TOPIC 7
Communication Complexity and Monotone Depth

(Related sections: 15.2, 18)

Summary: Minterm-maxterm games. Proof that s-t connectivity of directed n-vertex graph cannot be solved by monotone circuits of depth $o(\log^2 n)$. The amplification argument.

We will consider games between two players, Alice and Bob, where communication between players is allowed. However, we will assume that the players are far away from each other, so that each communicated bit costs a money. The goal is to pay (and hence to communicate) as few as possible. The players are not adversaries; they help and trust each other.

7.1. The Karchmer–Wigderson game

We will restrict ourselves with one particular type of games related to the depth of boolean circuits. Let A, B be two disjoint subsets of $\{0,1\}^n$. The Karchmer–Wigderson game on this pair is the following game.

- Alice gets a vector $a \in A$.
- Bob gets a vector $b \in B$.
- The goal is to find an i such that $a_i \neq b_i$.

Before the game starts, the players agree on a "protocol" for exchanging messages. After that, given an input pair (a, b), the protocol dictates to each player what messages to send at each point, based on his/her input and the messages be received so far. It also dictates when to stop, and how to determine the answer from the information received. There is no limit on the computational complexity of these decisions, which are free of charge. The cost of the protocol is the number of bits they have to exchange on the *worst case* choice of input pair (a, b). The *communication complexity* $C(A, B)$ of the game on the pair A, B is the cost of the best protocol for this game.

More formally, this measure can be defined as follows.

A *communication matrix* for this game is an $|A| \times |B|$ matrix M whose rows are indexed by vectors from A and columns are indexed by vectors from B. The entry, corresponding to the row a and column b, is the set $M(a, b) = \{i : a_i \neq b_i\}$. By sending bits 0 and 1, the players actually split the rows (if this bit is send by Alice) or columns (if this bit is send by Bob) into two disjoint parts.

A *communication protocol* (or a *communication tree*) T for this game is a binary tree, each inner node of which correspond to a decision made by one of the players at this node. The tree has the following properties:

1. The root is labeled by the whole communication matrix M.

2. If a node u is labeled by a matrix N, then the sons of u are labeled by the corresponding submatrices N_0 and N_1 of N. Moreover, these submatrices are obtained from N by splitting the rows of N (if u is Alice's node) or by splitting the columns of N (if u is Bob's node).
3. If w is a leaf and N is its label, then the entries of N must share a common element.[1]

Since at each node, the rows (or columns) of the corresponding submatrix are splitted into *disjoint* parts, the protocol is *deterministic*: each pair (a, b) will reach precisely one leaf. The depth of a tree is the maximum number of edges from the root to a leaf. The minimum depth of a communication tree is the *communication complexity* $C(A, B)$ of the game on the pair A, B. The *communication complexity* $C(f)$ of a boolean function f is $C(A, B)$ with $A = f^{-1}(0)$ and $B = f^{-1}(1)$.

7.1.1. Communication complexity and the cover number

Let M be the communication matrix of the KW game on a pair A, B. Recall that the entry, corresponding to the row $a \in A$ and column $b \in B$, is the set $M(a, b) = \{i : a_i \neq b_i\}$. A *rectangle* is a submatrix R of M such that all its entries share a common element. The cover number $\alpha(A, B)$ of a pair A, B is the minimum number of mutually disjoint rectangles whose union cover whole communication matrix M of this pair.

Proposition 7.1. *For every pair A, B, $C(A, B) \geq \log \alpha(A, B)$.*

> **Ex. 7.2.** Prove this proposition. *Hint*: Note that each leaf of a communication protocol gives us a rectangle.

7.1.2. Communication complexity and the depth of circuits

Let $D(f)$ be the minimum depth of a formula with AND, OR and NOT gates computing f. The following theorem connects communication with computation.

Theorem 7.3 (Karchmer–Wigderson 1988). *For every Boolean function f,*

$$D(f) = C(f).$$

We prove lower and upper bounds on $D(f)$ separately.

Lemma 7.4. $C(f) \leq D(f)$.

Proof. We may assume that Alice and Bob have agreed on a formula of depth $D(f)$ computing f. Further, we may assume, using de Morgan's laws, that negations are applied only to the variables (leaves) and, thus, that all gates are ANDs and ORs.

Now suppose Alice gets an input a such that $f(a) = 0$, and Bob gets an input b such that $f(b) = 1$. In oder to find an i such that $a_i \neq b_i$, the players use the information provided by the underlying formula. At AND gates speaks Alice, and at OR gates speaks Bob.

Suppose the output gate is an AND-gate, that is, we can write $f = f_0 \wedge f_1$. Then Alice sends a bit i corresponding to a function f_i such that $f_i(a) = 0$; if both $f_0(a)$ and $f_1(a)$ output 0, then Alice send 0. We know that we must have $f_i(b) = 1$. We can then repeat this step at the gate corresponding to the output gate of f_i, where Alice sends a bit if the gate is an AND gate and Bob sends a bit if the gate is an OR gate (he sends a bit corresponding to a function which outputs 1). Alice and

[1] Recall that entries are *subsets* of $\{1, \ldots, n\}$.

Bob repeat this process until they reach a leaf. This leaf is labeled by some variable x_i or its negation $\overline{x}_i$. Hence, $a_i \neq b_i$ implying that i is a correct answer. $\square$

Lemma 7.5. *If $A, B \subseteq \{0,1\}^n, A \cap B = \emptyset$, and $C(A,B) \leq c$, then there exists a function $f : \{0,1\}^n \to \{0,1\}$ such that $f(A) = 0, f(B) = 1$, and $D(f) \leq c$. In particular, $D(f) \leq C(f)$.*

Proof. We prove the lemma by induction on c. Suppose $c = 0$. Then we must have, for some index i, that $a_i \neq b_i$ for all pairs (a,b) in $A \times B$. Thus we may choose either $f = x_i$ or $f = \overline{x_i}$ according to which function satisfies $f(A) = 0$ and $f(B) = 1$.

Next, we prove the claim is true for c assuming it is true for $c-1$. Consider a protocol for the communication game on A, B that uses at most c bits. Let us assume Alice sends the first bit. Then there is a partition $A = A_0 \cup A_1, A_0 \cap A_1 = \emptyset$, such that for $a \in A_0$, Alice sends the bit 0 and for $a \in A_1$, Alice sends the bit 1. We then know that $C(A_0, B) \leq c-1$ and $C(A_1, B) \leq c-1$. Applying our induction hypothesis, we find there exists a function f_0 such that $f_0(A_0) = 0, f_0(B) = 1$, and $D(f_0) \leq c-1$, and there exists a function f_1 such that $f_1(A_1) = 0, f_1(B) = 1$, and $D(f_1) \leq c-1$. We define $f = f_0 \wedge f_1$. Then $f(A) = 0, f(B) = 1$, and

$$D(f) \leq 1 + \max\{D(f_0),\, D(f_1)\} \leq c$$

as desired. Note that, if Bob had sent the first bit, we would have partitioned B and defined $f = f_1 \vee f_2$. $\square$

In a *monotone version* of Karchmer–Wigderson game on a pair A, B, given an input pair (a,b) in $A \times B$, the players must find an i such that $a_i = 0$ and $b_i = 1$. Note that, in general, this game may be not well defined: if a is the all-1 vector and b is the all-0 vector, then no valid answer exists. However, if $A \subseteq f^{-1}(0)$, $B \subseteq f^{-1}(1)$ and if f is a *monotone* boolean function, then every pair (a,b) has a valid answer.

For a monotone boolean function f, let $C_+(f)$ be the communication complexity of a monotone version of the Karchmer–Wigderson game on the pair A, B with $A = f^{-1}(0)$ and $B = f^{-1}(1)$. Let $D_+(f)$ be the minimum depth of a boolean formula with AND and OR gates computing f.

Theorem 7.6. *For every monotone boolean function f, $C_+(f) = D_+(f)$.*

Ex. 7.7. Prove this theorem.

7.2. The st-connectivity problem

We consider directed graphs on $m+2$ vertices with two exhibited vertices s (the source) and t (the target). The *st-connectivity problem $STCON_n$* is, given a directed graph, to determine whether it contains a path from s to t. Obviously, $STCON_m$ is a monotone function: if we add edges we cannot disconnect an existing path from s to t.

In this lecture we will show that any monotone circuit solving this problem has depth $\Omega((\log m)^2)$. We will do this by proving this lower bound on the communication complexity of the corresponding Karchmer–Wigderson game: [2]

[2]For convenience, we have reversed the roles of players (now Alice gets inputs from $f^{-1}(1)$, not from $f^{-1}(0)$). This does not change anything: since we consider only *deterministic* games, a function and its complement have the same complexity.

- Alice gets a graph G with $STCON(G) = 1$.
- Bob gets a graph H with $STCON(H) = 0$.
- The goal is to find an edge (u, v) which is present in G but is absent in H.

Since we are interested in proving *lower* bounds on the communication complexity of this game, we can restrict our attention to special inputs.

- Alice gets a directed path p form s to t.
- Bob gets a coloring c of vertices by the colors 0 and 1 such that $c(s) = 0$ and $c(t) = 1$.
- The goal is to find an edge $(u, v) \in p$ such that $c(u) = 0$ and $c(v) = 1$.

Let $C(STCON_m)$ denote the communication complexity of this game. Note that every protocol for the KW game can be used to solve this (restricted) game: given a coloring c, Bob converts it into a graph H in which (u, v) is an edge iff $c(u) = c(v)$.

Ex. 7.8. Prove that $C(STCON_m) = O((\log m)^2))$. *Hint*: Use binary search; in fact one of the players may do most of the talking, with the other player communicating only $O(\log n)$ bits overall.

7.3. The fork game

For positive integers n and l, let $[n]^l$ denote the set of all strings $a = (a_1, \ldots, a_l)$ with $a_i \in [n] = \{1, \ldots, n\}$. A *fork position* of two strings $a = (a_1, \ldots, a_l)$ and $b = (b_1, \ldots, b_l)$, a *fork point* is a position i such that $a_i \neq b_i$ and, if $i > 1$ then $a_{i-1} = b_{i-1}$.

The fork game on a subset $S \subseteq [n]^l$ is the following game:

- Alice gets a string $a = (a_1, \ldots, a_l) \in S$.
- Bob gets a string $b = (b_1, \ldots, b_l) \in S$.
- The goal is to find a fork position i of a and b. If $a_l = b_l$ then $i = l$ is also a legal answer.

Given a pair of strings $a, b \in [n]^l$, the players follow some agreed-upon deterministic protocol of communicating bits back and forth until in the end they have agreed on an answer $i \in \{0, 1, \ldots, l\}$. The communication complexity of a protocol is the total number of communicated bits on the worth-case input pair a, b.

Let $C(FORK_{n,l})$ denote the communication complexity of the fork game game on the whole set $S = [n]^l$.

7.3.1. Reducing $FORK$ to $STCON$

We can relate this game to the previous (s-connectivity) game as follows. Look at $[n]^l$ as an $n \times l$ grid whose nodes are arranged into levels $V_1, \ldots, V_l$, each with n nodes. Add two new levels $V_0 = \{s\}$ and $V_{l+1} = \{t\}$. and each other level consists of n vertices. We restrict our attention to graphs on $m = n \cdot l$ vertices, where only edges, connecting nodes from adjacent levels, are allowed.

Lemma 7.9. $C(FORK_{n,l}) \leq C(STCON_{n \cdot l})$.

Proof. Suppose we have a protocol Π for the Karchmer–Wigderson game corresponding to $STCON_{n \cdot l}$. We will show that this protocol can be used for the game $FORK_{n,l}$. To use the protocol Π, the players must convert their inputs $a = (a_1, \ldots, a_l)$ and $b = (b_1, \ldots, b_l)$ (for the fork game) to inputs for the st-connectivity game.

The players convert their inputs in the following way. Alice converts her input $(a_1, \ldots, a_l)$ into a path $p = (u_0, u_1, \ldots, u_l, u_{l+1})$ where $u_0 = s$, $u_{l+1} = t$, and $u_i = a_i$ for $1 \le i \le l$. Bob converts his input $(b_1, \ldots, b_l)$ into a coloring c by assigning color 0 to all vertices $s, b_1, \ldots, b_l$, and assigning color 1 to the remaining vertices; hence, $c(s) = 0$ and $c(t) = 1$ (because $t \notin \{s, b_1, \ldots, b_l\}$).

The players now can use the protocol Π to get an edge $(u_{i-1}, u_i) \in p$ such that $c(u_{i-1}) = 0$ and $c(u_i) = 1$. This means that u_{i-1} is in $(s, b_1, \ldots, b_l)$ and u_i is not. We claim that i is a valid answer for the fork game on the pair a, b. Indeed, if $i = 1$ then $u_{i-1} = s$, and hence, $a_1 \neq b_1$. If $1 < i \le l$ then $u_{i-1} = a_{i-1}$ and $u_i = a_i$; hence, $a_{i-1} = b_{i-1}$ and $a_i \neq b_i$. Finally, if $i = l + 1$ then $a_l = b_l = u_l$, and l is a legal answer for the fork game. □

7.4. The lower bound

By Lemma 7.9 and Exercise 7.8 we know that $C(FORK_{n,n}) = O((\log n)^2)$. We will show that this trivial upper bound is almost optimal.

Theorem 7.10 (Grigni–Sipser 1991). *Any communication protocol for a string game on the set $[n]^l$ requires at least $\Omega((\log n) \cdot (\log l))$ bits of communication. That is,*

$$C(FORK_{n,l}) = \Omega((\log n) \cdot (\log l)).$$

Proof. Call a two-player protocol an (α, l)-*protocol* if there is some subset $S \subseteq [n]^l$ such that $|S| \ge \alpha n^l$ and for all $a, b \in S$ the protocol finds a fork position of a and b or outputs l if $a_l = b_l$. Denote by $C(\alpha, l)$ the minimum communication complexity of an (α, l)-protocol. That is, if $C(\alpha, l) = d$ then there exists a subset $S \subseteq [n]^l$ of $|S| \ge \alpha n^l$ strings and a protocol Π of communication complexity d such that Π works correctly on S. In particular, $C(1, n) = C(FORK_{n,l})$.

We start with two simple claims.

Claim 7.11. *For any $l \ge 1$ and any $\alpha > 1/n$, $C(\alpha, l) > 0$.*

Proof. Suppose that $C(\alpha, l) = 0$. Thus, there exists a subset of strings $S \subseteq [n]^l$ such that $|S| \ge \alpha n^l > n^{l-1} \ge 1$ such that the players know the answer $i \in \{0, 1, \ldots, l\}$ without any communication. Since $|S| \ge 2$, the set S contains at least two different strings $a \neq b$. But on input (a, a) the answer must be $i = l$, whereas on input (a, b) the answer must be some $i < l$, a contradiction. □

Claim 7.12. *For every $l \ge 1$ and α, if $C(\alpha, l) > 0$ then $C(\alpha, l) \ge C(\alpha/2, l) + 1$.*

Proof. Let $d = C(\alpha, l)$. Thus, there exists a subset $S \subseteq \{1, \ldots, n\}^l$ such that $|S| \ge \alpha n^l$ and there is a protocol Π such that for all $a, b \in S$, the protocol correctly solves the game on these inputs. Assume w.l.o.g. that Alice speaks first (the case when Bob speaks first is similar). Hence Alice sends either 0 or 1. After this (first) bit is communicated, the set S is splitted into two parts S_0 and S_1. Assume w.l.o.g. that $|S_0| \ge |S_1|$. Let Π_0 be the rest of the protocol Π, after assuming that the first bit send by Alice was 0. That is, Π_0 works exactly like Π, but without sending the first bit, and continuing as if the value of the first bit was 0. The communication complexity of Π_0 is at most $d-1$. Obviously, Π_0 must work correctly on S_0, because Π does this on the whole set $S = S_0 \cup S_1$. Hence, Π_0 is an $(\alpha/2, l)$-protocol. Thus, $C(\alpha/2, l) \le d - 1 = C(\alpha, l)$. □

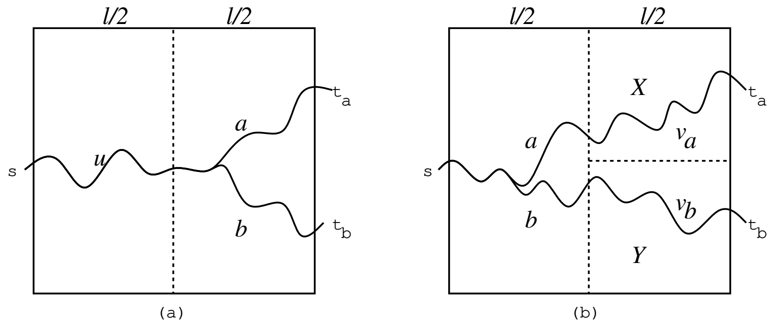

Figure 1. Two cases for constructing a protocol for strings a and b of length $l/2$.

Applying Claim 8.7 $t = (\log n)/2$ times, we obtain that $C(1,n) \geq C(\alpha, n) + t$ with $\alpha = 1/\sqrt{n}$. Since $\alpha > 1/n$, Claim 8.6 yields $C(1,n) = \Omega(\log n)$. This lower bound is, however, too weak.

We now turn to an amplification step: given an (α, l)-protocol (with $l \geq 2$ and α not too small), we convert it to a $(\sqrt{\alpha}/2, l/2)$-protocol. Thus α may be amplified greatly[3] while l is cut in half. By amplifying α after every $\Theta(\log n)$ steps of the traversal strategy, we may keep $\alpha > 1/n$ until l reaches 1, showing the protocol has a path of depth at least $\Omega(\log^2 n)$.

Ex. 7.13. Let $S \subseteq [n]^k$ and $|S| \geq \alpha \cdot n^k$. Show that then the projection of S onto some set of $k/2$ coordinates contains at least $\sqrt{\alpha} \cdot n^{k/2}$ strings.

Lemma 7.14. (Amplification Lemma) *For every $l \geq 2$ and $\alpha \geq 1/\sqrt{n}$,*

$$C(\alpha, l) \geq C(\frac{\sqrt{\alpha}}{2}, \frac{l}{2}).$$

Proof. We are given an (α, l)-protocol working correctly on some set $S \subseteq [n]^l$ of $|S| \geq \alpha n^l$ strings. Consider a bipartite graph G with parts U and V where U consists of all $n^{l/2}$ possible strings on the first $l/2$ levels, and V consists of all $n^{l/2}$ possible strings on the last $l/2$ levels. We connect u and v if their concatenation uv is a string in S; we say that v is an *extension* of u. Hence, G is a bipartite graph with parts of size $n^{l/2}$ and $|S| \geq \alpha n^l$ edges.

By Lemma 2.7 from the book (re-formulated in terms of bipartite graphs), at least one of the following two must hold:

(a) Some $u \in U$ has at least $\sqrt{\frac{\alpha}{2}}\,|V|$ extensions.

(b) There is an $U' \subseteq U$ such that $|U'| \geq \sqrt{\frac{\alpha}{2}}\,|U|$ and each $u \in U'$ has at least $\frac{\alpha}{2}\,|V|$ extensions.

In both cases (a) and (b), we show how to construct a $(\sqrt{\alpha}/2, l/2)$-protocol (see Fig. 1).

Case (a) In this case, we have one string u on the left that has many extensions v on the right such that $uv \in S$. Thus we can recover a $(\sqrt{\alpha/2}, l/2)$-protocol as follows: let S' be the set of extensions of u. Given two strings $v_a, v_b \in S'$, the two players can play the S'-game on these inputs by following the S-protocol for the

[3]Note that $\sqrt{\alpha}/2$ is at least twice larger than α, if $\alpha \leq 1/16$.

strings uv_a, uv_b. Since these strings are identical on the first $l/2$ coordinates, the answer i must correspond to a point where (the paths corresponding to) v_a and v_b diverge. [4]

Case (b) In this case, we take a random partition of the $nl/2$ nodes in the right $l/2$ levels. More precisely, take $n/2$ nodes at random from each of the right $l/2$ levels, and call their union X; call the set of remaining $nl/4$ right nodes Y.

Claim 7.15. *Given case (b) holds and we take a string $u \in U'$, then with probability at least $1 - 2e^{-\alpha n/4}$ there is an extension $v_X(u)$ of u entirely in set X and another extension $v_Y(u)$ of u entirely in set Y.*

Proof of Claim 7.15. We prove that such an extension $v_X(u)$ exists with probability $1 - e^{-\alpha n/4}$, and then add failure probabilities. We know that u has a set $S \subseteq [n]^{l/2}$ of $|S| \geq \frac{\alpha}{2}\,|V| = \frac{\alpha}{2}\,n^{l/2}$ extensions. Since the probability that a string $v \in S$ lies entirely in the set X is at least[5] $2^{-l/2}$, the probability that *none* of the strings from S lies entirely in X does not exceed

$$(1 - 2^{-l/2})^{|S|} < e^{-|S|/2^{l/2}} \leq e^{-(\alpha/2)\,(n/2)^{l/2}} \leq e^{-\alpha n/4}$$

Hence, the probability that X contains no extension of u does not exceed $e^{-\alpha n/4}$. Since the set of points Y has the same distribution as X, the same argument yields that Y contains no extension of u with probability at most $e^{-\alpha n/4}$. Adding these failure probabilities we obtain that, with probability at least $1 - 2e^{-\alpha n/4}$ there is an extension of u entirely in set X *and* another extension of u entirely in set Y. $\square$

Now if we have $\alpha \gg 1/n$, then $1 - o(1)$ of strings $u \in U'$ will have extensions in both X and Y.

> **Ex. 7.16.** Prove this. *Hint*: Say that $u \in U'$ is *good* if u has extensions in both X and Y. Use Markov's inequality to show that with probability at least $1/2$, at least $|U'|\left(1 - 4e^{-\alpha n/4}\right)$ nodes of U' are good.

In particular if we always keep $\alpha \geq n^{-1/2}$, then (for n large enough) there exists some choice of X and Y such that[6] $\sqrt{\alpha}/2$ of all possible left-half strings have extensions in both X and Y. This set of extendable strings is our new S'.

This yields a $(\sqrt{\alpha}/2, l/2)$-protocol as follows. Given strings $u_a, u_b \in S'$, the players follow the S-protocol on the inputs $u_a v_X(u_a), u_b v_Y(u_b)$. Since the S-protocol is correct on these strings, and since they share no vertices in the right $l/2$ levels, the protocol must return an answer i in the first $l/2$ levels, hence the answer is in fact valid for u_a and u_b.

This completes the proof of Lemma 7.14. $\square$

Now we can finish the proof of Theorem 7.10 as follows.

By $t = O(\log n)$ applications of Claim 8.7 and one application of the Amplification Lemma (Lemma 7.14), we obtain that

$$C(\frac{2}{\sqrt{n}}, l) \geq C(\frac{16}{n}, n) + \Omega(\log n) \geq C(\frac{2}{\sqrt{n}}, l) + \Omega(\log n).$$

---

[4]Strictly speaking, we must consider strings v_a, v_b as elements of $[n]^{l/2}$. Hence, if i is the answer of the S-protocol on uv_a, uv_b, then the answer of the S'-protocol is $i - l/2$.

[5]Because each v_i belongs to X with probability $1/2$, and this probability is the same for each i.

[6]Since $\frac{1}{2} \leq \frac{1}{\sqrt{2}}(1 - o(1))$ (we are quite generous here).

Applying the last inequality $O(\log l)$ times and using Claim 8.6, we obtain

$$C(\frac{2}{\sqrt{n}}, l) \geq C(\frac{2}{\sqrt{n}},\ 1) + \Omega(\log n) \cdot \Omega(\log l).$$

Hence,

$$C(FORK_{n,l}) = C(1,n) \geq C(\frac{2}{\sqrt{n}}, n) = \Omega((\log n)^2).$$

□