CNF Encodings of Symmetric Functions

Gregory Emdin
egd03072000@gmail.com
ITMO University

Alexander Kulikov
St. Petersburg Department of Steklov Institute of Mathematics

Ivan Mihajlin
St. Petersburg Department of Steklov Institute of Mathematics

Nikita Slezkin
St. Petersburg Department of Steklov Institute of Mathematics

---

Research Article

Keywords: encoding, parity, majority, lower bounds, circuits, CNF

Posted Date: July 26th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3171444/v1

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Theory of Computing Systems on March 26th, 2024. See the published version at https://doi.org/10.1007/s00224-024-10168-w.

CNF Encodings of Symmetric Functions

Gregory Emdin[1], Alexander S. Kulikov[2], Ivan Mihajlin[2], Nikita Slezkin[2]

[1]ITMO University.
[2]Steklov Mathematical Institute at St. Petersburg, Russian Academy of Sciences.

Contributing authors: egd03072000@gmail.com; kulikov@logic.pdmi.ras.ru; ivmihajlin@gmail.com; ne.slezkin@gmail.com;

Abstract

Many Boolean functions that need to be encoded as CNF in practice, have only exponential size CNF representations. To avoid this effect, one usually introduces nondeterministic variables. For example, whereas the minimum number of clauses in a CNF computing the parity function $x_1 \oplus x_2 \oplus \cdots \oplus x_n$ is 2^{n-1}, one can use $n-1$ nondeterministic variables to get a CNF encoding with $4n$ clauses. In this paper, we prove tradeoffs between various parameters (the number of clauses, the width of clauses, and the number of nondeterministic variables) of CNF encodings of various symmetric functions. In particular, we show that a folklore way of encoding parity as CNF is provably optimal. We do this by using a tight connection between CNF encodings and depth-3 circuits. This connection shows that CNF encodings is an interesting computational model for Boolean functions: on the one hand, it is routinely used in practice when translating a practical computational problem to a format acceptable by a SAT solver, on the other hand, lower bounds on the size of CNF encodings imply depth-3 circuit lower bounds.

Keywords: encoding, parity, majority, lower bounds, circuits, CNF

1 Introduction

A popular approach for solving a difficult combinatorial problem in practice is to encode it in conjunctive normal form (CNF) and to invoke a SAT solver. There are two main reasons why this approach works well for many hard problems: the state-of-the-art SAT solvers are extremely efficient and many combinatorial problems are expressed naturally in CNF. At the same time, a CNF encoding is not unique. Moreover, there is no such thing as the best way to translate a problem to CNF: different encodings have different number of clauses, number of variables, and width of clauses. For many real-world problems (e.g., product configuration [15], radio frequency assignment [4], or reconstructing images form computed tomographs [2]), a chosen encoding affects the time of solving them. The reason is that a straightforward representation for many Boolean functions has many clauses. To reduce the number of clauses, one can use nondeterministic variables (also known as guess or auxiliary variables). However, introducing nondeterministic variables forces a SAT solver to make potentially larger number of decisions. Thus, the best ratio between the number of variables and the number of clauses is determined experimentally. In [17], it is shown that modifications to a SAT solver can mitigate the drawbacks associated with the introduction of non-deterministic variables. Prestwich [21] gives an overview of various ways to translate a problem into CNF and discusses their desirable properties, both from theoretical and practical points of view.

Two of the most popular constraints that arise when translating a problem to CNF in practice are parity $(x_1 + x_2 + \cdots + x_n \bmod 2)$ and at-least $(x_1 + \cdots + x_n \geq k)$. The latter Boolean function is usually called a threshold function in the field of circuit complexity and is called a cardinality constraint in the field of SAT solving. A well known representative of the at-least class is the majority function $(x_1+\cdots+x_n > n/2)$. The `pysat` module [10] allows a user to select one of ten ways to encode the at-least constraint. See [6, 3, 13] for an experimental evaluation of different encodings of cardinality constraints.

The parity (PAR) and majority (MAJ) functions are also among the most frequently used in circuit lower bounds proofs. For example, many techniques for proving that parity and majority require constant depth circuits of exponential size are known (see [11, chapters 11 and 12] for an overview). At the same time, not much is known about CNF encodings from theoretical point of view. Sinz [22] proves lower and upper bounds on the number of clauses in a CNF encoding of at-least function: any CNF encoding has at least n clauses and there exists an encoding with $7n$ clauses. Kucera, Savický, Vorel [14] prove a lower bound $2n + o(n)$ on the number of clauses for at-most-one.

In this paper, we prove tradeoffs between the number m of clauses, the width k of clauses, and the number s of nondeterministic variables of CNF encodings of the parity and majority functions. With $s = O(n)$, the minimum number of clauses in a CNF encoding of parity is between $3n$ and $4n$, whereas any symmetric function can be encoded with at most $18n$ clauses. For any $s = s(n)$, the minimum k such that parity can be encoded as a k-CNF is $\frac{n}{s+1}$, up to a constant additive factor. Finally, when $s = n^{\alpha}$ (where $0 \leq \alpha \leq 1$ is a constant) the minimum number of clauses in a CNF encoding of both parity and majority is about $2^{n^{1-\alpha}}$.

The upper bounds are well-known and follow from a simple strategy: partition the input variables into blocks and encode the computed function for each block naively (we make it formal later in the text). Hence, our main contribution is lower bounds.

We derive them by using a tight connection between CNF encodings and depth-3 circuits as well as Satisfiability Coding Lemma due to Paturi, Pudlák, and Zane [19]. This lemma allows to prove a $2^{\sqrt{n}}$ lower bound on the size of depth-3 circuits computing the parity function. Interestingly, our lower bound on the number m of clauses (in any CNF encoding of parity) in terms of the number s of nondeterministic variables implies a lower bound $2^{\Omega(\sqrt{n})}$ for depth-3 circuits computing parity almost immediately, though it is not clear whether a converse implication can be easily proved. This connection provides an additional motivation for studying CNF encodings as a computational model for Boolean functions: on the one hand, it is routinely used in practice when translating a practical computational problem to a format acceptable by a SAT solver, on the other hand, lower bounds on the size of CNF encodings imply depth-3 circuit lower bounds.

2 General Setting

2.1 Computing Boolean Functions by CNFs

For a Boolean function $f(x_1, \dots, x_n)\colon \{0,1\}^n \to \{0,1\}$, we say that a CNF $F(x_1, \dots, x_n)$ *computes* f if $f \equiv F$, that is, for all $x_1, \dots, x_n \in \{0,1\}$, $f(x_1, \dots, x_n) = F(x_1, \dots, x_n)$. We treat a CNF as a set of clauses and by the *size* of a CNF we mean its number of clauses. It is well known that for every function f, there exists a CNF computing it. One way to construct such a CNF is the following: for every input $x \in \{0,1\}^n$ such that $f(x) = 0$, populate a CNF with a clause of length n that is falsified by x.

This method does not guarantee that the produced CNF has the minimal number of clauses: this would be too good to be true as the problem of finding a CNF of minimum size for a given Boolean function (specified by its truth table) is NP-complete as proved by Masek [18] (see also [1] and references herein). For example, for a function $f(x_1, x_2) = x_1$ the method produces a CNF $(\overline{x_1} \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2})$ whereas the function x_1 is already in CNF format.

It is well known that for many functions, the minimum size of a CNF is exponential. The canonical example is the parity function $\mathrm{PAR}_n(x_1, \dots, x_n) = x_1 \oplus \dots \oplus x_n$. The property of PAR_n that prevents it from being computable by short CNFs is its high *sensitivity*: by flipping *any* bit in *any* input $x \in \{0,1\}^n$, one flips the value of $\mathrm{PAR}_n(x)$.

Lemma 1. *The minimum size of a CNF computing* PAR_n *is* 2^{n-1}.

Proof. An upper bound follows from the method above by noting that $|\,\mathrm{PAR}_n^{-1}(0)| = 2^{n-1}$.

A lower bound is based on the fact that any clause of a CNF F computing PAR_n must contain all variables $x_1, \dots, x_n$. Indeed, if a clause $C \in F$ did not depend on x_i, one could find an input $x \in \{0,1\}^n$ that falsifies C (hence, $F(x) = \mathrm{PAR}_n(x) = 0$) and remains to be falsifying even after flipping x_i. As any clause of F has exactly n variables, it rejects exactly one $x \in \{0,1\}^n$. Hence, F must contain at least $|\,\mathrm{PAR}_n^{-1}(0)| = 2^{n-1}$ clauses. □

2.2 Encoding Boolean Functions by CNFs

We say that a CNF F *encodes* a Boolean function $f(x_1, \dots, x_n)$ if the following two conditions hold.

1. In addition to the input bits $x_1, \dots, x_n$, F also depends on s bits $y_1, \dots, y_s$ called *guess inputs* or *nondeterministic inputs*.
2. For every $x \in \{0,1\}^n$, $f(x) = 1$ iff there exists $y \in \{0,1\}^s$ such that $F(x,y) = 1$. In other words, for every $x \in \{0,1\}^n$,

$$f(x) = \bigvee_{y \in \{0,1\}^s} F(x,y)\,. \tag{1}$$

Such representations of Boolean functions are widely used in practice when one translates a problem to SAT. For example, the following CNF encodes PAR_4:

$$\begin{aligned}(x_1 \lor x_2 \lor \overline{y_1}) \land (x_1 \lor \overline{x_2} \lor y_1) \land (\overline{x_1} \lor x_2 \lor y_1) \land (\overline{x_1} \lor \overline{x_2} \lor \overline{y_1}) \land (y_1 \lor x_3 \lor \overline{y_2}) \land \\ (y_1 \lor \overline{x_3} \lor y_2) \land (\overline{y_1} \lor x_3 \lor y_2) \land (\overline{y_1} \lor \overline{x_3} \lor \overline{y_2}) \land (\overline{x_4} \lor y_2) \land (x_4 \lor \overline{y_2})\,.\end{aligned} \tag{2}$$

This example generalizes as follows. To encode $x_1 \oplus \dots \oplus x_n$ as CNF, one introduces s nondeterministic variables $y_1, \dots, y_s$ and partitions the set of input variables into $s+1$ blocks of size at most $\lceil n/(s+1) \rceil$: $\{x_1, x_2, \dots, x_n\} = X_1 \sqcup X_2 \sqcup \dots \sqcup X_{s+1}$. Then, one writes down the following $s+1$ parity functions in CNF:

$$\begin{aligned}&\left(y_1 = \bigoplus_{x \in X_1} x\right), \left(y_2 = y_1 \oplus \bigoplus_{x \in X_2} x\right), \dots, \\ &\qquad \left(y_s = y_{s-1} \oplus \bigoplus_{x \in X_s} x\right), \left(1 = y_s \oplus \bigoplus_{x \in X_{s+1}} x\right).\end{aligned} \tag{3}$$

The value for the parameter s is usually determined experimentally. For example, Prestwich [20] reports that taking $s = 10$ gives the best results when solving the minimal disagreement parity learning problem using local search based SAT solvers.

The construction above allows one to encode parity as a CNF with the following upper bounds on the number m of clauses, the number s of nondeterministic variables, and the width k of clauses.

Limited nondeterminism: using s nondeterministic variables, one can encode parity either as a CNF with at most

$$m \le (s+1)2^{\lceil n/(s+1) \rceil + 2 - 1} \le 4(s+1)2^{n/(s+1)} \tag{4}$$

clauses or as a k-CNF, where

$$k = 2 + \lceil n/(s+1) \rceil \le 3 + n/(s+1)\,. \tag{5}$$

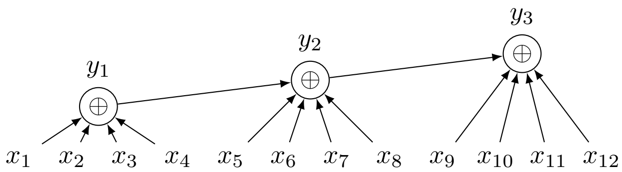
Unlimited nondeterminism: one can encode parity as a CNF with at most

$$m \le 4n \tag{6}$$

clauses (to do this, use $s = n-1$ nondeterministic variables; then, each of n functions in (3) can be written in CNF using at most four clauses).

2.3 Boolean Circuits and Tseitin Transformation

A natural way to get a CNF encoding of a Boolean function f is to take a Boolean circuit computing f and apply Tseitin transformation [23]. We describe this transformation using a toy example. The following circuit computes PAR_{12} with three gates: the fan-in of y_2 and y_3 is equal to five whereas the fan-in of y_1 is four. It has twelve inputs and three gates (one of which is an output gate), its depth is equal to three.



$$y_1 = x_1 \oplus x_2 \oplus x_3 \oplus x_4$$
$$y_2 = y_1 \oplus x_5 \oplus x_6 \oplus x_7 \oplus x_8$$
$$y_3 = y_2 \oplus x_9 \oplus x_{10} \oplus x_{11} \oplus x_{12}$$

To the right of the circuit, we show the functions computed by each gate. One can translate each line into CNF. Adding a clause (y_3) to the resulting CNF gives a CNF encoding of the function computed by the circuit. In fact, the CNF (3) can be obtained this way (after propagating the value of the output gate).

Observation 2. *If a function $f\colon \{0,1\}^n \to \{0,1\}$ can be computed by a circuit of fan-in two with g gates, then f can be encoded as a 3-CNF with $s = g$ nondeterministic variables and $m = 4g$ clauses.*

Proof. For every gate g computing $g_1 \circ g_2$, where $\circ$ is a binary Boolean operation and g_1 and g_2 are direct predecessors of g, one writes down four 3-clauses expressing the fact that $g = g_1 \circ g_2$. (More formally, one considers a Boolean function $h(g, g_1, g_2) = [g = g_1 \circ g_2]$. Then, $|h^{-1}(0)| = 4$ and it can be encoded as four 3-clauses.) □

2.4 Upper Bounds for Symmetric Functions

The parity and majority are symmetric functions. Recall that a Boolean function is called symmetric if its value depends on the sum (over integers) of the input bits only. To encode in CNF a symmetric function $f(x_1, \dots, x_n)$, one can use a construction similar to (3). Namely, partition the input variables into t blocks of size n/t: $\{x_1, x_2, \dots, x_n\} = X_1 \sqcup X_2 \sqcup \dots \sqcup X_t$. Let $Y_1, \dots, Y_t$ be t blocks each consisting of $\log n$ nondeterministic variables. Let Y_i be the bits of an integer $0 \le y_i \le n$. We then expand as a naive CNF each of the following identities:

$$\left(y_1 = \sum_{x \in X_1} x\right), \left(y_2 = y_1 + \sum_{x \in X_2} x\right), \dots, \left(y_t = y_{t_1} + \sum_{x \in X_t} x\right).$$

Then, y_s is equal to $\sum_{i=1}^{n} x_i$. Then, in at most $2^{|Y_s|} = 2^{\log n} = n$ additional clauses one can enforce the value of $f(x_1, \ldots, x_n)$. Thus, the total number of clauses is

$$m \le t \cdot \log n \cdot 2^{n/t + \log n} + n\,.$$

Thus, for any integer t, one can use $s = t \log n$ nondetermenistic variables to encode a symmetric function as a CNF with

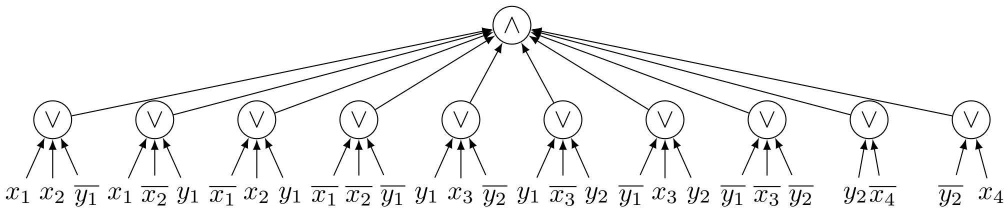
$$m \le s \cdot n \cdot 2^{\frac{n \log n}{s}} + n$$

clauses.

It is known that every symmetric Boolean function can be computed by a circuit (over the full binary basis) of size $4.5n + o(n)$ [5]. Observation 2 then implies that every Boolean function admits a 3-CNF encoding with $4.5n + o(n)$ nondeterministic variables and $18n + o(n)$ clauses.

2.5 Depth-3 Circuits

A CNF can be viewed as a depth-2 circuit where the output gate is an AND, all other gates are ORs, and the inputs are variables and their negations. For example, the following circuit corresponds to the CNF (2). Such depth-2 circuits are also denoted as AND $\circ$ OR circuits.

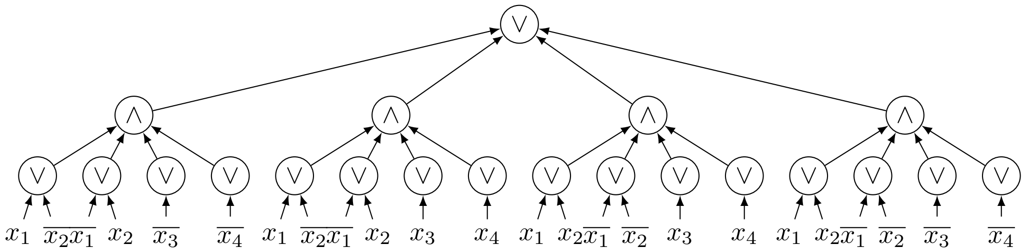


Depth-3 circuits is a natural generalization of CNFs: a Σ_3-*circuit* is simply an OR of CNFs. In a circuit, these CNFs are allowed to share clauses. A Σ_3-*formula* is a Σ_3-circuit whose CNFs do not share clauses (in other words, it is a circuit where the out-degree of every gate is equal to one).

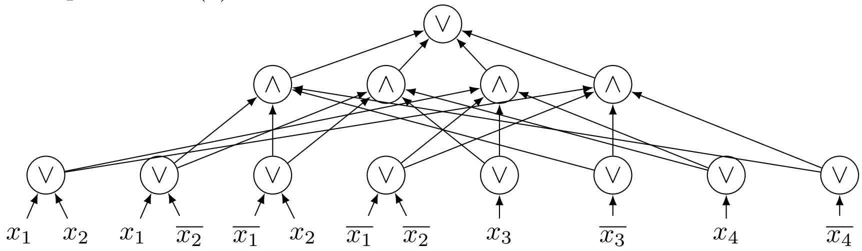
Equation (1) shows a tight connection between CNF encodings and depth-3 circuits of type OR $\circ$ AND $\circ$ OR. Namely, let $F(x_1, \ldots, x_n, y_1, \ldots, y_s) = \{C_1, \ldots, C_m\}$ be a CNF encoding of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$. Then, $f(x) = \vee_{y \in \{0,1\}^s} F(x, y)$. By assigning y's in all 2^s ways, one gets an Σ_3-formula that computes f:

$$f(x) = \bigvee_{j \in [2^s]} F_j(x)\,, \tag{7}$$

where each F_j is a CNF. We call this an *expansion* of F. For example, an expansion of the CNF (2) looks as follows. It is an OR of four CNFs.



An expansion is a formula: it is an OR of CNFs, every gate has out-degree one. One can also get a *circuit-expansion*: in this case, gates are allowed to have out-degree more than one; alternatively, CNFs are allowed to share clauses. For example, this is a circuit-expansion of (2).



Below, we show that CNF encodings and depth-3 circuits can be easily transformed one into the other. It will prove convenient to define the size of a circuit as its number of gates *excluding* the output gate. This way, the size of a CNF formula equals its number of clauses (a CNF is a depth-2 formula). By a $\Sigma_3(t,r)$-circuit we denote a Σ_3-circuit having at most t ANDs on the second layer and at most r ORs on the third layer (hence, its size is at most $t+r$).

Lemma 3. *Let $F(x_1,\ldots,x_n,y_1,\ldots,y_s)$ be a CNF encoding of size m of a function $f\colon\{0,1\}^n\to\{0,1\}$. Then, f can be computed by a $\Sigma_3(2^s, m\cdot 2^s)$-formula and by a $\Sigma_3(2^s,m)$-circuit.*

Proof. Let $F=\{C_1,\ldots,C_m\}$. To expand F as $\bigvee_{j\in[2^s]}F_j$, we go through all 2^s assignments to nondeterministic variables $y_1,\ldots,y_s$. Under any such assignment, each clause C_i is either satisfied or becomes a clause $C_i'\subseteq C_i$ resulting from C_i by removing all its non-deterministic variables. Thus, for each $j\in[2^s]$, $F_j\subseteq\{C_1',\ldots,C_m'\}$. The corresponding Σ_3-formula contains at most 2^s+m2^s gates: there are 2^s gates for F_j's, each F_j contains no more than m clauses. The corresponding Σ_3-circuit contains no more than 2^s+m gates: there are 2^s gates for F_j's and m gates for $C_1',\ldots,C_m'$ (each F_j selects which of these m clauses to contain). $\square$

Below, we show a converse transformation.

Lemma 4. *Let C be a $\Sigma_3(t,r)$-formula (circuit) computing a Boolean function $f\colon\{0,1\}^n\to\{0,1\}$. Then, f can be encoded as a CNF with $\lceil\log t\rceil$ nondeterministic variables of size r ($2rt$, respectively).*

Proof. Let $C=F_1\vee\cdots\vee F_t$ be a Σ_3-formula (hence, $r=\mathrm{size}(F_1)+\cdots+\mathrm{size}(F_t)$). Introduce $s=\lceil\log t\rceil$ nondeterministic variables $y_1,\ldots,y_s$. Then, for every assignment to $y_1, \dots, y_s$, take the corresponding CNF F_i ($1 \le i \le 2^s$ is the unique integer corresponding to this assignment) and add y_i's with the corresponding signs to every clause of F_i. Call the resulting CNF F_i'. Then, $F = F_1' \wedge \dots \wedge F_{2^s}'$ encodes f and F has at most r clauses.

If C is a Σ_3-circuit, we need to create a separate copy of every gate corresponding to a clause in each of 2^s CNFs. Hence, the size of the resulting CNF encoding is at most $r2^s \le 2rt$. $\square$

3 Lower Bounds for CNF Encodings

3.1 Connection to Circuit Lower Bounds

Before proving lower bounds for CNF encodings of parity and majority, we argue that establishing strong lower bounds for CNF encodings is a challenging task. Indeed, Lemma 4 and Tseitin transformation provide a simple way to transform a circuit into a CNF encoding. Through this transformation, lower bounds for CNF encodings translate to circuit lower bounds.

For the parity function, the best known lower bound on depth-3 circuits is $\Omega(2^{\sqrt{n}})$ [19]. If one additionally requires that a circuit is a formula, i.e., that every gate has out-degree at most 1, then the best lower bound is $\Omega(2^{2\sqrt{n}})$ [9]. Both lower bounds are tight up to polynomial factors. For the majority function, there is a depth-3 circuit lower bound $2^{\Omega(\sqrt{n})}$ [7, 8] and a depth-3 formula upper bound $2^{O(\sqrt{n \log n})}$ [9, 12]. Interestingly, these lower bounds show that the parameters of Lemma 4 cannot be substantially improved. Indeed, by plugging in a CNF encoding of PAR_n with $s = \sqrt{n}$ and $m = O(\sqrt{n}2^{\sqrt{n}})$ (see (3)), one gets a Σ_3-formula and a Σ_3-circuit of size $2^{2\sqrt{n}}$ and $2^{\sqrt{n}}$, respectively, up to polynomial factors. As discussed above, these bounds are known to be optimal.

Below (see (15)), we prove that, for any CNF encoding of PAR_n with s nondeterministic variables and m clauses, $m \ge \Omega\left(\frac{s+1}{n} \cdot 2^{n/(s+1)}\right)$. Now, let C be a $\Sigma_3(t,r)$-formula computing PAR_n. Lemma 4 guarantees that PAR_n can be encoded as a CNF of size r with $\lceil \log t \rceil$ nondeterministic variables. Then,

$$\mathrm{size}(C) = t + r \ge t + \Omega\left(\frac{1}{n} \cdot 2^{\frac{n}{\log t + 2}}\right) \ge \frac{1}{n}\left(t + \Omega\left(2^{\frac{n}{\log t+2}}\right)\right) \ge \Omega\left(\frac{2^{\sqrt{n}}}{n}\right).$$

Similarly, if C is a $\Sigma_3(t,r)$-circuit, Lemma 4 guarantees that PAR_n can be encoded as a CNF of size $2rt$ with $\lceil \log t \rceil$ nondeterministic variables. Then,

$$\mathrm{size}(C) = t + r \ge t + \Omega\left(\frac{1}{2tn} \cdot 2^{\frac{n}{\log t+2}}\right) \ge \Omega\left(\frac{2^{\sqrt{n/2}}}{n}\right).$$

Thus, lower bounds for CNF encodings imply lower bounds for depth-3 circuits. Note that for no Boolean function from NP, we know how to prove a $2^{\omega(\sqrt{n})}$ lower bound on the size of a depth-3 circuit computing it. (When saying that a Boolean function belongs to the class NP, we mean that we have an infinite sequence of functions $\{f_n\}_{n=1}^{\infty}$ such that the language $\bigcup_{n=1}^{\infty} f_n^{-1}(1)$ is in NP.)

Open Problem 5. *Find a Boolean function from* NP *that cannot be computed by depth-*3 *circuits of size* $2^{O(\sqrt{n})}$.

Another challenging open problem is to find a Boolean function that has no depth-3 circuits of size $2^{O(n/\log\log n)}$ where the bottom fan-in is bounded by n^{ε} for some constant $\varepsilon < 1$. As proved by Valiant [24], such a function cannot be computed by circuits having fan-in 2, size $O(n)$, and depth $O(\log n)$. This is a notoriously hard open problem in circuit complexity. Interestingly, in this reduction, Valiant essentially shows that a function that can be computed by a linear size and logarithmic depth binary circuit, admits a non-trivial CNF encoding.

Open Problem 6. *Find a Boolean function from* NP *that cannot be computed binary circuits of depth* $O(\log n)$ *and size* $O(n)$.

In fact, for fan-in two circuits, the best known lower bound is $3.1n$ [16] (even if one restricts depth to $O(\log n)$).

Open Problem 7. *Find a Boolean function from* NP *that cannot be computed fan-in two circuits of size* $3.2n$.

Below, we show that these open problems can be attacked from the CNF encodings angle.

Lemma 8. *Let* $f\colon \{0,1\}^n \to \{0,1\}$ *be a function from* NP.

1. *If f admits no CNF encoding with s non-deterministic variables and $m = O(2^{n/s})$, then f has no depth-*3 *circuits of size $2^{O(\sqrt{n})}$ (thus resolving Open Problem 5).*
2. *If f admits no CNF encoding with $s = O(\frac{n}{\log\log n})$ and $m = O(\frac{n}{\log\log n}2^{n^{\varepsilon}})$ for any constant $\varepsilon > 0$, then f has no fan-in two circuits of size $O(n)$ and depth $O(\log n)$ (thus resolving Open Problem 6).*
3. *If f admits no CNF encoding with $s = 3.2n$ and $m = 13n$, then f cannot be computed by fan-in two circuits of size $3.2n$ (thus resolving Open Problem 7).*

Proof. 1. Consider a $\Sigma_3(t,r)$-circuit C of size $t+r$. Lemma 4 guarantees that C can be encoded as a CNF of size $m = 2rt$ with $s = \lceil \log t \rceil$ nondeterministic variables. Since $m = O(2^{n/s})$, $r = \frac{1}{2t}O(2^{n/\log t})$. Hence,

$$\operatorname{size}(C) = t + r = t + \frac{1}{2t} \cdot O(2^{n/\log t}) = 2^{O(\sqrt{n})}$$

(either $t \geq 2^{\sqrt{n/2}}$ or $\frac{1}{2t} \cdot O(2^{n/\log t}) \geq 2^{\sqrt{n/2}}$).

2. We show that any circuit of size $O(n)$ and depth $O(\log n)$ can be transformed into a CNF with desired parameters. Take a circuit C of depth $d = O(\log n)$ with $O(n)$ fan-in 2 gates. Since each gate has fan-in 2, the number R of wires is at most $O(n)$.

   As proved by Valiant [25], for any directed graph of depth d (where the depth is the length of a longest path in the graph) with R edges and any integer $1 \leq r \leq \log d$, it is possible to remove $\frac{r}{\log d}R$ edges so that the depth of the resulting graph is at most $d/2^r$.

   For a parameter r to be specified later, apply Valiant's lemma to the circuit C. For each eliminated wire, we introduce a nondeterministic variable and to justify

its value, we add at most $2^{2^{d/2^r}}$ clauses. This way, we obtain a CNF encoding with at most $O(\frac{nr}{\log d})$ nondeterministic variables, and at most $O(\frac{nr}{\log d}2^{2^{d/2^r}})$ clauses. Since $d = O(\log n)$ and by taking $r \approx \log(1/\varepsilon)$ (a constant), we obtain a CNF encoding with $O(\frac{n}{\log\log n})$ nondeterministic variables, and $O(\frac{n}{\log\log n}2^{n^\varepsilon})$ clauses.
3. If f had a fan-in two circuit of size $3.2n$, then, using Observation 2, it could be encoded as CNF with $s = 3.2n$ and $m = 4 \cdot 3.2n \le 13n$.

□

To conclude this section, we note that, as it usually happens, proving *non-constructively* the existence of a Boolean function with no small CNF encoding is easy. Hence, the main challenge is to find an *explicit* such function, where by an explicit one usually means a function from NP (or E^{NP}). Indeed, there are

$$2^{(2n+2s)m}$$

CNF encodings with n input variables, s nondeterministic variables, and m clauses: there are m clauses, each of them is a subset of n input and s nondeterministic variables as well as their negations. Since there are 2^{2^n} Boolean functions, as long as

$$(2n+2s)m < 2^n,$$

there exists a Boolean function that cannot be encoded as CNF with s nondeterministic variables and m clauses.

3.2 Isolated Solutions

In this section, we prove two technical lemmas needed in the proof of lower bounds.

The essential property of PAR and MAJ functions used in our lower bound proofs is that they have a lot of isolated solutions. An assignment $x \in f^{-1}(1)$ is called *isolated in direction i* if flipping the i-th bit of x gives an assignment $x' \in f^{-1}(0)$. We say that x is *d-isolated* if there are d such directions. By $I_{f,x}$ we denote the set of directions for x. If a CNF F computes f, then for each d-isolated $x \in f^{-1}(1)$ and for each direction $i \in I_{f,x}$, F must contain a clause that is satisfied by x_i only. Following [19], we call such a clause *critical with respect to* (x,i). Fix a shortest critical clause w.r.t. (x,i) and denote it by $C_{F,x,i}$. Then, for a d-isolated satisfying assignment x, define its *weight* w.r.t. F as

$$w_F(x) = \sum_{i \in I_{f,x}} \frac{1}{|C_{F,x,i}|}\,.$$

The following lemma shows that a CNF cannot accept too many assignment of large weight. It was proved by [19] for the case $d = n$. In the Appendix, we show that minor modifications of the proof allows to extend the result to any d.

Lemma 9. *For any $\mu > 0$ and any integer $0 \le d \le n$, a CNF F over n variables has at most $2^{n-\mu}$ d-isolated satisfying assignments of weight at least μ.*

The notion of isolated solution extends to CNF encodings in a natural way. Namely, consider a function f and d-isolated assignment $x \in f^{-1}(1)$. Let $F(x,y)$ be a CNF encoding of f, and $y \in \{0,1\}^s$ be such that $F(x,y) = 1$. Then, for any $i \in I_{f,x}$,

F contains a clause that becomes falsified if one flips the bit x_i. We call it critical w.r.t. (x, y, i).

Lemma 10. *Let $F(x_1, \ldots, x_n, y_1, \ldots, y_s)$ be a CNF encoding of $f\colon \{0,1\}^n \to \{0,1\}$ with m clauses. Let $d \in [n]$ and $S = \{x \in f^{-1}(1)\colon x \text{ is } d\text{-isolated}\}$. Then, for every $0 < \varepsilon \le d \ln 2 - s - 1$,*

$$m \ge (s+1+\varepsilon)2^{\frac{d}{s+1+\varepsilon}}(|S|2^{-n} - 2^{-1-\varepsilon}).$$

Proof. Consider an expansion of F:

$$f(x) = \bigvee_{j \in [2^s]} F_j(x)\,.$$

We extend the definitions of $C_{F,x,i}$ and $w(x)$ to CNFs with nondeterministic variables as follows. Let $x \in f^{-1}(1)$ be d-isolated with directions $I = \{i_1, i_2, \ldots, i_d\}$. Let $j \in [2^s]$ be the smallest index such that $F_j(x) = 1$. For $i \in I$, let $C'_{F,x,i} = C_{F_j,x,i}$ (that is, we simply take the first F_j that is satisfied by x and take its critical clause w.r.t. (x, i)). Then, the weight $w'_F(x)$ of x w.r.t. to F is defined as $w_{F_j}(x)$:

$$w'_F(x) := w_{F_j}(x) = \sum_{i \in I} \frac{1}{|C'_{(F_j,x,i)}|} = \sum_{i \in I} \frac{1}{|C'_{(F,x,i)}|}\,.$$

For $l \in [n]$, let also $N_{l,F}(x) = |\{i \in [n]\colon |C'_{F,x,i}| = l\}|$ be the number of critical clauses (w.r.t. x) of length l. Clearly,

$$w'_F(x) = \sum_{l \in [n]} \frac{N_{l,F}(x)}{l}\,. \tag{8}$$

For a parameter ε, split $S \subseteq f^{-1}(1)$ into light and heavy parts:

$$H = \{x \in S\colon w'_F(x) \ge s+1+\varepsilon\}\,,$$
$$L = \{x \in S\colon w'_F(x) < s+1+\varepsilon\}\,.$$

We claim that

$$|H| \le 2^s \cdot 2^{n-s-1-\varepsilon}\,. \tag{9}$$

Indeed, for every $x \in H$, $w'_F(x) = w_{F_j}(x)$ for some $j \in [2^s]$, and by Lemma 9, F_j cannot accept more than $2^{n-s-1-\varepsilon}$ isolated solutions of weight at least $s+1+\varepsilon$.

Now we show that

$$|L| \le m \cdot 2^n 2^{\frac{-d}{s+1+\varepsilon}}(1/(s+1+\varepsilon)) \tag{10}$$

Let $F = \{C_1, \ldots, C_m\}$. For every $k \in [m]$, let $C'_k \subseteq C_k$ be the clause C_k with all nondeterministic variables removed. Hence, for every $j \in [2^s]$, $F_j \subseteq \{C'_1, \ldots, C'_m\}$. For $l \in [n]$, let $m_l = |\{k \in [m]\colon |C'_k| = l\}|$ be the number of such clauses of length l. Consider a clause C'_k and let $l = |C'_k|$. Then, there are at most $l2^{n-l}$ pairs (x, i), where

$x \in S$ and $i \in [n]$, such that $C'_{F,x,i} = C'_k$: there are at most l choices for i, fixing i fixes the values of all l literals in C'_k (all of them are equal to zero except for the i-th one), and there are no more than 2^{n-l} choices for the other bits of x. Recall that $N_{l,F}(x)$ is the number of critical clauses w.r.t. x of length l. Thus, we arrive at the following inequality:

$$m_l \cdot l \cdot 2^{n-l} \geq \sum_{x \in S} N_{F,l}(x) \geq \sum_{x \in L} N_{F,l}(x)\,.$$

Then,

$$m = \sum_{l \in [n]} m_l \geq \sum_{l \in [n]} \frac{\sum_{x \in L} N_{F,l}(x)}{l2^{n-l}} = \sum_{x \in L} \sum_{l \in [n]} \frac{N_{F,l}(x)}{l2^{n-l}} = \sum_{x \in L} d2^{-n} \sum_{l \in [n]} \frac{N_{F,l}(x)}{d} \cdot \frac{2^l}{l}\,. \tag{11}$$

To estimate the last sum, let

$$T(x) = \sum_{l \in [n]} \frac{N_{F,l}(x)}{d} \cdot \frac{2^l}{l} = \sum_{l \in [n]} \frac{N_{F,l}(x)}{d} \cdot g(l)\,,$$

where $g(l) = \frac{2^l}{l}$. Since $g(l)$ is convex (for $l > 0$) and $\sum_{l \in [n]} \frac{N_{F,l}(x)}{d} = 1$, Jensen's inequality gives

$$T(x) \geq g\left(\sum_{l \in [n]} \frac{N_{F,l}(x)}{d} \cdot l\right)\,. \tag{12}$$

Further, Sedrakyan's inequality[1] (combined with (8) and $\sum_{l \in [n]} N_{F,l}(x) = d$) gives

$$\sum_{l \in [n]} l N_{F,l}(x) = \sum_{l \in [n]} \frac{N^2_{F,l}(x)}{N_{F,l}(x)/l} \geq \frac{\left(\sum_{l \in [n]} N_{F,l}(x)\right)^2}{\sum_{l \in [n]} N_{F,l}(x)/l} = \frac{d^2}{w'_F(x)}\,. \tag{13}$$

Since $g(l)$ is monotonically increasing for $l \geq 1/\ln 2$ and $w'_F(x) < s + 1 + \varepsilon$ for every $x \in L$, combining (12) and (13), we get

$$T(x) \geq g\left(\frac{d}{w'_F(x)}\right) \geq g\left(\frac{d}{s+1+\varepsilon}\right)\,, \tag{14}$$

Last inequality is true since $\varepsilon \leq d \ln 2 - s - 1$.

Thus,

$$\begin{aligned} m &\geq \sum_{x \in L} d2^{-n} T(x) \geq && \text{(11 and 14)} \\ &\geq \sum_{x \in L} d2^{-n} g\left(\frac{d}{s+1+\varepsilon}\right) = && \text{(definition of } g\text{)} \end{aligned}$$

[1] Sedrakyan's inequality is a special case of Cauchy–Schwarz inequality: for all $a_1, \ldots, a_n \in \mathbb{R}$ and $b_1, \ldots, b_n \in \mathbb{R}_{>0}$, $\sum_{i=1}^n a_i^2/b_i \geq \left(\sum_{i=1}^n a_i\right)^2 / \sum_{i=1}^n b_i$.

$$= |L| 2^{-n} 2^{\frac{d}{s+1+\varepsilon}} (s+1+\varepsilon) \, .$$

Using (9), (10) and fact that $|H| + |L| = |S|$ we have

$$\begin{aligned} m &\geq (|S| - |H|) 2^{-n} 2^{\frac{d}{s+1+\varepsilon}} (s+1+\varepsilon) \\ &\geq (|S| - 2^{n-1-\varepsilon}) 2^{-n} 2^{\frac{d}{s+1+\varepsilon}} (s+1+\varepsilon) \\ &= (s+1+\varepsilon) 2^{\frac{d}{s+1+\varepsilon}} (|S| 2^{-n} - 2^{-1-\varepsilon}). \end{aligned}$$

□

3.3 Lower Bounds for Parity

In this section, we prove that the upper bounds (4)–(6) on m and k shown in Section 2.2 are essentially optimal.

Theorem 11. *Let F be a CNF encoding of* PAR_n *with m clauses, s nondeterministic variables, and maximum clause width k.*

1. *The parameters s and m cannot be too small simultaneously: if $s = O(n)$, then*

$$m \geq \Omega\left(\frac{s+1}{n}\right) \cdot 2^{\frac{n}{s+1}} \, . \tag{15}$$

2. *The parameters s and k cannot be too small simultaneously:*

$$k \geq \frac{n}{s+1} \, . \tag{16}$$

3. *The parameter m cannot be too small:*

$$m \geq 3n - 9 \, . \tag{17}$$

3.3.1 Limited Nondeterminism

The first inequality is a straightforward consequence of Lemma 10.

Proof of (15)*, $m \geq \Omega((s+1)2^{n/(s+1)}/n)$.* Consider two cases.

1. $s \leq n/2$. Let S be a set of n-isolated solutions of PAR_n. Note that $|S| = 2^{n-1}$. By Lemma 10, if

$$0 < \varepsilon \leq n \ln 2 - s - 1 \, , \tag{18}$$

then

$$m \geq (s+1+\varepsilon) 2^{\frac{n}{s+1+\varepsilon}} (1/2 - 2^{-1-\varepsilon}) = (s+1+\varepsilon) 2^{\frac{n}{s+1}} 2^{\frac{-n\varepsilon}{(s+1)(s+1+\varepsilon)}} (1/2 - 2^{-1-\varepsilon}) \, .$$

Set $\varepsilon = 1/n$ (the inequalities (18) are satisfied, since $s \leq n/2$). Then,

$$\left(\frac{1}{2} - \frac{1}{2^{\frac{1}{n}+1}}\right) = \Theta\left(\frac{1}{n}\right) \, .$$

Also,

$$\frac{1}{2} \leq 2^{\frac{-1}{(s+1)(s+1+1/n)}} \leq 1\,,$$

as $2^{-1/x}$ is increasing for $x > 0$. Thus,

$$m \geq \Omega\left(\frac{s+1}{n} \cdot 2^{\frac{n}{s+1}}\right)\,.$$

2. $n/2 < s = O(n)$. In this case, the lower bound becomes obvious.

□

3.3.2 Width of Clauses

To prove the lower bound $k \geq n/(s+1)$, we use the following corollary of the Satisfiability Coding Lemma.

Lemma 12 (Lemma 2 in [19])**.** *Any k-CNF $F(x_1, \ldots, x_n)$ has at most $2^{n-n/k}$ isolated satisfying assignments.*

Proof of (16)*, $k \geq n/(s+1)$.* Consider a k-CNF $F(x_1, \ldots, x_n, y_1, \ldots, y_s)$ that encodes PAR_n. Expand F to an OR of 2^s k-CNFs:

$$\mathrm{PAR}_n(x) = \bigvee_{j \in [2^s]} F_j(x)\,.$$

By Lemma 12, each F_j accepts at most $2^{n-n/k}$ isolated solutions. Hence,

$$2^s \geq \frac{2^{n-1}}{2^{n-n/k}} = 2^{n/k-1}$$

and thus, $k \geq n/(s+1)$. □

3.3.3 Unlimited Nondeterminism

In this subsection, we prove the lower bound $m \geq 3n - 9$.

Proof of (17)*, $m \geq 3n-9$.* We use induction on n. The base case $n \leq 3$ is clear. To prove the induction step, assume that $n > 3$ and consider a CNF encoding $F(x_1, \ldots, x_n, y_1, \ldots, y_s)$ of PAR_n with the minimum number of clauses. Below, we show that one can find k deterministic variables (where $k = 1$ or $k = 2$) such that assigning appropriately chosen constants to them reduces the number of clauses by at least $3k$, respectively. The resulting function computes PAR_{n-k} or its negation. It is not difficult to see that the minimum number of clauses in encodings of PAR and its negation are equal (by flipping the signs of all occurrences of any deterministic variable in a CNF encoding of PAR, one gets a CNF encoding of the negation of PAR, and vice versa). Hence, one can proceed by induction and conclude that F contains at least $3(n-k) - 9 + 3k = 3n - 9$ clauses.

To find the required k deterministic variables, we go through a number of cases. In the analysis below, by a d-literal we mean a literal that appears exactly d times in F, a d^+-literal appears at least d times. A (d_1, d_2)-literal occurs d_1 times positively and d_2 times negatively. Other types of literals are defined similarly. We treat a clause as a set of literals (that do not contain a literal together with its negation) and a CNF formula as a set of clauses.

Note that for all $i \in [s]$, y_i must be a $(2^+, 2^+)$-literal. Indeed, if y_i (or $\overline{y_i}$) is a 0-literal, one can assign $y_i \leftarrow 0$ ($y_1 \leftarrow 1$, respectively). It is not difficult to see that the resulting formula still encodes PAR. If y_i is a $(1, t)$-literal, one can eliminate it using resolution: for all pairs of clauses $C_0, C_1 \in F$ such that $\overline{y_i} \in C_0$ and $y_i \in C_1$, add a clause $C_0 \cup C_1 \setminus \{y_i, \overline{y_i}\}$ (if this clause contains a pair of complementary literals, ignore it); then, remove all clauses containing y_i or $\overline{y_i}$. The resulting formula still encodes PAR_n, but has a smaller number of clauses than F (we remove $1 + t$ clauses and add at most t clauses).

In the case analysis below, by l_i we denote a literal that corresponds to a deterministic variable x_i or its negation $\overline{x_i}$.

1. *F contains a 3^+-literal l_i.* Assigning $l_i \leftarrow 1$ eliminates at least three clauses from F.
2. *F contains a 1-literal l_i.* Let $l_i \in C \in F$ be a clause containing l_i. C cannot contain other deterministic variables: if $l_i, l_j \in C$ (for $i \neq j \in [n]$), consider $x \in \{0,1\}^n$ such that $\mathrm{PAR}_n(x) = 1$ and $l_i = l_j = 1$ (such x exists since $n > 3$), and its extension $y \in \{0,1\}^s$ such that $F(x, y) = 1$; then, F does not contain a critical clause w.r.t. (x, y, i). Clearly, C cannot be a unit clause, hence it must contain a nondeterministic variable y_j. Consider $x \in \{0,1\}^n$, such that $\mathrm{PAR}_n(x) = 1$ and $l_i = 1$, and its extension $y \in \{0,1\}^s$ such that $F(x, y) = 1$. If $y_j = 1$, then F does not contain a critical clause w.r.t. (x, y, i). Thus, for every $(x, y) \in \{0,1\}^{n+s}$ such that $F(x, y) = 1$ and $l_i = 1$, it holds that $y_j = 0$. This observation allows us to proceed as follows: first assign $l_i \leftarrow 1$, then assign $y_j \leftarrow 0$. The former assignment satisfies the clause C, the latter one satisfies all the clauses containing $\overline{y_j}$. Thus, at least three clauses are removed.
3. *For all $i \in [n]$, x_i is a $(2,2)$-literal.* If there is no clause in F containing at least two deterministic variables, then F contains at least $4n$ clauses and there is nothing to prove. Let $l_i, l_j \in C_1 \in F$, where $i \neq j$, be a clause containing two deterministic variables and let $l_i \in C_2 \in F$ and $l_j \in C_3 \in F$ be the two clauses containing other occurrences of l_i and l_j ($C_1 \neq C_2$ and $C_1 \neq C_3$, but it can be the case that $C_2 = C_3$).

   Assume that C_2 contains another deterministic variable: $l_k \in C_2$, where $k \neq i, j$. Consider $x \in \{0,1\}^n$, such that $\mathrm{PAR}_n(x) = 1$ and $l_i = l_j = l_k = 1$ (such x exists since $n > 3$), and its extension $y \in \{0,1\}^s$ such that $F(x, y) = 1$. Then, F does not contain a critical clause w.r.t. (x, y, i): C_1 is satisfied by l_j, C_2 is satisfied by l_k. For the same reason, C_2 cannot contain the literal l_j. Similarly, C_3 cannot contain other deterministic variables and the literal l_i. (At the same time, it is not excluded that $\overline{l_j} \in C_2$ or $\overline{l_i} \in C_3$.) Hence, $C_2 \neq C_3$. Note that each of C_2 and C_3 must contain at least one nondeterministic variable: otherwise, it would be possible to falsify F by assigning l_i and l_j.

(a) *At least one of C_2 and C_3 contains a single nondeterministic variable.* Assume that it is C_2:

$$\{l_i, y_1\} \subseteq C_2 \subseteq \{l_i, \overline{l_j}, y_1\}\,.$$

Assign $l_j \leftarrow 1$. This eliminates two clauses: C_1 and C_3 are satisfied. Also, under this substitution, $C_2 = \{l_i, y_1\}$ and l_i is a 1-literal. We claim that in any satisfying assignment of the resulting formula F', $l_i = \overline{y_1}$. Indeed, if (x, y) satisfies F' and $l_i = y_1$, then $l_i = y_1 = 1$ (otherwise C_2 is falsified). But then there is no critical clause in F' w.r.t. (x, y, i). Since in every satisfying assignment $l_i = \overline{y_1}$, we can replace every occurrence of y_1 ($\overline{y_1}$) by $\overline{l_i}$ (y_1, respectively). This, in particular, satisfies the clause C_2.

(b) *Both C_2 and C_3 contain at least two nondeterministic variables:*

$$\{l_i,\ l_j\} \subseteq C_1, \quad \{l_i,\ y_1,\ y_2\} \subseteq C_2, \quad \{l_j,\ y_3,\ y_4\} \subseteq C_3\,.$$

Here, y_1 and y_2 are different variables, y_3 and y_4 are also different, though it is not excluded that some of y_1 and y_2 coincide with some of y_3 and y_4. Let $Y \subseteq \{y_1, \dots, y_s\}$ be nondeterministic variables appearing in C_2 or C_3.

Recall that for every $(x, y) \in \{0,1\}^{n+s}$ such that $F(x, y) = 1$ and $l_i = l_j = 1$, it holds that $y = 0$ for all $y \in Y$. This means that if a variable $y \in Y$ appears in both C_2 and C_3, then it has the same sign in both clauses. Consider two subcases.

(i) $Y = \{y_1, y_2\}$:

$$\{l_i,\ l_j\} \subseteq C_1, \quad \{l_i,\ y_1,\ y_2\} \subseteq C_2, \quad \{l_j,\ y_1,\ y_2\} \subseteq C_3\,.$$

Assume that $\overline{y_1} \notin C_1$. Assign $l_i \leftarrow 1$, $l_j \leftarrow 1$. Then, assigning $y_1 \leftarrow 0$ eliminates at least two clauses. Let us show that there remains a clause that contains $\overline{y_2}$. Consider $x \in \mathrm{PAR}_n^{-1}(1)$, such that $l_i = l_j = 1$, and its extension $y \in \{0,1\}^s$, such $F(x, y) = 1$. We know that y_1 and y_2 must be equal to 0. However, flipping the value of y_2 results in a satisfying assignment. Thus, it remains to analyze the following case:

$$\{l_i,\ l_j, \overline{y_1}, \overline{y_2}\} \subseteq C_1, \quad \{l_i,\ y_1,\ y_2\} \subseteq C_2, \quad \{l_j,\ y_1,\ y_2\} \subseteq C_3\,.$$

Assume that $\overline{l_j} \notin C_2$ and $\overline{l_i} \notin C_1$. Assign $l_i \leftarrow 1$, then assign $y_1 \leftarrow 0$ and $y_2 \leftarrow 0$. Under this assignment, $C_3 = \{l_j\}$ (recall that C_3 cannot contain other deterministic variables, see Case 3). This would mean that $l_j = 1$ in every satisfying assignment of the resulting CNF formula which cannot be the case for a CNF encoding of parity. Thus, we may assume that either $\overline{l_j} \in C_2$ or $\overline{l_i} \in C_1$. Without loss of generality, assume that $\overline{l_j} \in C_2$.

Let us show that for every $(x, y) \in \{0,1\}^{n+s}$, such that $F(x, y) = 1$ and $l_i = 1$, it holds that $l_j \neq y_1$ and $l_j \neq y_2$. Indeed, if there is $(x, y) \in \{0,1\}^{n+s}$ such that $F(x, y) = 1$ and $l_i = l_j = 1$, then y_1 and y_2 must be equal to 0. If there is $(x, y) \in \{0,1\}^{n+s}$, such that $F(x, y) = 1, l_i = 1, l_j = 0$, then y_1 and y_2 must be equal to 0, otherwise F does not contain a critical clause w.r.t.

(x, y, i). Thus, assigning $l_i \leftarrow 1$ eliminates two clauses (C_1 and C_2). We then replace y_1 and y_2 with $\overline{l_j}$ and delete the clause C_3.

(ii) $|Y| \geq 3, \{y_1, y_2, y_3\} \subseteq Y$:

$$\{l_i,\ l_j\} \subseteq C_1, \quad \{l_i,\ y_1,\ y_2\} \subseteq C_2, \quad \{l_j,\ y_1,\ y_3\} \subseteq C_3\,.$$

Assigning $l_i \leftarrow 1, l_j \leftarrow 1$ eliminates C_1, C_2, C_3. Assigning $y_1 \leftarrow 0$ eliminates at least one more clause (y_1 appears positively at least two times, but it may appear in C_1). There must be a clause with $\overline{y_2}$ (otherwise we could assign $y_2 \leftarrow 1$). Assigning $y_2 \leftarrow 0$ eliminates at least one more clause. Similarly, assigning $y_3 \leftarrow 1$ eliminates another clause. In total, we eliminate at least six clauses.

□

3.4 Lower Bounds for Majority

Theorem 13. *Let F be a CNF encoding of MAJ_n with m clauses and $s = O(n)$ nondeterministic variables. Then the parameters s and m cannot be too small simultaneously:*

$$m \geq \Omega\left(\frac{s+1+\log n}{\sqrt{n}} \cdot 2^{\frac{n}{2(s+1+\log n)}}\right)\,. \tag{19}$$

Proof. Consider two cases.

1. $s \leq n/2$. Let $S = \{x : \sum_{i=1}^{n} x_i = \lceil n/2 \rceil\}$. Note that $S \subseteq \mathrm{MAJ}_n^{-1}(1)$, and

$$|S| = \binom{n}{\lceil n/2 \rceil} \geq \frac{2^n}{\sqrt{n}}\,.$$

By Lemma 10, if

$$\varepsilon \leq \frac{n}{2}\ln 2 - s - 1, \tag{20}$$

then

$$m \geq (s+1+\varepsilon)2^{\frac{n/2}{s+1+\varepsilon}}\left(\frac{1}{\sqrt{n}} - 2^{-1-\varepsilon}\right)$$

Set $\varepsilon = \frac{1}{2}\log n$ (the inequalities 20 are satisfied, since $s \leq n/2$). Then,

$$\left(\frac{1}{\sqrt{n}} - 2^{-1-1/2\log n}\right) = \left(2^{-1/2\log n} - 2^{-1/2\log n}/2\right) = 2^{-1/2\log n}/2 = \frac{1}{2\sqrt{n}} = \Theta\left(\frac{1}{\sqrt{n}}\right)\,.$$

Hence,

$$m \geq \Omega\left(\frac{s+1+\log n}{\sqrt{n}} \cdot 2^{\frac{n}{2(s+1+\log n)}}\right)\,.$$

2. $n/2 < s = O(n)$. In this case, we need to show that $m \geq \Omega(\sqrt{n})$. Indeed, the number of clauses must be at least $\frac{n}{2}$, otherwise we would be able to satisfy a formula by assigning less than $\frac{n}{2}$ variables.

□

4 Appendix

Here, we prove Lemma 9. Let $F(x_1, \dots, x_n)$ be a CNF computing $f\colon \{0,1\}^n \to \{0,1\}$ and $x \in f^{-1}(1)$. For a permutation $\sigma \in S_n$, define an encoding $\Phi_\sigma\colon \{0,1\}^n \to \{0,1\}^{\le n}$ of x as follows. Permute the bits of x according to σ. For each $i \in [n]$, delete the i-th bit of the permuted string, if there is a critical clause $C_{F,x,\sigma(i)}$ such that the variable $\sigma(i)$ occurs after all other variables in this clause (according to the ordering σ).

Recall that an encoding function $\Phi\colon S \to \{0,1\}^*$ is called *prefix-free*, if $f(s_1)$ is not a prefix of $f(s_2)$ for any $s_1 \neq s_2 \in S$. In [19, Fact 1], it is proved that for a prefix-free encoding Φ with average code length $l = \sum_{s \in S} \Phi(s)/|S|$, it holds that $|S| \le 2^l$. It is also shown that Φ_σ is a prefix-free encoding.

Proof of Lemma 9. We show that there exists a permutation σ such that the average description length under the encoding Φ_σ of a d-isolated solution of weight at least μ is at most $n - \mu$.

Take a random permutation σ. Let x be a d-isolated solution of weight $w(x) \ge \mu$. Since the bit in x corresponding to a variable i is deleted with probability at least $1/|C_{(F,x,i)}|$ while constructing the encoding Φ_σ, the expected number of bits deleted is at least $\sum_{i \in I_{f,x}} 1/|C_{(F,x,i)}| \ge \mu$. Hence, there exists a permutation σ such that the average (over all isolated solutions of weight greater than or equal to μ) of the description length under the encoding Φ_σ is at most $n - \mu$. Thus, the number of isolated solutions of weight at least μ is at most $2^{n-\mu}$. □

References

[1] Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008. doi:10.1137/060664537.

[2] Olivier Bailleux and Yacine Boufkhad. Efficient cnf encoding of boolean cardinality constraints. In Francesca Rossi, editor, *Principles and Practice of Constraint Programming – CP 2003*, pages 108–122, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[3] Paul Maximilian Bittner, Thomas Thüm, and Ina Schaefer. SAT encodings of the at-most-k constraint - A case study on configuring university courses. In Peter Csaba Ölveczky and Gwen Salaün, editors, *Software Engineering and Formal Methods - 17th International Conference, SEFM 2019, Oslo, Norway, September 18-20, 2019, Proceedings*, volume 11724 of *Lecture Notes in Computer Science*, pages 127–144. Springer, 2019. doi:10.1007/978-3-030-30446-1_7.

[4] Bertrand Cabon, Simon de Givry, Lionel Lobjois, Thomas Schiex, and Joost P. Warners. Radio link frequency assignment. *Constraints An Int. J.*, 4(1):79–89, 1999. doi:10.1023/A:1009812409930.

[5] Evgeny Demenkov, Arist Kojevnikov, Alexander S. Kulikov, and Grigory Yaroslavtsev. New upper bounds on the boolean circuit complexity of symmetric functions. *Inf. Process. Lett.*, 110(7):264–267, 2010. doi:10.1016/j.ipl.2010.01.007.

[6] Alan M. Frisch and Paul A. Giannaros. SAT encodings of the at-most-k constraint: Some old, some new, some fast, some slow. In *Proceedings of the 9th International Workshop on Constraint Modelling and Reformulation*, 2010.

[7] Johan Håstad. Almost optimal lower bounds for small depth circuits. In Juris Hartmanis, editor, *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 6–20. ACM, 1986. doi:10.1145/12130.12132.

[8] Johan Håstad, Stasys Jukna, and Pavel Pudlák. Top-down lower bounds for depth 3 circuits. In *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*, pages 124–129. IEEE Computer Society, 1993. doi:10.1109/SFCS.1993.366875.

[9] Shuichi Hirahara. A duality between depth-three formulas and approximation by depth-two. *Electron. Colloquium Comput. Complex.*, page 92, 2017. URL: https://eccc.weizmann.ac.il/report/2017/092.

[10] Alexey Ignatiev, António Morgado, and Joao Marques-Silva. Pysat: A python toolkit for prototyping with sat oracles. In *International Conference on Theory and Applications of Satisfiability Testing*, 2018.

[11] Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-24508-4.

[12] Maria Klawe, Wolfgang J. Paul, Nicholas Pippenger, and Mihalis Yannakakis. On monotone formulae with restricted depth. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, STOC '84, page 480–487, New York, NY, USA, 1984. Association for Computing Machinery. doi:10.1145/800057.808717.

[13] Stepan Kochemazov, Oleg Zaikin, and Alexander Semenov. The comparison of different sat encodings for the problem of search for systems of orthogonal latin squares. In *International Conference Mathematical and Information Technologies-MIT*, pages 155–165, 2016.

[14] Petr Kucera, Petr Savický, and Vojtech Vorel. A lower bound on CNF encodings of the at-most-one constraint. *Theor. Comput. Sci.*, 762:51–73, 2019. doi:10.1016/j.tcs.2018.09.003.

[15] Wolfgang Kuechlin and Carsten Sinz. Proving consistency assertions for automotive product data management. *J. Autom. Reasoning*, 24:145–163, 02 2000. doi:10.1023/A:1006370506164.

[16] Jiatu Li and Tianqi Yang. $3.1n$ - $o(n)$ circuit lower bounds for explicit functions. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1180–1193. ACM, 2022. doi:10.1145/3519935.3519976.

[17] Joao Marques-Silva and Inês Lynce. Towards robust cnf encodings of cardinality constraints. In Christian Bessière, editor, *Principles and Practice of Constraint Programming – CP 2007*, pages 483–497, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[18] William J. Masek. Some NP-complete set covering problems. Unpublished Manuscript, 1979.

[19] Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. *Chic. J. Theor. Comput. Sci.*, 1999, 1999. URL: http://cjtcs.cs.uchicago.edu/articles/1999/11/contents.html.

[20] Steven David Prestwich. SAT problems with chains of dependent variables. *Discret. Appl. Math.*, 130(2):329–350, 2003. doi:10.1016/S0166-218X(02)00410-9.

[21] Steven David Prestwich. CNF encodings. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 75–97. IOS Press, 2009. doi:10.3233/978-1-58603-929-5-75.

[22] Carsten Sinz. Towards an optimal CNF encoding of boolean cardinality constraints. In Peter van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005, Proceedings*, volume 3709 of *Lecture Notes in Computer Science*, pages 827–831. Springer, 2005. doi:10.1007/11564751_73.

[23] G. S. Tsejtin. On the complexity of derivation in propositional calculus. Semin. Math., V. A. Steklov Math. Inst., Leningrad 8, 115-125 (1970); translation from Zap. Nauchn. Semin. Leningr. Otd. Mat. Inst. Steklova 8, 234-259 (1968)., 1968.

[24] Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *International Symposium on Mathematical Foundations of Computer Science*, 1977.

[25] Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science 1977, 6th Symposium, Tatranska Lomnica, Czechoslovakia, September 5-9, 1977, Proceedings*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 1977. doi:10.1007/3-540-08353-7_135.