From feasible proofs to feasible computations

Jan Krajíček[*]

Faculty of Mathematics and Physics, Charles University
Sokolovská 83, 186 75, Prague, Czech Republic

Abstract. We shall discuss several situations in which it is possible to extract from a proof, be it a proof in a first-order theory or a propositional proof, some feasible computational information about the theorem being proved. This includes extracting feasible algorithms, deterministic or interactive, for witnessing an existential quantifier, a uniform family of short propositional proofs of instances of a universal quantifier, or a feasible algorithm separating a pair of disjoint NP sets.

1 Universal theories

Let L be a language that has a function symbol corresponding to every polynomial time algorithm, say as represented by clocked polynomial time Turing machines. We shall assume that polynomial time relations are represented by their characteristic functions and hence the only relation symbol is the equality, which we regard as a logical symbol.

Every function symbol from L has a canonical interpretation on the set of natural numbers $\mathbf{N}$ which we identify with the set of all binary words $\{0,1\}^*$; the resulting L-structure will be called the standard model. Let T be the universal theory of the standard model.

Theory PV of Cook [7] can be thought of as a subtheory of T: its language has a symbol for every polynomial time algorithm as they are built in Cobham's theorem [6] by repeated composition and limited recursion on notation, and its axioms are universal statements codifying the equations defining this process (see also [5, 8, 13]).

1.1 Witnessing existential formulas

Assume

$$T \vdash \exists y A(x,y)$$

where A is an open formula. Herbrand's theorem implies that there are terms $t_1(x), \ldots, t_k(x)$ such that

$$T \vdash \bigvee_i A(x, t_i(x)) \ .$$

[*] Supported in part by grants IAA100190902, MSM0021620839, LC505 (Eduard Čech Center) and by a grant from the John Templeton Foundation. Also partially affiliated with the Institute of Mathematics of the Academy of Sciences and grant AV0Z10190503.

As the class of polynomial time functions is closed under definitions by cases distinguished by a polynomial time property, it follows that there is one term $t(x)$ such that

$$T \vdash A(x, t(x)) .$$

In particular, the polynomial time algorithm defined by $t(x)$ witnesses in the standard model the validity of the sentence $\forall x \exists y A(x, y)$.

Consequences of this simple witnessing are various independence results (conditioned upon complexity theoretic hypotheses). For example, let h be a one-way permutation (cf. [1]). As h must be surjective the sentence

$$\forall y \exists x \; h(x) = y$$

is valid. But it is not provable in T as the witnessing algorithm would compute the inverse function to h (which is supposed to be impossible).

Another example can be given by a hash function g that always outputs less bits than the input has. Then there must be a collision pair of any length:

$$\forall x \exists y_1, y_2 \; [\; y_1 \neq y_2 \wedge |x| = |y_1| = |y_2| \wedge g(y_1) = g(y_2) \;]$$

but this sentence cannot be provable in T as the witnessing algorithm would find a collision pair (which is supposed to be hard).

For the final example let P be a propositional proof system in the sense of Cook and Reckhow [9], i.e. a polynomial time function whose range is exactly the set of propositional tautologies. Any π such that $P(\pi) = \tau$ is called a P-proof of τ. The completeness of P can be stated as

$$\forall x \exists y \; [\; Fla(x) \rightarrow \; (\; SatNeg(x, y) \vee P(y) = x \;) \;\;]$$

where $Fla(x)$ and $SatNeg(x, y)$ are open L-formulas formalizing that x *is a propositional formula* and that y *is a truth assignment satisfying the negation of* x, respectively. Then unless P $=$ NP this is unprovable in T as a witnessing algorithm could be used to decide SAT.

1.2 Witnessing $\exists_2$-formulas

Assume

$$T \vdash \exists y \forall z A(x, y, z)$$

where A is an open formula. Herbrand's theorem implies (see [22, 13]) that there are terms $t_1(x), t_2(x, y_1), \ldots, t_k(x, z_1, \ldots, z_{k-1})$ such that

$$T \vdash \bigvee_i A(x, t_i(x, z_1, \ldots, z_{i-1}), z_i) . \tag{1}$$

This can be interpreted as an interactive algorithm witnessing the existential quantifier in the following way (cf. [21]). The computation is performed by a polynomial time Student and a Teacher of unlimited powers. Upon receiving an input x the Student computes his first candidate witness $y_1 := t_1(x)$ and sends it to the Teacher. If indeed $\forall z A(x, y_1, z)$ holds she will acknowledge it and the computation stops. Otherwise she will send to the Student a counter-example z_1: a string such that $\neg A(x, y_1, z_1)$. The Student then computes the next candidate witness $y_2 := t_2(x, y_1)$ and sends it to the Teacher, and so on.

The validity of the disjunction in (1) implies that the Student will find a valid witness in at most k rounds. In order to guarantee that the total time of the Student is polynomial in the length of x one needs to assume that the universal quantifier is bounded: $\forall z < s(x)$.

An interesting statement of this logical form is the following maximization principle. Let $R(x, y)$ be a polynomial time relation and assume

$$\forall x R(x, 0) \ \wedge \ \forall x, y \ (R(x, y) \rightarrow |y| \leq |x|) \ .$$

The principle says that for any x there is a solution y of maximal size:

$$\forall x \exists y \forall z \ [\ R(x, y) \wedge (R(x, z) \rightarrow |z| \leq |y|) \ ] \ .$$

For example, the principle implies the existence of a maximal clique in a graph. Krajíček, Pudlák and Takeuti [22] proved that there are polynomial time relations R for which the principle is not provable in T unless NP $\subseteq$ P/poly.

1.3 Universal formulas

Assume

$$T \ \vdash \ A(x)$$

where A is an open formula. Herbrand's theorem implies that there are finitely many axioms $\forall y_1, \ldots, y_k \ B(y_1, \ldots, y_k)$ of T and for each of them finitely many k-tuples of terms $t_1(x), \ldots, t_k(x)$ such that $A(x)$ is provable already from all these finitely many instances

$$B(t_1, \ldots, t_k) \ .$$

To avoid an excessive notation we shall assume that instances of only one axiom $\forall y B(y)$ are used and that the open formula $B(y)$ has only one free variable y. Hence for some terms $s_1(x), \ldots, s_k(x)$

$$B(s_1) \wedge \ldots \wedge B(s_k) \ \vdash \ A(x) \ . \tag{2}$$

Formula $A(x)$ defines a polynomial time predicate and hence is computable by a (uniform) family of polynomial size circuits that we shall denote $||A(x)||^n$, and similarly $||B(s_i(x))||^n$ for the other formulas.

What we want to derive is a propositional implication

$$\bigwedge_i \ ||B(s_i(x))||^n \ \rightarrow \ ||A(x)||^n \ . \tag{3}$$

The qualification 'propositional' is not quite true as we use circuits rather than formulas. But one can either define a propositional calculus operating directly with circuits (cf. [10]) or replace circuits by formulas written using auxiliary variables used to define the circuit computations, as in the proof of the NP-completeness of SAT.

Implication (3) need not to be, in fact, a tautology as in the first-order derivation of (2) some equality axioms could have been used. Hence there is a finite set $\mathcal{E}$ of instances of equality axioms by terms in x such that

$$\bigwedge_{E\in\mathcal{E}} ||E||^n \wedge \bigwedge_i ||B(s_i(x))||^n \ \rightarrow\ ||A(x)||^n \ . \tag{4}$$

Moreover, (4) is now valid as a propositional formula. This means that if we interpret atomic formulas as propositional atoms then the implication is valid under all truth assignments. Hence there is a template propositional derivation (in any proof system containing at least resolution) and proofs of (4) for specific $n \geq 1$ are obtained by replacing in the whole template derivation propositional atoms by translations $||\dots||^n$ of the atomic formulas from (2) (and of the equality axioms from $\mathcal{E}$).

The interesting thing on which a correspondence between theories and propositional proof systems rests is that for a subtheory S of T whose set of axioms is in NP one can define a proof system P_S that admits p-bounded proofs of all $||\dots||^n$-translations of term instances of all axioms of S (and of all equality axioms), and hence also of all $||\dots||^n$-translations of the universal consequences of S. Moreover, for natural S such a proof system is often defined quite naturally as well. For example, for Cook's PV [7] the corresponding proof system is the Extended Frege system (or, equivalently) the Extended Resolution. See [18, 13] for details.

2 Theories of bounded induction

Let us recall first the concept of NP search problems.

2.1 NP search problems

A NP search problem is determined by a polynomial time relation $R(x, y)$ such that it holds

$$\forall x \exists y R(x,y) \ \wedge \ \forall x, y \ (R(x,y) \rightarrow |y| \leq |x|^{O(1)}) \ .$$

The task is: given x find a solution y such that $R(x, y)$. Note that the solution needs not to be unique, i.e. R is not necessarily a graph of a function.

The relative complexity of two NP search problems $R(x, y)$ and $S(u, v)$ can be compared by a *p-reduction.* A p-reduction of R to S is a pair of polynomial time functions $f(x)$ and $g(x, v)$ such that

$$S(f(x), v) \ \rightarrow\ R(x, g(x, v)) \ .$$

The interpretation of the reduction is this: if we want to find a solution y for x in R we take a solution v for $f(x)$ in S and compute from it $y := g(x, v)$.

There is a variety of classes of NP search problems that are p-equivalent (mutually p-reducible), often characterized by a problem with a transparent combinatorial meaning. A prominent example is the class PLS (polynomial local search), cf.[4, 24]. It is given by a polynomial time relation $S(x, y)$ defining the set of possible solutions $\{y \mid S(x, y)\}$ for x (we assume it contains 0 and the size of any possible solution is polynomially bounded in $|x|$), an integer valued cost function $c(x, y)$ and a neighborhood function $N(x, y)$ such that

$$S(x, y) \ \rightarrow \ [\ S(x, N(x, y)) \wedge c(x, N(x, y)) \leq c(x, y)\ .\ ]$$

Both functions $c(x, y)$ and $N(x, y)$ are polynomial time. The task is to find a locally optimal solution, i.e. a string y such that

$$S(x, y) \ \wedge c(x, N(x, y)) = c(x, y)\ .$$

2.2 NP induction

Extend T by adding as new axioms formulas expressing induction

$$[\ B(0) \ \wedge \ \forall i < x\ (B(i) \rightarrow B(i+1))\ ] \ \rightarrow \ B(x) \tag{5}$$

for all bounded existential formulas (the so called E_1-formulas) of the form

$$B(x) \ := \ \exists y < s(x)\ C(x, y)$$

with C an open formula. Formula C can have other free variables (parameters). Every E_1-formula defines an NP predicate and vice versa, every NP predicate can be defined by such a formula. We shall call this extension of T as $NP{-}IND$.

Assume

$$NP - IND \ \vdash \ \exists y A(x, y) \tag{6}$$

A open. The witnessing can be analyzed using the Herbrand's theorem from Subsection 1.2 although other methods are available (cf.[3, 13]). We shall explain the idea on a special case for which the easier form of Herbrand's theorem from Subsection 1.1 suffices.

Assume that the formula in (6) is provable from T using just one induction axiom up to $t(x)$

$$[\ B(0) \ \wedge \ \forall i < t(x)\ (B(i) \rightarrow B(i+1))\ ] \ \rightarrow \ B(t(x))$$

for a formula B without any parameters (this is what allows us to use the simpler version of Herbrand's theorem). Assume also that $C(0, 0)$ is valid, and that $C(x, y)$ implies $y < s(x)$ to ease the notation. Then (using the deduction lemma) T proves, in particular, the formula

$$[\ (i < t(x) \wedge C(i, z_i)) \rightarrow \exists z_{i+1}\ C(i+1, z_{i+1})\ ] \ \vee \ \exists y\ A(x, y)\ . \tag{7}$$

and

$$\forall z \neg C(t(x), z) \ \vee \ \exists y\ A(x, y)\ . \tag{8}$$

Using the witnessing from Subsection 1.1 to (7) we get a polynomial time algorithm $f(x, i, z_i)$ that either finds a witness y for $A(x, y)$ or a witness z_{i+1} for $C(i + 1, z_{i+1})$. Hence if we start from the triple $(x, 0, 0)$ and iterate f we either find suitable y or a witness z to $C(t(x), z)$. But in the latter case we apply a witnessing function for (8) to find y.

From this process we do not get a polynomial time witnessing algorithm. But it shows that the NP search problem defined by A is p-reducible to a PLS problem: interpret f as a neighborhood function and $t(x) - i$ as a cost function on a suitable set of possible solution (the triples above). See Buss and Krajíček [4] for details (and another proof).

2.3 NP length induction

One may extend T by a set of formulas expressing a weaker form of induction, the so called length induction (after Buss [3]):

$$[\ B(0)\ \wedge\ \forall i < |x|\ (B(i) \rightarrow B(i+1))\ ]\ \rightarrow\ B(|x|)$$

again for all E_1-formulas, and denote the resulting theory $NP - LIND$.

By Krajíček, Pudlák and Takeuti [22] the theory $NP - LIND$ is stronger than T unless NP $\subseteq$ P/poly. On the other hand Buss [3] has proved that $NP - LIND$ is conservative over T w.r.t. existential formulas. Hence existential formulas provable in $NP - LIND$ can be witnessed by polynomial time algorithms.

2.4 Witnessing E_2-formulas in NP-IND and NP-LIND

E_2-formulas are formulas of the form $\exists y < s(x) \forall z < t(x, y)\ A(x, y, z)$. E_2-formulas provable in $NP-IND$ can be witnessed by a polynomial time algorithm with an access to an NP oracle (cf.[3]). If an E_2-formula is proved using NP-LIND only it can be witnessed by a polynomial time algorithm which queries an NP oracle but only $(O(\log n)$ times (cf.[12]). It may be interesting to point out that the predicates in this class (i.e. 0/1-functions) are precisely those decidable in logarithmic space with an access to an NP oracle.

2.5 Induction for predicates in PH

Predicates in Polynomial Hierarchy (PH) are defined by bounded formulas (no restriction on the quantifier complexity). Theory PH-IND augments T by adding the induction axiom for all bounded formulas. By [3] the same theory would be obtained if we added only PH-LIND: an IND axiom for a bounded formula with k quantifiers can be deduced from a LIND axiom for a formula with $k + 1$ quantifiers.

Analysis of witnessing of existential formulas provable in PH-IND is done using various classes of NP search problems. The original characterization of [19] used NP search problems related to the soundness of fragments of quantified propositional calculus. These characterizations have been greatly simplified and their combinatorial structure has been made more transparent using NP search problems defined in terms of games, cf. [26, 23, 27, 28].

3 Propositional proof systems

A proof system for propositional logic (see Subsection 1.1) is p-bounded if there is a constant $c \geq 1$ such that any tautology τ has a P-proof of size at most $|\tau|^c$. Cook and Reckhow [9] noted that a p-bounded proof system exists iff NP is closed under complementation. It is thus expected that no such proof system exists and to prove this is the fundamental problem of proof complexity (cf. [15]).

Given a proof system P we would like to find an explicit infinite family of tautologies whose P-proofs cannot be polynomially bounded (such tautologies are called informally hard). For many proof systems explicit examples of hard tautologies are known but all these proof systems are weaker than the usual text-book propositional calculus based on a finite number of axioms schemes and inference rules (a Frege system in the terminology of [9]).

It is a very interesting problem to find such hard tautologies for Frege systems or even for stronger systems. In demonstrating the hardness of the formulas one would not hesitate to use a plausible hypothesis from the computational world, e.g. a hypothesis of the form that every circuit performing a specific task must be large. Several other fundamental problems of complexity theory were reduced to a hypothesis of this form. Examples include the conjectures that the classes P and NP differ, that a universal derandomization is possible or that a cryptographically strong pseudo-random generator exists (see [14] for a discussion).

3.1 Feasible interpolation

Let $\alpha_n(x, y)$ and $\beta_n(x, z)$ be size $n^{O(1)}$ propositional formulas having common variables $x = (x_1, \ldots, x_n)$. Consider disjunctions

$$\neg\alpha_n(x, y) \vee \neg\beta_n(x, z) \ . \tag{9}$$

These disjunctions express the disjointness of sets

$$U \ := \ \bigcup_n \{x \in \{0,1\}^n \mid \exists y \alpha_n(x, y)\}$$

and

$$V \ := \ \bigcup_n \{x \in \{0,1\}^n \mid \exists z \beta_n(x, z)\} \ .$$

These sets are in NP/poly and in NP if the formulas are defined uniformly in n.

The idea of feasible interpolation is that having short proofs of the disjunctions (9) in a proof system P it ought to be possible to separate sets U and V by some feasible algorithm. The intended use is in the opposite direction: having a pair of disjoint NP sets hard to separate (such pairs are conjectured to exists in cryptography) allows to define formulas (the disjunctions above) hard for P.

This lower bound method is quite successful and applies to the most varied class of proof systems among all lower bound methods. Proof systems admitting some form of feasible interpolation include resolution, cutting planes proof system, algebraic and geometric proof systems, or the OBDD proof system. Most recent references (as well as some history) can be found in [16, 25].

Feasible interpolation does not, however, work for strong proof systems. For example, assume $h(x)$ is a one-way permutation and $b(x)$ is its hard bit (cf. [1]). Assume in addition that there are polynomial size P-proofs of formulas expressing that h is injective; using the notation from Subsection 1.3 the formulas are

$$||x \neq y \to h(x) \neq h(y)||^n \ .$$

In such a case it is easy to see that the disjunctions expressing the disjointness of sets

$$U_i \ := \ \{y \in \{0,1\}^n \mid \exists x (h(x) = y \wedge b(x) = i\}, \text{ for } \ i = 0, 1$$

have short P-proofs as well. But an algorithm that would separate these sets would at the same time compute the hard bit, and that is (conjectured to be) impossible. This applies with $h = RSA$ to Extended Frege systems by [20], and there are similar constructions also for some weaker systems (cf.[2]).

3.2 Feasible disjunction property

A proof of feasible interpolation for a proof system P usually establishes a stronger property: There exists an algorithm that upon receiving a P-proof of a disjunction $\alpha \vee \beta$ of two formulas in disjoint sets of variables finds a P-proof of one of them.

Let us point out a property that can be assumed (for the purpose of proving lengths-of-proofs lower bounds) to hold for all proof systems. Following [17] we call it the *feasible disjunction property* (fdp).

A proof system P has fdp if there exists a constant $c \geq 1$ such that whenever a disjunction

$$\alpha_1 \vee \ldots \vee \alpha_k \qquad (10)$$

of k formulas with no two having a variable in common has a P-proof of size s then one of α_i has a P-proof of size at most s^c.

A simple observation is that a proof system P that does not have fdp cannot be p-bounded. Assume for a simplicity that a P-proof of a formula is at least as long as the formula. Then all formulas α_i in (10) have size at most s, at least one of them must be a tautology but it does not have p-bounded proofs.

The feasible disjunction property appeared in a connection with the so called proof complexity generators. These are propositional tautologies of a certain specific form and they are proposed as candidate hard formulas for strong proof systems. The analysis of their hardness is quite related to various forms of witnessing theorems as above. It is consistent with the present knowledge that although feasible interpolation does not apply to strong proof systems, some of its features can be salvaged even in strong systems for these specific formulas, enough to deduce lengths-of-proofs lower bounds (see [17] for a background and references).

Acknowledgements. I thank N. Thapen (Prague) for comments on the draft of this paper.

References

1. S. Arora and B. Barak, *Computational Complexity: A Modern Approach*, Cambridge University Press, (2009).
2. M. L. Bonet, T. Pitassi, and R. Raz, On Interpolation and Automatization for Frege Proof Systems, *SIAM J. of Computing*, **29(6)**, (2000), pp.1939-1967.
3. S. R. Buss, *Bounded Arithmetic*. Naples, Bibliopolis, (1986).
4. S. R. Buss, and J. Krajíček, An application of boolean complexity to separation problems in bounded arithmetic, *Proceedings of the London Mathematical Society*, **69(3)**, (1994), pp. 1-21.
5. P. Clote and E. Kranakis, *Boolean Functions and Models of Computation*, Springer-Verlag, 2002.
6. A. Cobham, The intrinsic computational difficulty of functions, in : *Proc. Logic, Methodology and Philosophy of Science*, ed. Y. Bar-Hillel, North-Holland, (1965), pp. 24-30.
7. S. A. Cook, Feasibly constructive proofs and the propositional calculus, in: *Proc. 7th Annual ACM Symp. on Theory of Computing*, (1975), pp. 83-97. ACM Press.
8. S. A. Cook, and P. Nguyen, *Logical foundations of proof complexity*, Cambridge U. Press, (2009).
9. S. A. Cook, and Reckhow, The relative efficiency of propositional proof systems, *J. Symbolic Logic*, **44(1)**, (1979), pp.36-50.
10. E. Jeřábek, Dual weak pigeonhole principle, Boolean complexity, and derandomization, *Annals of Pure and Applied Logic*, **129**, (2004), pp.137.
11. L. Kolodziejczyk, P. Nguyen and N. Thpen, The provably total NP search problems of weak second order bounded arithmetic, preprint 2009.
12. J. Krajíček, Fragments of bounded arithmetic and bounded query classes, *Transactions of the A.M.S.*, **338(2)**, (1993), pp.587-598.
13. J. Krajíček, *Bounded arithmetic, propositional logic, and complexity theory*, Encyclopedia of Mathematics and Its Applications, Vol. **60**, Cambridge University Press, (1995).
14. J. Krajíček, Hardness assumptions in the foundations of theoretical computer science, *Archive for Mathematical Logic*, **44(6)**, (2005), pp.667-675.
15. J. Krajíček, Proof complexity, in: Laptev, A. (ed.), European congress of mathematics (ECM), Stockholm, Sweden, June 27–July 2, 2004. Zurich: European Mathematical Society, (2005), pp.221-231.
16. J. Krajíček, A form of feasible interpolation for constant depth Frege systems, *J. of Symbolic Logic*, **75(2)**, (2010), pp.774-784.
17. J. Krajíček, On the proof complexity of the Nisan-Wigderson generator based on a hard NP $\cap$ coNP function, submitted (preprint March 2010). Preliminary version in *Electronic Colloquium on Computational Complexity*, Rep. No.**54**, (2010).

18. 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.
19. J. Krajíček and P. Pudlák, Quantified Propositional Calculi and Fragments of Bounded Arithmetic, *Zeitschr. f. Mathematikal Logik u. Grundlagen d. Mathematik*, Bd. **36(1)**, (1990), pp. 29-46.
20. J. Krajíček and P. Pudlák, Some consequences of cryptographical conjectures for S_2^1 and EF", *Information and Computation*, Vol. **140 (1)**, (January 10, 1998), pp.82-94.
21. J. Krajíček, P. Pudlák, and J. Sgall, Interactive Computations of Optimal Solutions, in: B. Rovan (ed.): *Mathematical Foundations of Computer Science* (B. Bystrica, August '90), Lecture Notes in Computer Science **452**, Springer-Verlag, (1990), pp. 48-60.
22. J. Krajíček, P. Pudlák and G. Takeuti, Bounded arithmetic and the polynomial hierarchy, *Annals of Pure and Applied Logic*, **52**, (1991), pp.143–153.
23. J. Krajíček, A. Skelley and N. Thapen, NP search problems in low fragments of bounded arithmetic, *J. of Symbolic Logic*, **72:2**, (2007), pp.649-672.
24. C. Papadimitriou, The Complexity of the Parity Argument and Other Inefficient proofs of Existence *J. of Computer and System Sciences*, **48(3)**, (1994), pp.498-532.
25. P. Pudlák, The lengths of proofs, in: Handbook of Proof Theory, S.R. Buss ed., Elsevier, (1998), pp.547-637.
26. P. Pudlák, Consistency and games - in search of new combinatorial principles, in: Proc. Logic Colloquium'03, Helsinki, eds. V. Stoltenberg-Hansen and J. Vaananen, Assoc. for Symbolic Logic, (2006), pp.244-281.
27. P. Pudlák, Fragments of Bounded Arithmetic and the lengths of proofs, *J. of Symbolic Logic*, **73(4)**, (2008), pp.1389-1406.
28. A. Skelley and N. Thapen, The provably total search problems of bounded arithmetic, (preprint 2007, revised March 2010).

This article was processed using the LaTeX macro package with LLNCS style