

The Cook-Reckhow definition

Jan Krajíček

Faculty of Mathematics and Physics
Charles University*

The Cook-Reckhow paper [6] introduced the notions of *propositional proof systems* and *polynomial simulations* among them, described several classes of logical propositional calculi and compared them with regards to their efficiency, and introduced the *pigeonhole principle tautology* PHP_n that is *the prime example of a tautology hard to prove in weaker systems* ever since, rivaled only by the tautology proposed earlier by Tseitin [34]. It also showed that *the central question whether there exist a propositional proof system allowing polynomial size proofs of all tautologies is equivalent to a central question of complexity theory whether the class \mathcal{NP} is closed under complementation*.

Classical proof theory of first-order logic developed in the first half of the twentieth century assigned to proofs several combinatorial characteristics and some of them can be perceived as measures of complexity; for example, the height of a proof tree. Primary emphasis was on constructions producing various normal forms of proofs and the combinatorial characteristic helped to measure the progress of normalization constructions. The question about the minimum length of a proof of a statement (measured by either the number of steps or by the size, i.e. the number of symbols) was also studied, primarily in the context of speed-up results; references [12, 27, 9, 28, 29] can serve as illustrations of this research. The results rest on constructions underlying the undecidability of the Halting problem or Gödel's Incompleteness theorem, and do not give any insight¹ into analogous problems in propositional logic.

It was the Cook-Reckhow 1979 paper [6] which defined the area of research we now call proof complexity. There were earlier papers which contributed to the subject as we understand it today, the most significant being Tseitin's [34]. But none of them introduced general notions² that would allow to make an explicit and universal link between lengths-of-proofs problems and computational complexity theory.

*Sokolovská 83, Prague, 186 75, The Czech Republic, krajicek@karlin.mff.cuni.cz

¹In fact, Parikh [28] introduced theory PB (called now $I\Delta_0$, cf. [20, 23]) which later in the 1980s turned out to be important for the development of proof complexity.

²Tseitin's paper [34] offers no motivation for the research reported there but one of the motivations were questions we now formulate as the \mathcal{P} vs. \mathcal{NP} problem (another motivation was computer processing of natural languages) and the special role the *Entscheidungsproblem* for *propositional calculus* plays in them (personal communication).

In this note we shall highlight three particular definitions from the paper: of proof systems, p-simulations and the formula PHP_n , and discuss their role in defining the field. We will also mention some related developments and open problems. In particular, we shall show that the general definition of proof systems that has seemingly little to do with how ordinary logical calculi are defined is actually equivalent to the calculi definition with a more general treatment of logical axioms than is usual (Section 1), we shall present the optimality problem stemming from the notion of simulations (Section 2), and we shall discuss the role of the PHP_n formula in proof complexity lower bounds, its limitations and modern variants aimed at strong proof systems (Section 3). Paper [6] also discusses few measures of complexity of proofs other than the proof size and describes some relations among them; we shall not discuss this and instead we refer the reader to available literature.

The Cook-Reckhow paper [6] had a precursor [7], an extended abstract that summarized some research presented later in [6], as well as from Reckhow's PhD Thesis [33]. This earlier paper differs from [6] in several aspects: it uses simulations (see Sec 2) as opposed to the latter finer notion of p-simulations, and it contains neither Extended Frege system nor the PHP tautology. It presents a rather succinct version of the construction underlying Reckhow's theorem that was replaced in [6] by a similar statement for Extended Frege systems with much easier (and more illuminating) proof (cf. Sec 2). It also treats in detail the sequent calculus that is only glanced over in [6], and it derives some super-polynomial lower bounds, using Tseitin's results in [34], for weak proof systems such as tree-like resolution or semantic trees.

The aim of this note is to give an idea to the non-expert reader about the main ideas stemming from [6] and about the fundamental problems of proof complexity. Further information about proof complexity, its basic as well as advanced parts, and about topics in mathematical logic and complexity theory it relates to can be found in [20, 23].

1 Definition of proof systems

The main example of a logical propositional calculus to keep in mind is a *Frege system*. It is any calculus operating with propositional formulas over a complete basis of logical connectives (i.e. all Boolean functions can be defined in the language), having a finite number of sound axiom schemes and inference rules that are *implicationally complete*. The latter term means that if a formula A is a logical consequence of formulas B_1, \dots, B_m then it can be derived from them in the calculus. An example of a complete language is the DeMorgan language with constants $0, 1$ and connectives \neg, \vee and \wedge . We shall denote by TAUT the set of tautologies in this language and we shall tacitly assume that $\text{TAUT} \subseteq \{0, 1\}^*$, with formulas being encoded by binary strings in some natural way.

There is a number of such systems described in logic text-books and they are often called *Hilbert-style*, referring to Hilbert's work in proof theory [14, 15, 16].

The form of calculi is based on Frege's [10], hence the name Cook and Reckhow [6] chose for this class of propositional calculi.

The calculi are *sound* (every provable formula is a tautology) and *complete* (every tautology is provable). In addition, the key property singled out by [6] is that to recognize whether a string of symbols is a valid proof in the calculus or not is computationally feasible: it can be done by a p-time algorithm. This leads to the following fundamental definition.

Definition 1.1 (Cook-Reckhow [6])

A **propositional proof system** is any p-time computable function

$$f : \{0, 1\}^* \rightarrow \{0, 1\}^*$$

such that

$$TAUT = Rng(f) .$$

Any $w \in \{0, 1\}^*$ such that $f(w) = A$ is called an **f -proof of A** .

Cook and Reckhow [6, Def.1.3] actually define more generally a proof system for any $L \subseteq \{0, 1\}^*$ by the condition $L = Rng(f)$, and consider proof systems for the set of tautologies in any fixed language.

A Frege system F can be represented by a function f which takes a string w and maps it to the last formula of w , if w is a sequence of formulas that forms a valid F -proof, or to constant 1 if w is not an F -proof. The soundness of F implies that $Rng(f) \subseteq TAUT$ and its completeness implies the opposite inclusion $TAUT \subseteq Rng(f)$.

It is easy to see that a number of other classes of propositional calculi considered in mathematical logic literature fit the definition in the same sense as Frege systems do. These calculi include resolution, sequent calculus or natural deduction. Less usual examples of propositional proof systems can be constructed as follows. Take a consistent first-order theory axiomatized by a finite number of axioms and axiom schemes that is sound and contains some simple base theory (in order to guarantee both the correctness and the completeness) and interpret it as a proof system: a proof of formula A is a proof in the theory of the formalized statement $A \in TAUT$. Yet another examples are logic calculi that are set-up to prove the unsatisfiability of formulas: these can be interpreted as proof systems by accepting a refutation of $\neg A$ as a proof of A .

In addition, the general form of the definition allows us to interpret various calculations in algebra as propositional proofs. Here it is more natural to speak about refutation systems. If we have a CNF formula that is a conjunction of clauses C_i , we can represent each C_i by a constraint of an algebraic form and use a suitable algebraic calculus to derive the unsolvability of the formula. For example, a clause

$$p \vee \neg q \vee r$$

together with the requirement that we look for 0 – 1 solution can be represented by polynomial equations

$$(1 - p)q(1 - r) = 0 , \quad p^2 - p = 0 , \quad q^2 - q = 0 , \quad r^2 - r = 0$$

the first equation states that the clause contains a true literal while the last three equations force 0 – 1 solutions over any integral domain. In this case we can use a calculus deriving elements of the ideal generated by the equations representing similarly all clauses of the formula, trying to derive 1 as a member of the ideal and thus demonstrating the unsolvability of the equations and hence the unsatisfiability of the formula.

Another approach is to represent the clause as integer linear inequalities

$$p + (1 - q) + r \geq 1, \quad 1 \geq p, q, r \geq 0$$

and use some integer linear programming algorithm to derive the unsolvability of the system of inequalities representing the whole CNF formula. It is a great advantage of Definition 1.1 that it puts all these quite different formal system under one umbrella.

Proof systems can be also defined equivalently in a relational form. A *relational propositional proof system* is a binary relation $P(x, y)$ that we interpret as the provability relation *y is a proof of x*. It is required that it is p-time decidable and that for any formula A it holds:

$$A \in \text{TAUT} \text{ iff } \exists w P(A, w) .$$

This is closer in form to logical calculi (and can be represented by the function version as Frege systems were before) but it is equally general: a functional proof system f is represented by the relation $f(y) = x$.

A proof system f is **p-bounded** iff there exists $c \geq 1$ such that for all A , $|A| > 1$,

$$A \in \text{TAUT} \Rightarrow \exists w (|w| \leq |A|^c) f(w) = A .$$

In the relational form this would read

$$A \in \text{TAUT} \Rightarrow \exists w (|w| \leq |A|^c) P(A, w)$$

and combining this with the soundness we get

$$x \in \text{TAUT} \Leftrightarrow \exists y (|y| \leq |x|^c) P(x, y) .$$

The right-hand side expression has the well-known general form in which any \mathcal{NP} set can be defined. Hence we get as a simple but important corollary to the definition the following statement (the second equivalence uses Cook's theorem: the \mathcal{NP} -completeness of SAT, cf. Cook [4]).

Theorem 1.2 (Cook - Reckhow [6])

A p-bounded proof system exists iff $\text{TAUT} \in \mathcal{NP}$ iff $\mathcal{NP} = \text{coNP}$.

This theorem determines

Problem 1.3 (Main problem of proof complexity)

Is there a p-bounded proof system for TAUT?

By Theorem 1.2 showing that no p-bounded proof system exists would imply, in particular, that $\mathcal{P} \neq \mathcal{NP}$ because \mathcal{P} is closed under complementation. On the other hand, defining a p-bounded proof system f would allow to witness various coNP -properties by short witnesses (f -proofs); [7] mentions the property that two graphs are not isomorphic.

One may consider variants of the definition of proof systems when the provability relation is not necessarily decidable by a p-time algorithm but only by more general algorithm; for example, using some randomness. My view is that this changes the basic problems of proof complexity substantially. While it may link propositional proof systems with various other proof systems considered in different parts of complexity theory, it is not clear that it will shed light on proof complexity proper. This may change if some of these other parts of complexity theory advance significantly on their own fundamental open problems.

The Cook-Reckhow definition is handy for establishing Theorem 1.2 and the connection to complexity theory but the reader may wonder if it does not deviate from logical form of calculi too much. In fact, it can be shown that every proof system can be p-simulated (in the sense of the next section) by a Frege system whose set of axioms is not given just by a finite number of axiom schemes but is possibly infinite but easy to recognize (in p-time, in particular) sparse subset of TAUT. Doing this precisely is rather technical and we refer the reader to [24, 20, 23].

2 Simulations among proof systems

When studying the problem whether some proof system is p-bounded it is useful to be able to compare two proof systems with respect to their efficiency. The following two notions³ are aimed at that.

Definition 2.1 (Cook-Reckhow [6])

*Let f, g be two proof systems. A **simulation** of g by f is any function*

$$h : \{0, 1\}^* \rightarrow \{0, 1\}^*$$

such that for all $w \in \{0, 1\}^$, $|h(w)| \leq |w|^c$, for some independent constant $c \geq 1$ and all $|w| > 1$, and such that*

$$f(h(w)) = g(w) .$$

*Simulation h is **p-simulation** if it is p-time computable.*

*Proof system f **(p-)simulates** g ($f \geq g$ and $f \geq_p g$ in symbols, respectively) iff there is a (p-)simulation of g by f .*

In other words, the statement that $f \geq g$ says that if we replace f by g we can speed-up proofs at most polynomially, while the statement that $f \geq_p g$ says that we can even efficiently translate g -proofs into f -proofs. Both these relations

³P-simulations are also defined in Cook [5].

are quasi-orderings (we get partial orderings after factoring by the equivalence relations of mutual simulations).

There are other options how to define a quasi-ordering of proof systems. In particular, if we did not insist in Definition 1.1 that all proof systems prove tautologies in the same language (we have defined TAUT using the DeMorgan language only) but allowed tautologies in different languages then a (p-)simulation should allow to translate also formulas and not just proofs. By insisting that the target set is TAUT we forced that such a translation of formulas is incorporated into the definition of particular proof systems that may operate with formulas in other languages or even with polynomials or other objects. In fact, considering instead of propositional proof systems proof systems for any coNP -complete set we ought to allow p-reductions between such sets and TAUT.

However, for positive results (as is Theorem 2.2 below) p-simulations allow to formulate the strongest possible statements while strongest negative results (obtained by proving a super-polynomial lower bound for f -proofs of formulas for which there are polynomial size g -proofs) talk about super-polynomial speed-ups and hence about the non-existence of simulations. Thus the two types of simulations serve their purpose very well.

Cook and Reckhow [6] compared various logical proof systems in terms of p-simulations; the following statement summarizes their most memorable results in this respect.

Theorem 2.2 (Cook-Reckhow [6])

1. *All Extended Frege systems in all languages p-simulate each other.*
2. *Frege systems and propositional parts of natural deduction and of sequent calculus mutually p-simulate each other.*
3. *Extended Frege system EF and Tseitin's Extended resolution ER are p-equivalent and they are p-simulated by any Frege system with the substitution rule.*

Extended Frege systems EF were defined in [6] in a direct analogy with Extended resolution ER of Tseitin [34]. Any such system starts with a Frege system and allows, in addition, to abbreviate formulas by new atoms and use these in proofs. In particular, during an EF-proof we can take a new atom q (an *extension atom*) not used so far and not occurring in the target formula A to be proved, any formula D not containing q , and introduce the equivalence $q \equiv D$ (represented in the language of the system) as a new *extension axiom*. Note that EF is not a Frege system as the introduction of extension axioms does not fit the schematic way Frege axioms are supposed to be defined. The first statement in the theorem is a weaker version of Reckhow's theorem [33] which is stated for Frege systems. The version for Extended Frege system is much easier to prove (see [20, 23] for published proofs of the stronger version).

For the definition of natural deduction see [31], for sequent calculus see any of [11, 20, 23] (the sequent calculus part of the statement is just mentioned in [6]

while natural deduction is treated in detail). The substitution rule allows to infer from a formula $B(p_1, \dots, p_m)$ its arbitrary substitution instance $B(C_1, \dots, C_m)$ in one inference. A Substitution Frege system SF is a Frege system augmented by this rule. It was proved later in [8] (indirectly) and in [24] (an explicit p-simulation) that EF actually p-simulates SF as well.

An illuminating description of EF is that it is essentially a Frege system that operates with circuits rather than with formulas; this has been made precise in [17]. Perhaps even more useful is the statement that the minimum size s of an EF-proof of formula A is proportional to the minimum number of steps in a Frege proof of A and $|A|$, or to the minimum number of different formulas that need to occur as subformulas in any Frege proof of A and $|A|$, cf. [6] or [20, 23]. Hence moving from F to EF means that we are replacing the size as the measure of complexity of Frege proofs by the number of steps. This is interesting because from the point of view of mathematical logic the number of steps is a very natural complexity measure.

Extended Frege system is also important because of its relation to a particular theory PV introduced by Cook [5] at the same time (he used ER in his paper). This is discussed in S. Buss's article in this volume. Theory PV (stands for Polynomially Verifiable) allows to formalize a number of standard computational complexity constructions and arguments. Understanding the power of proof system EF and, in particular, showing that it is not p-bounded, is considered in the field as the pivotal step towards solving the Main problem and proving that $\mathcal{NP} \neq \text{co}\mathcal{NP}$. In particular, it is also known that any super-polynomial lower bound for EF implies that $\mathcal{NP} \neq \text{co}\mathcal{NP}$ is consistent with PV (cf. [23, Sec.12.4]).

We shall mention one problem formulated only later in [24] which is, however, natural and is implicit in the definition of simulations.

Problem 2.3 (Optimality problem)

Is there a proof system that (p-)simulates all other proof systems?

Such a maximal proof system is called **(p-)optimal** after [24]. We have (names for) three types of proof systems whose existence is considered by most researchers unlikely: p-bounded, p-optimal and optimal. Every p-bounded or p-optimal proof system is also optimal and this rules out three out of eight possibilities for the existence/non-existence of objects of these three types. At present we cannot rule out any of the remaining five scenarios:

- *A p-bounded, p-optimal proof system P_1 exists.*

Having such an ideal proof system we do not need to consider any other: even searching for proofs in any other proof system can be reduced to searching for P_1 -proofs. (We ignore here that p-reductions themselves increase polynomially the time complexity of a proof search algorithm and may transform a combinatorially transparent one into a complex one, cf. the last paragraph of this section.)

- *A p -bounded proof system P_2 exists but no p -optimal does.*

While p -size P_2 -proofs would exist for each tautology, finding them may be difficult and it may help to consider different proof systems for different (classes of) tautologies.

- *A p -optimal proof system P_3 exists but no p -bounded does.*

Here we can restrict our attention to P_3 : it is also optimal and search for proofs in any proof system can be replaced by a search for P_3 -proofs.

- *An optimal proof system P_4 exists but no p -bounded or p -optimal does.*

Proving lengths-of-proofs lower bounds (or upper bounds, for that matter) can be restricted to P_4 but proof search may benefit from considering different proof systems for different classes of tautologies.

- *None of these ideal objects exist.*

This appears to be the most likely scenario.

At present we cannot rule out that a Frege system is one of P_1, \dots, P_4 . The Optimality problem is related to a surprising number of varied topics in proof theory (quantitative Gödel's theorem), finite model theory, structural complexity, and some other (cf. [23, Chpt.21]).

An interesting question left-out by [6] as well as in later literature is how to compare proof search algorithms. A tentative definition was proposed in [23, Sec.21.5].

3 Hard tautologies and the PHP_n formula

In order to prove lengths-of-proofs lower bounds for a proof system we start with a suitable candidate tautology that we conjecture to be hard to prove (i.e. requiring long proofs) therein. A particular tautology for this purpose based on the pigeon-hole principle was proposed in [6]. The formula, to be denoted PHP_n , is built from atoms p_{ij} with $i \in [n] := \{1, \dots, n\}$ and $j \in [n-1]$, for $n \geq 2$. Thinking of p_{ij} as representing the atomic statement that i maps to j , we can express that the map is defined at i by the clause

$$\bigvee_j p_{ij} \tag{1}$$

the fact that j can be the value of at most on i by

$$\bigvee_{i_1 \neq i_2} \neg p_{i_1 j} \vee \neg p_{i_2 j} \tag{2}$$

and the fact that i maps to at most one value by

$$\bigvee_{j_1 \neq j_2} \neg p_{ij_1} \vee \neg p_{ij_2} . \tag{3}$$

Taking the conjunction of these clauses for all choices of i and j states that

$$\{(i, j) \in [n] \times [n - 1] \mid p_{ij} = 1\}$$

is the graph of an injective map from $[n]$ into $[n - 1]$. No such map exists and hence the negation of the conjunction is a tautology. This leads to the following definition.

Definition 3.1 (Cook-Reckhow [6])

For any $n \geq 2$, PHP_n is the disjunction of negations of clauses in (1) for all $i \in [n]$ and in (2) for all $j \in [n - 1]$ and in (3) for all $i \in [n]$.

In fact, to reach a contradiction we do not need the assumption that it is the graph of a function, a multi-function suffices (if i occupies more values j it is harder to be injective). In other words, we do not need to include the clauses from (3) and [6] did not include them. Nowadays the definition of PHP_n as formulated above is more customary and proving lower bounds for it yields stronger results than for the more economical version (the principle assumes more and hence it is logically weaker).

Cook and Reckhow [6] showed that it is possible to prove PHP_n in Extended Frege systems by a proof of size polynomial in n (note that the size of PHP_n is also polynomial in n). In fact, they introduced EF in order to formalize smoothly the inductive argument: from an assignment violating PHP_n we can define (using the extension rule) an assignment violating PHP_{n-1} . Hence PHP_n has also a proof in Frege systems with a polynomial number of steps (but having large size). Buss [3] improved the result (by a different construction formalizing counting) and proved that Frege systems actually also admit polynomial size proofs of PHP_n .

On the other hand, in a breakthrough result, Haken [13] proved a first lower bound for resolution using PHP_n and the same formula was proved to be hard for constant depth subsystems of any Frege system in the DeMorgan language by Ajtai [1] (Haken's lower bound was exponential while Ajtai's super-polynomial - its rate was later improved to exponential too by [25, 30]). The same formula (represented by polynomial equations similarly as in Section 1.1) served to Razborov [32] for his lower bound for polynomial calculus, an algebraic proof system manipulating polynomials.

There is an important variant of the PHP formula considered first by Paris, Wilkie and Woods [26] in the context of bounded arithmetic: allow i to range over a much bigger set than $[n]$; for example, over $[2n]$ or even $[n^2]$. Similarly as the PHP principle is related to counting these *weak* PHP principles relate to approximate counting and [26] showed that they can be sometimes used in place of PHP proper and that, crucially, they are easier to prove. Their proof (formulated using bounded arithmetic) gives quasi-polynomial size proofs in constant depth Frege systems of formulas formalizing these weaker principles for $n \geq 2$.

Even if the formula PHP_n itself cannot be used as a hard example for proof systems like F or EF, formulas formalizing a form of a weak PHP in a different

way possibly can. It has been an insight of Wilkie (result reported in [20, Sec.7.3]) that the *dual weak PHP* for p-time functions is important in bounded arithmetic (this has been much extended by Jeřábek [17, 18, 19]). The principle says that no p-time function g when restricted to any $\{0, 1\}^n$ can be onto $\{0, 1\}^{2n}$. Now take an arbitrary $b \in \{0, 1\}^{2n} \setminus \text{Rng}(g_n)$, where g_n is the restriction of g to $\{0, 1\}^n$, and define propositional formula

$$\tau_b(g_n)$$

expressing $\forall x \in \{0, 1\}^n g_n(x) \neq b$. The formula uses n atoms for bits of x and further $\text{poly}(n)$ atoms for bits of the computation y of g_n on x and says, in a DNF form, that either y is not a valid computation on input x or the output of the computation differs from b . Clearly

$$\tau_b(g_n) \in \text{TAUT} \Leftrightarrow b \notin \text{Rng}(g_n) .$$

These formulas were defined in [2, 21] and lead to the theory of proof complexity generators proposing several candidate tautologies of the form above as possibly hard for strong (or all) proof systems. The reader may find an overview of the theory in [22, Chpts.29 and 30] (no need to read the first 28 chapters).

Acknowledgements:

I thank Sam Buss, Bruce Kapron, Igor C. Oliveira and Jan Pich for comments on earlier versions of this paper.

References

- [1] M. Ajtai, The complexity of the pigeonhole principle, in: *Proc. IEEE 29th Annual Symp. on Foundation of Computer Science*, (1988), pp. 346-355.
- [2] M. Alekhovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson, Pseudorandom generators in propositional proof complexity, *SIAM J. on Computing*, **34(1)**, (2004), pp.67-88.
- [3] S. R. Buss, Polynomial size proofs of the propositional pigeonhole principle, *J. Symbolic Logic*, **52**, (1987), pp.916-927.
- [4] S. A. Cook, The complexity of theorem proving procedures, in: *Proc. 3rd Annual ACM Symp. on Theory of Computing (STOC)*, (1971), pp. 151-158. ACM Press.
- [5] S. A. Cook, Feasibly constructive proofs and the propositional calculus, in: *Proc. 7th Annual ACM Symp. on Theory of Computing (STOC)*, (1975), pp. 83-97. ACM Press.
- [6] S. A. Cook and R. A. Reckhow, The relative efficiency of propositional proof systems, *J. Symbolic Logic*, **44(1)**, (1979), pp.36-50.

- [7] S. A. Cook and R. A. Reckhow, On the lengths of proofs in the propositional calculus, in: *Proc. of the Sixth Annual ACM Symposium on the Theory of Computing*, May 1974, pp.135-148.
Corrections in *SIGACT News*, **6**, (1974), pp.15-22.
- [8] M. Dowd, *Propositional representations of arithmetic proofs*, PhD Thesis, University of Toronto, (1979).
- [9] A. Ehrenfeucht and J. Mycielski, Abbreviating proofs by adding new axioms, *Bull. A.M.S.*, **77**, (1971), pp.366-367.
- [10] G. Frege, *Begriffsschrift: eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*, Halle, (1879).
- [11] G. Gentzen, Die Widerspruchsfreiheit der reinen Zahlentheorie, *Mathematische Annalen*, **112**, (1936), pp.493-565.
- [12] K. Gödel, Über die Länge von Beweisen, *Ergebnisse ihres Mathematischen Kolloquiums*, Heft 7, (1936), pp.23-24.
- [13] A. Haken, The intractability of resolution, *Theoretical Computer Science*, **39**, (1985), pp.297-308.
- [14] D. Hilbert and W. Ackermann, *Principles of Mathematical Logic*. Chelsea. (1950). Translation of 1938 German edition.
- [15] D. Hilbert and P. Bernays, *Grundlagen der Mathematik. I*, in: Die Grundlehren der mathematischen Wissenschaften 40, Berlin, New York: Springer-Verlag, (1934).
- [16] D. Hilbert and P. Bernays, *Grundlagen der Mathematik. II*, in: Die Grundlehren der mathematischen Wissenschaften 50, Berlin, New York: Springer-Verlag, (1939).
- [17] E. Jeřábek, Dual weak pigeonhole principle, Boolean complexity, and derandomization, *Annals of Pure and Applied Logic*, **129**, (2004), pp.1-37.
- [18] E. Jeřábek, Approximate counting in bounded arithmetic, *J. of Symbolic Logic*, **72(3)**, (2007), pp.959-993.
- [19] E. Jeřábek, Approximate counting by hashing in bounded arithmetic, *J. of Symbolic Logic*, **7493**, (2009), pp.829-860.
- [20] J. Krajčček, *Bounded arithmetic, propositional logic, and complexity theory*, Encyclopedia of Mathematics and Its Applications, Vol. **60**, Cambridge University Press, (1995).
- [21] J. Krajčček, On the weak pigeonhole principle, *Fundamenta Mathematicae*, Vol.**170(1-3)**, (2001), pp.123-140.

- [22] J. Krajíček, *Forcing with random variables and proof complexity*, London Mathematical Society Lecture Note Series, No. **382**, Cambridge University Press, (2011).
- [23] J. Krajíček, *Proof complexity*, Encyclopedia of Mathematics and Its Applications, Vol. **170**, Cambridge University Press, to appear in 2019.
- [24] J. Krajíček and P. Pudlák, Propositional proof systems, the consistency of first-order theories and the complexity of computations, *J. Symbolic Logic*, **54(3)**, (1989), pp.1063-1079.
- [25] J. Krajíček, P. Pudlák, and A. Woods, An Exponential Lower Bound to the Size of Bounded Depth Frege Proofs of the Pigeonhole principle", *Random Structures and Algorithms*, **7(1)**, (1995), pp.15-39.
- [26] J. Paris, A. J. Wilkie and A. Woods, Provability of the Pigeonhole Principle and the Existence of Infinitely Many Primes, *Journal of Symbolic Logic*, **53(4)**, (1988), pp.1235-1244.
- [27] A. Mostowski, Sentences undecidable in formalized arithmetic, North-Holland, Amsterdam, (1952).
- [28] R. Parikh, Existence and feasibility in arithmetic, *J. Symbolic Logic*, **36**, (1971), pp.494-508.
- [29] R. Parikh, Some results on the length of proofs, *Trans. A.M.S.*, **177**, (1973), pp.29-36.
- [30] T. Pitassi, P. Beame, and R. Impagliazzo, Exponential lower bounds for the pigeonhole principle, *Computational Complexity*, **3**, (1993), pp.97-308.
- [31] D. Prawitz, *Natural deduction*, A proof-theoretic study, Stockholm, (1965).
- [32] A. A. Razborov, Lower Bounds for the Polynomial Calculus, *Computational Complexity*, **7(4)**, (1998), pp.291-324.
- [33] R. A. Reckhow, *On the lengths of proofs in the propositional calculus*, PhD.Thesis, Dept. of CS, University of Toronto, (1976).
- [34] G. S. Tseitin, On the complexity of derivations in propositional calculus, in: *Studies in mathematics and mathematical logic, Part II*, ed. A.O.Slisenko, (1968), pp.115-125.