Simple reductions to circuit minimization

Vishal Ramesh, Sasha Sami, and Noah Singer
Mentor: Eric Allender, DIMACS REU 2021

July 24, 2021

Outline

Boolean Circuit

- A *Boolean circuit* is composed of logic gates and wires, and computes a Boolean function $f : \{0,1\}^k \to \{0,1\}$.

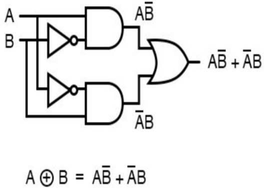

Figure: Circuit for XOR

- $C(S)$ denotes the size or complexity of a circuit S, and is usually defined to be the number of gates in S.
- The circuit depth is the length of the longest path from an input to an output gate.

Circuit complexity classes

Definition

- $\mathbf{AC}^0$ corresponds to the set of problems solvable by constant-depth, unbounded fan-in, polynomial-sized family of circuits with AND, OR, and NOT gates.
- $\mathbf{NC}^0$ is defined similarly to $\mathbf{AC}^0$, with the exception that the AND and OR gates have a fan-in of two, and thus each output gate depends on a constant number of input gates.
- **Projections** are functions computed by $\mathbf{NC}^0$ circuits, where each output bit is a constant $0/1$, or, same as or negation of an input bit.

Circuit complexity classes (contd.)

Uniformity

Circuits are non-uniform model of computation, inputs of different lengths are computed by different circuits. A family of circuits $\{C_n\}_{n\in\mathbb{N}}$ (where C_n is applicable for inputs of length n) is uniform if the description of C_n, can be generated in some resource bound manner, given n.

Example

A family of circuits is **DLOGTIME**-uniform, if description of C_n, can be generated in $\mathcal{O}(\log n)$ time, give n.

Reductions and Hardness

Many-one reduction

Given two languages L_1 and L_2, and a complexity class $\mathcal{C}$, L_1 is **many-one** reducible to L_2, $L_1 \leq_m^{\mathcal{C}} L_2$, if $\exists$ a $\mathcal{C}$-computable function f, such that $x \in L_1 \Leftrightarrow f(x) \in L_2$.

Example

$L_1 = \{\text{binary strings with odd number of } 1\}$
$L_2 = \{\text{binary strings with even number of } 1\}$
$L_1 \leq_m^{\mathbf{P}} L_2$.

Turing reduction

Given two languages L_1 and L_2, and a complexity class $\mathcal{C}$, L_1 is **Turing** reducible to L_2, $L_1 \leq_T^{\mathcal{C}} L_2$, if L_1 is $\mathcal{C}$-computable, given access to an oracle $\mathcal{O}$ for L_2.

Reductions and Hardness (contd.)

Adaptive vs Non-Adaptive Turing reduction

In a non-adaptive Turing reduction, a query asked to the oracle $\mathcal{O}$ does not depend on the result of a previously asked query (whereas in an adaptive reduction it does). A non-adaptive reduction can be thought of as presenting $\mathcal{O}$ with a single list of queries.

Definition

A language L is hard under reduction $\mathcal{R}$, for some complexity class $\mathcal{C}$, if all languages in $\mathcal{C}$ are reducible to L under $\mathcal{R}$.

Minimum Circuit Size Problem

MCSP

Let $T(S)$ denote the binary string of length $N = 2^n$, representing the truth table of the Boolean function computed by circuit S, with n input bits. Then for $x \in \{0,1\}^*, \theta \in \mathbb{N}$

$$\mathsf{MCSP} = \{(x, \theta) \mid \exists \text{ circuit } S \text{ s.t. } C(S) \leq \theta \text{ and } T(S) = x\}$$

$\mathbf{AC}^0_d$-MFSP (Minimum formula size problem)

$\mathbf{AC}^0_d$-MFSP is defined similarly to MCSP, except that S is an $\mathbf{AC}^0$ circuits of constant depth d. And $C(S)$ is measured as the number of leaf nodes in S.

Majority Problem

Definition

Majority (Maj) is the Boolean function that evaluates to false when half or more inputs are false and true otherwise.

Example

$\text{Maj}(110) = 1$ and $\text{Maj}(100) = 0$.

Known lower bound

$\text{Maj} \notin \mathbf{AC}^0$.

Coin Problem

Definition

(p, q)-*coin problem* is to distinguish a p-biased N-bit string from a q-biased N-bit string with high probability, where a p-biased N-bit string is sampled so that each bit is independently set to 1 with probability p.

Limitations on $\mathbf{NP}$-hardness for MCSP/MKTP

Results of [MW17]:

- MCSP/MKTP *unconditionally* cannot be hard for $\mathbf{NP}$ under *very simple* reductions
- If MCSP/MKTP are hard for $\mathbf{NP}$ under *any* deterministic polynomial-time many-one reductions, $\mathbf{EXP} \neq \mathbf{ZPP}$

Hardness of MKTP

- Recent results [AH17,ABM20,AGHR21]: MKTP is hard for $\mathbf{DET}$ and even $\mathbf{coNISZK_L}$ under non-uniform projections
- Results exploit properties of MKTP which are lacking in MCSP, specifically, bounds on hardness of tightest function

Reduction from MCSP to coin problem

- Result of [GII+19]: MCSP does not have small $\mathbf{AC}^0[p]$ circuits
  - Replicates result of [AH17] for MKTP, using different techniques
  - Exploits difference in circuit complexity of random biased functions
- Constructs reduction from coin problem to MCSP
- Combines with [SV10] reduction from Maj to coin problem

Our first result

- Crucial observation of [SV10]: Given $x \in \{0,1\}^N$, sampling an M-bit string of random bits of x is equiv. to sampling a $\mathrm{wt}(x)/N$-biased string
- We make assumption on monotonicity of expected complexity of biased functions, and build on [GII+19] and [SV10] to prove:

Theorem

(Assuming assumption above,) there exists a non-uniform projection from Maj *to* MCSP*.*

How important is adaptivity?

- $\mathbf{AC}^0_d$-MFSP is $\mathbf{NP}$-complete under quasipolynomial, randomized, *adaptive* reductions [Ila20]
- MCSP cannot be $\mathbf{ZPP}$-complete under polynomial-time, deterministic, *non-adaptive* reductions, unless $\mathbf{ZPP} = \mathbf{EXP}$ [Fu20]

(Slightly) improving [Fu20]

We show:

Theorem

If MCSP *is* $\mathbf{ZPP}$*-complete under quasipolynomial-time, deterministic, non-adaptive reductions, then* $\mathbf{ZPP} \neq \mathbf{EXP}$.

Same seems to hold for MFSP. We also give a slightly cleaner exposition than [Fu20].

Analyzing the reduction of [Ila20]

We give evidence that the reduction of [Ila20] can be implemented in $\mathbf{AC}^0$. [Ila20]'s reduction occurs in three stages...

- Reducing depth-d formula minimization to $O(1)$-approximating depth-d $\vee$-top formula minimization
- Reducing $O(1)$-approximating depth-d $\vee$-top formula minimization to $O(1)$-approximating depth-$(d-1)$ $\vee$-top formula minimization
- Invoking pre-existing hardness reductions for DNF minimization ($=$ depth-2 $\vee$-top formula minimization)

Acknowledgments

We would like to thank our adviser, Dr. Eric Allender.

This work was carried out while the authors were participants in the 2021 DIMACS REU program. V.R. and S.S. were supported by CoSP, a project funded by European Union's Horizon 2020 research and innovation programme, grant agreement No. 823748, while N.S. was supported by NSF grant CCF-1852215.