COMS 6998: **Spring 2024**
Unconditional Lower Bounds & Derandomization

Lecture 2: January 23, 2024

Lecturer: Rocco Servedio *Scribes: Yiming Fang*

1 Introduction

1.1 Last time

- Overview of some restricted computational models that we will consider
- Defined worst-case lower bounds, average-case lower bounds, and PRGs
- Proved that PRGs imply worst-case lower bounds

1.2 Today

- Finish overview: prove that PRGs imply average-case lower bounds
- Deterministic Approximate Counting
- Start unit on Boolean formulas; prove various worst-case lower bounds

1.3 Notation Conventions

- $a \approx_\varepsilon b$ means $|a - b| \le \varepsilon$
- $\mathcal{U}_s$ means uniform distribution over $\{0,1\}^s$ or $\{-1,1\}^s$
- $\mathbb{E}[f]$ means $\mathbb{E}_{\mathbf{U} \sim \mathcal{U}_n}[f(\mathbf{U})]$
- Use bold to denote random variables, *e.g.,* $\mathbb{E}[f(\mathbf{U})]$

2 Overview (continued)

Lemma 1. *(PRG $\Rightarrow$ average-case lower bounds)*
Let $\mathcal{C}$ be a class of functions $f : \{0,1\}^n \mapsto \{0,1\}$ that is closed under restrictions. Note that all classes that we will consider satisfy this natural condition.
Let $G : \{0,1\}^s \mapsto \{0,1\}^n$ be an ε-PRG for $\mathcal{C}$, where $s < n, \varepsilon < 1/2$.
Let $r = s + \log \frac{1}{\varepsilon}$, and assume $r \leq n$.
Define function $h : \{0,1\}^r \mapsto \{0,1\}$ that outputs 1 on x if and only if some string in the range of G starts with x:

$$h(x) = 1 \iff \exists y \in \{0,1\}^s, z \in \{0,1\}^{n-r} \text{ such that } G(y) = (x,z).$$

Let $\mathcal{C}'$ be all restrictions of functions in $\mathcal{C}$, by fixing the last $n-r$ bits, leaving the first r bits alive.
Let $\mathcal{D} = \frac{1}{2}\mathcal{U}_r + \frac{1}{2}G(\mathcal{U}_s)_{1\ldots r}$, which is a distribution constructed by a balanced convex combination of the distributions $\mathcal{U}_r$ and $G(\mathcal{U}_s)_{1\ldots r}$.
Then, h is ε-hard for $\mathcal{C}'$ with regard to $\mathcal{D}$.

Proof. Let $\mathbf{U} \sim \mathcal{U}_r$, $\mathbf{U}' \sim \mathcal{U}_s$. Fix any $f \in \mathcal{C}'$, and by definition of $\mathcal{C}'$, we know that $f(x) = f_0(x,a)$ for some $f_0 \in \mathcal{C}, a \in \{0,1\}^{n-r}$. Then,

$$\begin{aligned}
\Pr_{\mathbf{X}\sim\mathcal{D}}[f(\mathbf{X}) = h(\mathbf{X})] &= \frac{1}{2}\Pr[f(\mathbf{U}) = h(\mathbf{U})] + \frac{1}{2}\Pr[f(G(\mathbf{U}')_{1\ldots r}) = h(G(\mathbf{U}')_{1\ldots r})] \\
&= \frac{1}{2}\Pr[f(\mathbf{U}) = h(\mathbf{U})] + \frac{1}{2}\Pr[f(G(\mathbf{U}')_{1\ldots r}) = 1] \\
&\leq \frac{1}{2}\Pr[f(\mathbf{U}) = 0] + \frac{1}{2}\Pr[h(\mathbf{U}) = 1] + \frac{1}{2}\Pr[f(G(\mathbf{U}')_{1\ldots r}) = 1] \\
&\leq \frac{1}{2}\Pr[f(\mathbf{U}) = 0] + \frac{1}{2}\Pr[h(\mathbf{U}) = 1] + \frac{1}{2}(\Pr[f(\mathbf{U}) = 1] + \varepsilon) \\
&= \frac{1}{2} + \frac{\varepsilon + \mathbb{E}[h]}{2} \\
&\leq \frac{1}{2} + \varepsilon.
\end{aligned}$$

The first inequality follows by upper bounding $\Pr[f(\mathbf{U}) = h(\mathbf{U}) = 0]$ and $\Pr[f(\mathbf{U}) = h(\mathbf{U}) = 1]$. The second inequality uses the fact G ε-fools f because $f \in \mathcal{C}' \subseteq \mathcal{C}$ and G ε-fools $\mathcal{C}$. The last inequality follows because h outputs 1 on 2^s out of $2^r = 2^{s+\log\frac{1}{\varepsilon}} = \frac{2^s}{\varepsilon}$ inputs, so $\mathbb{E}[h] \leq \varepsilon$.

$\Pr_{\mathbf{X}\sim\mathcal{D}}[f(\mathbf{X}) = h(\mathbf{X})] \geq \frac{1}{2} - \varepsilon$ can be proven similarly. ∎

With Lemma 1 and previous results, we haven shown a hierarchy of lower bounds:

$$\text{PRG} \implies \text{Average-Case Lower Bounds} \implies \text{Worst-Case Lower Bounds}$$

3 Deterministic Approximate Counting

So far, we have considered PRGs, which are useful because they let us derandomize *obliviously*. PRGs are oblivious in the sense that the 2^s many n-bit outputs fool *every* $f \in \mathcal{C}$ without knowing f explicitly. However, for most natural scenarios that we are interested in, we do know f explicitly. Then, we could conceivably use this knowledge to aid the derandomization process, which leads us to the concept of Deterministic Approximate Counters (DAC).

Definition 2. *(DAC)*
Let $\mathcal{C}$ be a class of representations of functions $\{0,1\}^n \mapsto \{0,1\}$.
An algorithm is an ε-DAC for $\mathcal{C}$ if on any input $F \in \mathcal{C}$, it deterministically outputs a value in

$$[\mathbb{E}[f(\mathbf{U}) - \varepsilon, \mathbb{E}[f(\mathbf{U}) + \varepsilon].$$

Fact 3. *DAC is weaker than PRG, but it still suffices for derandomization. So, if there is a poly(n)-time ($\varepsilon = 0.1$)-DAC for $\mathcal{C} = \{$all poly(n)-size circuits$\}$, then $\mathsf{P} = \mathsf{BPP}$.*

Fact 4. *For various classes of functions, the best known ε-DAC is faster than 2^spoly(n), where s is the seed length of the best known ε-PRG.*

One concrete example classes of functions are CNFs/DNFs: the best known PRG runs in $O(n^{\log n})$ time, whereas the best known DAC runs in $O(n^{\log\log n})$ time.

4 Boolean Formulas

4.1 Basics

Definition 5. *(Boolean formula)*
A De Morgan Boolean formula F is a rooted binary tree with

- *leaf nodes (labeled with x_i or $\overline{x_i}$)*
- *internal nodes (labeled with $\vee$ or $\wedge$)*

Definition 6. *(Size and depth of Boolean formulas)*
Given a Boolean formula F,

$$\begin{aligned} size(F) &= \text{ number of leaves} \\ &= \text{ number of internal nodes} + 1 \\ depth(F) &= \text{ length of the longest root-to-leaf path} \end{aligned}$$

Remark 1. *Without loss of generality, we assume that all negations are at the leaves, since we can always use De Morgan's Law to push negations down to the bottom level.*

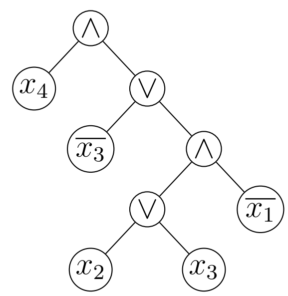


Figure 1: The example Boolean formula computes the function f in an obvious way: $f(x_1, x_2, x_3, x_4) = x_4 \wedge (\overline{x_3} \vee ((x_2 \vee x_3) \wedge \overline{x_1})$

Definition 7. *(Formula size and formula depth of functions)*
Given a function $f : \{0, 1\}^n \mapsto \{0, 1\}$,

- *Formula size:* $size(f)$ = *the size of the smallest formula* F *computing* f.
- *Formula depth:* $depth(f)$ = *the depth of the shallowest formula* F *computing* f.

Remark 2. *Sometimes, gates with unbounded fan-in are allowed. This relaxation does not affect the size of the formula, but may affect its depth, as shown in Figure 2.*

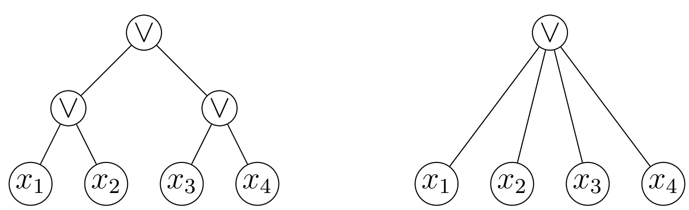
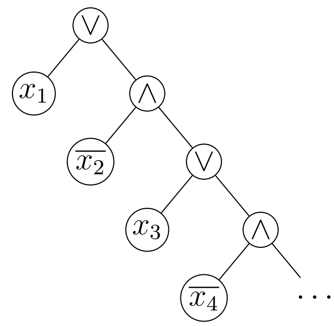


Figure 2: Allowing gates with unbounded fan-in changes the depth but not the size.

Example 8. *(Decision List)*
Consider the following decision list function with n *variables:*

$$\begin{aligned} \textit{if } x_1 &\longrightarrow 1 \\ \textit{else if } x_2 &\longrightarrow 0 \\ \textit{else if } x_3 &\longrightarrow 1 \\ \textit{else if } x_4 &\longrightarrow 0 \\ &\cdots \end{aligned}$$

This function is computed by the following Boolean formula F:



It is clear that size$(F) = n$ *and depth*$(F) = n$.

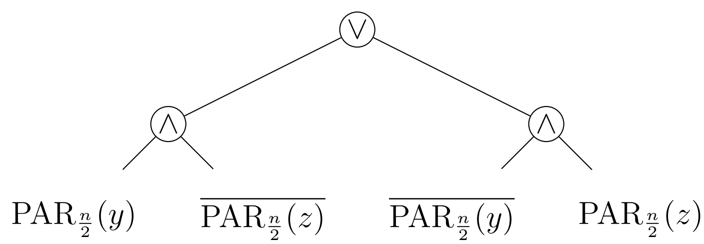
For any n-variable decision list function, the size of $O(n)$ is the minimum possible. However, the $O(n)$ depth is not minimal, and there exists a Boolean formula with $O(\log n)$ depth: the given decision list function can be computed by a depth-2 DNF (which has unbounded fan-in), and we can convert such a DNF into a Boolean formula with depth of $2\log n$ and size of $\Theta(n^2)$.

Example 9. *(Parity)*
The parity function on n variables is defined as

$$\mathrm{PAR}_n(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i \mod 2.$$

We can recursively construct a Boolean formula that computes PAR_n:

$\mathrm{PAR}_n(y, z) =$



where $y = (x_1, \ldots, x_{\frac{n}{2}})$, $z = (x_{\frac{n}{2}+1}, \ldots, x_n)$.

We can compute the size of the resulting formula inductively. Let $s(i)$ denote the size of a Boolean formula computing PAR_i, then $s(1) = 1$, $s(2i) = 4s(i)$. Hence $s(n) = n^2$, which is in fact optimal, i.e., size$(\mathrm{PAR}_n) = \Omega(n^2)$.

The proof of the $\Omega(n^2)$ lower bound is highly non-trivial. In the following sections, we will instead prove a weaker lower bound of $\Omega(n^{\frac{3}{2}})$ using techniques that are more relevant for PRGs.

Fact 10. *There exist $O(n)$-size Boolean circuits computing PAR_n.*

4.2 Shannon's Lower Bound via Counting Arguments

Idea: The following result uses counting arguments to establish lower bounds for nonexplicit problems. The strategy is to demonstrate that there are many n-variable Boolean functions, while there are not too many Boolean formulas with small size. Therefore, some fraction of Boolean functions necessarily have large formula size.

Fact 11. *There are 2^{2^n} many functions $f : \{0,1\}^n \mapsto \{0,1\}$ (since each function has 2^n possible inputs, and has 2 possible output values for each input).*

Theorem 12. *[Sha49] At least a $1-1/2^n$ fraction of all functions $f : \{0,1\}^n \mapsto \{0,1\}$ have formula size*

$$size(f) \geq \frac{2^n}{2\log n}.$$

Proof. Let $s = \frac{2^n}{2\log n}$. We will show that the number of Boolean formulas with size s over n variables is much fewer than $2^{2^n}/2^n$.

To upper bound the number of possible Boolean formulas with size s, it suffices to consider the number of possible *specifications* of such formulas. Any size-s Boolean formula F can be specified by:

(i) *The structure of the binary tree*

Since $size(F) = s$, F has s leaves and $s-1$ internal nodes. We can describe any such tree as an element of

$$\{II, IL, LI, LL\}^{s-1},$$

where L and I represent internal and leaf node respectively, and each tuple represents the node type of the left and right children. Any tree structure can be unambiguously parsed into a sequence of $s-1$ tuples, by traversing every internal node in an orderly fashion and recording the node types of the current node's children. Therefore, the number of possible tree structures is at most 4^{s-1}.

(ii) *Labels of all nodes*

Each internal node can be labeled as $\vee$ or $\wedge$, so there are 2^{s-1} possible labelings for the $s-1$ internal nodes.

Each leaf node can be labeled as x_i or $\overline{x_i}$, for any $i \in [n]$, so there are $(2n)^s$ possible labelings for the s leaf nodes.

Thus, the number of size-s formulas over n variables can be upper bounded by

$$4^{s-1} \cdot 2^{s-1} \cdot (2n)^s \leq (16n)^s.$$

For $s = \frac{2^n}{2\log n}$ and sufficiently large n,

$$(16n)^s = 2^{\log(16n)\cdot 2^n/2\log(n)} = 2^{\frac{4+\log(n)}{2\log(n)}\cdot 2^n} < 2^{0.51\cdot 2^n} \ll \frac{1}{2^n} \cdot 2^{2^n}.$$

Therefore, at most a $1 - 1/2^n$ fraction of functions have formula size larger than s. ■

Fact 13. *For any function $f : \{0,1\}^n \mapsto \{0,1\}$, there is a DNF/CNF computing f which has at most 2^{n-1} terms/clauses. Since there are at most n literals in each term/clause, we can upper bound its size with by $n2^{n-1}$.*

Homework Problem 1. *Extend Theorem 12 to average-case lower bound under $\mathcal{U}_n$.*

Remark 3. *Theorem 12 provides an exponential lower bound for the formula size of almost all functions. However, the lower bound is unsatisfying because it is not explicit: from it, we cannot derive lower bounds for the formula size of explicit functions, such as functions in NP or P.*

4.3 Subbotovskaya's Lower Bound via Random Restrictions

[Sub61] analyzed the effect of random restrictions on Boolean formulas.

Idea: Given a Boolean formula F computing function $f : \{0,1\}^n \mapsto \{0,1\}$, we can shrink the size of F by restriction, *i.e.,* randomly and independently set some of the input variables x_i to be 0/1, leaving the rest free. Then, we argue that this restriction operation shrinks $size(F)$ by a significant amount. Thus, if the size of the restricted formula is at least 1, then $size(F)$ must have been large.

Definition 14. *(Restriction)*
A restriction (or a partial assignment) is a mapping $\rho : [n] \mapsto \{0,1,*\}$, where we understand $\rho(i) = *$ to mean that x_i is left unassigned.

We abuse the notation and also use $\rho \in \{0,1,*\}^n$ to denote a specific partial assignment yielded by the mapping ρ.

Definition 15. *(Restriction of function)*
A restriction of function f by ρ is defined as

$$f \upharpoonright \rho = f(\rho(1), \ldots, \rho(n)).$$

Note that $f \upharpoonright \rho$ is a subfunction of f, on variables x_i for which $\rho(i) = *$.

Example 16. *Let $\rho = (1, 0, 1, *, *, 1)$, and let f be a 6-variable function.*
Then, $f \restriction \rho = f(1, 0, 1, x_4, x_5, 1)$, which is a 2-variable function.
The restriction can be interpreted as "zooming in" on a subcube of f.

Definition 17. *Let $\mathcal{R}_k$ denote the following distribution over restrictions:*

- *Choose k variables uniform randomly and leave them unassigned (as $*$)*
- *Assign $0/1$ values randomly to the remaining $n - k$ variables*

Theorem 18. *[Sub61]*
Let $f : \{0,1\}^n \mapsto \{0,1\}$. Let $\boldsymbol{\rho} \sim \mathcal{R}_k$. Let $\Gamma = \frac{3}{2}$. Then,

$$\Pr_{\boldsymbol{\rho} \sim \mathcal{R}_k}\left[size(f \restriction \boldsymbol{\rho}) \leq 4 \cdot \left(\frac{k}{n}\right)^{\Gamma} \cdot size(f) \right] \geq \frac{3}{4}. \quad (1)$$

Application: Before delving into the proof, we consider the following application to see how Theorem 18 is useful in proving lower bounds on formula size.

Let $k = 1$, $f = \mathrm{PAR}_n$. $f \restriction \boldsymbol{\rho}$ is x_i or $\overline{x_i}$ for some $i \in [n]$, so $size(f \restriction \boldsymbol{\rho}) = 1$. Thus,

$$1 \leq 4 \cdot \left(\frac{1}{n}\right)^{\frac{3}{2}} \cdot size(f)$$
$$size(f) \geq \frac{1}{4} \cdot n^{\frac{3}{2}}.$$

Proof. (Theorem 18)
Let F be the optimal (*i.e.*, smallest) formula computing f. Let $s = size(F)$.

We can view $\boldsymbol{\rho}$ being constructed in $n - k$ stages. At each stage, choose a variable randomly from the remaining unassigned variables, and randomly assign $0/1$ to it. We will analyze the effect of this restriction stage-by-stage.

First stage: Choose x_i and randomly assign $b \in_r \{0,1\}$ to it. Fixing $x_i = b$ makes all instances of x_i and $\overline{x_i}$ vanish from F. By an averaging argument,

$$\mathbb{E}[\#\text{occurrence of } x_i \text{ or } \overline{x_i}] = s/n.$$

So $\mathbb{E}[size(f)]$ shrinks by at least s/n.

In fact, $\mathbb{E}[size(f)]$ shrinks more due to the following "secondary effect":

- Consider any occurrences of x_i or $\overline{x_i}$ in F, *i.e.*, $x_i \wedge G$ (A similar argument can be made for $x_i \vee G$), where G is a subformula of F and $size(G) \geq 1$.

- The key insight is that G cannot contain any x_i or $\overline{x_i}$:

  Suppose for sake of contradiction that G contains at least one x_i or $\overline{x_i}$. Then, since G only matters if $x_i = 1$, we can replace every occurrence of x_i or $\overline{x_i}$ with 1 or 0, respectively. This operation reduces $size(G)$ and hence $size(F)$, but does not change the function computed by F. However, this reduction in size contradicts the optimality of F.
- So, with probability 1/2, setting x_i to a constant makes G vanish, reducing $size(F)$ by $size(G) \geq 1$.

Therefore, the expected number of literals that vanish from F due to this "secondary effect" is at least $\frac{s}{2n}$. In total,

$$\mathbb{E}[size(f \restriction \boldsymbol{\rho}) \text{ after 1 stage}] \leq s - \frac{3}{2} \cdot \frac{s}{n} = s\left(1 - \frac{3}{2n}\right) \leq s \cdot \left(1 - \frac{1}{n}\right)^{\frac{3}{2}}.$$

Second stage: Proceed after making F optimal, which can only decrease the formula size. Since $f \restriction \boldsymbol{\rho}$ in this stage is a function on $n-1$ variables,

$$\mathbb{E}[size(f \restriction \boldsymbol{\rho}) \text{ after 2 stages}] \leq s \cdot \left(1 - \frac{1}{n}\right)^{\frac{3}{2}} \cdot \left(1 - \frac{1}{n-1}\right)^{\frac{3}{2}} = s \cdot \left(\frac{n-1}{n}\right)^{\frac{3}{2}} \cdot \left(\frac{n-2}{n-1}\right)^{\frac{3}{2}}.$$

After $n-k$ stages: (on optimal formula at each stage)

$$\mathbb{E}[size(f \restriction \boldsymbol{\rho})] \leq s \cdot \left(\frac{n-1}{n}\right)^{\frac{3}{2}} \cdot \left(\frac{n-2}{n-1}\right)^{\frac{3}{2}} \cdots \left(\frac{k}{k+1}\right)^{\frac{3}{2}} = s \cdot \left(\frac{k}{n}\right)^{\frac{3}{2}}.$$

Finally, by Markov's inequality,

$$\Pr_{\boldsymbol{\rho} \sim \mathcal{R}_k}\left[size(f \restriction \boldsymbol{\rho}) \leq 4 \cdot s \cdot \left(\frac{k}{n}\right)^{\frac{3}{2}}\right] \geq \frac{3}{4}.$$

∎

History: Let Γ denote the best shrinkage exponent such that (1) holds.

- $size(\mathrm{PAR}_n) = \Omega(n^2)$ implies $\Gamma \leq 2$
- [IN93]: $\Gamma \geq 1.55$
- [PZ93]: $\Gamma \geq 1.63$
- [Hås98]: $\Gamma \geq 2 - o(1)$
- [Tal14]: better $o(1)$

Project Topic 1. *Study [Hås98] and [Tal14].*

4.4 Andre'ev's Lower Bound

[And87] proposed a clever way to obtain $\widetilde{\Omega}\left(n^{\Gamma+1}\right)$ lower bound for the formula size of an explicit function. Note that the notation $\widetilde{\Omega}(\cdot)$ hides polylogarithmic factors.

4.4.1 Warmup to Andre'ev's function

From Theorem 12, we know that there exists an n-variable function with formula size at least $\frac{2^n}{2\log n}$. By replacing n with $\log n$, we know that there exists a $(\log n)$-variable function that requires formula size at least $\frac{n}{2\log\log n}$. Since this lower bound is nonexplicit, we only know the existence of such a function:

$$\exists \psi : \{0,1\}^{\log n} \mapsto \{0,1\} \text{ such that } \mathit{size}(\psi) \geq \frac{n}{2\log\log n}.$$

Let $b = \log n$. Let $m = n/b$. We can view $x_1, \ldots, x_n$ as b blocks of variables, with m variables in each block. Let $f_\psi : \{0,1\}^n \mapsto \{0,1\}$ be defined as

$$f_\psi(x_1, \ldots, x_n) = \psi\left(x_1 \oplus \cdots \oplus x_m, x_{m+1} \oplus \cdots \oplus x_{2m}, \ldots, x_{n-m+1} \oplus \cdots \oplus x_n\right). \tag{2}$$

Let $k = b\ln(4b)$, so that $\boldsymbol{\rho} \sim \mathcal{R}_k$ leaves $b\ln(4b)$ variables unassigned.

Lemma 19. *With probability at least $3/4$ over $\boldsymbol{\rho} \sim \mathcal{R}_k$*

$$\mathit{size}(f_\psi \restriction \boldsymbol{\rho}) \leq 4 \cdot \left(\frac{k}{n}\right)^{\Gamma} \cdot \mathit{size}(f_\psi). \tag{3}$$

Proof. It suffices to apply Theorem 18 to f_ψ. ■

Lemma 20. *With probability at least $3/4$ over $\boldsymbol{\rho} \sim \mathcal{R}_k$*

$$\forall i \in [b], \textit{block } i \textit{ gets at least one } * . \tag{4}$$

Proof. By direct calculation.

There are $\binom{n}{k}$ ways to assign k $*$'s to n variables. Fix a block $i \in [b]$. The event that no variables in block i gets a $*$ happen if and only if all k $*$'s are assigned to the other $n-m$ variables, and there are $\binom{n-m}{k}$ such assignments. Thus,

$$\begin{aligned}\Pr_{\boldsymbol{\rho}\sim\mathcal{R}_k}[\text{block } i \text{ gets no } *] &= \frac{\binom{n-m}{k}}{\binom{n}{k}} \leq \left(\frac{n-m}{n}\right)^k \\ &= \left(1 - \frac{1}{b}\right)^{b\ln(4b)} \leq \frac{1}{4b}.\end{aligned}$$

By union bound over all b blocks,

$$\Pr_{\boldsymbol{\rho}\sim\mathcal{R}_k}[\exists i, \text{block } i \text{ gets no } *] \le b \cdot \frac{1}{4b} = \frac{1}{4}.$$

■

By Lemmas 19 and 20, with probability at least $1/2$, $\boldsymbol{\rho} \sim \mathcal{R}_k$ satisfies 3 and 4. For any such ρ, $f_\psi \upharpoonright \rho$ contains ψ as a subfunction. So

$$\begin{aligned}
\frac{n}{2\log\log n} \le size(\psi) &\le size(f_\psi \upharpoonright \rho) \\
&\le 4 \cdot \left(\frac{k}{n}\right)^{\Gamma} \cdot size(f_\psi) \\
&= 4\left(\frac{(\log n)\cdot \ln(4\log n)}{n}\right)^{\Gamma} \cdot size(f_\psi) \\
size(f_\psi) &= \widetilde{\Omega}\left(n^{\Gamma+1}\right).
\end{aligned}$$

4.4.2 Andre'ev's actual function

Idea: There is an issue with f_ψ as defined in (2): it is not explicit. The function f_ψ depends on a nonexplicit function ψ, whose existence is known from Theorem 12. We can fix this issue viewing $\psi(x_1, \dots, x_b)$ as an $(2^b = n)$-bit string, and feed it as input into Andre'ev's function.

Theorem 21. *[And87]*
Let $A(x, y) : \{0,1\}^{2n} \mapsto \{0,1\}$ be Andre'ev's function. A is explicit and satisfies

$$size(A(x,y)) \le \widetilde{\Omega}\left(n^{\Gamma+1}\right).$$

Proof. For any string $y \in \{0,1\}^n$, y can be viewed as a b-bit function. Let $f_y : \{0,1\}^n \mapsto \{0,1\}$ be defined as

$$f_y(x_1, \dots, x_n) = y\left(x_1 \oplus \cdots \oplus x_m, x_{m+1} \oplus \cdots \oplus x_{2m}, \dots, x_{n-m+1} \oplus \cdots \oplus x_n\right).$$

Let Andre'ev's actual function $A(x,y) : \{0,1\}^{2n} \mapsto \{0,1\}$ be defined as

$$A(x,y) = f_y(x_1, \dots, x_n).$$

There exists some string $y \in \{0,1\}^n$ that is equivalent to the string representing ψ, so

$$size(A(x,y)) \ge size(f_\psi) \ge \widetilde{\Omega}\left(n^{\Gamma+1}\right).$$

■

Homework Problem 2. *Show that $A(x, y)$ is computable by $O(n)$-size circuit.*

Fact 22. $A \in \mathsf{P}$.

Fact 23. *A exhibits the largest known gap between circuit size and formula size:*

- *Circuit size $O(n)$*
- *Formula size $\widetilde{\Omega}(n^3)$*

Project Topic 2. *KRW conjecture [KRW95].*

References

[And87] Alexander E Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of π-schemes. *Moscow Univ. Math. Bull.*, 42(1):63–66, 1987. 4.4, 21

[Hås98] Johan Håstad. The shrinkage exponent of de morgan formulas is 2. *SIAM Journal on Computing*, 27(1):48–64, 1998. 4.3, 1

[IN93] Russell Impagliazzo and Noam Nisan. The effect of random restrictions on formula size. *Random Structures & Algorithms*, 4(2):121–133, 1993. 4.3

[KRW95] Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Computational Complexity*, 5:191–204, 1995. 2

[PZ93] Michael S Paterson and Uri Zwick. Shrinkage of de morgan formulae under restriction. *Random Structures & Algorithms*, 4(2):135–150, 1993. 4.3

[Sha49] Claude E. Shannon. The synthesis of two-terminal switching circuits. *The Bell System Technical Journal*, 28:59–98, 1949. 12

[Sub61] B. A. Subbotovskaya. Realizations of linear functions by formulas using $+, *, -$. *Soviet Mathematics Doklady*, 2:110–112, 1961. 4.3, 18

[Tal14] Avishay Tal. Shrinkage of de morgan formulae by spectral techniques. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 551–560. IEEE, 2014. 4.3, 1