CS 2429 - Foundations of Communication Complexity

Lecturer: Toniann Pitassi

1 Monotone Circuit Depth Lower Bounds

Today we will finish the proof that there are monotone functions that require near linear monotone circuit depth.

Recall from last class that we will start with the CNF search problem associated with Tseitin, and lift it to obtain a lifted search problem. From the theorem from last class, this implies that there is a corresponding monotone function such that the KW communication complexity of this monotone function is large, if the communication complexity of the lifted Tseitin search problem is large.

In particular, $Tseitin$ on a graph with n vertices has n constraints, and we will select a gadget g with $c = 2$. Thus, our monotone function will have $4n$ inputs and its monotone depth will be at least the communication complexity of $S(Tseitin(G))og^n$.

By using a different family of unsatisfiable CNF formulas, it is also possible to obtain a function that is in monotone P and that requires monotone circuits of depth $\sqrt{n}$; however we will not do this here.

For the remainder of the lecture, we will prove that for appropriate graphs G, the communication complexity of $S(Tseitin(G))og^n$ is large.

2 Block sensitivity, Critical Block Sensitivity, Decision Tree Depth

Define block sensitivity, and critical block sensitivity.

Fix a search problem $S \subseteq \{0,1\}^n \times Q$. An assignment α is said to be *critical* if it has a unique feasible solution.

Let $f \subset S$ be a total function – for each input α, f picks out some feasible solution $f(\alpha)$ for α.

The block sensitivity of f at α, $bs(f,\alpha)$, is the maximal number bs such that there are disjoint blocks of coordinates $B_1, \ldots, B_{bs} \subset [n]$ satisfying $f(\alpha) \neq f(\alpha^{B_i})$ for all i. Here, α^{B_i} represents the assignment α where the block of variables B_i has been flipped.

The block sensitivity of f, $bs(f)$ is then the maximal block sensitivity over all assignments α. The block sensitivity of a search problem S is just the minimum over all total functions $f \subset S$ of the block sensitivity of f.

The critical block sensitivity of S, $bs_{crit}(S)$ is the minimum over all $f \subset S$ of the maximum over all critical assignments α, of $bs(f,\alpha)$. So the critical block sensitivity is just like the block sensitivity but we only look over critical assignments.

If a search problem has high critical block sensitivity, then it has high block sensitivity, which in turn implies high decision tree complexity.

It turns out that for the appropriate choice of graph G, that the Tseitin CNF formulas have very high critical block sensitivity.

Theorem 1. *There are bounded-degree graphs G_n such that $bs_{crit}(S(Tseitin(G_n)) = \Omega(n/\log n)$.*

3 Defining the Gadget

A "nice" two-party gadget g will have some nice properties. Let f_1 and f_2 be two-party functions. We say that f_1 reduces to f_2, written $f_1 \leq f_2$ if the communication matrix of f_1 appears as a submatrix of f_2. This is equivalent to their being functions π_A and π_B such that $f_1(x, y) = f_2(\pi_A(x), \pi_B(y))$ for all x, y. In other words, if f_1 reduces to f_2, then from a short communication complexity for f_2, Alice and Bob can solve f_1. (Furthermore, the reduction is of the simple type where Alice without communication maps her input x to some x' and similarly Bob without communication maps y to y' and then they run f_2 on (x', y') and the answer returned will be equal to $f(x, y)$.)

We want our gadget g to have three reducibility properties. First, $\neg g$ is reducible to g. Secondly the AND function is reducible to g. Thirdly, g is random self reducible. A gadget g satisfying these three properties will be called *versatile*.

Lemma 2. *The following 2-party function g from $X \times Y \rightarrow \{0, 1\}$ is versatile. Let $X = \{0, 1, 2, 3\}$, $Y = \{0, 1, 2, 3\}$. Then $g(x, y) = 1$ if and only if $x + y (mod 4)$ equals 2 or 3.*

4 Communication Complexity Lower Bound for Lifted Tseitin via Set Disjointness

We first give a warmup reduction that should help to explain the main high level idea. Suppose that f is a function (not a search problem) with block sensitivity bs and let α be an assignment that witnesses this block sensitivity. Let the blocks be $B_1, \ldots, B_{bs}$. Now consider fog^n where g is some inner gadget such that both AND and $\neg$ are reducible to g (that is, both of these functions occur as subrectangles of the matrix for g).

The players, upon input x, y to set disjointness, $|x| = |y| = bs$ want to solve unique disjointness with a protocol for fog^n. They want to map input x_i, y_i to values for block B_i. If $x_i = y_i = 1$ then they want to set the i^{th} block to the value specified by α and otherwise (if $AND(x_i, y_i) = 0$) then they want to toggle the values of the i^{th} block. All other indices that are not in a critical block will take on a value consistent with α. They can do this as long as both AND and $\neg$ are reducible to g. Now if x, y are disjoint, then g^n will be equal to α, but if x, y are not disjoint and intersect uniquely at location i, then g^n will equal α^{B_i}. Since f is sensitive on every block on α, the protocol for fog^n will either output $f(\alpha)$ and this happens whenever x, y are disjoint, or it will output $f(\alpha^{B_i})$ for some i and this will happen whenever x, y uniquely intersect at location i. Thus Alice and Bob can determine whether or not x, y intersect from a protocol for fog^n.

In this section, S will be the search problem associated with Tseitin formula over a fixed highly expanding graph on $O(n)$ vertices. By the above theorem, we know that the critical block sensitivity of S is very large – $\Omega(n/\log n)$.

Let Sog^n be the lifted search problem, where Alice gets a binary vector x of length $4n$, and similarly Bob gets a binary vector y of length $4n$, and their job is to solve the search problem for S on input $z_1 = g(x^1, y^1), \ldots, z_n = g(x^n, y^n)$, where g is the versatile gadget described in the previous section.

We want to show that if there is a short protocol for Sog^n, then there is also a short protocol for unique set disjointness.

Let Π be the short protocol for Sog^n. We will record for each $\alpha \in \{0, 1\}^n$ the most likely feasible output of Π on inputs (x, y) that encode α. That is, for each α, we define μ_α to be the uniform distribution on the set of preimages of α, i.e., μ_α is uniform over all (x, y) such that $g^n(x, y) = \alpha$. The most likely feasible solution output by Π on inputs from distribution μ_α is now captured by a total function $f \subset S$.

Going back to our reduction, we want to show that from the short protocol Π for Sog^n described above (and corresponding total function $f \subset S$ described above), that Alice and Bob can solve unique set disjointness. To this aim, let $a = a_1, \ldots, a_m$ be Alice's input to unique set disjointness, and let $b = b_1, \ldots, b_m$ be Bob's input to unique set disjointness. We will eventually be setting m equal to the block sensitivity of S, so m will be about $n/\log n$.

Since S has critical block sensitivity $\Omega(n/\log n)$, there exists some critical input γ to S that witnesses this high block sensitivity with respect to f. Let the blocks be denoted by $B_1, \ldots, B_{bs} \subset [n]$ be the sensitive blocks with $f(\alpha^{B_i}) \neq f(\alpha)$.

Lemma 3. *The protocol* Π *can distinguish between* μ_α *and* $\mu_{\alpha^{B_i}}$ *in the sense that if* (x, y) *is chosen randomly from* μ_α*, then the probability that* $\Pi(x, y) = f(\alpha)$ *is at least* $1 - \epsilon$*, but if* (x, y) *is chosen randomly from* $\mu_{\alpha^{B_i}}$*, then the probability that* $\Pi(x, y) = f(\alpha)$ *is at most 1/2.*

Proof. Since $f(\alpha)$ is the most likely feasible solution output by Π over inputs consistent with α, this solution must be correct for these inputs. On the other hand, the inputs consistent with α^{B_i} cannot give this same answer because f is sensitive to B_i at α_i. Since the protocol is correct (most of the time), this means that the protocol better give an answer other than $f(\alpha)$ a lot of the time.

4.1 The Reduction

- On input $(a, b) = (a_1, \ldots, a_m, b_1, \ldots, b_m)$ to $UDISJ_m$, we first take each pair (a_i, b_i) through the reduction $AND \leq g$ to obtain instances $(a'_1, b'_1), \ldots, (a'_m, b'_m)$ of g. Note that if $UDISJ(a, b) = 0$ then $g(a'_i, b_i) = 0$ for all i, and if $UDISJ(a, b) = 1$ then there is a unique i with $g(a'_i, b'_i) = 1$.

- Next the instances (a'_i, b'_i) are used to populate a vector $(x, y) = (x_1, \ldots, x_n, y_1, \ldots, y_n)$ carrying n instances of g as follows. Instance (a'_i, b'_i) is plugged in for the coordinates $j \in B_i$ with the c opies corresponding to $\alpha_j = 1$ flipped. That is, we define for $j \in B_i$:

  - if $\alpha_j = 0$ then $(x_j, y_j) = (a'_i, b'_i)$;
  - if $\alpha_j = 1$ then $(x_j, y_j) = (\pi_A(a'_i), \pi_B(b'_i))$ where (π_A, π_B) is the reduction $\neg g \leq g$.

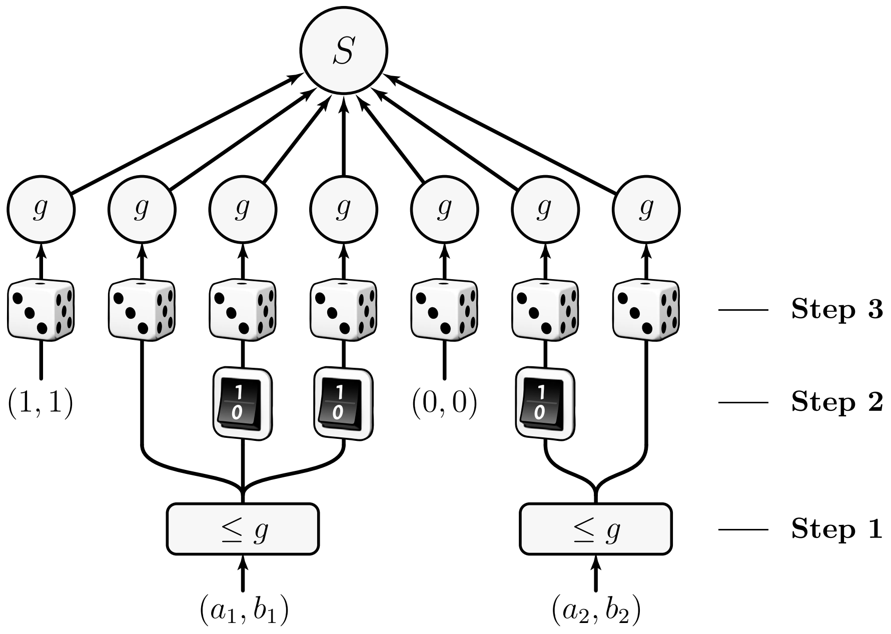
  For j not in any of the blocks, we simply fix an arbitrary $(x_j, y_j) \in g^{-1}(\alpha_j)$.

  We now have that if $UDISJ(a, b) = 0$, then $g^n(x, y) = \alpha$, and if $UDISJ(a, b) = 1$ with $a_i = b_i = 1$, then $g^n(x, y) = \alpha^{B_i}$.

- Finally, we apply a random self reduction independently for each component (x_i, y_i) of (x, y): this maps a z-input (x_i, y_i) to a uniformly random z-input (x_i, y_i) from μ_z. The result is: if $UDISJ(a, b) = 0$, then (x, y) is a random vector such that $g^n(x, y) = \alpha$, and if $UDISJ(a, b) = 1$ with $a_i = b_i$ then (x, y) is a random vector such that $g^n(x, y) = \alpha^{B_i}$.

  By the above lemma, the protocol Π distinguishes between these 2 vectors, and thus Alice and Bob can determine whether or not (a, b) are disjoint with high probability.

Figure 1: The reduction mapping (a, b) to a distribution (x, y). In this example $bs = 2$ and $n = 7$. The critical input is $\alpha = 1011010$ and the two sensitive blocks are $B_1 = \{2, 3, 4\}$ and $B_2 = \{6, 7\}$. The input pair (a_i, b_i), $i = 1, 2$, is plugged in for the block B_i.


S
g
Step 3
$(1,1)$
$(0,0)$
Step 2
$\leq g$
Step 1
(a_1, b_1)
(a_2, b_2)