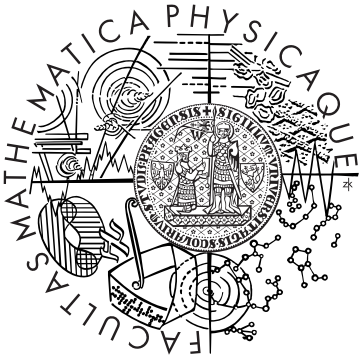
Charles University in Prague
Faculty of Mathematics and Physics

BACHELOR THESIS



Ján Pich

Bounded Arithmetic and Theory of Razborov and Rudich

Department of Algebra

Supervisor: Prof. RNDr. Jan Krajíček, DrSc.

Study Program: Mathematics

2009

Acknowledgements

I would like to thank my supervisor Jan Krajíček for careful guidance, patience and support. I would also like to thank my parents, for encouraging me through the whole studies.

I declare that I have written this bachelor thesis on my own and listed all used sources. I agree with lending of the thesis.

Prague, May 2009 Ján Pich

Contents

Title: Bounded Arithmetic and Theory of Razborov and Rudich
Author: Ján Pich
Department: Department of Algebra
Supervisor: Jan Krajíček
Supervisor's email address: krajicek@math.cas.cz

Abstract: We present a known result that the bounded arithmetic theory $S^2_2(\alpha)$ cannot separate complexity classes P from NP by proving circuit lower bounds unless hard pseudorandom generators do not exist. This was originaly proved by Razborov [17] but we present a simplyfied proof by Krajíček [12].

Keywords: Bounded Arithmetic, Independence, Natural Proofs

Chapter 1

Introduction

The P versus NP problem is considered as one of the most important open problems in contemporary mathematics. There were many unsuccessful attempts to solve it. Generally, if a problem seems to be unsolvable, one can try to prove that used techniques cannot work. In this thesis we present some known results of this kind concernig the P vs NP problem (also interpreted as reasons why the problem is difficult).

Our main goal is to prove that an arithmetical theory $S_2^2(\alpha)$ cannot separate P from NP by proving circuit lower bounds unless hard pseudorandom generators do not exist.

The unprovability result is originaly by Razborov [17] but we present simplyfied proof by Krajíček [12].

In the next chapter we investigate the circuit complexity of boolean functions. It turns out that lower bounds on circuit size can lead, in principle, to separation of complexity classes like P and NP. However, an important theorem of Razborov and Rudich, which we focus on, gives a serious limitation of proof techniques in circuit complexity. This is one of the key facts yielding the main unprovability result of the thesis.

The third chapter introduces the theory $S_2^2(\alpha)$ among other weak theories of arithmetic called collectively Bounded Arithmetic. These theories are in close relation to complexity theory and have many nice properties that will be used in the main proof.

In the fourth chapter we explain a translation of the bounded arithmetic theory into propositional logic that allows to prove and apply certain interpolantion theorems. This will be connected together with the rest of the thesis into the proof of unprovability of superpolynomial circuit lower bounds in the theory of bounded arithmetic $S_2^2(\alpha)$ under the mentioned cryptographical assumption.

Finally, in the last chapter we discuss possible improvements of the obtained result.

A reader having basic knowledge of Computational Complexity and Mathematical Logic should have no problem in reading this thesis.

Chapter 2

Natural Proofs

The notion of natural proofs refers to a certain kind of proof techniques in a theory of boolean circuits. There is a strong evidence that natural proofs cannot settle the P vs NP question.

Let us start with a description of a boolean circuit, a natural model for nonuniform computation[1].

Definition 2.1 [1]. *A* **boolean circuit** C *is a directed acyclic graph. The nodes of indegree* 0 *are called inputs, and are labeled with one of variables* $x_1, ..., x_n$ *or with a constant* 0 *or* 1*. The nodes of indegree* $k > 0$ *are called gates and are labeled with a boolean function on* k *inputs. We restrict to the boolean functions AND, OR (indegree* 2*) and NOT (indegree* 1*). One of the nodes is designated the output node. Denote the output by* $C(x_1, ..., x_n)$ *for inputs* $x_1, ..., x_n$*. The size* $S(C)$ *is the number of gates.*

A boolean circuit represents a boolean function of $x_1, ..., x_n$ in a natural way.

Definition 2.2. *A language* L *is in* $\mathbf{P/poly}$ *iff there is a family of circuits* $\{C_0, C_1, ...\}$ *where* C_n *has inputs* $x_1, ..., x_n$ *such that* $S(C_n) \leq n^{O(1)}$ *and for all* n *and all* $x = x_1, ..., x_n$*:* $x \in L \Leftrightarrow C_n(x_1, ..., x_n) = 1$*.*

It is known that $P \subseteq P/poly$, so one approach how to show $P \neq NP$ is to show a superpolynomial lower bound on circuit size for an NP-problem. Actually, it may also be that $NP \subseteq P/poly$ even though $P \neq NP$, but then the polynomial-time hierarchy collapses as Karp and Lipton [8] showed. This is generally not expected. Therefore trying to prove strong lower bounds for circuits seems to be a promissing approach to the P vs NP problem.

On the other hand, Razborov and Rudich show in their celebrated paper [18] that all known proofs of lower bounds for circuit size of boolean functions fit under a common concept called natural proofs. Moreover, as mentioned above, they prove that no such proof can ever settle the P vs NP question unless the strong pseudorandom generator conjecture fails. We explain this result in detail now.

[1] Nonuniform means that different "algorithm" is allowed for each input size.

Firstly, denote $F_n := \{f; f : \{0,1\}^n \to \{0,1\}\}$ and the set of all boolean functions $F = \bigcup_n F_n$. We write (f) for the truth table of $f \in F_n$ (hence its size is 2^n).

Definition 2.3. *A* **combinatorial property** *is a set $C \subseteq F$. Denote $C_n = C \cap F_n$.*

A combinatorial property C is

- **P/poly-natural** *iff the following two conditions holds:*
  Constructivity: $\{(f); f \in C\} \in P/poly$
  Largeness: $|C_n| \geq 2^{-O(n)}|F_n|$
- **useful against P/poly** *iff it satisfies*
  Usefulness: for any sequence $f_1, f_2, ..., f_n, ...$ where $f_n \in C_n$, and for any k there exists f_n such that the circuit size of f_n is greater then n^k.

Informally, a proof of a lower bound is called **P/poly-natural against P/poly** if it implies the existence of P/poly-natural combinatorial propety useful against P/poly.

The limitations of natural proofs in the $P \neq NP$ question arise from an assumption of the existence of a strong pseudorandom generator, roughly said, a function that passes any feasible test for randomness. Their existence is not proved at present but it is a general conjecture accepted by researchers in cryptography. However, it implies $P \neq NP$ and thus is expected to be hard.

Definition 2.4. *Let $G_k : \{0,1\}^k \to \{0,1\}^{2k}$ be a function. The hardness, $H(G_k)$ of G_k is the least S such that for some circuit C of size $\leq S$,*

$$|\mathbb{P}[C(G_k(\mathbf{x})) = 1] - \mathbb{P}[C(\mathbf{y}) = 1]| \geq \frac{1}{S}$$

where $\mathbf{x}$ is taken at random from $\{0,1\}^k$, and $\mathbf{y}$ is taken at random from $\{0,1\}^{2k}$ ($\mathbb{P}$ denotes a probability measure).

SPRG Conjecture. *There is a function G_n (pseudorandom generator), computed by polynomial size circuits, with hardness $H(G_n) \geq 2^{n^\epsilon}$, for some $\epsilon \geq 0$.*

Theorem 2.5 (Razborov, Rudich [18]). *There is no P/poly-natural proof against P/poly unless the SPRG conjecture fails.*

Proof: Suppose there is a combinatorial property C which is $P/poly$-natural and useful against $P/poly$. We will show that the hardeness of any pseudorandom generator $G_k : \{0,1\}^k \to \{0,1\}^{2k}$ in $P/poly$ is $H(G_k) \leq 2^{k^{o(1)}}$.

Let $\epsilon > 0$, $n = \lceil k^\epsilon \rceil$ and $G_0, G_1 : \{0,1\}^k \to \{0,1\}^k$ be the first and the last k bits of G_k, respectively.

Consider a binary tree T of height n. Arrange all its nodes $v_1, ..., v_{2^n-1}$ in such a way that if v_i is a son of v_j then $i < j$. There are also 2^n leaves in the last level corresponding to the values from $\{0,1\}^n$. Let T_i by the union of subtrees of T made by $\{v_1, ..., v_i\}$ along with all leaves. For a leaf y let $v_i(y)$ be the (highest) root of the subtree T_i containing y. $h(i,y)$ is the distance between y and $v_i(y)$.

For each assignment $x_v \in \{0,1\}^k$ of roots v of trees in T_i define function $f_{i,n}(y)$ as the first bit of $G_{y_n} \cdot ... \cdot G_{y_{n-h(i,y)+1}}(x_{v_i(y)})$. $\mathbf{f}_{i,n}$ denotes a random function from $f_{i,n}$, the set of all functions obtained as above by all choices x_v in T_i. $\mathbf{f}_{0,n}$ stands for a random function from F_n.

Every function from $f_{2^n-1,n}$ is in P/poly: Let $f \in f_{2^n-1,n}$. G_0, G_1 are in P/poly so there exists a circuit composed of n circuits polynomial in k, computing a function from f for all $y \in \{0,1\}^n$. The circuits are connected through small circuits computing the first or the last k bits depending on the sequence y.

C is useful against P/poly, hence for any sufficiently large k any $\mathbf{f}_{2^n-1,n}$ is not in C_n. Therefore the second condition of P/poly-natural property (Largeness) implies:

$$\begin{aligned} 2^{-O(n)} &\leq |\mathbb{P}[\mathbf{f}_{2^n-1,n} \in C_n] - \mathbb{P}[\mathbf{f}_{0,n} \in C_n]| \\ &= |\sum_{i=1}^{2^n-1} \mathbb{P}[\mathbf{f}_{i,n} \in C_n] - \mathbb{P}[\mathbf{f}_{i-1,n} \in C_n])| \\ &\leq \sum_{i=1}^{2^n-1} |\mathbb{P}[\mathbf{f}_{i,n} \in C_n] - \mathbb{P}[\mathbf{f}_{i-1,n} \in C_n]| \end{aligned}$$

hence there exists $i \in \{1, ..., 2^n - 1\}$ such that

$$|\mathbb{P}[\mathbf{f}_{i,n} \in C_n] - \mathbb{P}[\mathbf{f}_{i-1,n} \in C_n]| \geq 2^{-O(n)} \cdot 2^{-n} = 2^{-O(n)}$$

Functions from $f_{i,n}, f_{i-1,n}$ are determined by roots of trees in T_i and T_{i-1}, respecitvely. Denote fix_j the j-th possible assignment of common roots from T_i and T_{i-1}. Then,

$$\begin{aligned} 2^{-O(n)} &\leq |\mathbb{P}[\mathbf{f}_{i,n} \in C_n] - \mathbb{P}[\mathbf{f}_{i-1,n} \in C_n]| \\ &= |\sum_{\text{fix}_j} \mathbb{P}[\mathbf{f}_{i,n} \in C_n | \text{fix}_j] \cdot \mathbb{P}[\text{fix}_j] - \sum_{\text{fix}_j} \mathbb{P}[\mathbf{f}_{i-1,n} \in C_n | \text{fix}_j] \cdot \mathbb{P}[\text{fix}_j]| \end{aligned}$$

The probability $\mathbb{P}[\text{fix}_j]$ is the same nonzero number for each fix_j, hence

$$\frac{2^{-O(n)}}{\mathbb{P}[\text{fix}_j]} \leq \sum_{\text{fix}_j} |\mathbb{P}[\mathbf{f}_{i,n} \in C_n | \text{fix}_j] - \mathbb{P}[\mathbf{f}_{i-1,n} \in C_n | \text{fix}_j]|$$

Therefore there exists a fixed assignment of roots of trees in T_i except x_{v_i} such that

$$2^{-O(n)} \leq |\mathbb{P}[\mathbf{f}_{i,n} \in C_n | \text{fix}_j] - \mathbb{P}[\mathbf{f}_{i-1,n} \in C_n | \text{fix}_j]| \quad (*)$$

Now we construct a statistical test for strings $x \in \{0,1\}^{2k}$ distinguishing random and pseudorandom outputs.

Let $x \in \{0,1\}^{2k}$ represent the two sons of x_{v_i}(corresponding to the first and the last k bits). For each $y \in \{0,1\}^n$ obtain by a polynomial circuit(as in $f_{2^n-1,n}$) the value $G_{y_n} \cdot ... \cdot G_{y_n - h(i,y)+1}(z)$ where $z = x_{v_i}$ if $v_i(y) \neq v_i$ and z is the relevant son of x_{v_i} otherwise. Therefore we obtain values for each possible y by a circuit of size $2^{O(n)}$. The function obtained in this way belongs to $f_{i,n}$ for the fixation fix_j if x is output of $G(x_{v_i})$ and belongs to $f_{i-1,n}$ for the fixation fix_j if x is random.

According to the P/poly-natural combinatorial property there exists a circuit of size $2^{O(n)}$ recognizing whether $f_n \in F_n$ is in C_n. Therefore, it follows from $(*)$ that for each x we decide correctly with probability $\geq 2^{-O(n)}$ by a circuit of size for $2^{O(n)}$(composed by the circuit of previous paragraph and the circuit deciding belonging into C_n) whether it is output of generator.

Thus $H(G_k) \leq 2^{O(n)} \leq 2^{O(k^\epsilon)}$. The theorem is proved since ϵ was arbitrary.

■

Remark 2.6. If the effectivity condition is weakened to the requirement that C is computable in quasi-polynomial size, the proof works as well. Denote the modified form of the property **quasi-natural property against P/poly**.

Chapter 3

Bounded Arithmetic

3.1 Theories of Bounded Arithmetic

The theories of bounded arithmetic are theories for the natural numbers which are axiomatized by induction axioms restricted to the so called bounded formulas. The axiomatization can be done in various ways. We follow a formulation developed by Buss [2] which allows to express a precise relation between the quantifier complexity of a formula and the computational complexity of the relation it defines.

This is very brief introduction into these theories pointing out properties relevant for the main Theorem 4.3.1, mostly the definition of the investigated theory $S_2^2(\alpha)$. For more information about the extensive field of Bounded Arithmetic see [2] or [10].

Definition 3.1.1. *The first-order language L for theories of bounded arithmetic contains:*

- *non-logical predicate symbols:* $=, \leq$
- *function symbols:* $0, S$ *(successor)* $, +, \cdot, \lfloor \frac{x}{2} \rfloor, |x|, \#$

Remark 3.1.2 The intended value of $|x|$ is $\lceil \log_2(x+1) \rceil$. Therefore it will be the length of the binary representation of x. The intended value of $x\#y$ is $2^{|x|\cdot|y|}$. The function $\#$ will allow to express $2^{p(|x|)}$ for any polynomial p with positive integer coefficients.

The syntax of bounded arithmetic is also enlarged to contain bounded and sharply bounded quantifiers:

Definition 3.1.3 [3]. *A bounded quantifier is a quantifier of the form* $(Qx \leq t)$ *where t is a term not involving x and Q is* $\exists$ *or* $\forall$*. A sharply bounded quantifier is one of the form* $(Qx \leq |t|)$*.* $(\forall x)$ *and* $(\exists x)$ *are unbounded quantifiers. A bounded formula is one with no unbounded quantifiers.*

To express the mentioned relation to computational complexity we need to define a hierarchy of classes Σ_k^b, Π_k^b of bounded formulas.

The hierarchy is defined by counting alternations of bounded quantifiers, ignoring sharply bounded, analogously to definition of the arithmetic hierarchy, by counting alternations of unbounded quantifiers, ignoring bounded quantifiers:

Definition 3.1.4 (Buss [2]).
1. $\Sigma_0^b = \Pi_0^b$ *is the set of formulas with only sharply bounded quantifiers.*
2. *For* $i \geq 0$ *the classes* Σ_{i+1}^b *and* Π_{i+1}^b *are the smallest classes satisfying*
   a) $\Sigma_i^b \cup \Pi_i^b \subseteq \Sigma_{i+1}^b \cap \Pi_{i+1}^b$
   b) Σ_{i+1}^b *and* Π_{i+1}^b *are closed under sharply bounded quantification, disjunction* $\vee$*, and conjunction* $\wedge$
   c) Σ_{i+1}^b *is closed under bounded existential quantification*
   d) Π_{i+1}^b *is closed under bounded universal quantification*
   e) *the negation of a* Σ_{i+1}^b*-formula is* Π_{i+1}^b*, and the negation of a* Π_{i+1}^b*-formula is* Σ_{i+1}^b
3. *The class* Σ_∞^b *of bounded L-formulas is the union* $\bigcup_i \Sigma_i^b = \bigcup_i \Pi_i^b$.
4. *A* Σ_i^b*-formula is* Δ_i^b *in a theory T iff it is equivalent to a* Π_i^b*-formula in T.*

One of the main reasons for the previous definition is the following theorem, where Σ_k^p stands for the k-th class of predicates in the polynomial hierarchy. In particular, $\Sigma_1^p = NP$.

Theorem 3.1.5 [9,19,21]. *Let* $k \geq 1$. *A predicate Q is in* Σ_k^p *iff it is definable by a* Σ_k^b*-formula.*

Now we introduce a specific systems of Bounded Arithmetic formulated by Buss [2] that share some similar properties with the polynomial hierarchy.

A fundamental theory of bounded arithemtic is a set of open axioms called **BASIC**. It defines simple properties of the function and relation symbols. Note that **BASIC** also contains Robinson's arithmetic **Q**.

Definition 3.1.6. [10]. **BASIC** *is the following 32 axioms in the language L:*

1. $a \leq b \rightarrow a \leq b + 1$
2. $a \neq a + 1$
3. $0 \leq a$
4. $(a \leq b \wedge a \neq b) \rightarrow a + 1 \leq b$
5. $a \neq 0 \rightarrow 2a \neq 0$
6. $a \leq b \vee b \leq a$
7. $(a \leq b \wedge b \leq a) \rightarrow a = b$
8. $(a \leq b \wedge b \leq c) \rightarrow a \leq c$
9. $|0| = 0$
10. $a \neq 0 \rightarrow (|2a| = |a| + 1 \wedge |2a + 1| = |a| + 1)$
11. $|1| = 1$
12. $a \leq b \rightarrow |a| \leq |b|$
13. $|a\#b| = |a| \cdot |b| + 1$

14. $0\#a = 1$
15. $a \neq 0 \to (1\#(2a) = 2(1\#a) \wedge 1\#(2a+1) = 2(1\#a))$
16. $a\#b = b\#a$
17. $|a| = |b| \to a\#c = b\#c$
18. $|a| = |b| + |c| \to a\#d = (b\#d) \cdot (c\#d)$
19. $a \leq a + b$
20. $(a \leq b \wedge a \neq b) \to (2a+1 \leq 2b \wedge 2a+1 \neq 2b)$
21. $a + b = b + a$
22. $a + 0 = a$
23. $a + (b+1) = (a+b) + 1$
24. $(a+b) + c = a + (b+c)$
25. $a + b \leq a + c \to b \leq c$
26. $a \cdot 0 = 0$
27. $a \cdot (b+1) = a \cdot b + 1$
28. $a \cdot b = b \cdot a$
29. $a \cdot (b+c) = (a \cdot b) + (a \cdot c)$
30. $1 \leq a \to ((a \cdot b \leq a \cdot c) \equiv (b \leq c))$
31. $a \neq 0 \to |a| = |\lfloor (a/2) \rfloor| + 1$
32. $a = \lfloor (b/2) \rfloor \equiv (2a = b \vee 2a + 1 = b)$

Definition 3.1.7 [2]. *Let $i \geq 0$.*

S_2^i is the first-order theory with language L and axioms **BASIC** *and the polynomial induction axiom scheme Σ_i^b-***PIND***:*

$$A(0) \wedge \forall x(A(\lfloor \tfrac{x}{2} \rfloor) \to A(x)) \to \forall x A(x)$$

for all $A \in \Sigma_i^b$.

T_2^i is the first-order theory with language L and axioms **BASIC** *and the induction axiom scheme Σ_i^b-***IND***:*

$$A(0) \wedge \forall x(A(x) \to A(x+1)) \to \forall x A(x)$$

for all $A \in \Sigma_i^b$.

Recall several important theorems proved in [2] that demonstrate importance of these theories.

Theorem 3.1.8 [2]. *Every polynomial-time function is Σ_1^b-definable by S_2^1 and every polynomial-time predicate is Δ_1^b-definable by S_2^1*

Theorem 3.1.9 [2]. *Let $i \geq 1$. Let A be a Σ_i^b-formula. Suppose $S_2^i \vdash \forall x \exists y A(x,y)$. Then there is a Σ_i^b-formula B and function f computable in polynomial-time with an acces to a Σ_{i-1}^p-oracle and a term t so that*
1. $S_2^i \vdash \forall x, y(B(x,y) \to A(x,y))$
2. $S_2^i \vdash \forall x \exists! y B(x,y)$
3. $S_2^i \vdash \forall x \exists y \leq t B(x,y)$

4. for all n, $N \models B(n, f(n))$

Theorem 3.1.10 [2]. *Let* $i \geq 1$. $T_2^i \vdash S_2^i$ *and* $S_2^i \vdash T_2^{i-1}$.

It is an open question whether the inclusions $S_2^1 \subseteq T_2^1 \subseteq S_2^2 \subseteq T_2^2 ...$ are proper.

The theories S_2^i, T_2^i play central role in the field of bounded arithmetic. We will now consider their second-order extensions. The second-order language $L(\alpha)$ is an extension of L by adding symbols for countably many unary predicates $\alpha, \beta ....$ Definition of classes $\Sigma_i^b(\alpha), \Pi_i^b(\alpha)$ is straightforward generalization to the language $L(\alpha)$. Many other nice properties like characterization from Theorem 3.1.5 generalize too. For example $\Sigma_1^b(\alpha)$-formulas define exactly the subsets of natural numbers which are NP^α, i.e. NP with oracle $\{w; \alpha(w)\}$.

Definition 3.1.11.
$S_2^i(\alpha)$ *is the second-order theory with language* $L(\alpha)$ *and axioms* **BASIC** *and* $\Sigma_i^b(\alpha)$-**PIND**
$T_2^i(\alpha)$ *is the second-order theory with language* $L(\alpha)$ *and axioms* **BASIC** *and* $\Sigma_i^b(\alpha)$-**IND**.

There is a similar concept of a generalization in complexity classes, called relativization. Thanks to Yao [22] and Hastad [7] we know how to separate these relativized complexity classes of the polynomial hierarchy. A reader familiar with this result may find the following theorem composed of theorems from Krajíček [11] and Krajíček, Pudlák, Takeuti [15] natural. Actually, the result is used in its proof.

Theorem 3.1.12 [11,15]. *Let* $i \geq 1$. $S_2^i(\alpha) \subsetneq T_2^i(\alpha) \subsetneq S_2^{i+1}(\alpha)$

However, Buss [4] proved that $S_2^2(\alpha)$ is conservative over $T_2^1(\alpha)$ with respect to a special subset of $L(\alpha)$-formulas. We use this fact in the main proof. Let us describe it precisely.

Definition 3.1.13. *A bounded* $L(\alpha)$*-formula is* $E_1(\alpha, \Sigma_\infty^b)$ *iff it has the form* $\exists^{\leq} \bigvee_i \bigwedge_j A_{i,j}$*, where each* $A_{i,j}$ *is an atomic formula or negation of an atomic formula or a* Σ_∞^b*-formula. Note that the* Σ_∞^b*-formulas do not contain predicates* $\alpha, \beta ....$ $U_1(\alpha, \Sigma_\infty^b)$*-formulas are defined dually, replacing* $\exists^{\leq}$ *by* $\forall^{\leq}$ *and a disjunction of conjunctions by a conjunction of disjunctions.*

Theorem 3.1.14 [4]. $S_2^2(\alpha)$ *is* $U_1(\alpha, \Sigma_\infty^b)$*-conservative over* $T_2^1(\alpha)$.

3.2 The Sequent Calculus

We specify the syntax and rules of deduction, the so called sequent calculus, for the theories introduced in the previous section. This is necessary for any formal reasoning about the theories.

We use the following formulation of the *sequent calculus LK* from Krajíček [12].

The propositional language contains logical symbols: $0, 1, \neg, \bigvee, \bigwedge$. The negation is allowed only in front of atoms. The conjuction and the disjunction are of unbounded arity. $\neg A$ denotes the formula obtained from the formula A by interchanging 0 and 1, $\bigvee$ and $\bigwedge$, p_i and $\neg p_i$.

The size $|A|$ of A is the number of occurences of connectives and atoms in it. The *depth* $dp(A)$ of A is the maximal nesting of $\bigvee$ and $\bigwedge$ in A:

$$dp(0) = dp(1) = dp(p_i) = dp(\neg p_i) = 0$$
$$dp(\textstyle\bigwedge_i A_i) = dp(\bigvee_i A_i) = 1 + max_i(dp(A_i))$$

A *cedent* is a finite (possibly empty) sequence of formulas denoted $\Gamma, \Delta, ...$ A *sequent* is an ordered pair of cedents written $\Gamma \to \Delta$.

A sequent is satisfied if at least one formula in Δ is satisfied or at least one formula in Γ is falsified. For example, the empty sequent cannot be satisfied.

The inference rules are the following:

1. The *initial sequents*:

$$\begin{array}{cccc} \to 1 & \neg 1 \to & 0 \to & \to \neg 0 \\ p \to p & \neg p \to \neg p & p, \neg p \to & \to p, \neg p \end{array}$$

2. The *weak structural rules*:

$$\frac{\Gamma \to \Delta}{\Gamma' \to \Delta'}$$

   where Γ', Δ' are any permutations of Γ, Δ (the exchange), or Γ', Δ' are obtained from Γ, Δ by deleting any multiple occurences of formulas (the contraction), or $\Gamma \subseteq \Gamma'$ and $\Delta \subseteq \Delta'$ (the weakening).

3. The *propositional rules*:
   $\bigwedge$: introduction

$$\frac{A, \Gamma \to \Delta}{\bigwedge A_i, \Gamma \to \Delta} \qquad \frac{\Gamma_0 \to \Delta_0, A_1 \quad \dots \quad \Gamma_m \to \Delta_m, A_m}{\Gamma_0, ..., \Gamma_m, \to \Delta_0, ..., \Delta_m, \bigwedge_{i \leq m} A_i}$$

where A is one of A_i in the left rule.

$\bigvee$: introduction

$$\frac{\Gamma \to \Delta, A}{\Gamma \to \Delta, \bigvee A_i} \qquad \frac{A_1, \Gamma_0 \to \Delta_0 \quad \dots \quad A_m, \Gamma_m \to \Delta_m}{\bigvee_{i \le m} A_i, \Gamma_0, ..., \Gamma_m \to \Delta_0, ..., \Delta_m}$$

where A is one of A_i in the left rule.

4. The *cut rule*:

$$\frac{\Gamma \to \Delta, A \qquad A, \Pi \to \Lambda}{\Gamma, \Pi \to \Delta, \Lambda} \text{ cut}$$

An *LK-proof* of a sequent S from the sequents $S_1, ..., S_m$ is a sequence of sequents s.t. each element is either an initial sequent or from $\{S_1, ..., S_m\}$ or derived from the previous sequents by an inference rule.

A proof π is *tree-like* if every sequent in π is a hypothesis of at most one inference. $k(\pi)$ denotes the number of sequents in π.

Remark 3.2.1 It is well-known that the LK calculus is complete even without the cut inference. See Takeuti [20].

We now formalize the theories of bounded arithmetic as sequent calculus systems by enlarging the LK calculus in the following way taken in [3]. However, we do it simultaneously for first-order and second-order theories:

(1) Add equality, logical and BASIC axioms as initial sequents.

A logical axiom is a sequent of the form:

- $A \to A$ where A is an atomic formula.

An equality axiom is a sequent of the form:

- $\to t_1 = t_1$
- $t_1 = s_1, ..., t_n = s_n \to f(t_1, ..., t_n) = f(s_1, ..., s_n)$
- $t_1 = s_1, ..., t_n = s_n, p(t_1, ..., t_n) \to p(s_1, ..., s_n)$

  where the t_i's and s_i's are arbitrary terms and f or p is any n-ary function or predicate symbol, respectively.

(2) Add inferences for quantifiers (the first-order variable a occurs only as indicated, s and t are arbitrary terms):

$$\frac{A(a), \Gamma \to \Delta}{\exists x A(x), \Gamma \to \Delta} \qquad \frac{\Gamma \to \Delta, A(t)}{\Gamma \to \Delta, \exists x A(x)}$$

$$\frac{A(t), \Gamma \to \Delta}{\forall x A(x), \Gamma \to \Delta} \qquad \frac{\Gamma \to \Delta, A(a)}{\Gamma \to \Delta, \forall x A(x)}$$

$$\frac{a \leq s, A(a), \Gamma \to \Delta}{\exists x \leq sA(x), \Gamma \to \Delta} \qquad \frac{\Gamma \to \Delta, A(t)}{t \leq s, \Gamma \to \Delta, \exists x \leq sA(x)}$$

$$\frac{A(t), \Gamma \to \Delta}{t \leq s, \forall x \leq sA(x), \Gamma \to \Delta} \qquad \frac{a \leq s, \Gamma \to \Delta, A(a)}{\Gamma \to \Delta, \forall x \leq sA(x)}$$

(3) Add relevant induction inferences (for all A from a set of formulas Λ):

$$\Lambda\text{-PIND} \quad \frac{A(\lfloor \frac{x}{2} \rfloor), \Gamma \to \Delta, A(x)}{A(0), \Gamma \to \Delta, A(t)}$$

$$\Lambda\text{-IND} \quad \frac{A(x), \Gamma \to \Delta, A(x+1)}{A(0), \Gamma \to \Delta, A(t)}$$

where x is always a first-order variable.

The induction inferences are equivalent to the induction axioms (in LK). Moreover, $S_2^i, T_2^i, S_2^i(\alpha)$ and $T_2^i(\alpha)$ are equivalently formulated as sequent calculus systems with BASIC axioms as initial sequents and with Σ_i^b-PIND, Σ_i^b-IND, $\Sigma_i^b(\alpha)$-PIND and $\Sigma_i^b(\alpha)$-IND inference rules, respectively. We use these later formulations.

However, note that to let the formulation above work we must express all axioms with respect to our propositional language. For example, negation is allowed only in front of atomic formulas, and there are no connectives like implication. Of course, this does not affect any common property of these theories.

Now we should also mention a nice property of $T_2^i(\alpha)$-proofs which we use in the main proof. The next theorem holds for other theories as well.

Theorem 3.2.2 [10]. *Let $i \geq 1$. Assume that the sequent $\Gamma \to \Delta$ is provable in $T_2^i(\alpha)$. Then there is a proof of $\Gamma \to \Delta$ in the same theory in which every formula is either a subformula of a $\Sigma_i^b(\alpha)$- or $\Pi_i^b(\alpha)$-formula or a subformula of a formula from $\Gamma \to \Delta$.*

Chapter 4

Independence result

4.1 Translation from Bounded Arithmetic into Propositional Logic

The main idea of the independence result presented in the thesis is to use a certain interpolation property of propositional logic to obtain a quasi-natural property from each proof of superpolynomial circuit lower bound in the theory $S^2_2(\alpha)$. Therefore, roughly said, all methods provided by the theory $S^2_2(\alpha)$ quasi-naturalize and thus cannot succeed unless SPRG conjecture fails.

To show this we need to translate bounded formulas into propositional formulas in a way that allow to prove and apply interpolation theorems in propositional logic. This section is dedicated to this procedure. We follow Krajíček [12].

Definition 4.1.1. *For a bounded formula $A(a, \alpha_1, ..., \alpha_k)$ with the predicate parameters α_i and the number parameter a and for any value $a = N$, denote by $\langle A \rangle_N(p^1, ..., p^k)$, a constant-depth, size $2^{(\log N)^{O(1)}}$ propositional formula obtained in the following way:*

- *the atomic sentence $j \in \alpha_i$ translates into the atom p^i_j*
- *a true (resp. false) first-order atomic sentence translates into 1 (resp.0)*
- *a bounded universal (resp. existential) quantifier $\forall x < tB(x)$ (resp. $\exists x < tB(x)$) translates into a conjunction (resp. disjunction) of the translations of $B(x), x = 0, ..., t - 1$.*

Remark 4.1.2 Observe that each $\Sigma^b_i(\alpha)$-formula translates into a formula which is equivalent (in predicate calculus) to a formula of depth at most $i+1$.

The crucial property of the previous translation is the following theorem.

Theorem 4.1.3 [12]. *Assume that $\forall x \leq s(a)A(a, x, \alpha_1, ..., \alpha_k)$ is a bounded $U(\alpha_1, ..., \alpha_k, \Sigma^b_\infty)$-formula and that $\exists x \leq t(a)B(a, y, \alpha_1, ..., \alpha_k)$ is a bounded $E(\alpha_1, ..., \alpha_k, \Sigma^b_\infty)$-formula.*

If the theory $S^2_2(\alpha)$ proves the sequent

$$\forall x \leq s(a) A(a, x, \alpha_1, ..., \alpha_k) \rightarrow \exists y \leq t(a) B(a, y, \alpha_1, ..., \alpha_k)$$

Then there exists a constant c such that for every N the propositional sequent

$$\langle A \rangle_{N,0}, ..., \langle A \rangle_{N,s(N)} \rightarrow \langle B \rangle_{N,0}, ..., \langle B \rangle_{N,t(N)}$$

has an LK-proof π_N satisfying the following conditions:
1. π_N is tree-like
2. $k(\pi_N) = 2^{(\log N)^{O(1)}}$
3. every formula in π_N has depth at most 2
4. every sequent in π_N contains at most c depth 2 formulas

Proof: There is a $T^1_2(\alpha)$-proof of the same sequent due to Theorem 3.1.14. By Theorem 3.2.2 we can also assume that the proof contains only $\Sigma^b_1(\alpha)$-formulas or subformulas of $\forall\ x \leq s(a)\quad A(a, x, \alpha_1, ..., \alpha_k)$ or $\exists y \leq t(a) B(a, y, \alpha_1, ..., \alpha_k)$. Expand this proof so that it is tree-like, by adding redundant inferences if necessary. Denote the obtained proof π.

Now translate π into propositional proofs π_N satisfying $1.-4.$ by induction on the number of inferences in π:

Initial sequents of π translate into initial sequents of propositional logic or

$$\begin{array}{cc} 1, ..., 1 \rightarrow 1 & ..., 0, ... \rightarrow 0 \\ p^i_j \rightarrow p^i_j & 1, p^i_j \rightarrow p^i_j \\ 0, p^i_j \rightarrow p^i_s & \end{array}$$

All of them have constant size tree-like proofs in propositional logic satisfying other conditions too.

Induction step: if the last inference in π is

$$\frac{A(x), \Gamma \rightarrow \Delta, A(x+1)}{A(0), \Gamma \rightarrow \Delta, A(t)}$$

then by induction hypothesis there are relevant proofs of
$\langle A(i) \rangle_N, \Gamma^T \rightarrow \Delta^T, \langle A(i+1) \rangle_N$ for all i where Γ^T denotes Γ after translations of all of its formulas into propositional logic. We connect them to a relevant proof with last inferences:

$$\frac{\left\{ \langle A(i) \rangle_N, \Gamma^T \rightarrow \Delta^T, \langle A(i+1) \rangle_N \right\}^{t-1}_{i=0}}{\langle A(0) \rangle_N, \Gamma^T \rightarrow \Delta^T, \langle A(t) \rangle_N} \text{ cut rules}$$

Other inferences are managed similarly, for example:

$$\frac{a \leq s, \Gamma \to \Delta, A(a)}{\Gamma \to \Delta, \forall x \leq s A(x)} \quad \longmapsto \quad \frac{\left\{1, \Gamma^T \to \Delta^T, \langle A(i) \rangle_N \right\}_{i=0}^{s}}{\Gamma^T \to \Delta^T, \bigwedge_{i=0}^{s} \langle A(i) \rangle_N}$$

$$\frac{\Gamma \to \Delta, A(t)}{t \leq s, \Gamma \to \Delta, \exists x \leq s A(x)} \quad \longmapsto \quad \frac{\Gamma^T \to \Delta^T, \langle A(t) \rangle_N}{1, \Gamma^T \to \Delta^T, \bigvee_{i=0}^{s} \langle A(i) \rangle_N}$$

Since the proof π has constant size, and since the values of all terms are $2^{(\log N)^{O(1)}}$, the size bound holds as well as the other conditions.

■

4.2 The Craig Interpolation Theorem

As we mentioned, the method of interpolation play crucial role in the main independence result, Theorem 4.3.1. It provides a property which can separate disjoint classes of boolean functions. We will specify it in the proof of Theorem 4.3.1. Now we discuss how to obtain such an effective interpolant.

Definition 4.2.1. *An interpolant of a valid implication $A(p,q) \to B(p,r)$, where $p = (p_1, ..., p_n)$ are the atoms occuring in both A and B, while $q = (q_1, ..., q_s)$ occur only in A and $r = (r_1, ..., r_t)$ only in B, is a formula $I(p)$ such that the both implications*

$$A(p,q) \to I(p) \text{ and } I(p) \to B(p,r)$$

are tautologies.

Theorem 4.2.2 (Craig [5,6]). *Let π be a cut-free LK-proof of the sequent:*

$$A_1(p,q), ..., A_m(p,q) \to B_1(p,r), ..., B_l(p,r)$$

with $p = (p_1, ..., p_n)$ the atoms occurring simultaneously in some A_i and B_j and $q = (q_1, ..., q_s)$ and $r = (r_1, ..., r_t)$ all other atoms occurring only in some A_i or in some B_j respectively.

Then there is an interpolant $I(p)$ of the implication

$$\bigwedge_{i \le m} A_i \to \bigvee_{j \le l} B_j$$

whose circuit-size is at most $(k(\pi))^{O(1)}$.

Recall that the sequent $A_1, ..., A_l \to B_1, ..., B_m$ represents the implication $\bigwedge_i A_i \to \bigvee_j B_j$. Also, the circuit-size of a formula stands for the circuit-size of a circuit representing the boolean function expressed by the formula.

The original statement and proof of the Craig interpolation theorem is simplier than the presented one. This is because we need to consider circuit-size of the interpolant there.

Proof (Krajíček [12]):

Let S be a sequent in the cut-free proof π. We define the interpolant $I^S(p)$ for S by induction:

If S is initial sequent then $I^S(p)$ is one of $0, 1, p_i, \neg p_i$. Note that the only initial sequents in π where q_i or r_i occur are $q_i, \neg q_i \to$ and $\to r_i, \neg r_i$, which have interpolants 0 and 1, respectively.

If S is derived from one hypothesis S_1 put $I^S(p) = I^{S_1}(p)$.

If S is derived from the $S_1, ..., S_l$ by the right $\bigwedge$:introduction (resp. by the left $\bigvee$:introduction) then put $I^S(p) = \bigwedge_{i \leq l} I^{S_i}(p)$ (resp. $I^S(p) = \bigvee_{i \leq l} I^{S_i}(p)$).

Finally, observe that the circuit-size of the last interpolant is $(k(\pi))^{O(1)}$, because of our restriction to the circuits with gates of fanin at most 2.

■

Theorem 4.2.3. (Krajíček [12]). *Let π be an LK-proof of the sequent:*

$$A_1(p,q), ..., A_m(p,q) \rightarrow B_1(p,r), ..., B_l(p,r)$$

with atoms $p = (p_1, ..., p_n), q = (q_1, ..., q_s), r = (r_1, ..., r_t)$ occurring as displayed and such that the formulas A_i (resp. B_j) are literals or disjunctions (resp. conjunctions) of literals.

Assume that π satisfies:

1. *π is tree-like*
2. *every formula in π has the depth at most 2*
3. *every sequent in π contains at most c depth 2 formulas, where c is an independent constant.*

Then there is an interpolant $I(p)$ of the implication

$$\bigwedge_{i \leq m} A_i \rightarrow \bigvee_{j \leq l} B_j$$

whose circuit-size is at most $k(\pi)^{O(1)}$.

Proof:

The given proof π can be transformed into a proof π' with $k(\pi') = k(\pi)^{O(1)}$ in which every formula is of depth at most 1.

The transformation of individual sequents goes in the following way (where depth 2 formulas are as indicated):

$$\begin{array}{lcl}
\Gamma \rightarrow \Delta, \bigwedge_{i=1}^k A_i & \longmapsto & \Gamma \rightarrow, A_i, \text{ for all } i \leq k \\
\Gamma \rightarrow \Delta, \bigvee_{i=1}^k A_i & \longmapsto & \Gamma \rightarrow, A_1, ..., A_k \\
\bigvee_{i=1}^k A_i, \Gamma \rightarrow \Delta & \longmapsto & A_i, \Gamma \rightarrow \Delta, \text{ for all } i \leq k \\
\bigwedge_{i=1}^k A_i, \Gamma \rightarrow \Delta & \longmapsto & A_1, ..., A_k, \Gamma \rightarrow \Delta
\end{array}$$

We build a new proof from the obtained sequents in a straightforward way. For example, the cut rule in the original proof:

$$\frac{\Gamma \rightarrow \Delta, \bigwedge_{j=1}^k A_j \qquad \bigwedge_{j=1}^k A_j, \Pi \rightarrow \Lambda}{\Gamma, \Pi \rightarrow \Delta, \Lambda}$$

can be replaced by k cuts:

$$\dfrac{\Gamma' \to \Delta', A_1 \qquad \dfrac{A_1, ..., A_k, \Pi' \to \Lambda' \qquad \dfrac{A_2, ..., A_k, \Gamma', \Pi' \to \Delta', \Lambda' \qquad \dfrac{A_3, ..., A_k, \Gamma', \Pi' \to \Delta', \Lambda'}{\vdots}}{\Gamma', \Pi' \to \Delta' \Lambda'}}{\Gamma' \to \Delta', A_2}}{\Gamma' \to \Delta', A_3}$$

Other inferences are managed analogously. The only reason why the new proof is not tree-like is the weakening rule introducing a formula of depth 2 which we might transform into inferences that use one sequent repetitively. However, for each cut rule the subproofs of input sequents are disjoint. Define an *almost tree-like* proof as a proof such that for each cut rule with a cut formula of depth 1 the subproofs of input sequents are disjoint. So our proof is almost tree-like. Observe also that without a loss of generality the number of conjuncts (resp. disjuncts) in each formula of depth 2 is less than $k(\pi)$: if such a formula is not created by any introduction rule from previous formulas than it can be replaced by a formula with less conjuncts (resp. disjuncts). This concludes the transformation into π' with right size bound.

Now we transform the almost tree-like proof π' into a proof π'', again not necessarily tree-like, in which every cut formula is literal.

Firstly, we describe a procedure of elimination of one cut rule with cut formula of depth 1.

Consider a cut rule as above, but this time A_j's replace by literals l_j's:

$$\dfrac{\Gamma \to \Delta, \bigwedge_{j=1}^{k} l_j \qquad \bigwedge_{j=1}^{k} l_j, \Pi \to \Lambda}{\Gamma, \Pi \to \Delta, \Lambda}$$

The cut formula $\bigwedge_{j=1}^{k} l_j$ could be created in the following ways

$$\text{1. weakening } \dfrac{\Gamma' \to \Delta'}{\Gamma' \to \Delta', \bigwedge_{j=1}^{k} l_j}$$

$$\text{2. introduction } \dfrac{\Gamma'_0 \to \Delta'_0, l_1 \qquad \dots \qquad \Gamma'_0 \to \Delta'_k, l_k}{\Gamma'_0, ..., \Gamma'_k \to \Delta'_0, ..., \Delta'_k, \bigwedge_{j=1}^{k} l_j}$$

or in more steps by one weakening rule followed (not necessarily immediately) by introduction rules:

$$\text{1'. weakening } \dfrac{\dfrac{\Pi' \to \Lambda'}{\bigwedge_{j \in J} l_j, \Pi'' \to \Lambda''}}{\bigwedge_{j=1}^{k} l_j, \Pi''' \to \Lambda'''}$$

where $J \subseteq [k] = \{1, ..., k\}$

or in more steps merely by applying introduction rules:

$$\text{2'. introduction } \dfrac{\dfrac{l_{j_i}, \Pi' \to \Lambda'}{\bigwedge_{j \in J} l_j, \Pi'' \to \Lambda''}}{\bigwedge_{j=1}^{k} l_j, \Pi''' \to \Lambda'''}$$

where $J \subseteq [k] = \{1, ..., k\}$ and $j_i \in [k]$.

The procedure of elimination:

Firstly, we get a correct proof of $l_1, ..., l_k, \Pi \to \Lambda$ of the same size as the original proof of $\bigwedge_{j=1}^{k} l_j, \Pi \to \Lambda$ replacing all inferences of type $1'$. and $2'$. in the proof of $\bigwedge_{j=1}^{k} l_j, \Pi \to \Lambda$ by weakening as follows (showed for the case 1'.):

$$\dfrac{\dfrac{\Pi' \to \Lambda'}{l_{j_1}, ..., l_{j_d}, \Pi'' \to \Lambda''}}{l_1, ..., l_k, \Pi''' \to \Lambda'''}$$

where $d = |J|$ and $j_1 < ... < j_d$ are from J.

Then, for each inference of type 2. we use $k \leq k(\pi')$ cuts on the sequent $l_1, ..., l_k, \Pi \to \Lambda$ to obtain the sequent

$$\Gamma'_0, ..., \Gamma'_k, \Pi \to \Lambda, \Delta'_0, ..., \Delta'_m$$

and continue straightforwardly the original proof of $\Gamma \to \Delta, \bigwedge_{j=1}^{k} l_j$ (the original proof is almost tree-like, so we do not need sequents replaced in the first step of the procedure). Therefore, because of the side cedents, we get a proof of $\Gamma, \Pi \to \Delta, \Lambda$ with the size of proof increased by at most $k(\pi')^2$ sequents.

However, if there is no inference of type 2., we can eliminate the cut rule at all by ignoring weakening that introduce $\bigwedge l_i$ and using weakening in place of cut rule to obtain $\Gamma, \Pi \to \Delta, \Lambda$.

The elimination of cut formula $\bigvee_{j=1}^{k} l_j$ is similar. This concludes the elimination procedure of one cut inference.

Applying the previous procedure in a way that preserve almost tree-likeness we obtain a proof π'' that has cut formulas of depth at most 0 and $k(\pi'') = k(\pi)^{O(1)}$.

Note that the number of added sequents in each cut elimination step depends only on the number of introductions of type 2 in the original proof π' and the number of conjuncts (resp. disjuncts) in cut formula. If we had not almost tree-like proof we could not replace rules 1'. and 2'. but just add the new inferences what would increase the size of the final proof π'' exponentionaly.

In the proof π'' replace cut inference with the cut formula $p_i, \neg p_i, \neg q_i$ or q_i by introduction of $p_i \vee \neg p_i$ or $q_i \vee \neg q_i$ into the antecedent, and the cut inference with the cut formula r_i or $\neg r_i$ by introduction $r_i \wedge \neg r_i$ to the succedent. For example, to introduce $p_i \vee \neg p_i$ replace the proof of say $\Gamma \to \Delta, p_i$ by a proof of $\neg p_i, \Gamma \to \Delta$ (of the same size) and then obtain $p_i \vee \neg p_i, \Gamma, \Pi \to \Delta, \Lambda$ by weakening and introduction rule.

Denote Γ_a (antecedent) and Δ_a (succedent), the cedents created by these new formulas, where a refers to the a-th sequent in the proof π''.

The new proof of the implication:

$$\bigwedge \Gamma_{k(\pi'')} \wedge \bigwedge_{i \leq m} A_i \to \bigvee_{j \leq l} B_j \vee \bigvee \Delta_{k(\pi'')}$$

is cut-free, hence by Theorem 4.2.2 it has an interpolant $I(p)$ of circuit-size $k(\pi)^{O(1)}$. This $I(p)$ is also an interpolant for the implication:

$$\bigwedge_{i \leq m} A_i \to \bigvee_{j \leq l} B_j$$

because $\bigwedge \Gamma_{k(\pi'')}$ is a tautology and $\bigvee \Delta_{k(\pi'')}$ is unsatisfiable.

■

4.3 An Independence Result for Bounded Arithmetic Theory $S_2^2(\alpha)$

Finally, we almost get into the proof of the main independence result. All we need is to formalize the statement.

Let $N = 2^n$. Any $f \subseteq \{1, ..., N\}$ is thought of as truth table of a boolean function $f : \{0,1\}^n \to \{0,1\}$. A first-order bounded formula $E(a, x)$ determines for every N the function $E_N = \{1 \leq i \leq N | E(i, N)\}$.

The following formalization comes from Razborov [16]. Boolean circuits are expressed as second-order objects while their inputs as first-order objects.

$Circuit(\alpha, N, t(N))$ means that α codes a circuit with $\vee, \wedge$ fan-in two of size $t(N)$ in n inputs:

$$
\begin{aligned}
&Circuit(\alpha, N, t(N)) \rightleftharpoons \\
&\quad \forall u < t(N) \\
&\qquad \big[\exists i < |N| \, \forall x < N \quad (\alpha(\langle u, x\rangle) \Leftrightarrow Bit(i, x)) \\
&\qquad\ \vee \exists u_1, u_2 < u \big(\quad \forall x < N (\alpha(\langle u, x\rangle) \Leftrightarrow \alpha(\langle u_1, x\rangle) \wedge \alpha(\langle u_2, x\rangle)) \\
&\qquad\qquad\qquad \vee \forall x < N (\alpha(\langle u, x\rangle) \Leftrightarrow \alpha(\langle u_1, x\rangle) \vee \alpha(\langle u_2, x\rangle)) \\
&\qquad\qquad\qquad \vee \forall x < N (\alpha(\langle u, x\rangle) \Leftrightarrow \neg\alpha(\langle u_1, x\rangle))\big)\big]
\end{aligned}
$$

where $\langle x, y\rangle$ is a pairing function $\frac{(x+y)(x+y+1)}{2} + x$ and $Bit(i, x)$ is a relation which holds iff the i-th bit in the binary notation for x is 1. It is known to be Δ_1^b-definable in S_2^1.

$Comp(\alpha, N, t(N), f)$ asserts that α codes a circuit as above and that the circuit computes the function $f \subset \{1, ..., N\}$ expressed by a bounded formula $E(x, N)$:

$$
\begin{aligned}
&Comp(\alpha, N, t(N), f) \rightleftharpoons \\
&\quad Circuit(\alpha, N, t(N)) \wedge \forall x < N (\alpha(\langle t(N), x\rangle) \Leftrightarrow E(x, N))
\end{aligned}
$$

Observe that it can be expressed as a $U_1(\alpha, \Sigma_\infty^b)$-formula.

Finally, the formalization of the lower bound $t(N)$ to the circuit-size of f is the formula:

$$LB(N, t(N), f) \rightleftharpoons \forall \alpha \neg Comp(\alpha, N, t(N), f)$$

Theorem 4.3.1 (Razborov [17]). *Let $E(x,a)$ be a first-order bounded formula, $t(N)$ a function such that $t(N) = (\log N)^{\omega(1)}$ holds in the natural numbers. Then, assuming the SPRG conjecture, it is true that*

$$S_2^2(\alpha) \nvdash \forall a\, LB(a, t(a), E_a)$$

Proof (Krajíček [12]):

Let $s(N)$ be any function such that $s(N) = (\log N)^{\omega(1)}$ and the parity of two circuits of size $s(N)$ is computable in size $\leq t(N)$. The parity of f and E_N is defined by the bounded formula: $f \oplus E_N(x) \rightleftharpoons (f(x) \Leftrightarrow \neg E_N(x))$.

Assume that $LB(N, t(N), E_N)$ holds. Now, if α codes a circuit of size $s(N)$ computing f, then β does not code a circuit of size $s(N)$ computing $f \oplus E_N$. Otherwise, the two circuits joined by $\oplus$ compute E_N contradicting the assumtion. It means that the following holds:

$$Comp(\alpha, N, s(N), f) \rightarrow \neg Comp(\beta, N, s(N), f \oplus E_N)$$

Hence it is sufficient to show that $S_2^2(\alpha)$ does not prove the implication above.

Assume, for the sake of contradiction, that it does. Denote

$$A_i(p_1, ..., p_N, q_1, ..., q_{s(N)})$$

the propositional formula formalizing that the computation on i coded in α yields the value $f(i)$ with the atoms p_i translating $i \in f$ and q_j translating $j \in \alpha$. Similarly denote by

$$B_i(p_1, ..., p_N, r_1, ..., r_{s(N)})$$

the formula formalizing that the computation on i coded in β does not yield the value $f(i) \oplus E_N(i)$ (here r_j translate $j \in \beta$).

By Theorem 4.1.3 the sequent:

$$A_1, ..., A_N \rightarrow B_1, ..., B_N$$

has an LK-proof π satisfying the hypothesis of Theorem 4.2.3. By that Theorem the implication admits an interpolant

$$I(p_1, ..., p_N)$$

whose circuit-size is $\leq (2^{(\log N)^{O(1)}})^{O(1)} = 2^{(\log N)^{O(1)}}$.

The bits $p_1, ..., p_N$ naturally define the function $f \in \{0,1\}^N$. Define the sets U, V by:

$$U = \{f \in \{0,1\}^N | \exists q \bigwedge_i A_i(p,q)\}$$

$$V = \{f \in \{0,1\}^N | \exists r \bigwedge_j \neg B_j(p,r)\}$$

The sets U, V are disjoint and the following holds for any $f \in \{0,1\}^N$:

$$f \in U \Leftrightarrow f \oplus E_N \in V$$

This implies the following properties of the interpolant:
1. $\neg I(f) \Rightarrow f \notin U$, for any f
2. $I(f \oplus E_N) \Rightarrow f \notin U$, for any f

Therefore, the following property $P(f)$ of functions $f \in \{0,1\}^N$ is the quasi-natural property against $P/poly$:

$$P(f) = \begin{cases} \neg I(f) & \text{if at least a half of functions satisfy } \neg I \\ I(f \oplus E_N) & \text{otherwise} \end{cases}$$

This is contradiction with SPRG Conjecture according to Theorem 2.5 in modified quasi-natural form (Remark 2.6).

■

Chapter 5

Concluding Remarks

The most satisfactory result, but possibly intractable, would be an independence proof from a strong theory like Peano Arithmetic or ZFC.

Basically there are three ways how one could improve the independence result obtained in the previous section.

The first one is to prove the conservativeness of stronger theories than $S^2_2(\alpha)$ over $T^1_2(\alpha)$ what would allow to prove and use stronger form of Theorem 4.1.3. However, this is not possible since it is known that even $T^2_2(\alpha)$ is not $U_1(\alpha, \Sigma^b_\infty)$-conservative over $T^1_2(\alpha)$, cf. [10].

The second one is to improve the interpolation Theorem 4.2.3 by effective elimination of cut formula of depth more than 2. Again, it turns out that this cannot work. Krajíček [13] proved that such elimination, in general, requires superpolynomial speed-up.

The third one is to prove that even stronger proof systems admit existence of polynomial size interpolant in terms of the size of the shortest proof of given proposition. In fact, the presented method is based on this property. Unfortunately, Krajíček and Pudlák [14] showed that the property fails for strong systems like Extended Frege system assuming that RSA is secure against an adversary computing functions in P/poly.

Therefore it seems that one needs to discover new methods.

Bibliography

[1] Boppana R.B., Sipser M.: *The Complexity of Finite Functions*; NY, MIT, 1989.

[2] Buss, S.R.: *Bounded Arithmetic*; Bibliopolis, 1986. Revision of 1985 Princeton University Ph.D. thesis.

[3] Buss, S.R.: *Bounded Arithmetic and Propositional Proof Complexity*; University of California.
http://math.ucsd.edu/ sbuss/ResearchWeb/marktoberdorf95/index.html

[4] Buss, S.R.: *Axiomatizations and conservation results for fragments of bounded arithmetic*; in Logic and Computation, proceedings of a Workshop held Carnegie-Mellon University, 1987, vol. 106 of Contemporary Mathematics, American Mathematical Society, 1990, pp. 57-84.

[5] Craig, W.: *Linear reasoning: A new form of the Herbrand-Getzen theorem*; Journal of Symbolic Logic, 22(3), 1957, pp. 250-268.

[6] Craig, W.: *Three uses of the Herbrand-Getzen theorem in relating model theory and proof theory*; Journal of Symbolic Logic, 22(3), 1957, pp. 269-285.

[7] Hastad, J.: *Almost optimal lower bounds for smart depth circuits*; 18th Symposium on Theory of Computing, 1986, pp. 6-20.

[8] Karp R.M., Lipton R.: *Turing machines that take advive*; L'enseignment Mathematique, 28, 1982, pp. 191-209.

[9] Kent C.F., Hodgson B.R.: *An arithmetic characterization of NP*; Theoretical Comput. Sci, 21, 1982, pp. 255-267.

[10] Krajíček, J.: *Bounded Arithmetic, Propositional Logic, and Complexity Theory*; Cambridge University Press, 1994.

[11] Krajíček, J.: *Fragments of Bounded Arithmetic and Bounded Query Classes*; Transactions of the AMS, 338(2), 1993, pp. 587-598.

[12] Krajíček, J.: *Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithemtic*; Journal of Symbolic Logic, 66(2), 1997, pp. 457-486.

[13] Krajíček, J.: *Lower Bounds to the Size of Constant-depth Propositional Proofs*; Journal of Symbolic Logic, 59(1), 1994, pp. 73-86.

[14] Krajíček J., Pudlák P.:*Some consequences of cryptographical conjectures for S_1^2 and EF*; Logic and Computational Complexity, Ed. D.Leivant, Springer-Verlag, Lecture Notes in Computer Science, Vol. 960, 1995, pp.

210-220. Revised version in *Information and Computation*, Vol. 140(1), 1998, pp. 82-94.

[15] Krajíček J., Pudlák P., Takeuti G.: *Bounded Arithmetic and the Polynomial Hierarchy*; Annals of Pure and Applied Logic, 52, 1991, pp. 143-153.

[16] Razborov, A.A.: *Bounded arithmetic and lower bounds in Boolean complexity*; in: Feasible Mathematics II, eds. P.Clote and J.Remmel, Birkhauser, pp. 344-386.

[17] Razborov, A.A.: *Unprovability of lower bounds on the circuit size in certain fragments of bounded arithmetic*; Izvestiya of the R.A.N., 59(1), 1995, pp. 201-224.

[18] Razborov A.A., Rudich S.: *Natural Proofs*; in Proceedings of the 26-th Annual ACM Symposium on Theory of Computing, 1994, pp. 204-213.

[19] Stockmeyer, L.J.: *The polynomial-time hierarchy*; Theoretical Comput. Sci., 3, 1976, pp. 1-22.

[20] Takeuti, G.: *Proof theory*; North Holland, 1975.

[21] Wrathall, C.: *Complete sets and the polynomial-time hierarchy*; Theoretical Comput. Sci., 3, 1976, pp. 23-33.

[22] Yao, A.: *Separating the polynomial-time hierarchy by oracles*; 26th Foundations of Computer Science, 1985, pp. 1-10.