On the Bounded Version of Hilbert’s Tenth Problem

Chris Pollett

214 MacQuarrie Hall
Dept. of Math and Computer Science
San Jose State University
1 Washington Square, San Jose CA 95192
USA

Abstract. The paper establishes lower bounds on the provability of $D \equiv \text{NP}$ and the MRDP theorem in weak fragments of arithmetic. The theory $I^1E_1$ is shown to be unable to prove $D \equiv \text{NP}$. This non-provability result is used to show that $I^1E_1$ cannot prove the MRDP theorem. On the other hand it is shown that $I^1E_1$ proves $D$ contains all predicates of the form $(\forall i \leq b)P(i, x) \equiv Q(i, x)$ where $\equiv$ is $=, <, \leq$, and $I^1E_1$ proves $D$ contains all predicates of the form $(\forall i \leq b)P(i, x) = Q(i, x)$. Here $P$ and $Q$ are polynomials. A conjecture is made that $D$ contains NLOGTIME. However, it is shown that this conjecture would not be sufficient to imply $D = \text{NP}$. Weak reductions to equality are then considered as a way of showing $D \equiv \text{NP}$. It is shown that the bit-wise less than predicate, $\leq_2$, and equality are both co-NLOGTIME complete under FLOGTIME reductions. This is used to show that if the FLOGTIME functions are definable in $D$ then $D = \text{NP}$.

1 Introduction

The Matiyasevich-Robinson-Davis-Putnam (MRDP) theorem [15] says that the Diophantine sets are precisely the recursively enumerable sets and provides a negative answer to Hilbert’s Tenth Problem concerning the decidability of the Diophantine sets. The MRDP theorem has been used to show many problems from calculus and differential equations are undecidable (see Matiyasevich [16]).

The study of what bounded versions of this theorem hold was begun by Adleman and Manders [1]. They show a bounded form of MRDP Theorem which says a set $A$ is in $E_n$ for $n \geq 3$ of the Grzegorczyk Hierarchy iff it is of the form:

$$A = \{x | (\exists y \leq f(x)) P(x, y) = Q(x, y)\},$$

where $P, Q$ are polynomials with coefficients in $\mathbb{N}$ and $f$ is in $E_n$. These bounds are quite large as $E_3$ is already the class of elementary functions. Adleman and Manders also pose the question of whether $NP$ is equal to the class $D$ of predicates given by formulas of the form:

$$\exists y [(\sum_j y_j \leq 2^{\sum_i |x_i|^n}) \land P(x, y) = Q(x, y)]$$
where $P, Q$ are polynomials with coefficients in $\mathbb{N}$. (It should be noted that many papers on this class work with an equivalent formulation where integer coefficients are used and $P$ and $Q$ are pulled-over to the same side of the equality and a check is made if this resulting polynomial is zero. It will be convenient however to work with natural number coefficients in this paper.) They show that the class resulting from the use of only a single existential quantifier is contained in $P$ and moreover if the two variable case is in $P$ then primality checking is p-time. Adleman and Manders [2] define a notion of $D$- reducibility and give four very weak languages which are $D$-complete for $NP$. Jones and Matiyasevich [8] have shown that $D = NP$ if the predicate $x \leq y$ is in $D$. Here $x \leq y$ if the $i$ bit of $x$ is less than the $i$ bit of $y$ for all $i$. Venkatesan and Rajagopalan [21] conjecture that given a number $A$ and a product $\pi$ of powers of distinct odd primes $P_i$ then testing whether $A$ has even residues modulo all the $P_i$’s is in $D$. Under this conjecture, they show that $D = NP$ and that the Randomized Diophantine Problem (RDP) is average case complete for $NP$. On the other hand, $D = NP$ implies their conjecture, so it follows that $D = NP$ implies the average case completeness of RDP. Given the scarcity of natural average case complete problems for $NP$, this lends impetus to resolving the $D = NP$ problem.

The question of what bounded forms of the MRDP theorem hold is closely connected with how strong a formal system is needed to prove the MRDP theorem. Gaifman and Dimitracopoulos [6] have shown that $\Pi_1 \Delta_0 + exp$ can prove the MRDP theorem. $\Pi_1 \Delta_0 + exp$ essentially allows one usual induction on bounded formulas in the language of arithmetic together with the axiom that exponentiation is total. This result was improved by Kaye [11] who showed $IE_1 + E$ could prove the MRDP theorem and used this to show $IE_1 + E = \Pi_1 \Delta_0 + exp$. Here $E$ was an axiom that asserted that a certain Diophantine equation had solutions. The bounds on these solutions are known to be exponential in the real world. These results correspond well with the Adleman and Manders result since Clote and Takeuti [3] have shown that in some sense $\Pi_1 \Delta_0 + exp$ is the “right” theory to reason about about the elementary functions.

Just as it is unknown if one can get a more than exponentially bounded form of the MRDP theorem, it is also unknown what weaker theories can prove this theorem. It is known that if a theory $T \subseteq S_2$ proves the MRDP theorem then $NP = co-NP$ (see Hájek and Pudlák [7] for a proof). Here $S_2$ corresponds roughly to a theory which has induction on formulas in the polynomial hierarchy. However, no such complexity consequences are known concerning the provability of $D = NP$ in subsystems of $S_2$. It is also still open whether $S_2$ or a subsystem might be able to prove the MRDP theorem. That is, no one has proven $S_2$ cannot prove the MRDP theorem. It is known $IOpen$ in the language of $0, S, +, \cdot$ cannot prove MRDP [12].

The first result of this paper is to give a “reasonably strong” theory which cannot prove $D = NP$. Essentially by results of Kent-Hodgson [14], every $NP$-predicate can be expressed in the language $L_2$ mentioned in the abstract by a $\Sigma_1^1$-formula. That is, it can be expressed by a formula of the form $B := (\exists y \leq t(x)) (\forall i \leq |s(x)|) A(x, y, i)$ where $A$ is open (see Pollett [17]). Since $D$ is closed
under bounded existentiation, to show $\mathcal{D} = \mathcal{NP}$ it suffices to show $\mathcal{D}$ contains the class of formulas $\Pi^0_3$ of the form $(\forall i \leq |s(x)|)A(x, y, i)$. It seems likely that such a proof should be formalizable in a weak arithmetic using induction over values $0 \leq i \leq |s|$ for some $\Pi^0_3$-formula and for some $\mathcal{D}$-formula. The theory $I^1E_1$ which has length induction on bounded existential $(E_1)$ formulas should thus be able to formalize this proof. This is because standard tricks $I^1E_1$ can prove length induction on boolean combination of $E_1$-formulas (so also bounded universal $(U_1)$ formulas). As evidence for this belief, in this paper it is shown that $I^1E_1$ proves $\mathcal{D}$ contains the predicates $(\forall i \leq |b|)P(i, x) \circ Q(i, x)$ where $\circ$ is $=, \leq, \text{or} < \text{and} P, Q$ are polynomials and $i \leq |b|$ means each $i_j \leq |b|, j = 1, \ldots, m$. Replacing the $L_2$-base functions of the open predicate $A$ in the formula $B$ above to get a matrix of the form $P = Q$ for $P, Q$ polynomials would in general introduce bounded existential quantifiers inside the length-bounded universal. So this result does not imply $\mathcal{D} = \mathcal{NP}$. In the case of $=, \text{one can formalize an old result of Raphael Robinson [19]}$ to show $I^1E_1$ proves $\mathcal{D}$ contains all predicates of the form $(\forall i \leq |b|)P(i, x) = Q(i, x)$, which has five-lengths induction on $E_1$-predicates and cannot prove $\mathcal{D} = \mathcal{NP}$. The way this is proved is to show $I^1E_1$ can $\Sigma^0_4$-define the function $[\langle i \rangle]$ but it cannot $E_1$-define it. Since $\mathcal{D} \subseteq E_1$ this gives the result. The $\Sigma^0_4$-definability argument in $I^5E_1$ essentially follows from a use of excluded middle. The $E_1$-nondifiability argument is based on a new more sensitive variant of the block counting argument of Johannsen [9, 10] and Pollett [18].

One can slightly sharpen the $\Sigma^0_4$-characterization of $\mathcal{NP}$ given above. $\mathcal{NP}$ can be represented as those formulas of the form $(\exists y \leq t(x))\text{co-NLOGTIME}$ (The PCP Theorem [3] says something stronger still: that $\mathcal{NP}$ is the predicates of the form $(\exists y \leq t(x))\text{co-RLOGTIME}$ where the co-RLOGTIME machine can only query $y$ constantly many times on a given branch). This is sharper since neither addition nor multiplication is in co-NLOGTIME. It also illustrates how weak a class needs to be shown in $\mathcal{D}$ in order to prove $\mathcal{D} = \mathcal{NP}$ and thus illustrates how surprising it is that one can get any reasonable independence proof. This is because this result implies it suffices to show co-NLOGTIME $\subseteq \mathcal{D}$ to get $\mathcal{D} = \mathcal{NP}$. Nevertheless, at present the co-NLOGTIME $\subseteq \mathcal{D}$ inclusion seems hard to show; however, the author conjectures that it is reasonably likely that NLOGTIME is in $\mathcal{D}$. This is because the known techniques (machine-based, Chinese Remaindering, or generalized geometric progressions) for removing a bounded universal in front of a block of existentials make two nested uses of exponentiation. The first use is to create a sequence of values and the second nested use is a call usually to $\binom{n}{k}$ to verify some correctness property of this sequence. It seems possible using these techniques that an NLOGTIME computation could be verified by a quantifier of the form $(\forall i \leq |t|)$ in front of a block of small existentials and polynomial equalities. If two nested exponentials are actually required to do sequence coding with Diophantine equations than this might be the largest machine class one could hope to directly in $\mathcal{D}$. If NLOGTIME $\subseteq \mathcal{D}$, though, one might conjecture that $\mathcal{D} = \mathcal{NP}$ because ENLOGTIME = NP. Here $EC'$ denotes the class of languages expressed by predicates of the form $(\exists y \leq s)(\langle x, y \rangle \in L)$ for some
s an $L_2$-term and $L$ in $C$. It is shown however that $\text{ENPOLYLOGTIME} \subseteq \text{NP}$. Thus, if $D = \text{NP}$, the proof will need to rely more on the fact that $\leq$, $+$, and $\cdot$ lie outside of $\text{NLOGTIME}$.

Pursuing this last idea, the possibility that $=$ might be hard for co-$\text{NLOGTIME}$ under reductions computable in $D$ is investigated. A function is said to be in $\text{FDLOGTIME}$ if it has a polynomial sized bit-graph whose bits are in $\text{DLOGTIME}$. Since $\text{DLOGTIME}$ is likely to be in $D$, it is conjectured that the $\text{FDLOGTIME}$ functions are definable in $D$. It is shown in this paper that both the $\leq_2$ of Jones and Matiyasevich and equality are co-$\text{NLOGTIME}$-complete under $\text{FDLOGTIME}$ reductions. So if the conjecture holds then using equality one can show co-$\text{NLOGTIME} \subseteq D$ and so $D = \text{NP}$. Since it is easy to see that $D = \text{NP}$ implies $\leq_2$ in $D$, Venkatesan and Rajagopalan [21]'s conjecture, $\leq_2$ in $D$, the conjecture above, and $D = \text{NP}$ are all equivalently hard problems. In the conclusion we discuss a stronger conjecture which implies but is not equivalent to $D = \text{NP}$.

The remainder of this paper is organized as follows: In Section 2 some preliminary definitions are presented. In Section 3, it is proven that $\text{EBASIC}$ can define $\lceil \frac{1}{2^n} x \rceil$ and the function algebras $A^m$ are defined and these algebras are shown to contain all the $E_1$-definable functions of $P^m E_1$. Then in Section 4, it is established that $\lceil \frac{1}{2^n} x \rceil$ is not in $A^0$ and so $P E_1$ does not prove $D = \text{NP}$. Section 5 gives the proof that the $(\forall i \leq |s|)P = Q$ predicates are in $D$. In Section 6 has the results about $\text{EPOLOGYTIME}$ and co-$\text{NLOGTIME}$. Finally, the last section contains a conclusion.

## 2 Preliminaries

The language $L_2$ contains the non-logical symbols: $0$, $S$, $+$, $\cdot$, $=$, $\leq$, $\geq$, $\lceil \frac{1}{2^n} x \rceil$, $\lceil x \rceil$, $\text{MSP}(x,i)$ and $\#$. The symbols $0$, $S(x) = x + 1$, $+$, $\cdot$, and $\leq$ have the usual meaning. The intended meaning of $x + y$ is $x$ minus $y$ if this is greater than zero and zero otherwise, $\lceil \frac{1}{2^n} x \rceil$ is $x$ divided by $2$ rounded down, and $[x]$ is $\lceil \log_2 (x+1) \rceil$, that is, the length of $x$ in binary notation. $\text{MSP}(x,i)$ stands for ‘most significant part’ and is intended to mean $\lceil x / 2^i \rceil$. Finally, $x \not\equiv y$ reads ‘$x$ smash $y$’ and is intended to mean $2^{\text{length}}$. The notation 1 is used for $S(0)$, 2 for $S(S(0))$, etc. A quantifier of the form $(\forall x \leq t)$ or $(\exists x \leq t)$ where $t$ is a term not containing $x$ is called a **bounded quantifier**. A formula is bounded or $\Delta_0$ if all its quantifiers are. A quantifier of the form $(\forall x \leq |t|)$ or of the form $(\exists x \leq |t|)$ is called **sharply bounded** and a formula is sharply bounded if all its quantifiers are. Given a language $L$, the hierarchy of formulas $E_{i,L}$ and $U_{i,L}$ are defined as follows: $E_{1,L}$ are those formulas of the form $(\exists x \leq t)\phi$ and $U_{1,L}$ are those formulas of the form $(\forall x \leq t)\phi$ where $\phi$ is an open formula. $E_{i,L}$ are those formulas of the form $(\exists x \leq t)\phi$ where $\phi \in U_{i-1,L}$-formula. $U_{i,L}$ are those formulas of the form $(\forall x \leq t)\phi$ where $\phi \in E_{i-1,L}$. The notations $E_i$ and $U_i$ are used when $L$ is understood. The class of quantifier-free formulas is denoted by $\text{open}$. For $i > 0$, a $\Sigma_i^0$-formula (resp. $\Pi_i^0$-formula) is defined to be a $E_{i+1}$-formula (resp. $U_{i+1}$-formula) whose innermost quantifier is sharply bounded. Kent and Hodgeson [14] (see also Pollett [17]) have shown the sets defined by $\Sigma_i^0$ (resp.
\( \hat{P}^1 \)-formulas in the language of \( L_2 \) are precisely the \( \Sigma^1_1 \)-(resp. \( \Pi^1_1 \)-)predicates. Thus, the \( \hat{\Sigma}^1_1 \)-formulas correspond to the NP predicates.

The theory \( \text{BASIC} \) is axiomatized by all substitution instances of a finite set of quantifier free axioms for the non-logical symbols of \( L_2 \). These are listed in Buss [4] except for the axioms for \( \text{MSP} \) and \( \neg \) which are listed in Takeuti [20].

The \( L_2 \)-terms below will be useful for this paper:

\[
\begin{align*}
2|b| &= 2|b|^1 := 1 \\
2|b|^n &= 2^{|1|b|^n-1} \\
2^{|1|b|^n} &= 2|b|^n \cdot 2^{(k-1)}|b|^n \\
2^{|\min(|b|, x)|} &= \text{MSP}(2|b|, |y| + x) \\
K_-(x) &= 1 - x \\
K_<(x, y) &= x \cdot y \\
K_<(x, y) &= K_-(y - x) \\
\text{LS}(x, i) &= x - \text{MSP}(x, i) \cdot 2^{|\min(|x|, i)|} \\
\beta(x, |t|, w) &= \text{MSP} \left( \text{LS}(w, Sx|t|), x|t| \right) \\
\text{Bit}(i, x) &= \beta(i, 1, x) \\
\beta(x, |t|, s, w) &= \min(\beta(x, |t|, w), s) \\
\text{cond}(x, y, z) &= K_-(x) \cdot y + K_-(K_-(x)) \cdot z \\
\max(x, y) &= \max(K_<(x, y), y, x) \\
\min(x, y) &= \min(K_<(x, y), x, y)
\end{align*}
\]

The \( k \) and \( n \) in \( 2^{|1|b|^n} \) are fixed integers. Taking products of terms \( 2^{|1|b|^n} \) one can construct terms representing \( 2^{|n|b|l|} \) where \( p \) is any polynomial. Roughly, \( \beta(x, |t|, w) \) projects out the \( x \)th block (starting with a 0th block) of \( |t| \) bits from \( w \). \( \beta(x, |t|, s, w) \) returns the minimum of \( \beta(x, |t|, w) \) and \( s \). For brevity, this paper uses \( 2^{|t|} \) for \( 2^{|\min(|t|, x|t|, |t|)|} \), if \( |t| \) is a term which is obviously less than \( |t(x)| \) for some \( t \in L_2 \).

A pairing operation which will sometimes be more convenient than block coding can be defined as follows. Let \( B = 2^{|\max(|x|, y)|} + 1 \). Thus, \( B \) will be longer than either \( x \) or \( y \). Define an ordered pair as \( (x, y) := (2^{|\max(|x|, y)|} + y) \cdot B + (2^{|\max(|x|, y)|} + x) \). To project out the coordinates from such an ordered pair, use \( \beta(1, w) := \beta(0, [\frac{1}{2}|w| - 1, \beta(0, [\frac{1}{2}|w|], w)) \) and \( \beta(2, w) := \beta(0, [\frac{1}{2}|w| - 1, \beta(1, [\frac{1}{2}|w|], w)) \) which return the left and right coordinates of the pair \( w \). (The real Gödel beta function projects out \( \beta(i, w) \), the \( i \)th element of a sequence \( w \). However, as this function is never used in this paper, the suggestive notation should not cause confusion.) To check if \( w \) is a pair the function \( \text{ispair}(w) := \text{Bit}(w, [\frac{1}{2}|w| + 1) = 1 \land 2 \cdot \max(\beta(1, w), \beta(2, w))] + 2 = |w| \)

is used. Notice the above functions are all \( L_2 \)-terms.

**Definition 21** \( \text{E} \text{BASIC} \) is the theory obtained from \( \text{BASIC} \) by adding the following axioms:

1. \( b < 2^{|\min(k, |l|)|} \supseteq \text{MSP}(a, 2^{|\min(k, |l|)|} + b, \min(k \cdot |d|, |d|^2)) = a \).
2. \( b < 2^{|l|} \land a < 2^{|d|} \supseteq (\beta(0, |d|, a \cdot 2^{|d|} + b) = b \land \beta(1, |d|, a \cdot 2^{|d|} + b) = a) \).


(3) \( Si \cdot |a| \leq k \supset \hat{\beta}(i, |a|, w) = \hat{\beta}(i, |a|, LSP(w, k)) \)

The new axioms of \( EBASIC \) were used in Pollett [17] to show \( EBASIC \) can do pairing. In particular, the following lemma was shown.

**Lemma 22** Let \( m = \max(s(a), t(a, s)) \), and let \( t^+ := t(a, \hat{\beta}(0, |m|, s(a), w)) \) where \( s(a), t(a, b) \in L_2 \). Then the theory \( EBASIC \) proves:

(a) \((\exists x \leq 2^{2 |m|}) A(\hat{x}(0, |m|, s, w), \hat{\beta}(1, |m|, t^+, w)) \)
\( \iff (\exists x \leq s)(\exists y \leq t) A(x, y) \)

(b) \((\forall x \leq 2^{2 |m|}) A(\hat{x}(0, |m|, s, w), \hat{\beta}(1, |m|, t^+, w)) \)
\( \iff (\forall x \leq s)(\forall y \leq t) A(x, y) \).

\( BASIC \) and \( EBASIC \) proofs will be formalized in the system \( LKB \) of Buss [4] where one has equality axioms and where one takes the axioms of \( BASIC \) (\( EBASIC \)) as initial sequents. The main point of \( LKB \) is it treats bounded quantifiers syntactically. One defines stronger theories by adding various types of induction rules to \( BASIC \) and \( EBASIC \).

**Definition 23** A \( \Psi \cdot L^mIND \) inference is an inference

\[
\frac{A(b), \Gamma \rightarrow A(Sb), \Delta}{A(0), \Gamma \rightarrow A([t(x)]_m), \Delta}
\]

where \( b \) is an eigenvariable and must not appear in the lower sequent, \( t \) is a term in the language, and \( A \in \Psi \). Here \( |a|_0 = a \) and \( |a|_m = ||a||_{m-1} \). We call \( [t(x)]_m \) the principal term of the induction inference.

The notations \( IND \), \( LIND \) and \( LLIND \) will be used instead of \( L^0IND \), \( L^1IND \), and \( L^2IND \).

**Definition 24** (\( i \geq 0 \)) \( I^mE_i \) is defined to be \( EBASIC + E_i \cdot L^mIND \) and \( S^m_2 \) is defined to be \( BASIC + \Sigma^b_i \cdot LIND \). Finally, \( S_2 := \cup_i S^m_2 \).

### 3 Definability Results

The proof that \( I^5E_1 \) cannot prove \( D = \mathbb{NP} \) requires first showing that \( I^5E_1 \) can \( \hat{\Sigma}^b_4 \)-define the function \( \|x\| \) which will be done in this section. Then a function algebra \( A^m \) which is an upper bound on the \( E_1 \)-definable functions of \( I^mE_1 \) is given. Since the \( \hat{\Sigma}^b_4 \)-predicates are the \( NP \)-predicates and the \( E_1 \)-predicates contain the \( D \)-predicates, these two results will serve as the basis for the proof that \( I^5E_1 \) cannot prove \( D = \mathbb{NP} \).

Let \( \Psi \) be a set of formulas. A theory \( T \) can \( \Psi \)-define a function \( f(x) \), if there is a \( \Psi \)-formula \( A_f(x, y) \) such that \( T \vdash \forall x \exists y A_f(x, y) \) and \( \mathbb{N} \models A_f(x, f(x)) \). The notions of \( E_1 \) and \( \Sigma^b_4 \)-definability will be the most useful for this paper.

**Theorem 1.** \( \|x\| \) is \( \hat{\Sigma}^b_4 \)-definable in \( EBASIC \).

**Proof.** Using excluded middle,

\[
EBASIC \vdash (\forall x)(\exists y) A_{\chi \|x\|}(x, y)
\]
where $A_{|\frac{R}{x}|}$ is:

\[(\exists z \leq |x|)(B(x, z) \land y = z)\]

\[
\lor (y = |x| + 1 \land \neg(\exists z \leq |x|)B(x, z))
\]

and

\[B(x, z) := 3z = |x| \lor 3z + 1 = |x| \lor 3z + 2 = |x|.
\]

This is easily seen to be equivalent in $EBASIC$ to a $\tilde{\Sigma}^0_1$-formula. Uniqueness follows since $EBASIC$ can prove $y$ must be either less than or equal to $|x|$ or greater than $|x|$ but not both. In the latter case its value is forced to be $|x| + 1$. In the former case, $EBASIC$ can prove only one of $3z = |x|$, $3z + 1 = |x|$, and $3z + 2 = |x|$ can hold by using the axiom $a + b \leq a + c \iff b \leq c$ and equality axioms. It is obvious that $\mathbb{N} = A_{|\frac{R}{x}|}(x, \frac{1}{3}|x|)).$

**Definition 31** ($BPR^m$) $f$ is defined from functions $g$, $h$, $t$, and $r$ by $m$-length bounded primitive recursion if

\[
F(0, x) = g(x)
\]

\[
F(n + 1, x) = \min(h(n, x, F(n, x)), r(n, x))
\]

\[
f(n, x) = F([t(n, x)]_m, x)
\]

for some $r \in L_2$ and for some $t \in L_2$.

If $g$, $h$, $t$, and $r$ are multifunctions then $f$ obtained by $BPR^m$ results by viewing each step in the above iteration as a composition of multifunctions.

**Definition 32** $A^m$ is the smallest class containing the $L_2$-terms, closed under composition, and closed under $BPR^m$.

The next theorem gives an upper bound on the $E_1$-definable functions of $I^m E_1$.

**Theorem 2.** The $E_1$-definable functions of $I^m E_1$ are contained in $A^m$.

It is not claimed that $I^m E_1$ can define all the functions in $A^m$.

**Proof.** This is proved by a Buss-style [4] “witnessing argument”. Let $f$ be $E_1$-defined by $A_f(x, y)$. If $I^m E_1$ proves $(\exists x)(\exists y)A_f$, then since $I^m E_1$ is bounded theory by Parikh’s Theorem there is a term $t \in L_2$ such that $I^m E_1$ proves $(\exists x)(\exists y \leq t)A_f$. So $I^m E_1$ proves $(\exists y \leq t(a))A_f(a, y)$. By cut-elimination, one can assume every formula of a sequent in this last proof is either of the form $(\exists x_1 \leq t_1)(\exists x_2 \leq t_2)\phi$ where $\phi$ is an open formula or can be made into this form by padding on dummy quantifiers. Call such a formula $LEE_1$-formula (lexicographically an existential followed by a $E_1$-formula). A witness predicate for such an $LEE_1$-formula is defined as follows: If $A$ is open $Wit_A(w, a) := w = 0 \land A(a)$. If $A$ is $(\exists x \leq t(a))B$ and $A$ is $E_1$ then $Wit_A(w, a) := w \leq t(a) \land B(w, a)$. Finally, if $A(a)$ is $(\exists x_1 \leq t_1)(\exists x_2 \leq t_2)B$ and $A \in EE_1$ then

\[
Wit_A(w, a) := ispair(w) \land \beta(1, w) \leq t_1 \land \beta(2, w) \leq t_2 \land
\]

\[B(\beta(1, 2), \beta(2, w), a).
\]
For a cedent \( \Gamma = \{ A_1, \ldots, A_N \} \) of LEE\(_1\)-formulas, let \( \wedge \Gamma \) denote their conjunction and \( \forall \Gamma \) their negation. Let \( \Delta \) be another such cedent. Following Pollett [18] one can define open witness predicates \( \text{Wit}_{\wedge \Gamma}(w, a) \), \( \text{Wit}_{\forall \Delta}(w, a) \) for such cedents and also terms \( t_\Gamma \) and \( t_\Delta \) such that \( \mathbb{N} \models \Gamma \rightarrow \Delta \) iff
\[
\mathbb{N} \models (\exists w \leq t_\Gamma) \text{Wit}_{\wedge \Gamma}(w, a) \rightarrow (\exists w \leq t_\Delta) \text{Wit}_{\forall \Delta}(w, a).
\]

The predicate \( \text{Wit}_{\forall \Delta}(w, a) \) in Pollett [18] has the property that if \( \Delta \) consists of just one formula \( A \) then \( \text{Wit}_{\forall \Delta}(w, a) = \text{Wit}_A(w, a) \). The predicate \( \text{Wit}_{\wedge \Gamma}(w, a) \) in the case where \( \Gamma \) is empty is defined in Pollett [18] as \( 0 = 0 \).

Next it is argued by induction on proofs of sequents of LEE\(_1\)-formulas that if \( I^m E_1 \) proves \( \Gamma \rightarrow \Delta \) then there is an \( A^m \) function \( g \) such that
\[
\mathbb{N} \models \text{Wit}_{\wedge \Gamma}(w, a) \rightarrow \text{Wit}_{\forall \Delta}(g(w, a), a).
\]

This induction breaks into cases depending on the last inference in the proof of \( \Gamma \rightarrow \Delta \). All of the cases in this witnessing argument can be handled in the same way as the \( i = 1 \) case of Theorem 22 in Pollett [18] where \( \tau = \{ |x|_m \} \). The only difference between the algebra here and the multilocation algebra \( B \) given there is that the latter contains the \( (W j \leq |t|)(s = 0) \) operator where \( s, t \) are \( L_2 \)-terms. This operator was used only to handle the sharply bounded universal quantifier inferences that arose since LEE\(_1\)-formulas were being considered in that paper. Since all the sequents in our proofs will only involve LEE\(_1\)-formulas this kind of inference never arises and so we can get away without these operators.

Thus, if \( I^m E_1 \vdash (\exists y \leq t) A_f(a, y) \), then by the above there is an \( A^m \) function \( g \) such that \( \mathbb{N} \vdash \text{Wit}_{A_f}(a, g(a)) \). So by the definition of witness, \( \mathbb{N} \vdash A_f(a, \beta(1, g(a))) \). Hence, one obtains \( f = \beta(1, g(a)) \in A^m \).

4 \( \lfloor \frac{1}{3} |x| \rfloor \) is not \( E_1 \)-definable

In this section, the proof that \( I^5 E_1 \) cannot prove \( D = \text{NP} \) is completed by showing that \( I^5 E_1 \) cannot \( E_1 \)-define \( \lfloor \frac{1}{3} |x| \rfloor \). To show this it suffices by Theorem 2 to show that \( A^5 \) cannot define \( \lfloor \frac{1}{3} |x| \rfloor \). The idea of the proof is to introduce a notion called the plus-complexity of a natural number. It is shown that if one feeds low plus-complexity inputs into a \( A^5 \) function then the plus-complexity of the output of the function will not be too “large”. On the other hand, the plus-complexity of \( \lfloor \frac{1}{3} |x| \rfloor \) on these inputs will be larger than any \( A^5 \) function could produce.

Plus-complexity as defined below will be a more sensitive measure than counting the number of alternations of 1’s and 0’s in a number that was used in Johannsen [9, 10] and Pollett [18]. Because \( L_2 \) contains \( |x| \), even if all the inputs to a term \( t \) had “few” alternations of 1’s and 0’s, all one could say about the number of alternations in the output was that it was polynomial in \( | \ldots | \) of the inputs. This is not be sufficient to show the non-definability of \( \lfloor \frac{1}{3} |x| \rfloor \) which is why the more complicated notion of plus-complexity needed to be invented.

Before plus-complexity can be defined, the notion of a stack code must first be defined:
**Definition 41** 0 is a stack code. If $\sigma$ and $\omega$ are stack codes, then the expressions $(\sigma + \omega)$ and $(2^\sigma)$ and $(-\sigma)$ are stack codes. Nothing else is a stack code. The natural number $v$ represented by a stack code $\sigma$ is defined to be the number obtained by evaluating the stack code according to the usual definitions of $+, -, 2^\sigma$ on the natural numbers.

Since $2^0$ evaluates to 1 and stack codes are closed under addition it follows that every natural number is represented by at least one stack code. In general, there are many stack codes for any given number. For example, 3 can be represented by $2^2 - 2^0$ and by $2^0 + 2^0 + 2^0$. Here parentheses are omitted from the codes, as will be done often in this paper, to improve readability. Also, the convention above in the definition that barred variables are stack codes and the same variable without the bar is what it evaluates to will be followed throughout this paper. The plus-complexity of a stack code $\sigma$, $C_+(\sigma)$, is defined below.

**Definition 42** If $\nu, \omega$ are stack codes then:

1. $C_+ (0) := 1$.
2. $C_+ ((2^\nu)) := C_+ (\nu)$.
3. $C_+ ((-\nu)) := C_+ (\nu)$.

For a natural number $x$ we define $C_+(x)$ to be

$$\min(C_+ (\nu)|\nu \text{ is a stack code evaluating to } x).$$

**Lemma 43** Let $x \in \mathbb{N}$, $x > 0$. (1) There is stack code $\pi$ such that $C_+(\pi) = C_+(x)$ and $\pi$ has the form:

$$(\cdots (2^{x_0} \pm 2^{x_1}) \pm \cdots \pm 2^{x_n}).$$

where evaluating codes $\pi_i$ to numbers $x_i$ gives $x_0 > x_1 > \ldots > x_n$. (2) If $\pi$ is of the form

$$(\cdots (2^{x_0} - 2^{x_1}) \cdots)$$

then we can take $x_0 \geq x_1 + 2$.

**Proof.** (1) follows by noticing that stack codes with subcodes of the form $(2^{x_0} + (2^{x_1} + \cdots + 2^{x_n}))$ cannot be minimal. In the minus case, this is because one could replace this subcode with 0. In the plus case, one could replace such a code with $2^{x_0 + x_1}$ which has a smaller plus-complexity. To get the $2^{x_1}$ ordered in the correct way we can use the fact that $(\pi + \omega)$ and $(\pi + \omega)$ evaluate to the same number and so do $((\pi + \omega) + \pi)$ and $(\pi + (\pi + \pi))$ and also $(-((\pi + \pi))$ and $((-\pi) + (\pi))$. Finally, since $x > 0$ the first term does not have a minus in front of it.

For (2) suppose $x_0 = x_1 + 1$ then for the sub-stack code $(2^{x_0} - 2^{x_1})$ we could have used the code $2^{x_1}$ which has lower complexity.

The next lemma gives a simple upper bound on $C_+(x)$. 
Lemma 44 For all $x \in \mathbb{N}$, $C_+(x) \leq |x|^3 + 1$.

Proof. The proof is by induction on $x$. $C_+(0) = 1$, $C_+(1) = 1$, $C_+(2) = 1$, $C_+(3) = C_+(2^2 + 2^0) = 3$, $C_+(4) = 1$. So assume for all $n < x$ that $C_+(n) \leq |n|^3 + 1$. Let $x$ be the representation of $x$ from Lemma 43. There are fewer than $|x|$ many summands. The plus-complexity rule for plus adds 1 for each of these. By the induction hypothesis, each $x_i$ will have plus-complexity less than $|x_i|^3 + 1 \leq \|x\|^3 + 1$. So one can bound the plus-complexity of $x$ by $|x| + |x| \cdot (\|x\|^3 + 1) = |x| + |x||x|^3 \leq |x|^3 + 1$. The last inequality holds if the weaker inequality $1 + \|x\|^3 \leq |x|^2$ holds and this in turn holds if $|x| \geq 3$. i.e., $x \geq 4$.

Definition 45 The function $\#_B(x)$ returns the number of alternating bits between 1 and 0 in reading the binary number $x$ from left to right. The counting of this number is started at 1 so $\#_B(1) = 1$.

As an example, let $x$ be the binary number 1110011 then $\#_B(x) = 3$. The next lemma, gives a lower bound on the plus-complexity of a number.

Lemma 46 For all $x \in \mathbb{N}$, $C_+(x) \geq \#_B(x) - 1$.

Proof. Consider a minimal stack code $\tau$ for $x$. Assume it is written in the form given by Lemma 43. From the definition of plus-complexity in the case of sum, and the fact that each summand has plus-complexity at least 1, we get $C_+(x) \geq 2 \cdot \text{number of summands} - 1 + 1$ in $\tau$. On the other hand it is easy to see $\#_B(w + 2^0) \leq \#_B(w) + 2$ and also $\#_B(w - 2^0) \leq \#_B(w) + 2$ and so we get the result.

In the next lemma, the plus-complexity of the output of each of the base functions of $L_2$ is compared with the plus-complexity of its inputs.

Lemma 47 Let $x, y \in \mathbb{N}$.

1. $C_+(x + y) \leq C_+(x) + C_+(y) + 1 \leq 2(C_+(x) + C_+(y))$.
2. $C_+(x - y) \leq C_+(x) + C_+(y) + 1 \leq 2(C_+(x) + C_+(y))$.
3. $C_+(S(x)) \leq C_+(x) + 2 \leq 3C_+(x)$.
4. $C_+(x \cdot y) \leq (C_+(x) + C_+(y) + 1)^2 \leq 8(C_+(x) + C_+(y))^3$.
5. $C_+(|x|) \leq C_+(x) + 2 \leq 3C_+(x)$.
6. $C_+(x \# y) \leq (C_+(x) + C_+(y) + 5)^3 \leq 4(C_+(x) + C_+(y))^3$.
7. $C_+(\text{MSP}(x, y)) \leq (C_+(x) + C_+(y) + 1)^2 \leq 4(C_+(x) + C_+(y))^2$.
8. If $t(x_1, \ldots, x_k)$ is an $L_2$-term, then one can find fixed constants $k, d$ such that $C_+(t(x)) \leq k(\sum_i C_+(x_i))^d$.

Proof. (1) follows directly from Definition 42.

(2) if $y > x$ then $C_+(x - y) = C_+(0) = 1$ and the bound holds; otherwise, the bound holds because of Definition 42 cases (3) and (4).

(3) follows from the fact $C_+(1) = C_+(2^0) = 1$ and the definition of plus-complexity for +.
(4) First consider the case where \( x \) and \( y \) have minimal stack codes of the form \( 2^{2^x} \) and \( 2^{2^y} \), respectively. Then a stack code for \( x \cdot y \) is \( 2^{2^{x+y}} \) and so \( C_+(x \cdot y) \leq C_+(x) + C_+(y) + 1 \). Notice if there had been minus signs in front of either \( x \) or \( y \) it would not have affected the plus-complexity of the output. In the general case, a minimal stack code of \( x \) (resp. \( y \)) will be additions of less than \( C_+(x) \) (resp. \( C_+(y) \)) terms of form \( 2^{2^x} \) or \( -2^{2^x} \) each of plus-complexity less than \( C_+(x) \) (resp. \( C_+(y) \)). So if the minimal stack codes of \( x \) and \( y \) are “multiplied out” a stack code for \( x \cdot y \) is obtained with less than \( C_+(x) \cdot C_+(y) \) terms each of which has plus-complexity less than \( C_+(x) + C_+(y) + 1 \). Using the definition of plus-complexity for addition and subtraction one obtains

\[
C_+(x \cdot y) \leq C_+(x)C_+(y)(C_+(x) + C_+(y) + 1) + C_+(x)C_+(y)
\]

which is less than

\[
(C_+(x)C_+(y) + 1)(C_+(x) + C_+(y) + 1).
\]

This in turn is seen to be less than \( (C_+(x) + C_+(y) + 1)^3 \).

(5) Assume \( x \) is a minimal stack code for \( x \) and is in the form given by Lemma 43, i.e., \( x = 2^{x'} + \bar{v} \) and evaluating \( \bar{v} \) gives a number between \(-2^{x'-1} \) and \( 2^{x'} \). Then either \( x' \), \( (x'+2^n) \), or \( (x'+2^{n'}) \) is a code for \(|x| \). Since \( C_+(x') \leq C_+(x) \) this gives the bound.

(6) follows from the definition of \( x \# y \), (5), (4) and the definitions of \( C_+(2^n) \).

(7) Let \( x := (\cdots (2^{x_n} \pm 2^{x_1}) \pm \cdots \pm 2^{x_n}) \) and \( \bar{y} \) be stack codes for \( x \) and \( y \) of minimal complexity as given in Lemma 43. A stack code for \( u = x/2^y \) (not necessarily a natural number) is

\[
(\cdots (2^{x_n-y} \pm 2^{x_1-\bar{y}}) \pm \cdots \pm 2^{x_n-y})
\]

This code \( \bar{a} \) can be split into two parts: a part \( \bar{a} \) with \( x_1 - y \geq 0 \) and a part \( \bar{v} \) with \( x_1 - y < 0 \). To obtain a stack code for \( MSP(x, y) := \lfloor x/2^y \rfloor \), delete \( \bar{v} \) from \( \bar{a} \) if it evaluates to a positive real and if \( \bar{u} \) evaluates to a negative real replace it with \((-2^y))\). In the resulting stack code there are at most \( C_+(x) \) summands of plus-complexity less than \( C_+(x) + C_+(y) + 1 \). So one can bound the plus-complexity of adding these summands by \( (C_+(x)+1)(C_+(x)+C_+(y)+1) \leq (C_+(x)+C_+(y)+1)^2 \).

(8) This is straightforward to prove by induction on the complexity of \( t \).

Now the effects of recursion on plus-complexity are considered.

**Lemma 48** Let \( C_S(x) := C_+(x) + ||x|| \). Let \( f \) be defined by \( BPR^5 \) using \( g, h, r, t \) satisfying

1. \( C_+(g(x)) \leq c \cdot (\sum_i C_S(x_i))^{2^{(1 + r + s_1 \cdot t_1)}} \)
2. \( C_+(h(x, z, w)) \leq c \cdot (C_S(z) + C_S(w) + \sum_i C_S(x_i))^{\psi} \) where \( \psi \) is defined to be \( 3^{(1 + r + s_1 + s_1 \cdot t_1)} \).
3. \( C_+(r(x, y)) \leq c \cdot (C_S(y) + \sum_i C_S(x_i))^{k} \) where \( s, s_1 \in L_2 \) and \( c, k \) are constants.
Then there is a constant $k'$ such that

$$C_+(f(n, x)) \leq c \cdot (C_S(n) + \sum_i C_S(x_i))^{3^{|1 + n + \sum_i x_i|^{k'}}}.$$ 

\textbf{Proof.} Note there is loss of generality in assuming the constants $c, k$ are the same in the bound of each $g, h$, and $r$ since if they differed one could always take the maximum of the three values. It follows from the bound on $h$ that

$$C_+(F(n + 1, x)) \leq c \cdot [C_S(n) + C_S(F(n, x))] + \sum_{i=0}^{m} C_S(x_i)]^\psi$$

where $\psi := 3^{|1 + n + f(n, x) + \sum_i x_i|^{k'}}$. By the definition of $BPR^5$, $F(n, x) \leq r(n, x)$. So also $|F| \leq |r|$. Since $r \in L_2$, there is a constant $k'$ such that

$$|r(n, x)| \leq k' \cdot (|n| + \sum_{i=0}^{m} |x_i|) \leq k' \cdot (C_S(n) + \sum_{i=0}^{m} C_S(x_i))$$

Let $k := k' + 1$. Thus, $C_S(F(n + 1, x))$ is less than

$$c[k(C_S(n) + \sum_{i=0}^{m} C_S(x_i)) + C_+(F(n, x))]^{3^{|1 + n + f(n, x) + \sum_i x_i|^{k'}}}.$$ 

Using the bound on $C_+(h)$, one can then expand $C_+(F(n, x))$ and so on. Doing this gives a bound on $C_+(f(n, x)) = C_+(F[\ell(n, x)|, x))$ of $Y :=

$$c \cdot \left(\sum_{i=0}^{m} C_S(x_i) + \sum_{j=0}^{k} k \cdot (C_S(j) + \sum_{i=0}^{m} C_S(x_i))^{3^\psi}\right)$$

where $\psi$ is

$$|1 + \sum_i x_i|^{k} + \sum_{j=0}^{k} (W_j)^k$$

and $W_j$ is

$$|1 + j + F(j, x) + \sum_i x_i|^{k}.$$

The leftmost $\sum_{i=0}^{m} C_S(x_i)$ in $Y$ and the $(|1 + \sum_i x_i|^{k})$ in $\psi$ come from the composition with $g$. Since $t \in L_2$ one can bound $|\ell(n, x)|^{k} \leq M := (|1 + n + \sum_i x_i|^{k''})$ for some constant $k''$ and $n \geq 1$. As $F \leq r \leq 2^{|1 + n + \sum_i x_i|^{k''}}$ for some $d$ one can bound $\psi$ by $\psi' :=

$$(|1 + \sum_i x_i|^{k} + M \cdot (|1 + M + 2^{|1 + M + \sum_i x_i|^{k''}} + \sum_i x_i|^{k})).$$
By choosing $d'$ sufficiently large one can bound $\psi'$ by $\psi'' := (1 + n + \sum_{i} x_i |_{3})^{d'}$. (Notice the term $2^{1 + M + \sum_{i} x_i |_{2}}$ is contained within the $|\cdot|_{5}$.) Now consider the term $W$ under the exponent in $Y$. Since $|f|_{5} \leq M$ and by Lemma 44, $W$ can be bounded by

$$\sum_{i=0}^{m} C_S(x_i) + M \cdot k \cdot (M^3 + 1 + \sum_{i=0}^{m} C_S(x_i)).$$

Notice since $C_{+}(x_i) \geq 1$ and $M \leq C_S(n) + \sum_{i} C_S(x_i)$, there are $a, b$ such that the above and $W$ can be bounded by $(W')^b := a \cdot (C_S(n) + \sum_{i} C_S(x_i))^b$. Thus, $C_{+}(f) \leq (W')^{b \cdot 3^v}$ and one can make this bound into the form of the theorem by choosing a slightly larger value than $d'$ in $\psi''$.

**Corollary 1.** If $f(x) \in A^5$ and $C_{+}(x_i) \leq |x_i|_{3}$ then

$$C_{+}(f(x)) \leq c \cdot (\sum_{i} |x_i|_{3})^{3(1+\varepsilon \cdot 1)^b}.$$  

**Proof.** Follows from Lemma 47 and Lemma 48.

The main result of this section can now be established.

**Theorem 3.** The function $\frac{1}{h}|x|$ is not $E_{1}$-definable in $I^5 E_{1}$. Hence, $I^5 E_{1}$ cannot prove $E_{1} = \Sigma^0_{1}$ and also cannot $D = \Sigma^0_{1}$.

**Proof.** By Theorem 2, the $E_{1}$-functions of $I^5 E_{1}$ are contained in $A^5$. Consider $x := 2^{2^{n+1} - 1}$ for any $n$. So for large enough $x$,

$$C_{+}(x) \leq (|n|)^3 + 7 \leq |x|_{3}$$

since by Lemma 44, $C_{+}(n) \leq (|n|)^3 + 1$. On the other hand, $\frac{1}{h}|x| = \frac{1}{h}|2^{2^{n+1} - 1} - 1|$ is a number of length $||x|| - 1$ of the form $1010 \cdots$. Hence, by Lemma 46,

$$C_{+}(\frac{1}{h}|x|) \geq ||x|| - 2 \geq 2^{2^{1+\varepsilon - 1} - 1} - 2 > c \cdot (|x|_{3})^{3(1-\varepsilon)}$$

for fixed $c, k$ and large enough $x$. The last inequality follows because

$$c \cdot (|x|_{3})^{3(1-\varepsilon)} \leq c \cdot 2^{2^{2^{k} - 1}}$$

for fixed $c, k$, and large enough $x$. So by Corollary 1, $\frac{1}{h}|x|$ is not in $A^5$ and hence not $E_{1}$-definable in $I^5 E_{1}$. But $\frac{1}{h}|x|$ is $\Sigma^0_{1}$-definable in $EBASIC$. So if $I^5 E_{1}$ could prove every $\Sigma^0_{1}$-predicate equivalent to some $E_{1}$-predicate, it could $E_{1}$-define $\frac{1}{h}|x|$. Thus, $I^5 E_{1}$ cannot prove $E_{1} = \Sigma^0_{1}$. It also cannot prove $D = \Sigma^0_{1}$ since every $D$ predicate can be expressed as an $E_{1}$-predicate provably in $I^5 E_{1}$ since even $EBASIC$ can do pairing.
The following is a variation on a result mentioned in the introduction, which is used later to show $I^5E_1$ cannot prove MRDP.

**Theorem 4.** Suppose $T \subseteq S_2$. If $T$ proves the MRDP theorem then $T$ proves $D = NP = PH = \text{co-D}$.  

**Proof.** Suppose $T$ proves the MRDP theorem. Then for every $U_1$-formula $A(x)$ there is a formula $E(x) := (\exists y) P(x,y) = Q(x,y)$ where $P,Q$ are polynomials such that $T \vdash A \equiv E$. In particular, $T$ proves $A \equiv (\exists y) P(x,y) = Q(x,y)$. By Parikh’s theorem (see Hájek and Pudlák [7] for a proof), since $T$ is a bounded theory one can bound the $y$’s by an $L_2$-term $t$. Since $A$ implies this bounded form of $E$ it will also imply $E_2 := (\exists y)[\sum_i y_i] \leq 2^{\sum_j z_i t^j} \land P(x,y) = Q(x,y)]$ for large enough $k$. Note $E_2 \supset E \supset A$ so $A \equiv E_2$. Hence, every $U_1$-formula is equivalent to an $D$-formula. Using this one can show $D = \Delta_0 = \cup_i E_i = \cup_i \Sigma_i = PH$, i.e., the polynomial hierarchy (PH) collapses. Since $\Delta_0$ is closed under complement we get $D = \text{co-D}$.

The next result follows from the previous two theorems.

**Corollary 2.** $I^5E_1 \not\vdash$ the MRDP theorem.

In fact, a stronger theory cannot prove the MRDP theorem.

**Corollary 3.** Let $Z := \cup_i Z_i$ where $Z_i = EBASIC + \Sigma_i + \Pi^{i+3} IND$. Then $Z$ cannot prove the MRDP theorem.

**Proof.** In Pollett [18], it was shown $Z$ cannot prove the collapse of the polynomial hierarchy, but $D = NP = \text{co-D}$ implies the collapse of the polynomial hierarchy.

## 5 Towards $D = NP$

In this section, several statements in the direction of trying to show $D = NP$ will be proven. It will be shown that $I^1E_1$ proves $D$ contains predicates of the form $(\forall i \leq |s|) P \circ Q$ for $\circ$ either $=, \leq$, or $\prec$. A $D$-predicate $A(x) := (\exists y)(\sum_j y_j \leq 2^{\sum_i z_i t^i}) \land P = Q$ is $uD$ in $T$, if $T$ proves $(\exists y) (P = Q) \supset A$. It is shown below that $I^1E_1$ proves its $uD$-functions closed under composition, and under $\land$, and $\lor$. This is used to show $I^1E_1$ can $D$-define $L_2$-terms. Lastly, $D$ is shown to be closed under certain sums of polynomial length and this is used to prove the result.

**Lemma 51** Let $P_i, Q_i$ be polynomials $EBASIC$ proves (1) $(P_1 = Q_1 \land P_2 = Q_2) \iff P_1 + P_2 = Q_1 + Q_2$. (2) $(P_1 = Q_1 \lor P_2 = Q_2) \iff P_1P_2 + Q_1Q_2 = P_1Q_2 + P_2Q_1$ (3) $I^1E_1$ proves its $uD$-definable functions are closed under composition.
Proof. (1) and (2) are straightforward. For (3), let $f$, $g$ be two Diophantine functions. By Parikh's theorem, if $I^0 E_1$ proves $(\forall x)((\exists y)A_f(x, y))$ then there is an $L_2$-term $t_f$ bounding $y$ such that $I^0 E_1 \vdash (\forall x)(\exists y \leq t_f)A_f(x, y)$. Since $y$ was originally unbounded and unique one can choose $t_f$ of the form $2^{|x|} k^2$ for some fixed $k$. So if and $I^0 E_1 \vdash (\forall y)((\exists z)\leq 2^{|y|}k^2)A_f(y, z)$, let $B$ be what's inside the scope of the $(\exists z)$. Then $B$ can be made into a $D$ predicate by choosing a bound of the correct form larger than the bounds on $y$ and in $A_f$ and $A_y$, prenexifying the existentials of $A_f$ and $A_y$ and using (1). To show that this new predicate, $C$, is equivalent to $B$, we use that $A_f$ and $A_y$ are $uD$ for $I^0 E_1$ to show $C \subseteq B$.

Lemma 52. $I^0 E_1$ can $uD$-define $\prec$, $\leq$, $x - y$, and $[x/y]$ and prove basic properties of them.

Proof. Define $LE(x, y)$ as $(\exists w)(w \leq 2|x| + y + w = y)$ and $x < y$ as $LE(Sx, y)$. The goal is to show $I^0 E_1$ proves $LE(x, y) \iff x \leq y$ where $\leq$ is the $L_2$-predicate symbol. That $LE(x, y)$ implies $x \leq y$ involves checking that $LE$ satisfies all of the $EBASIC$ axioms for $\leq$. For instance, $LE(0, y)$ holds assuming $I^0 E_1$ proves $x = y \iff x < y$. For $x \leq y$ implies $LE(x, y)$, use the $EBASIC$ axioms (4) and (20) from Buss [4] and $LIND$ on these axioms to binary search for the $w$ such that $x + w = y$. Define $Sub(x, y) = z$ as $(x = y + z \land (z = 0 \land x < y))$. Since $\prec$ is in $L_1$, $I^0 E_1$ can use this to prove $z$ exists and is unique and in fact Takeuti's [20] axiom for $\prec$ is precisely how $Sub(x, y)$ is defined. Define $[x/y] = z$ as $(\exists w)[w \leq 2|x| + y \land ((y = 0 \land z = 0) \land (w < y \land x + z = y + w))].$ The proof of existence of $z$ in $I^0 E_1$ can be done using $LIND$ to binary search for a value. Uniqueness and being in $uD$ follow from $EBASIC$ axioms.

The next argument is based on an observation of Raphael Robinson [19].

Theorem 5. Let $P$, $Q$ be polynomials. (1) $I^0 E_1$ proves $D$ contains the predicate $B_m(b) := (\forall i \leq b)P(i, x) = Q(i, x)$. (2) $I^0 E_1$ proves $D$ contains the predicate $B_m([b])$.

Proof. First, note the predicate $B_m(b)$ above holds if and only if $P'(b, x) := \sum_{i=0}^{b} P(i, x)$ is equal to $Q'(b, x) := \sum_{i=0}^{b} Q(i, x)$. $P'$ and $Q'$ can be written as polynomials. This is because $\sum_{i=0}^{b} (t_i)^m$ for fixed $m$ is expressible as a degree $m + 1$ polynomial over a fixed number. For example $\sum_{i=0}^{b} (i)^m = b(b + 1)/2$. Let $C(c)$ be the formula: $B_m(c) \iff P'(c, x) = Q'(c, x)$.

A Diophantine representation of exponentiation was given in Matiyasevich [15] by studying the zeros of the equation $p(a, x, y) := x^2 + y^2 - 2ay - 1$. Adleman
and Manders [1] have shown the predicate \( E(x, y, z) := x = y^2 \) is in \( C \) by bounding the existentials in this representation in terms of \( x \). Using \( E(x, y, z) \) one can define \( h(x) = z \) as \((\exists y)(x \leq y \land y \leq 2x \land E(y, 2, z))\) and one can define \( x \# y = z \) as \( E(z, 2, h(x) \cdot h(y)) \). (These are not quite \( C \) predicates but one can combine the bound on \( y \) in \( h \) with that of \( E \) to make it in \( C \).) \( I^1 E_1 \) can prove the existence of \( z \) in \( h(x) = z \) using \( \text{LIND} \) on this predicate and can also show \( h(x) = |x| \) and show \( E \) and \( h \) are \( C \)-predicates. The details about verifying the correctness of the \( E \) predicate in weak theories of arithmetic can be found in Kaye [11]. Together with Lemma 52, \( I^1 E_1 \) can show all the \( L_2 \) base functions are in \( C \) and, hence, also the \( L_2 \)-terms listed in the preliminaries.

**Lemma 53** Fix \( n \) and suppose \(|n| < |i| \) for all \( i \leq |a| \). Then (1) \( I^1 E_1 \) can \( C \)-define \( S_n(|a|) := \sum_{i=0}^{|n|} i^2 2^{i|n|} \) and (2) prove \( S_n(2a) = S_n(|a| + 1) = S_n(|a|) + (|a| + 1)^n \).

**Proof.** The proof is by induction (in the real world) on \( n \). For \( n = 0 \) note \( I^1 E_1 \) can \( C \)-define \( S_0(|a|) \) using the formula for summing a geometric series: \( \frac{2^{2|a|}(|a|) - 1}{2|a| - 1} \).

Now assume for \( n' < n + 1 \) we have defined \( S_{n'}(|a|) \) in \( I^1 E_1 \) and proven (2). Then one can define \( S_{n+1}(|a|) = x \) as

\[
(2^{x|a|} - 1)x = |a|^{n+1} 2^{|n|} = \sum_{i=0}^n (-1)^{n+1-i} \binom{n+1}{i} (S_i(2a) - S_i(|a|)).
\]

The value of an \( x \) satisfying the above can be bounded by the right hand side of the equation. \( I^1 E_1 \) can verify (2) for \( S_{n+1} \) by \( \text{LIND} \) on \( c \) for the equation for \( S_{n+1}(\min(c, |2a|)) \), the induction step using that \( S_n \) is correctly defined. The above definition is motivated by the appendix on summing geometric equations in Matiyasevich [16] and by Kaye [13]. Note that \( n \) in the above (and so the \((n+1)\)’s) are fixed natural numbers.

**Lemma 54** Suppose \(|P(i, y)| < |i| \) for all \( i \leq |i| \). Then \( I^1 E_1 \) can (1) \( C \)-define \( S(|a|) = \sum_{|i|=0}^{|a|} P(i, y)^i \cdot 2^{i|n|} \) and (2) prove \( S(2a) = S(|a| + 1) = S(|a|, y) + P(|a| + 1, y) \).

**Proof.** Rewrite \( P \) as \( \sum_{j=0}^d P_j(y)^i \) where the \( P_j \)'s are polynomials in the remaining variables. So \( S \) equals

\[
\sum_{i=0}^{|a|} \left( \sum_{j=0}^d P_j(y)^i \right) \cdot 2^i|n|
\]

which in turn equals

\[
\sum_{j=0}^d P_j(y) \left( \sum_{i=0}^{|a|} i^j \cdot 2^i|n| \right).
\]

Each of the outer summands is just a product of a polynomial with the kind of sums we showed \( C \)-definable in Lemma 53 (1). So \( S \) is \( C \)-definable in \( I^1 E_1 \). Similarly using Lemma 53 (2) establishes (2) above.
**Theorem 6.** Let \( P, Q \) be polynomials. \( I^1 E_1 \) proves \( B_\prec ([b], y) := (\forall i \leq |b|) P(i, y) \circ Q(i, y) \) is equivalent to a \( D \)-predicate where \( \circ \) is either \(< \) or \( \leq \).

**Proof.** Only the \( B_\prec \) case is shown as the proofs of the two cases are essentially the same. Since \( P, Q \) have nonnegative integer coefficients they are nondecreasing in each variable. Hence, \( I^1 E_1 \) proves \( P(i, y) \leq P([b], \ldots, [b], y) \) and \( Q(i, y) \leq Q([b], \ldots, [b], y) \). Let \( m := |P([b], \ldots, [b], y) \cdot Q([b], \ldots, [b], y)] \). Now \( B_\prec \) holds iff we have

\[
|b| \sum_{i_j = 0}^{v} \cdots \sum_{i_1 = 0}^{v} (P(i, y) + 1 + \beta(v, m, w)) 2^v
\]

where \( v = \sum_{j=0}^{k} i_j \cdot m^j \) is equal to

\[
|b| \sum_{i_1 = 0}^{v} \cdots \sum_{i_k = 0}^{v} Q(i, y) 2^\sum_{j=0}^{k} i_j \cdot m^j
\]

for some polynomial length string \( w \). Let \( L([b], y, w) \) and \( R([b], y) \) be the first and second sums respectively. Using Lemma 54 and that the \( L_2 \)-terms are \( aD \)-definable, \( I^1 E_1 \) can \( D \) define \( L \) and \( R \). So LIND on \( c \) in \( C(c) := B_\prec (\min(c, [b]), y) \Leftrightarrow (\exists w \leq t)(L(\min(c, [b]), y, w) = R(\min(c, [b]), y)) \)

for some large enough term \( t \) gives the result. The induction step uses Lemma 54(2).

As mentioned in the introduction \( I^1 E_1 \) will have LIND for this predicate since it is a boolean combination of \( E_1 \)-formulas.

### 6 Logtime classes and \( D \)

As was mentioned in the introduction the author conjectures that techniques similar to those in the last section might be able to show NLOGTIME is contained in \( D \). In this section, it is shown that this result would not suffice to prove \( D = \text{NP} \). The possibility of proving \( D = \text{NP} \) by showing equality is hard for \( \text{co-NLOGTIME} \) under some kind of reduction computable in \( D \) is then considered. The model of sublinear time used in this section is the standard one: The machines used are allowed to write an index of an input tape square on a query tape, enter a query state, and in one step get the value of that square of the input.

**Definition 61.** Let \( C \) be a class of languages. Then \( \text{E}C \) denotes the class of languages in \( (\exists y \leq s) \langle x, y \rangle \in L \) for some \( s \) an \( L_2 \)-term and \( L \) in \( C \).

One might conjecture that \( \text{ELOGTIME} = \text{NP} \) or the weaker conjecture that \( \text{ENPOLYLOGTIME} = \text{NP} \). It is established next that this is not possible.

**Theorem 7.** Let \( t(|x|) \) be monotonic \( L_2 \)-terms so that \( t(|x|) \geq \log |x|, t(|x|^k) \in O(|t(|x|)|^k) \subseteq o(|x|) \) for all \( k \). Then

\[
\text{ENTIME}[t^{O(1)}] = \text{NTIME}[t^{O(1)}].
\]
The case of a single input variable \(x\) is shown but the method can be generalized to any number of inputs.

**Proof.** By the definition if \(L\) is in \(\text{ENTIME}[t^{O(1)}]\), there is some \(L_2\)-term \(s\) and \(L'\) in \(\text{NTIME}[t^{O(1)}]\) such that

\[
x \in L \iff (\exists y \leq s)(x, y) \in L'.
\]

Note since \(s\) is an \(L_2\)-term then \(s\) can be bounded above by \(2^p(|x|)\) for some polynomial \(p\). Thus, there is some nondeterministic machine \(M\) for \(L'\) such that on inputs \(x\) and \(y \leq s(x)\) decides if \(x \in L\) in time bounded by \([t(|x| + p(|x|))]^k \leq t(|x|)^k'\) for some constants \(k, k'\). Without loss of generality one can assume \(M\) never queries the \(|s| - 1\)st bit of \(y\), so if \(M\) makes a query to the \(i\)th bit of \(y\) there are values of \(y \leq s\) so that this value is both 1 and 0. Let \(N\) be the machine that simulates \(M\) on input \(x\) but whenever \(M\) makes a query of a bit value of \(y\), \(N\) checks in a table it stores on another tape if that query has been made before. If it has it uses the value it stored for that query. Otherwise, \(N\) nondeterministically guesses 1 or 0 as a returned value. Storing the value of which bit position was queried takes less than \(\log|s| \leq t(|x|)^m\) bits for some \(m\) so \(N\)'s runtime is in \(\text{NTIME}[t^{O(1)}]\). If there was a value of \(y\) such that \(M\) would accept, then the bits used in this \(y\) give a guess path to an accepting path in \(N\). Since all possible bit settings for \(y\)'s occur in values less than \(s\) if there were some nondeterministic choices that make \(N\) accept, one could set the appropriate bits of \(y\) accordingly and make \(M\) accept.

**Corollary 4.** \(\text{ENPOLYLOGTIME} \subseteq \text{NP}.\)

**Proof.** By the Theorem 7

\[
\text{ENPOLYLOGTIME} \subseteq \text{NLIN} \subseteq \text{NP}.
\]

The last strict inclusion is by the nondeterministic time hierarchy theorem.

It might be possible to show \(\mathcal{D} = \text{NP}\) by showing equality is hard for \(\text{co-NLOGTIME}\) under some kind of reduction. As defined in the introduction, a polynomially bounded functions is in \(\text{FDLOGTIME}\) if each bit of its output is \(\text{DLOGTIME}\) computable. If \(\text{NLOGTIME}\) is in \(\mathcal{D}\) then computing any individual bit of an \(\text{DLOGTIME}\) functions will be in \(\mathcal{D}\); however, it would still be unclear if given a \(\text{DLOGTIME}\) function whether its graph would be in \(\mathcal{D}\). The author conjectures it will be and the next theorem shows that this conjecture would imply \(\mathcal{D} = \text{NP}\). Recall from the introduction that the predicate \(x \leq_2 y\) holds if the \(i\)th bit of \(x\) is less than the \(i\)th bit of \(y\) for all \(i\).

**Theorem 8.** \(x \leq_2 y\) and equality are \(\text{co-NLOGTIME}\)-complete under \(\text{FDLOGTIME}\) reductions.

**Proof.** Both problems are in \(\text{co-NLOGTIME}\) as one can universally guess a bit position \(i\) and then in \(\text{DLOGTIME}\) check if either \(\text{BIT}(i, x) \leq \text{BIT}(i, y)\) or
\(BIT(i, x) = BIT(i, y)\). Let \(M\) be a co-NLOGTIME machine and \(x\) an input. Without loss of generality one can assume each path in the computation tree of \(M\) is of length \(k \log |x|\). The FDLOGTIME reduction outputs two numbers \(w, v\) to be input to the \(\leq_2\) predicate as \(w \leq_2 v\). It sets the \(i\)th bit of \(w\) to 1 if \(M\) on path \(i\) rejected and to 0 otherwise. It sets the \(i\)th bit of \(v\) always to 0. Since on a fixed path \(M\)'s computation is DLOGTIME, this reduction will be in FDLOGTIME. Thus, \(w \leq_2 v\) iff for every bit of \(w\) is 0 and this happens iff \(M\) accepted on every path. This in turn implies \(x \in L(M)\). To reduce \(x, M\) to equality let the \(i\)th bit of \(w\) be 1 iff \(M\) accepts on path \(i\) and let the \(i\)th bit of \(v\) be 1. Again, this reduction is in FDLOGTIME and \(w = v\) iff \(M\) accepts on every path iff \(x \in L(M)\).

7 Conclusion

Several lines of attack on the \(D = \text{NP}\) question seem promising. It is quite likely one can find a better complexity notion than plus-complexity such that for all the \(L_2\) base functions the complexity of the output can be bounded linearly in the outputs. This measure would also have to be useful in that one should be able to use the excluded middle trick to define some function whose complexity cannot easily be bounded this new measure. Nevertheless, such a measure might exist and allow one to show \(I^m E_1\) cannot prove \(D = \text{NP}\) for some \(m < 5\).

Towards showing that \(D = \text{NP}\) holds, the approach of considering reductions to equality might still bear fruit. The first step would be to show a complexity class like NLOGTIME or DLOGTIME is contained in \(D\). This would not immediately imply FDLOGTIME is contained in \(D\). But maybe some weaker reduction would suffice. In any case, it is useful to make conjectures which aren't already equivalent to \(D = \text{NP}\) in the hopes that they either lead one towards a proof of \(D = \text{NP}\) or their refutations gives some new insight on the problem. For instance, one reducibility which is presumably weaker than FDLOGTIME reducibility is to consider functions whose graphs are recognizable in DLOGTIME (or NLOGTIME). A reasonable conjecture is that equality testing is hard for co-RLOGTIME under this kind of reducibility. If DLOGTIME is in \(D\), this would suffice to show \(D = \text{NP}\). On the hand, \(D = \text{NP}\) does not necessarily imply this conjecture. Unfortunately, these kinds of reductions are probably too weak to show the desired hardness result.

References