 for the RHS. This implies that

$$\|\pi\| = \Omega(nt) = \Omega(n \log N),$$

by the choice of $t = 200 \log n = \log N$. This concludes the proof. □

2.2 A BP Lower Bound for Hamming Distance: Proof of Theorem 12

We are now ready to complete the proof of Theorem 12. We follow the strategy in the proof of Theorem 2 for oblivious BPs but now have to account for the non-obliviousness of the BP we work with.

Fix a BP $\mathcal{B}$ for HD with parameters N, n with $N \geqslant n^{400}$. Let depth (runtime) of $\mathcal{B}$ be $T = kn$ for some $k \geqslant 1$ and its space be S. Partition the T layers of $\mathcal{B}$ into b blocks as before and consider partitions $F_A, F_B \subseteq [b]$. Also recall the definitions of $\texttt{read}, \texttt{core}$ sets for $x \in [N]^n$ and F_A, F_B. Finally, defines O as the set of x's in $[N]^n$ with $\mathsf{HD}(x) = 1$.

We use the following corollary of Lemma 9 via the probabilistic method (this is the reason we proved the lemma for all choices of x even though for oblivious BPs we did not need it).

Corollary 17. *For any BP with runtime $T = kn$ and $b \geqslant 16k^2 \cdot 2^k$ time blocks, there exists a choice of F_A, F_B of $[b]$ such that for at least half of x's in O,*

$$|\texttt{core}(F_A, x)| \geqslant \frac{n}{2^{k+1}} \quad \text{and} \quad |\texttt{core}(F_B, x)| \geqslant \frac{n}{2^{k+1}}.$$

We fix a choice of F_A and F_B and let O^* be a half x's of O that satisfy the guarantees of Corollary 17. We further define $m := n/2^{k+1}$ for every $x \in O^*$, only consider its first m indices in $\texttt{core}(F_A, x)$ and $\texttt{core}(F_B, x)$ in the following.

Since $\mathcal{B}$ is not an oblivious BP, we do not have any guarantee that $\texttt{core}(F_A, x) = \texttt{core}(F_A, y)$ for $x, y \in O^*$ (and similarly for $\texttt{core}$ of F_B); this prevents us from implementing a reduction to the communication complexity as was done in the proof of Theorem 2. However, we do a "partial reduction" in the following, using a quite similar strategy.

Now that we have fixed $\texttt{core}(F_A, x), \texttt{core}(F_B, x) \subseteq [n]$ for $x \in O^*$ to be of size exactly m (or rather consider only m indices in each), the number of choices for the pair $(\texttt{core}(F_A, x), \texttt{core}(F_B, x))$ is at most

$$\binom{n}{m}^2 \leqslant \left(\frac{4n}{m}\right)^{2m} = \left(2^{k+2}\right)^{2m} = 16 \cdot 4^{km}.$$

Thus, by pigeonhole principle, there exists a choice of $O^{**} \subseteq O^*$ with

$$|O^{**}| \geqslant \frac{|O^*|}{16 \cdot 4^{km}} \geqslant \frac{N^n}{64 \cdot 4^{km}}, \tag{2}$$

such that for all $x, y \in O^{**}$, $\texttt{core}(F_A, x) = \texttt{core}(F_A, y) := S_A$ and $\texttt{core}(F_B, x) = \texttt{core}(F_B, y) := S_B$; here, the second inequality is by Lemma 14 and since O^* is of size at least half of O (by Corollary 17). We fix O^{**} and choices S_A and S_B. Finally, for each $z \in [N]^{n-2m}$, define

$$O^{**}_z := \left\{x \in O^{**} \mid x_{[n]\setminus(S_A \cup S_B)} = z\right\},$$

namely, O^{**}_z are all those x's in O^{**} whose values outside S_A, S_B is fixed to be z. We have,

$$\mathop{\mathbb{E}}_{z \in [N]^{n-2m}} |O^{**}_z| = N^{-(n-2m)} \cdot \sum_z |O^{**}_z| = N^{-(n-2m)} \cdot |O^{**}| \geqslant \frac{N^{2m}}{64 \cdot 4^{km}},$$

by Eq (2). Thus, by an averaging argument, there exists some choice of $z \in [N]^{n-2m}$ such that

$$|O^{**}_z| \geqslant \frac{N^{2m}}{64 \cdot 4^{km}}. \tag{3}$$

We further fix the choice of z. Now, consider the following protocol for $\mathsf{HD}_{A,B} : [N]^{2m} \to \{0, 1\}$.

Algorithm 2.

To get the protocol π on input $y \in [N]^{2m}$, we define $x \in [N]^n$ where $x_{S_A} = y_A$, $x_{S_B} = y_B$ and $x_{[n]\setminus(S_A \cup S_B)} = z$ is fixed (we write $x = x(y_A, y_B, z)$ to denote this unique x from here on). This can be done with no communication. The protocol now works as follows:

1. The players simulate $\mathcal{B}$ on x by simply "hoping" it belongs to O^{**}_z: whenever a time block is in F_A, Alice simulates the execution of $\mathcal{B}(x)$ for that block – if at any point, $\mathcal{B}$ queries an index not consistent with $\texttt{core}(F_A, x)$ being S_A, Alice terminates the protocol and outputs 'fail' instead. Otherwise, every query she makes are from indices in $[n] \setminus S_B$, all of which are known to her. Once she finishes the time block, Alice communicates the name of the vertex $\mathcal{B}$ that the computation lies on to Bob (assuming the next time block is in F_B, otherwise, she simply continues as above).
2. Bob can continue the simulation on time blocks in F_B the same way, outputting 'fail' when needed.
3. The players continue the simulation this way until they compute $\mathcal{B}(x)$. They will then return the same answer to $\mathsf{ED}_{A,B}(y)$ as what $\mathcal{B}(x)$ outputs.

We should note right away that protocol π in Algorithm 2 is *not* (necessarily) a correct protocol for $\mathsf{HD}_{A,B}$ as there are many cases that it outputs 'fail' instead of solving the problem. Moreover, when it outputs 1, it is always correct, but when it outputs 0, it is either the case that there exists $i, j \in [2m]$ such that $\|y_i - y_j\|_0 \leqslant 1/4 \log N$, *or*, there exists $i \in [2m]$ and $j \in [n] \setminus [2m]$ such that $\|y_i - z_j\|_0 \leqslant 1/4 \log N$. In the following, we will prove that even such a protocol requires a large communication cost. The proof follows that of Theorem 13 closely.

Claim 18. *π outputs 1 on at least $N^{2m}/(64 \cdot 4^{km})$ inputs (y_A, y_B) such that $x = x(y_A, y_B, z)$ is in O^{**}_z.*

Proof. Any input (y_A, y_B) such that the resulting $x = x(y_A, y_B, z)$ is inside O^{**}_z will result in π outputting 1, as by the definition of O^{**}, $\texttt{core}(F_A, x)$ and $\texttt{core}(F_B, x)$ will be consistent with S_A and S_B and thus the protocol does not output 'fail' and answers correctly. The claim follows from Eq (3) now. □

Claim 19. *Let v be any leaf-node of the communication tree of π and $Y^v_A \times Y^v_B$ be the set of inputs consistent with this node (as per Proposition 5). If there exists $y_A \in Y^v_A$ and $y_B \in Y^v_B$ such that $x = x(y_A, y_B, z)$ is in O^{**}_z, then, the protocol π outputs 1 on the entire $Y^v_A \times Y^v_B$.*

Proof. By Proposition 5, the inputs at each node of the tree is always a combinatorial rectangle. At a leaf-node, the answer of the protocol is fixed to be 0, 1, or 'fail' (the protocol terminates at the node and always answers the same). Since we know π will output 1 on the input (y_A, y_B) specified in the claim, it should output 1 on the entire rectangle $Y^v_A \times Y^v_B$. □

We can now use the above to bound the communication cost of π.

Lemma 20. *For the protocol π in Algorithm 2,*

$$\|\pi\| \geqslant 0.01m \cdot \log N - 3k \cdot m.$$

Proof. By Claim 18, π outputs 1 on at least $N^{2m}/(64 \cdot 4^{km})$ inputs all of which, by Claim 19 belong to 1-rectangles in the communication tree of π. On the other hand, by Lemma 16, every 1-rectangle here can only have size $2^{1.99tm}$ where $t = \log N$ as otherwise π will be forced to output something other than 1 on one of its entries. Therefore, we have,

$$2^{\|\pi\|} \cdot N^{1.99m} \geqslant N^{2m}/(64 \cdot 4^{km}),$$

which implies that

$$\|\pi\| \geqslant 0.01m \cdot \log N - 3k \cdot m,$$

concluding the proof. □

We can now finalize the proof of Theorem 12.

Proof of Theorem 12. Consider the BP $\mathcal{B}$ specified earlier and the resulting protocol π of Algorithm 2 from it. By construction, we have that $\|\pi\| \leqslant b \cdot S$ as the players communicate at most b times, each needing to specify one of 2^S vertices in a layer. By Lemma 20, this implies that

$$b \cdot S \geqslant 0.01m \cdot \log N - 3k \cdot m.$$

Suppose we ensure that $k \leqslant \log N/400$, then, given the choice of $b = 16k^2 \cdot 2^k$ in Corollary 17 and $m = n/2^{k+1}$,

$$S \geqslant \Omega(1) \cdot \frac{n \log N}{2^{3k}},$$

which in turn, since $k = T/n$, implies that

$$2^{3T/n} = \Omega\left(\frac{n \log n}{S}\right),$$

which implies that

$$T = \Omega\left(n \log\left(\frac{n \log n}{S}\right)\right).$$

Finally, the guarantee that $k \leqslant \log N/400$ is without loss of generality because we have $T \leqslant n \log n$ always in the above and $N = n^{400}$. This concludes the proof. □

Remark. The best time lower bound one can achieve using Theorem 12 is of the type $T = \Omega(n \log n)$ (when $S = O(\log n)$ bits). This may sound underwhelming at first glance. For instance, in the last lecture, we saw a lower bound of $T \approx n^2$ at least for algorithms with $O(\log n)$ space. However, it is worth noting that the situation for *decision* problems is a lot trickier than multi-output problems we study in the last lecture. To put this in perspective, note that to date, we still do not have *any* $\omega(n)$ time lower bounds for any explicit problem even in **NP** (with no restriction on the space) – Theorem 12 shows that at least when focusing on $O(\log n)$ space (or even $n^{0.99}$ space) algorithms, one can prove $\omega(n)$ time lower bounds. It is also worth noting that the lower bounds of [BSSV03] remain more or less the state-of-the-art for decision problems in branching programs for over 20 years now.

References

[BSSV03] Paul Beame, Michael E. Saks, Xiaodong Sun, and Erik Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *J. ACM*, 50(2):154–195, 2003. 1, 2, 9, 14