DD2242 Seminars on TCS: Proof Complexity Sep 19, 2016

Lecture 5

Lecturer: Marc Vinyals *Scribe: Robert Alnesjö, Susanna F. de Rezende*

1 Introduction

In the last lecture we began by introducing the cutting planes proof system but then switched our focus back to resolution when we wanted to illustrate how to use the interpolation technique to prove lower bounds. Today we want to extend the interpolation technique to cutting planes and show how to obtain exponential lower bounds on proof length for this proof system.

Let us recall the strategy for proving the lower bound for resolution via interpolation. We start with an unsatisfiable CNF formula on the form $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$ for disjoint sets of variables $\mathbf{p},\mathbf{q},\mathbf{r}$, and suppose for the sake of contradiction that $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$ has a short resolution refutation. We then prove that we can construct a small *interpolating circuit* for $A(\mathbf{p},\mathbf{q}) \wedge B(\mathbf{p},\mathbf{r})$, that is, a Boolean circuit $I(\mathbf{p})$ such that for any assignment ρ to the variables $\mathbf{p}$, which splits the formula into two subformulas $A(\mathbf{p},\mathbf{q})\restriction_\rho = A(\rho,\mathbf{q})$ and $B(\mathbf{p},\mathbf{r})\restriction_\rho = B(\rho,\mathbf{r})$ over disjoint sets of variables, it holds that $I(\rho) = 0$ implies that $A(\rho,\mathbf{q})$ is unsatisfiable and $I(\rho) = 1$ implies that $B(\rho,\mathbf{r})$ is unsatisfiable. Finally, we appeal to an already known circuit complexity lower bound that states that no such small circuit can exist. For this final step to work and yield unconditional lower bounds it is important that the interpolating circuit is *monotone*, because it is currently not known how to prove strong enough lower bounds for general, non-monotone, circuits.

The proof strategy for cutting planes follows the same idea, but in order to extract interpolating circuits from cutting planes refutations we have to use a more general class of *real circuits*, which are circuits that compute with real numbers and have arbitrary real functions as gates. The plan for today is to first show how to build interpolants in the form of monotone real circuits from short cutting planes refutations of clique-coclique formulas, and then present the main ideas behind the proof that these formulas do not have such small interpolating circuits. In the next lecture we will see the full proof of the latter result.

Recall that the *clique-coclique* formulas are unsatisfiable CNF formulas encoding the contradictory claim that there exist undirected graphs $G = (V,E)$ on $n = |V|$ vertices which have an m-clique but are also $(m-1)$-colourable. The encoding uses the following Boolean variables:

- $p_{i,j}$, for $i,j \in [n]$, $i < j$ representing the truth value of “there is an edge between vertices i and j;”
- $q_{k,i}$, for $i \in [n]$ and $k \in [m]$ representing the truth value of “vertex i is the kth member of the m-clique;” and
- $r_{i,\ell}$, for $i \in [n]$ and $\ell \in [m-1]$ representing the truth value of “vertex i is has colour ℓ.”

The clique-coclique formula consists of the following clauses (where we write $(x_1 \wedge \cdots \wedge x_m) \rightarrow y$ to denote the equivalent clause $\overline{x}_1 \vee \cdots \vee \overline{x}_m \vee y$ to highlight that a clause on this form can be viewed as encoding an implication):

$$q_{k,1} \vee q_{k,2} \vee \cdots \vee q_{k,n} \qquad k \in [m] \tag{1.1a}$$
$$\overline{q}_{k,i} \vee \overline{q}_{k',i} \qquad i \in [n];\ k,k' \in [m], k \neq k' \tag{1.1b}$$
$$(q_{k,i} \wedge q_{k',j}) \rightarrow p_{i,j} \qquad i,j \in [n], i < j;\ k,k' \in [m], k \neq k' \tag{1.1c}$$
$$r_{i,1} \vee r_{i,2} \vee \cdots \vee r_{i,m-1} \qquad i \in [n] \tag{1.1d}$$
$$(r_{i,\ell} \wedge r_{j,\ell}) \rightarrow \overline{p}_{i,j} \qquad i,j \in [n], i < j;\ \ell \in [m-1] \tag{1.1e}$$

We observe that the clauses in the clique-coclique formula can indeed be partitioned into two sets: one set, which we denote $A(\mathbf{p},\mathbf{q})$, consisting of clauses that contain $\mathbf{p}$- and $\mathbf{q}$-variables, i.e., the clauses in (1.1a)–(1.1c); and a second set, which we denote $B(\mathbf{p}, \mathbf{r})$, consisting of clauses that contain $\mathbf{p}$- and $\mathbf{r}$-variables, i.e., the clauses in (1.1d)–(1.1e).

We want to use refutations of the clique-coclique formula in in (1.1a)–(1.1e) to build an interpolating circuit, which is a circuit computing with $\mathbf{p}$-variables encoding edges and non-edges in a graph G with n vertices. This interpolating circuit should be able to distinguish between a graph G that has no m-clique (i.e., an assignment ρ to the $\mathbf{p}$-variables such that $A(\rho, \mathbf{q})$ is unsatisfiable) and one that has no $(m-1)$-colouring (i.e., an assignment such that $B(\rho, \mathbf{r})$ is unsatisfiable). Rephrasing the conditions on interpolants discussed above, for the clique-coclique formula the interpolating circuit has to output 1 if the graph G encoded by ρ contains an m-clique and 0 if G has an $(m-1)$-colouring, and can answer with either value if none of these conditions hold.

2 Clique-Coclique Formulas and Interpolation for Cutting Planes

As explained in Lecture 4, when we want to refute a CNF formula in cutting planes we translate clauses to inequalities by replacing connectives $\lor$ with addition and negated variables $\overline{x}$ with $(1-x)$. When converting the clauses (1.1b) in clique-coclique formulas to inequalities, however, we can note that a set of clauses

$$\left\{ \overline{x}_i \lor \overline{x}_j \,\middle|\, 1 \leq i < j \leq s \right\} \ , \tag{2.1a}$$

which is translated to

$$\left\{ -x_i - x_j \geq -1 \,\middle|\, 1 \leq i < j \leq s \right\} \ , \tag{2.1b}$$

is just another way to encode the constraint

$$\sum_{i=1}^{s} x_i \leq 1 \ , \tag{2.1c}$$

and since (2.1c) can be written natively as just one linear constraint

$$\sum_{i=1}^{s} -x_i \geq -1 \tag{2.1d}$$

in the context of cutting planes (where we switched to negations just to keep the format $\sum_i c_i x_i \geq C$ of greater-than inequalities that we have adopted for cutting planes) it seems more natural to use the latter encoding rather than a quadratic number of constraints as in (2.1b). For our purposes, it does not really matter which form we use since one can derive (2.1b) from (2.1d) and vice-versa in polynomial length. In one direction (the one we care about in order to get lower bounds for refutations of CNF formulas) this is obvious—from $\sum_{i=1}^{s} -x_i \geq -1$ one can derive $-x_{i_1} - x_{i_2} \geq -1$ for any $i_1, i_2 \in [s]$, $i_1 \neq i_2$, by just adding axioms $x_i \geq 0$ for all $i \in [s] \setminus \{i_1, i_2\}$; the other direction is also straightforward. This means that superpolynomial length lower bounds for clique-coclique formulas using the encoding in (2.1d) imply superpolynomial bounds for the encoding using (2.1b) and the other way round. In view of this, we will define the clique-coclique formula in cutting planes as consisting of the inequalities

$$\begin{aligned}
q_{k,1} + q_{k,2} + \cdots + q_{k,n} &\geq 1 && k \in [m] && \text{(2.2a)}\\
-q_{1,i} - q_{2,i} - \cdots - q_{m,i} &\geq -1 && i \in [n] && \text{(2.2b)}\\
p_{i,j} - q_{k,i} - q_{k',j} &\geq -1 && i,j \in [n], i < j;\ k,k' \in [m], k \neq k' && \text{(2.2c)}\\
r_{i,1} + r_{i,2} + \cdots + r_{i,m-1} &\geq 1 && i \in [n] && \text{(2.2d)}\\
-p_{i,j} - r_{i,\ell} - r_{j,\ell} &\geq -2 && i,j \in [n], i < j;\ \ell \in [m-1] && \text{(2.2e)}
\end{aligned}$$

and prove a lower bound on the length of cutting planes derivations establishing that this set of linear constraints is inconsistent.

Since lines in a cutting planes proofs consist of linear inequalities and the coefficients can be arbitrary integers, we need a more general computational model than standard Boolean circuits when we want to construct interpolants from cutting planes proofs.

Definition 2.1 (Real circuit [Pud97]). A *real circuit* C is a directed acyclic graph (DAG) with s sources, which are vertices labelled by variable inputs, and a unique sink. Every non-source vertex v with fan-in d_v is labelled by a function $f_v : \mathbb{R}^{d_v} \to \mathbb{R}$ of arity matching the fan-in of the vertex, and we will also refer to such vertices v as *gates*. The incoming edges from the predecessors of v are ordered in some fixed way, so that the function f_v labelling v is well-defined, and every gate v computes the value of the function f_v applied to the values provided by its predecessors. We require all vertices to have bounded fan-in $\mathrm{O}(1)$, and although the exact bound is not too important in what follows we will insist on fan-in at most 2. The output of the circuit C is the value computed by the sink vertex, and this defines a function $f_C : \mathbb{R}^s \to \mathbb{R}$ on the inputs in the natural way, which is the function computed by the circuit. The *size* of a circuit is the total number of vertices in the DAG.

A real circuit is *monotone* if all gates in it compute non-decreasing functions. We say that a real circuit computes a Boolean function if when restricted to inputs in $\{0, 1\}^s$ it produces an output in $\{0, 1\}$ (but in intermediate steps the circuit can still work with arbitrary real numbers).

Gates in a monotone real circuit can emulate Boolean gates such as $\wedge$ and $\vee$ with for example $\max$ and $\min$, respectively, so monotone real circuits are at least as strong as monotone Boolean circuits. Are they stronger, and if so how much? Well, we actually do not know! All we know is that there are functions computable by small real monotone circuits that require large monotone Boolean circuits, but we do not know which functions these are.

We can now formally state the theorem we will prove today.

Theorem 2.2 ([Pud97]). *Let π be a cutting planes refutation of the inequalities*

$$\sum_a e_{i,a} p_a + \sum_b f_{i,b} q_b \geq D_i \qquad i \in I \tag{2.3a}$$

$$\sum_a e_{j,a} p_a + \sum_c g_{j,c} r_c \geq D_j \qquad j \in J \tag{2.3b}$$

and let w be the maximal number of $\mathbf{p}$-variables that appear in any of these inequalities. Then if all coefficients $e_{i,a}$ for $i \in I$ are non-negative or if all coefficients $e_{j,a}$ for $j \in J$ are non-positive, it holds that the formula has an interpolant in the form of a monotone real circuit of size at most $w \cdot L(\pi)$.

Since in the clique-coclique formulas there is at most one $\mathbf{p}$-variable in each inequality, we immediately get the following corollary.

Corollary 2.3 ([Pud97]). *Let π be a cutting plane proof refutation of a clique-coclique formula. Then there exists a real monotone interpolating circuit for this clique-coclique formula of size at most $L(\pi)$.*

3 Proof of Cutting Planes Interpolation Theorem

We proceed to prove Theorem 2.2. Let π be a cutting planes refutation of a clique-coclique formula as in the statement of the theorem and let ρ be an assignment to the $\mathbf{p}$-variables. We will refer to inequalities (2.3a) as $\mathbf{q}$-axioms and inequalities (2.3b) as $\mathbf{r}$-axioms.

Our plan is to split the refutation π in to two derivations, one from $\mathbf{q}$-axioms and one from $\mathbf{r}$-axioms, in such a way that at least one of these will be deriving contradiction. We will then build an interpolating real circuit for the formula using one of these derivations and finally argue that this can be done by a monotone circuit.

In order to build two derivations, one from $\mathbf{q}$-axioms and one from $\mathbf{r}$-axioms, we will replace each line

$$\sum_a e_a \rho(p_a) + \sum_b f_b q_b + \sum_c g_c r_c \geq D \tag{3.1}$$

in $\pi{\restriction}_\rho$ with two inequalities

$$\sum_b f_b q_b \geq D_q \tag{3.2a}$$

and

$$\sum_c g_c r_c \geq D_r \tag{3.2b}$$

such that (3.2a) only contains $\mathbf{r}$-variables and (3.2b) only contains $\mathbf{q}$-variables. Note that since all variables in $\mathbf{p}$ have been assigned by ρ, the expression $\sum e_a p_a$ is replaced by the integer $\sum e_a \rho(p_a)$.

We will refer to D_q (the constant term in the inequality only containing $\mathbf{q}$-variables) as the $\mathbf{q}$-part, and similarly refer to D_r as the $\mathbf{r}$-part. We want inequalities (3.2a) and (3.2b) to be valid cutting planes lines, and therefore require D_q and D_r to be integers. Moreover, since we still want to have a refutation of the restricted formula, we make sure that (3.2a) and (3.2b) imply (3.1). We can do this by requiring that the inequality

$$D_q + D_r \geq D - \sum_a e_a \rho(p_a) \tag{3.3}$$

holds.

3.1 Extracting a Cutting Planes Refutation of One of the Subformulas

We now describe how to build the two independent derivations from the refutation π. We prove by forward induction over the cutting planes refutation π that there is a refutation of the restricted formula in at most the same length from either only $\mathbf{q}$-axioms or only $\mathbf{r}$-axioms.

We will go through π line by line maintaining the invariant that for all lines seen so far, written on the form (3.1), we have two split inequalities (3.2a) and (3.2b) that satisfy (3.3) and that can be obtained by two independent and valid derivations, one from $\mathbf{q}$-axioms and the other from $\mathbf{r}$-axioms. Moreover every step in π will correspond to at most one step in each of the new derivations, so the length of these derivations will be bounded by the length of π.

Since the final inequality in π is $0 \geq 1$, the split inequalities will be $0 \geq D_q$ and $0 \geq D_r$ with D_q and D_r satisfying $D_q + D_r \geq 1$. Because D_q and D_r are integers, at least one of D_q and D_r must be greater than or equal to 1. This implies that a contradiction has been derived from either $\mathbf{q}$-axioms or $\mathbf{r}$-axioms.

Suppose that all lines so far satisfy the invariant. All we need to show is how to obtain the two split inequalities for the next line while maintaining the invariant. We do so by a case analysis over the cutting planes derivation rules:

Axioms Say $\sum e_a p_a + \sum f_b q_b + \sum g_c r_c \geq D$ is an axiom in the formula or a variable axiom (i.e., $x \geq 0$ or $-x \leq -1$ for some variable x).

Since no axiom (neither variable axioms nor axioms in the formula) contains both $\mathbf{q}$- and $\mathbf{r}$-variables we only have to consider two cases. If the axiom does not contain $\mathbf{r}$-variables, then it holds that $\sum g_c r_c = 0$ and we can set $D_q = D - \sum e_a p_a$ and $D_r = 0$. Note that the inequality with the $\mathbf{r}$-part is $0 \geq 0$. If the axiom contains $\mathbf{r}$-variables, it does not contain $\mathbf{q}$-variables and hence we can set $D_q = 0$ and $D_r = D - \sum e_a p_a$. In both cases it is clear that D_q and D_r satisfy (3.3) and clearly the invariant holds.

Addition Say the refutation adds $\sum e_a p_a + \sum f_b q_b + \sum g_c r_c \geq D$ and $\sum e'_a p_a + \sum f'_b q_b + \sum g'_c r_c \geq D'$. By the induction hypothesis, we have that each inequality is associated to two split inequalities of the form (3.2a) and (3.2b) such that (3.3) holds.

We can add the split inequalities separately to obtain

$$\frac{\sum f_b q_b \geq D_q \qquad \sum f'_b q_b \geq D'_q}{\sum (f_b + f'_b) q_b \geq D_q + D'_q} \quad \text{and} \quad \frac{\sum g_c r_c \geq D_r \qquad \sum g'_c r_c \geq D'_r}{\sum (g_c + g'_c) r_c \geq D_r + D'_r} \,. \tag{3.4}$$

Since by the induction hypothesis $D_q + D_r \geq D - \sum e_a p_a$ and $D'_q + D'_r \geq D' - \sum e'_a p_a$, clearly the invariant still holds.

Multiplication Say we wish to multiply $\sum e_a p_a + \sum f_b q_b + \sum g_c r_c \geq D$ by λ. By the induction hypothesis, we have two split inequalities of the form (3.2a) and (3.2b) such that (3.3) holds.

By multiplying each split inequality by λ separately we get

$$\frac{\sum f_b q_b \geq D_q}{\sum (\lambda f_b) q_b \geq \lambda D_q} \quad \text{and} \quad \frac{\sum g_c r_c \geq D_r}{\sum (\lambda g_c) r_c \geq \lambda D_r} , \tag{3.5}$$

and it is easy to see that the invariant holds.

Division Say we wish to divide $\sum e_a p_a + \sum f_b q_b + \sum g_c r_c \geq D$ by λ. By the induction hypothesis, we have two split inequalities of the form (3.2a) and (3.2b) such that (3.3) holds.

We can divide each split inequality by λ separately to obtain

$$\frac{\sum f_b q_b \geq D_q}{\sum (f_b/\lambda) q_b \geq \lceil D_q/\lambda \rceil} \quad \text{and} \quad \frac{\sum g_c r_c \geq D_r}{\sum (g_c/\lambda) r_c \geq \lceil D_r/\lambda \rceil} . \tag{3.6}$$

Since the coefficients of the $\mathbf{q}$- and the $\mathbf{r}$-variables are the same, they are divisible by λ and therefore this is a valid application of the division rule. Moreover, since by the induction hypothesis, $D_q + D_r \geq D - \sum e_a p_a$, we have that

$$\left\lceil \frac{D_q}{\lambda} \right\rceil + \left\lceil \frac{D_r}{\lambda} \right\rceil \geq \left\lceil \frac{D_q + D_r}{\lambda} \right\rceil \geq \left\lceil \frac{D - \sum e_a p_a}{\lambda} \right\rceil = \left\lceil \frac{D}{\lambda} \right\rceil - \sum \frac{e_a}{\lambda} p_a \tag{3.7}$$

and hence the invariant still holds.

We have completed our goal to produce two independent derivation, one of $0 \geq D_r$ from $\mathbf{r}$-axioms and one of $0 \geq D_q$ from $\mathbf{q}$-axioms, such that $D_r + D_q \geq 1$.

3.2 Writing Down the Interpolating Circuit

Nice, but how do we build an interpolating circuit? Observe that, since $D_r + D_q \geq 1$ it is enough for the circuit to compute either D_r or D_q in order to answer correctly, i.e., to answer 1 if $D_r \geq 1$ and 0 if $D_q \geq 1$. Indeed, if for example the circuit computes D_r and answers one if $D_r \geq 1$ and zero otherwise, then the fact that $D_r + D_q \geq 1$ guarantees that when the circuit answers zero $D_q \geq 1$. In the concrete case of the clique-coclique formulas in equations (2.2a)–(2.2e) what this boils down to is that if the circuit answers 1 the graph defined by ρ has no $(m-1)$-colouring and if it answers 0 the graph has no m-clique.

Let us see how to build a circuit based on the derivation we obtain from the $\mathbf{r}$-axioms. We assume here that $\mathbf{p}$-variables appear only negatively in $\mathbf{r}$-axioms and leave the other case as an exercise to the reader. The circuit has a gate for every step in the derivation. Each gate receives as input one or two previously computed $\mathbf{r}$-parts and outputs the value of the $\mathbf{r}$-part of the new line derived. For each step in the derivation that is not an axiom, the circuit has a gate that either adds two inputs, or multiplies the input by a positive constant, or divides the input by a positive constant (and rounds up). For axioms, the gate must compute the $\mathbf{r}$-part which is either 0 or $D - \sum e_a p_a$. At the end we will also need a threshold gate that will output 1 if $D_r \geq 1$. Let us see each of these cases in a bit more detail and it will be clear that all gates are monotone.

Axioms with $\mathbf{r}$-variables To compute the value of the $\mathbf{r}$-part of such axioms we need a circuit that computes $D - \sum e_a p_a$. By assumption, for all axioms that contain $\mathbf{r}$-variables it holds that the coefficients e_a are non-positive. This means that we can compute $D - \sum e_a p_a$ with a monotone circuit. Since there are at most w $\mathbf{p}$-variables in any axiom, this means that this subcircuit has at most w gates.

Axioms with no $\mathbf{r}$-variables The value of the $\mathbf{r}$-part in this case is 0, so we use the constant gate 0 (and if we do not want constants in our circuits, it is easy to remove them in a postprocessing step, just as in Lecture 4).

Addition The value of the $\mathbf{r}$-part is the sum of two $\mathbf{r}$-parts, so we use a binary addition gate.

Multiplication The value of the $\mathbf{r}$-part is multiplied by a positive constant, hence we use a unary gate that computes this multiplication.

Division The value of the $\mathbf{r}$-part is divided by a positive constant and rounded up, so we use a unary gate that computes this operation.

Output gate At the end if the $\mathbf{r}$-part is at least 1, then the $\mathbf{r}$-axioms are unsatisfiable and the circuit can answer 1. If the $\mathbf{r}$-part is 0, then the $\mathbf{q}$-axioms are unsatisfiable and the circuit can answer 0. This can be done by using a unary threshold gate which answers 1 if the input is greater or equal to 1.

It is easy to see that all gates described above are monotone, so we can conclude that the circuit is monotone. This completes the proof of Theorem 2.2.

4 Outline of Monotone Real Circuit Lower Bound

Now that we know how to build interpolating circuits from cutting planes refutations of clique-coclique formulas, we want to show that monotone circuits computing interpolants for clique-coclique formulas must have exponential size. In what remains of today's lecture, we will outline a proof of the following theorem.

Theorem 4.1 ([Pud97]). *There exists a constant $\delta > 0$ such that for all large enough n it holds that any monotone real circuit computing an interpolant for the clique-coclique formula over n $\mathbf{p}$-variables has size* $\exp\left(n^{\delta}\right)$.

One of the very few known techniques to prove circuit lower bounds is Razborov's approximation method [Raz85]. Razborov's original method, which yields superpolynomial lower bounds for monotone boolean circuits, was later extended in [AB87] to obtain exponential lower bounds and generalized in [Pud97] to capture monotone real circuits. We present the basic idea of this method, and further clarify it next lecture.

We wish to find some class of functions $F : \{0,1\}^n \to \mathbb{R}$, for example polynomials of low degree, such that:

- Inputs to the circuit are in the class i.e., for every $i \in [n]$ the class contains a function which given input $(b_1, b_2, \ldots, b_n) \in \{0,1\}^n$ outputs b_i.

- If the inputs to a gate are the outputs of two functions F_i, F_j in the class, then the output of the gate can be very well approximated by another function $\tilde{F}$ in the class (i.e., $\tilde{F}$ errors—disagrees with the output of the gate—only on a small fraction of the inputs).

- The output of circuit cannot be approximated within the class (i.e., there is no function in the class that only errors—disagrees with the output of the circuit—on a small fraction of the inputs).

So if we have such a class of functions then we can argue that we have functions that compute the inputs to the bottom gates with no error, for every gate there is a function that introduces only a small error, but in the end all functions in the class make a huge error on the circuit output! From this we can conclude that the circuit must have many gates.

But what do we mean by errors then? To define this we introduce the notion of 1- and 0-inputs. We say an input x is a 1-input (resp. a 0-input), if the circuit returns 1 (resp. 0) for this input. We have two types of errors:

- If x is a 1-input and the approximation $\tilde{F}(x)$ is less than the original $F(x)$ we have a 1-error.

- Analogously, if x is a 0-input and the approximation $\tilde{F}(x)$ is more than the original $F(x)$ we have a 0-error.

Note that we only care about errors in one direction. For example, if x is a 1-input and the approximation $\tilde{F}(x)$ is more than the original $F(x)$, then we do not count this as an error. In some sense, since the circuit is monotone, we are only doing better.

We observe that when measuring errors we can choose the distribution of inputs that suits us best, and prove that the class of functions satisfy the properties we require for this distribution. In our case, we will only consider *critical* inputs x: a 0-input is critical if flipping the value of any variable in x from 0 to 1 yield a 1-input; analogously, a 1-input is critical if flipping the value of any variable in x from 1 to 0 yield a 0-input. For clique-coclique, the critical inputs are graphs of two types: complete $(m-1)$-partite graphs, because these are 0-inputs but adding any edge will ruin the colouring property and change the answer from 0 to 1; and m-cliques (i.e., graphs that contain an m-clique and all vertices not in the clique are isolated vertices), because these are 1-inputs but removing any edge makes it $(m-1)$-colourable.

Proof sketch of Theorem 4.1. In the next lecture we will prove that there is a class of functions $F : \{0,1\}^n \to \mathbb{R}$ such that:

(1) for every $i \in [n]$, it contains a function which given input $(b_1, b_2, \dots, b_n) \in \{0,1\}^n$ outputs b_i;

(2) if the inputs to a gate are the outputs of two functions F_i, F_j in the class, then there is a function $\tilde{F}$ in the class that differs from the output of the gate in at most a $\epsilon = \exp\left(-n^{\delta'}\right)$ fraction of critical 0-inputs, for some $\delta' > 0$;

(3) if the inputs to a gate are the outputs of two functions F_i, F_j in the class, then there is a function $\tilde{F}$ in the class that differs from the output of the gate in at most a $\epsilon' = \exp\left(-n^{\delta''}\right)$ fraction of critical 1-inputs, for some $\delta'' > 0$;

(4) for a function F in the class, either $F \geq 1$ on all critical inputs, or $F < 1$ on $2/9$ fraction of cliques.

If $F \geq 1$ for all critical inputs, then F is far from approximating the circuit since it errors on all critical 0-inputs; if $F < 1$ for $2/9$ fraction of cliques, then F errors on a large fraction of critical 1-inputs and is also far from approximating the circuit.

By (2) and (3), if there are functions in the class that approximate the inputs to a gate, say F_i and F_j, then there is a function in the class that approximates the output of the circuit making only a few more errors than F_i and F_j. Together with (4) this means that the circuit has many gates. How many gates? We must consider two cases.

If $F \geq 1$ on all critical inputs, then the circuit errors on all critical 0-inputs, so there must be at least $\frac{1}{\epsilon} = \exp\left(n^{\delta'}\right)$ gates. If $F < 1$ on $2/9$ fraction of cliques, then there must be at least $\frac{2}{9} \cdot \frac{1}{\epsilon'} = \frac{2}{9} \cdot \exp\left(n^{\delta''}\right)$ gates. In both cases we conclude that the circuit has exponentially many gates. □

From Corollary 2.3 and Theorem 4.1 we immediately get the following corollary.

Corollary 4.2 ([Pud97]). *There exists a constant $\delta > 0$ such that for any N large enough, every cutting plane refutation of the clique-coclique formula of size N has length* $\exp\left(N^{\delta}\right)$.

5 Summing up

In this lecture we proved that from a short cutting planes refutation of clique-coclique we can construct a small monotone interpolating real circuit for clique-coclique. We argued (but did not formally prove yet) that any such circuit must be of exponential size, and this implies that the cutting planes refutation has exponential length. To complete the proof of Theorem 4.1, we need to define the class of functions used to approximate the circuit and then prove that this class has the properties we need.

References

[AB87] Noga Alon and Ravi B. Boppana. The monotone circuit complexity of Boolean functions. *Combinatorica*, 7(1):1–22, March 1987.

[Pud97] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *Journal of Symbolic Logic*, 62(3):981–998, September 1997.

[Raz85] Alexander A. Razborov. Lower bounds for the monotone complexity of some Boolean functions. *Soviet Mathematics Doklady*, 31(2):354–357, 1985. English translation of a paper in *Doklady Akademii Nauk SSSR*.