II. Introduction to Bounded Arithmetic and Witnessing

Sam Buss, UCSD
sbuss@math.ucsd.edu

Prague, September 2009

Bounded arithmetic and bounded quantifiers

Language of first-order theory of bounded includes:

$$0,\ S,\ +,\ \cdot,\ \leq,\ |x| := \lceil \log_2(x+1) \rceil,\ \lfloor \tfrac{1}{2}x \rfloor,\ x \# y := 2^{|x|\cdot|y|}.$$

Sometimes also add all polynomial time functions and relations.

Axioms can include (among others):

(a) Defining (equational) axioms for functions and relations, "BASIC".
(b) Restricted forms of induction.

Definition

A *bounded* quantifier is of the form $(\forall x \le t)$ or $(\exists x \le t)$. It is *sharply bounded* provided t has the form $|s|$. A formula is *bounded* or *sharply bounded* provided all its quantifiers are bounded or sharply bounded (resp.).

Definition

$\Delta_0^b = \Sigma_0^b = \Pi_0^b$: Sharply bounded formulas

Σ_{i+1}^b: Closure of Π_i^b under existential bounded quantification and arbitrary sharply bounded quantification, modulo prenex operations.

Π_{i+1}^b is defined dually.

Σ_i^b, Π_i^b define exactly the predicates at the i-th level of the polynomial hierarchy (PH), if $i \ge 1$.
Thus, Σ_1^b and Π_1^b define exactly the NP and coNP sets.

Induction axioms

Let formulas A be in Ψ, we have the following kinds of induction:

Ψ-IND: $$A(0) \land (\forall x)(A(x) \rightarrow A(x+1)) \rightarrow (\forall x)A(x).$$

Ψ-PIND: $$A(0) \land (\forall x)(A(\lfloor \tfrac{1}{2}x \rfloor) \rightarrow A(x)) \rightarrow (\forall x)A(x).$$

Ψ-LIND: $$A(0) \land (\forall x)(A(x) \rightarrow A(x+1)) \rightarrow (\forall x)A(|x|).$$

Definition (Fragments of bounded arithmetic, B'85)

S_2^i: $\textsc{Basic} + \Sigma_i^b$-PIND.
T_2^i: $\textsc{Basic} + \Sigma_i^b$-IND.

$S_2 = \cup_i S_2^i$ and $T_2 = \cup_i T_2^i$.

Note T_2 is essentially $I\Delta_0 + \Omega_1$. [Parikh'71, Wilkie-Paris'87]

Theorem (B'85, B'90)

(a) $S^1_2 \subseteq T^1_2 \preccurlyeq_{\forall\Sigma^b_2} S^2_2 \subseteq T^2_2 \preccurlyeq_{\forall\Sigma^b_3} S^3_2 \subseteq \cdots$

(b) *Thus,* $S_2 = T_2$.

(c) $S^1_2 + \Sigma^b_i$*-LIND equals* S^i_2.

More axioms:

Φ-MIN: $(\exists x)A(x) \rightarrow (\exists x)(A(x) \wedge (\forall y < x)\neg A(y))$.

Φ-LMIN: $(\exists x)A(x) \rightarrow (\exists x)(A(x) \wedge (\forall y)(|y| < |x| \rightarrow \neg A(y)))$.

Φ-replacement:

$$(\forall x \le |t|)(\exists y \le s)A(x,y) \rightarrow (\exists w)(\forall x \le |t|)A(x, \beta(x,w)).$$

Φ-strong replacement:

$$(\exists w)(\forall x \le |t|)[(\exists y \le s)A(x,y) \leftrightarrow A(x, \beta(x,w))].$$

$$\begin{array}{l}
\Sigma^b_i\text{-IND} \iff \Pi^b_i\text{-IND} \iff \Sigma^b_i\text{-MIN} \iff \Pi^b_{i-1}\text{-MIN} \iff \Delta^b_{i+1}\text{-IND} \\
\qquad \Downarrow \\
\Sigma^b_i\text{-PIND} \iff \Pi^b_i\text{-PIND} \iff \Sigma^b_i\text{-LIND} \iff \Pi^b_i\text{-LIND} \\
\qquad \Updownarrow \\
\Sigma^b_i\text{-LMIN} \iff (\Sigma^b_{i+1} \cap \Pi^b_{i+1})\text{-PIND} \iff_1 \text{strong } \Sigma^b_i\text{-replacement} \\
\qquad \Downarrow \\
\Sigma^b_{i-1}\text{-IND}
\end{array}$$

$$S^i_2 \preccurlyeq_{\forall \Sigma^b_i} T^{i-1}_2 \qquad\qquad S^i_2 \preccurlyeq_{\forall \mathcal{B}(\Sigma^b_i)} T^{i-1}_2 + \Sigma^b_i\text{-replacement}$$

$$\Sigma^b_1\text{-PIND} + \Sigma^b_{i+1}\text{-replacement} \implies \Sigma^b_i\text{-PIND} \implies \Sigma^b_i\text{-replacement}$$

Open: The exact relative strength of Σ^b_i-replacement.

Provably total functions and Σ_i^b-definable functions

Definition

Let R be a bounded theory. A function $f : \mathbb{N} \to \mathbb{N}$ is *provably total* in R provided there is a formula $A_f(x, y)$ that defines the graph of f such that R proves $(\forall x)(\exists! y)A_f(x, y)$, with A_f polynomial time computable.

Definition

f is Σ_i^b-definable by R, provided there is a Σ_i^b-formula $A(x, y)$ such that

1. $R \vdash (\forall x)(\exists y \leq t)A(x, y)$ for some term t.
2. $R \vdash (\forall x)(A(x, y) \wedge A(x, z) \to y = z)$.
3. $A(x, y)$ defines the graph of f.

Thm. Any Σ_1^b-definable function in S_2^i or T_2^i can be introduced conservatively into the language of the theory with its defining axiom, and be used freely in induction formulas.

Theorem (B'85)

1. *S_2^1 can Σ_1^b-define every polynomial time function.*
2. *S_2^i can Σ_i^b-define every function which is polynomial time computable with an oracle from Σ_{i-1}^p.*

(The converse holds too.)

Hence, we can w.l.o.g. assume that all polynomial time functions are present in the language of bounded arithmetic.

Similar definitions and results hold for predicates.

Definition

A predicate P is Δ_i^b-definable in R provided there are a Σ_i^b-formula A and Π_i^b-formula B which are R-provably equivalent and which define the predicate P.

Theorem (B'85)

Every polynomial time predicate is Δ_1^b-definable by S_2^1.
Every predicate which is polynomial time computable with an oracle from Σ_{i-1}^b is Δ_i^b-definable in S_2^i.

(Again, a converse holds.)

Thus, every polynomial time predicate can be conservatively introduced to S_2^i or T_2^i with its defining axioms, and used freely in induction axioms.

Witnessing Theorem for S_2^i

Theorem (Main Theorem for S_2^i, B'85)

Let $i \geq 1$. Suppose f is Σ_i^b-defined by S_2^i. Then f is computable in $P^{\Sigma_{i-1}^p}$, that is, in polynomial time with an oracle for Σ_{i-1}^p.
For $i = 1$, f is in P, polynomial time computable.

This gives an exact characterization of the functions that are Σ_i^b-definable in S_2^i.

For $i = 1$, the Σ_1^b-definable functions of S_2^1 are precisely the polynomial computable functions.
Likewise, the Δ_1^b-definable predicates of S_2^1 are precisely the predicates that are provably in $\mathrm{NP} \cap \mathrm{coNP}$.

Open: Give a more satisfactory account of the functions that are Σ_1^b-definable in S_2^i, $i > 1$. That is, of the provably total functions of these theories. (Note the uniqueness condition.)

We now start the proof of the Main Theorem.

Proof idea: Form a free-cut free proof, in which all formulas are in Σ_i^b. The free-cut free proof is then essentially an algorithm for the function f.

The proof is considerably simplified by working with *strict* Σ_i^b*-formulas*, denoted $s\Sigma_i^b$ for short. These are of the form:

$$(\exists x_1 \le t_1)(\forall x_2 \le t_2) \cdots (Qx_i \le t_i) B(\vec{x})$$

where B is sharply bounded, and the quantifiers alternate in type (and subformulas of these formulas).

Thm. S_2^i can equivalently be formulated with $s\Sigma_i^b$-PIND, provided $\dot{-}$ and MSP are added to the language.

Proof idea: Careful bootstrapping, plus use of replacement.

To prove the witnessing theorem, by free-cut elimination, it suffices to consider sequent calculus proofs in which every formula is an $s\Sigma_i^b$-formula (including, via pairing functions, the final, proved formula). Henceforth, fix $i > 0$.

Definition

Let $A(\vec{c})$ be $s\Sigma_i^b$. The predicate $Wit_A(\vec{c}, u)$ is defined so that

- If A is $(\exists x \le t)B(\vec{c}, x)$, $B \notin \Sigma_{i-1}^b$, then $Wit_A(\vec{c}, u)$ is the formula $u \le t \wedge B(\vec{c}, u)$.
- If A is in Π_{i-1}^b, then $Wit_A(\vec{c}, u)$ is just $A(\vec{c})$.

The following is trivial since we are working with strict formulas.

Fact: $A(\vec{c}) \leftrightarrow (\exists u)Wit_A(\vec{c}, u)$.

Fact: Wit_A is a Π_{i-1}^b-formula (or Δ_1^b, when $i = 1$.)

A cedent is a set of formulas. If Γ and Δ are cedents, then $\Gamma \longrightarrow \Delta$ is a *sequent*. Its meaning is that the conjunction of Γ implies the disjunction of Δ.

Letting $\Gamma = A_1, \ldots, A_k$, then $Wit_\Gamma(\vec{c}, u)$ is the statement:

$$\bigwedge_{i=1}^{k} Wit_{A_i}(\vec{c}, (u)_i).$$

For $\Delta = B_1, \ldots, B_\ell$, $Wit_\Delta(\vec{c}, u)$ is the statement

$$\bigvee_{j=1}^{\ell} ((u)_1 = j \wedge Wit_{B_j}(\vec{c}, (u)_2))$$

The notation $(u)_i$ means $\beta(i, u)$, the i-entry in the sequence coded by u. That is, $u = \langle u_1, \ldots, u_k \rangle$ in the first case, and $u = \langle u_1, u_2 \rangle$ in the second case.

Theorem (Witnessing Lemma)

If $\Gamma \longrightarrow \Delta$ is an S^i_2-provable sequent of $s\Sigma^b_i$ formulas with free variables $\vec{c}$, then there is a function $f(\vec{c}, u)$ which is Σ^b_i-definable in S^i_2 and computable in polynomial time with an oracle for Σ^b_{i-1} such that S^i_2 proves

$$Wit_\Gamma(\vec{c}, u) \rightarrow Wit_\Delta(\vec{c}, f(\vec{c}, u)).$$

The theorem is proved by induction on the number of lines in a free-cut free S^i_2-proof P of $\Gamma \longrightarrow \Delta$. The base cases are the equational axioms defining the symbols of the language. Since witnesses for Δ^b_0-formulas are trivial, these cases are all trivial.

The induction step splits into cases depending on the last inference of the proof P.

Case (1): Last inference is $\exists \leq$:right.

$$\frac{\Gamma \longrightarrow \Delta, A(\vec{c}, s)}{s \leq t, \Gamma \longrightarrow \Delta, (\exists x \leq t) A(\vec{c}, x)}$$

The formula A is $s\Pi^b_{i-1}$. The induction hypothesis gives a function f, which accepts witnesses for Γ and produces a witness either making a formula in Δ true or making $A(\vec{c}, s)$ true. Modify f, so that in the latter case, it returns $\langle \ell, s \rangle$.

$$g(\vec{c}, u) = \begin{cases} f(\vec{c}, cdr(u)) & \text{if } (f(\vec{c}, cdr(u)))_1 < \ell \\ \langle \ell, s(\vec{c}) \rangle & \text{if } (f(\vec{c}, cdr(u)))_1 = \ell. \end{cases}$$

(The "*cdr*" operation strips the first entry from a sequence.)

Case (2): Last inference is $\exists \le$:left.

$$\frac{b \le t, A(\vec{c}, b), \Gamma \longrightarrow \Delta}{(\exists x \le t) A(\vec{c}, x), \Gamma \longrightarrow \Delta}$$

where A is $s\Pi^b_{i-1}$ but not $s\Sigma^b_{i-1}$. Let f be given by the induction hypothesis. Define g by

$$g(\vec{c}, u) \;=\; f(\vec{c}, (u)_1, \langle 0 \rangle * u)$$

(The "$*$" operation is sequence concatenation.)

Case (2'): Last inference is $\exists \le$:left.

$$\frac{b \le t, A(\vec{c}, b), \Gamma \longrightarrow \Delta}{(\exists x \le t) A(\vec{c}, x), \Gamma \longrightarrow \Delta}$$

where A is $s\Pi^b_{i-2}$. Let f be given by the induction hypothesis. Let $\mu_A(\vec{c})$ equal the least $x \le t(\vec{c})$ such that $A(\vec{c}, x)$ is true, or equal $t+1$ if no such x exists.

Define g as

$$g(\vec{c}, u) \;=\; f(\vec{c}, \mu_A(\vec{c}), \langle 0 \rangle * u).$$

Note that μ_A is computable in polynomial time with an oracle for $s\Sigma^b_{i-1}$.

A similar argument applies for $\forall \le$: *right* inferences.

Case (3): Last inference is PIND.

$$\frac{A(\lfloor \frac{1}{2}b \rfloor), \Gamma \longrightarrow \Delta, A(b)}{A(0), \Gamma \longrightarrow \Delta, A(t)}$$

where $A \in \Sigma_i^b \setminus \Sigma_{i-1}^b$. Let f be given by the induction hypothesis. Define

$$h(\vec{c}, b, u) = \begin{cases} h(\vec{c}, \lfloor \frac{1}{2}b \rfloor, u) & \text{if } (h(\vec{c}, \lfloor \frac{1}{2}b \rfloor, u))_1 < \ell \\ f(\vec{c}, b, \langle (h(\vec{c}, \lfloor \frac{1}{2}b \rfloor, u))_2 \rangle * cdr(u)), & \text{otherwise} \end{cases}$$

and $h(\vec{c}, 0, u) = \langle \ell, (u)_1 \rangle$. h can be defined by *limited iteration on notation* and is polynomial time computable relative to f. Here, ℓ is the number of formulas in the antecedent.

Then set $g(\vec{c}, u) = h(\vec{c}, t(\vec{c}), u)$. Q.E.D.

TFNP problems of S^1_2

Corollary

If $R(x,y) \in \mathrm{P}$ and $S^1_2 \vdash (\forall x)(\exists y)R(x,y)$, then $R(x,y)$ is computable by some polynomial time function, provably in S^1_2. That is, for some Σ^b_1-defined, hence ptime, function f, $S^1_2 \vdash \forall x R(x, f(x))$.

Proof: Parikh's theorem gives a polynomial bound on y that is provable in S^1_2. Then, the corollary is immediate from the Witnessing Lemma. $\square$

$T^i_2 \preccurlyeq_{\Sigma^b_{i+1}} S^{i+1}_2$

Next we sketch the proof of the fact that S^{i+1}_2 is $\forall\Sigma^b_{i+1}$-conservative over T^i_2.

Lemma

$T^i_2 \vdash \Pi^b_i$*-IND.*

Proof. Given $A(x)$ in Π^b_i, instead of using induction on $A(x)$ from $x = 0$ up to $x = t$, use induction on $\neg A(t \dot{-} x)$ with t fixed.

Lemma

$T^i_2 \vdash \Sigma^b_i$*-minimization.*

Proof. Suppose $(\exists x)A(x)$, but there is no least such x. Use induction on the Π^b_i-formula $(\forall x < a)\neg A(x)$ to get a contradiction.

Lemma

T^i_2 can Σ^b_{i+1}-define every function in $P^{\Sigma^b_i}$.

Proof. (Idea.) Let f be in $P^{\Sigma^b_i}$. Without loss of generality, f is computed using a "witness oracle" that when queried "$\exists x \le t.A(x,n)$?" either returns a value for $x \le t$ that makes A true, or returns $t+1$ indicating no such x exists.

A consistent computation for f is a computation based on a sequence of oracle answers such that any response $x \le t$ does satisfy A (but answers "$t+1$" may be incorrect).

The property of being a consistent computation is Π^b_{i-1}. Order consistent computations lexicographically; T^i_2, via Σ^b_i-minimization, proves there exists a minimum consistent computation. And, that this consistent computation has all oracle answers correct. It is straightforward to check that the minimum consistent computation is Σ^b_{i+1}-definable.

Theorem (B'90)

S_2^{i+1} is $\forall\Sigma_{i+1}^b$-conservative over T_2^i.

Proof. (Idea) Repeat the proof of the Witnessing Lemma for S_2^{i+1}, but now the conclusion is that T_2^i proves the witnessing sequent (instead of S_2^{i+1}):

$$Wit_\Gamma(\vec{c}, u) \rightarrow Wit_\Delta(\vec{c}, f(\vec{c}, u)).$$

It can be checked that T_2^i can formalize all the reasoning that was earlier formalized in S_2^{i+1}.

T^1_2 and PLS [BK'94]

A *Polynomial Local Search* PLS is formalized in S^1_2 provided its feasible set, initial point function, neighborhood function, and cost function are Σ^b_1-defined (as ptime functions).

Theorem

T^1_2 can prove that any (formalized) PLS problem is total.

Proof: By Σ^b_1-minimization, T^1_2 can prove there is a minimum cost value c_0 satisfying

$$(\exists s \leq b(x))(F(x,s) \wedge c(x,s) = c_0).$$

Choosing s that realizes the cost c_0 gives either a solution to the PLS problem or a place where the PLS conditions are violated. $\square$

Open: Can T^1_2 witness any PLS problem with a Σ^b_1-definable (single-valued) function?

Theorem (BK'94)

If $A \in \Sigma_1^b$ and $T_2^1 \vdash (\forall x)(\exists y)A(x,y)$, then there is a PLS *problem R such that T_2^1 proves*

$$(\forall x)(\forall y)(R(x,y) \rightarrow A(x,(y)_1)).$$

If $A \in \Delta_1^b$, then can replace "$(y)_1$" with just "y".

This gives an exact complexity characterization of the $\forall\Sigma_1^b$-definable functions of T_2^1, in terms of PLS-computability.

Theorem (Witnessing Lemma)

If $\Gamma \longrightarrow \Delta$ is a T^1_2-provable sequent of $s\Sigma^b_1$ formulas with free variables $\vec{c}$, then there is a PLS problem $R(\langle \vec{c}, u\rangle, v)$ so that T^1_2 proves

$$Wit_\Gamma(\vec{c}, u) \wedge R(\langle \vec{c}, u\rangle, v) \rightarrow Wit_\Delta(\vec{c}, v).$$

Proof idea: Use a free-cut free T^1_2-proof, proceed by induction on number of inferences in the proof. Arguments are similar to to what was used to prove the witnessing lemma for S^i_2 ($i = 1$ case). Most cases just require closure of PLS under polynomial time operations. However, induction (Σ^b_1-IND inference) now requires exponentially long iteration: this is handled via the exponentially many possible cost values. $\square$

The Theorem and Witnessing Lemma generalize to $i > 1$ with $\mathrm{PLS}^{\Sigma^b_{i-1}}$. The fourth talk will improve on this, however.

Some selected references

- R. Parikh, Existence and Feasibility in Arithmetic, JSL, 1971.
- A. Wilkie, J. Paris, On the Scheme of Induction for Bounded Arithmetic Formulas, APAL, 1987.
- S. Buss, *Bounded Arithmetic*, Ph.D. thesis, 1985. Bibliopolis, 1986. Also available online.
- S. Buss, Axiomatizations and conservation results for fragments of bounded arithmetic. In *Logic and Computation*,AMS Contemp. Math. 106 (1990) 57-84.
- S. Buss, J. Krajíček, An application of Boolean complexity to separation problems in bounded arithmetic. Proc. LMS 69 (1994) 1-21.