

Boolean Programs and Quantified Propositional Proof Systems

To appear in the Bulletin of the Section of Logic

Stephen Cook and Michael Soltys

A propositional proof is a derivation of a propositional tautology (e.g. $\neg P \vee P$) using a set of rules.

A nice example of a propositional proof system is Gentzen's PK system, where we have sequents of propositional formulas of the form:

$$A_1, \dots, A_n \rightarrow B_1, \dots, B_m \quad (*)$$

and the semantics of sequents is given as follows. We say that a truth assignment τ satisfies the sequent (*) iff either τ falsifies some A_i or τ satisfies some B_i .

For example, if τ is given by $P = 0$, $Q = 1$, $R = 0$ then τ satisfies

$$P \vee Q, \neg R \rightarrow Q \wedge \neg R$$

A sequent is equivalent to the formula

$$(A_1 \wedge \cdots \wedge A_n) \supset (B_1 \vee \cdots \vee B_m)$$

In other words, the conjunction of the A_i 's implies the disjunction of the B_i 's.

We say that a sequent is valid if it is true under all truth assignments. For example, the following are valid sequents:

$$A \rightarrow A$$

$$\rightarrow A, \neg A$$

$$A, \neg A \rightarrow$$

$$\rightarrow A \vee \neg A$$

$$A, A \supset B \rightarrow B$$

A PK proof is a sequence of sequents, where all the sequents are derived from previous sequents or are logical axioms of the form $A \rightarrow A$, $\rightarrow 1$ or $0 \rightarrow$.

PK Rules of inference:

weak	$\frac{\Gamma \rightarrow \Delta}{A, \Gamma \rightarrow \Delta}$	$\frac{\Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta, A}$
exch	$\frac{\Gamma_1, A, B, \Gamma_2 \rightarrow \Delta}{\Gamma_1, B, A, \Gamma_2 \rightarrow \Delta}$	$\frac{\Gamma \rightarrow \Delta_1, A, B, \Delta_2}{\Gamma \rightarrow \Delta_1, B, A, \Delta_2}$
cont	$\frac{\Gamma, A, A \rightarrow \Delta}{\Gamma, A \rightarrow \Delta}$	$\frac{\Gamma \rightarrow \Delta, A, A}{\Gamma \rightarrow \Delta, A}$
“¬”	$\frac{\Gamma \rightarrow \Delta, A}{\neg A, \Gamma \rightarrow \Delta}$	$\frac{A, \Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta, \neg A}$
“∧”	$\frac{A, B, \Gamma \rightarrow \Delta}{(A \wedge B), \Gamma \rightarrow \Delta}$	$\frac{\Gamma \rightarrow \Delta, A \quad \Gamma \rightarrow \Delta, B}{\Gamma \rightarrow \Delta, (A \wedge B)}$
“∨”	$\frac{A, \Gamma \rightarrow \Delta \quad B, \Gamma \rightarrow \Delta}{(A \vee B), \Gamma \rightarrow \Delta}$	$\frac{\Gamma \rightarrow \Delta, A, B}{\Gamma \rightarrow \Delta, (A \vee B)}$
cut	$\frac{\Gamma \rightarrow \Delta, A \quad A, \Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta}$	

Any valid sequent can be derived from initial sequents of the form $A \rightarrow A$, $\rightarrow 1$ and $0 \rightarrow$, using the above rules (*completeness* of PK), and only valid sequents can be derived (*soundness* of PK).

As an example we give a proof of one of DeMorgan's laws:

$$\neg(P \wedge Q) \rightarrow \neg P \vee \neg Q$$

Here is the derivation:

$$\frac{\frac{\frac{P \rightarrow P}{\rightarrow P, \neg P}}{\rightarrow P, \neg P, \neg Q}}{\frac{\frac{\frac{Q \rightarrow Q}{Q \rightarrow Q, \neg P}}{\rightarrow Q, \neg P, \neg Q}}{\rightarrow P \wedge Q, \neg P, \neg Q}}}{\frac{\rightarrow P \wedge Q, \neg P \vee \neg Q}}{\neg(P \wedge Q) \rightarrow \neg P \vee \neg Q}}$$

Abstract Definition: A proof system is a feasible (i.e. easily computable) map

$$f : \{0, 1\}^* \xrightarrow{\text{onto}} \{\text{tautologies}\}$$

Where $\{\text{tautologies}\}$ is just the set of propositional tautologies, and if $f(x) = A$ then x is the proof of the propositional tautology A .

A proof system is polynomially bounded if there exists a polynomial p such that for every $A \in \{\text{tautologies}\}$, there exists an x such that $|x| \leq p(|A|)$, i.e. every tautology has a proof which is “proportional” to its size.

See “The Complexity of Propositional Proofs”, by Alasdair Urquhart, in The Bulletin of Symbolic Logic, December 1995, for all the details.

Interest in propositional proof complexity arose from two fields connected with computers: automated theorem proving (artificial intelligence) and computational complexity theory.

Some Complexity Theory

P is the class of problems that can be solved with algorithms that run in polynomial time in the size of the input (e.g. given a truth assignment τ , and a sequent S , does τ satisfy S ?).

NP is the class of problems whose solutions can be verified by an algorithm that runs in polynomial time (e.g. given a sequent S , is it satisfiable?)

co-NP is the class of problems for which a counter-example can be verified by an algorithm that runs in polynomial time (e.g. given a sequent S , is it valid?).

Theorem: $NP = co-NP$ iff there is a polynomially bounded proof system for the propositional tautologies.

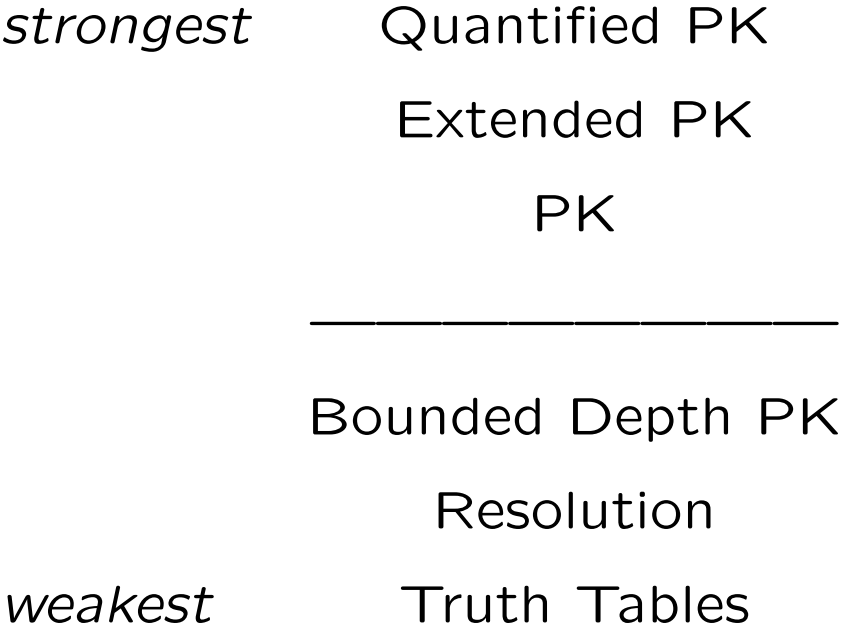
Also $P = NP$ implies $NP = co-NP$, thus if we prove that there is no polynomially bounded proof system (which is what experts conjecture), then it will follow that $NP \neq co-NP$, and therefore $P \neq NP$.

The problem

$$P \stackrel{?}{=} NP$$

has been listed by Steve Smale in the top three “Mathematical Problems for the Next Century” (Math Intelligencer 20, 1998).

How could we prove that there is no polynomially bounded proof system? It seems like a very difficult problem. The standard approach is to consider stronger and stronger proof systems, and to exhibit superpolynomial lower bounds for each of them.



However, we can only prove superpolynomial lower bounds for the weaker systems (the ones below the line). The systems above the line have no known lower bounds.

Quantified PK

Quantified propositional PK is formed by introducing propositional quantifiers:

$\forall x A(x)$ whose meaning is $A(0) \wedge A(1)$

$\exists x A(x)$ whose meaning is $A(0) \vee A(1)$

The propositional quantifiers do not increase the expressive power of formulas. Instead, they allow us to shorten some propositional formulas. For example, the formula

$$\bigvee_{(x_1, \dots, x_n) \in \{0,1\}^n} A(x_1, \dots, x_n)$$

has size in the order of $2^n |A|$, but the equivalent quantified formula

$$\exists x_1 \dots \exists x_n A(x_1, \dots, x_n)$$

has size in the order of $n + |A|$.

The quantified propositional calculus is first discussed by B. Russell in “The Theory of Implication”, American Journal of Mathematics in 1906. Also, the Polish mathematician Leśniewski treated it in his “Protothetic” .

Martin Dowd investigated the connection between quantified propositional proof systems and the complexity class PSPACE in his PhD thesis in 1979.

A modest source of information is also a 5 page section in Jan Krajíček’s book “Bounded Arithmetic, Propositional Logic, and Complexity Theory”, 1995.

Still, little is known about quantified propositional proof systems.

G is PK with \exists and \forall

I follow the results and notation from Jan Krajíček's "Bounded Arithmetic, Propositional Logic, and Complexity Theory".

The quantified propositional proof system G extends LK by allowing quantified propositional formulas in sequents, and by adding the following quantifier rules:

$$\frac{A(B), \Gamma \rightarrow \Delta}{\forall x A(x), \Gamma \rightarrow \Delta} \quad \frac{\Gamma \rightarrow \Delta, A(p)}{\Gamma \rightarrow \Delta, \forall x A(x)}$$
$$\frac{A(p), \Gamma \rightarrow \Delta}{\exists x A(x), \Gamma \rightarrow \Delta} \quad \frac{\Gamma \rightarrow \Delta, A(B)}{\Gamma \rightarrow \Delta, \exists x A(x)}$$

where B is any formula, and with the restriction that the atom p does not occur in the lower sequents of \forall -right and \exists -left.

Let G_1 be the restriction of G where we allow only strict Σ_1^q formulas, that is, all formulas must begin with a block of existential quantifiers (possibly empty) followed by a quantifier free formula. In other words, all formulas are either quantifier free, or of the form

$$\exists x_1 \dots \exists x_n A(x_1, \dots, x_n)$$

where $A(x_1, \dots, x_n)$ is quantifier free.

For example, the sequent

$$\exists x \exists y ((\neg x \vee p) \wedge (\neg y \vee q)) \rightarrow \exists x (x \vee \neg p \vee q), p \wedge q$$

has only Σ_1^q formulas.

Witnessing

Given a sequent S , we want to have a way of computing the values of the existential variables on the right in terms of the free variables, and the quantified variables on the left. For example, consider the sequent

$$\exists y A(x, y) \rightarrow \exists y A(x, y) \quad (*)$$

where x, y are the only variables in this sequent. In this case it is very easy. The quantified variable y on the right can be witnessed by the following function:

$$f(x, y) = y$$

that is, the sequent

$$A(x, y) \rightarrow A(x, f(x, y))$$

is valid if $(*)$ was valid.

Suppose that π is a G_1 proof. We show that the existential quantifiers can be witnessed by *Boolean programs*. We do this by induction on the number of sequents in π .

For example, consider the contraction rule:

$$\frac{C(\bar{p}) \rightarrow D(\bar{p}), \exists x B(x), \exists x B(x)}{C(\bar{p}) \rightarrow D(\bar{p}), \exists x B(x)}$$

By Induction Hypothesis, we know that there are Boolean programs g, h such that the sequent

$$C(\bar{p}) \rightarrow D(\bar{p}), B(g(\bar{p})), B(h(\bar{p}))$$

is valid. Now, define f as follows:

$$f(\bar{p}) = \begin{cases} g(\bar{p}) & \text{if } B(g(\bar{p})) \\ h(\bar{p}) & \text{otherwise} \end{cases}$$

The bottom sequent can thus be converted to the valid sequent

$$C(\bar{p}) \rightarrow D(\bar{p}), B(f(\bar{p})),$$

Main Idea

If π is a G_1 proof of a propositional tautology, and π is tree-like, then the existential quantifiers can be feasibly computed by evaluating Boolean circuits. This was already shown in Jan Krajíček's book "Bounded Arithmetic, Propositional Logic, and Complexity Theory".

If, on the other hand, π is not tree-like, then it seems that evaluating the existential quantifiers requires a lot of computational power, i.e. Boolean programs whose evaluation is a PSPACE complete problem. This is shown in our paper "Boolean Programs and Quantified Propositional Proof Systems".

References

“Boolean Programs and Quantified Propositional Proof Systems”, by S.Cook and M. Soltys, in this paper we introduce the notion of Boolean programs, and we characterize PSPACE in terms of families of Boolean programs. We then show how to use Boolean programs to witness quantifiers in G_1 .

“The Complexity of Propositional Proofs”, by Alasdair Urquhart, in The Bulletin of Symbolic Logic, December 1995, a comprehensive introduction to the complexity of propositional proofs.

“Bounded Arithmetic, Propositional Logic, and Complexity Theory”, by Jan Krajíček. A short introduction to quantified propositional proof systems.