Bounded Queries and the NP Machine Hypothesis

Richard Chang[†]
chang@umbc.edu

Suresh Purini[†]
suresh1@umbc.edu

University of Maryland Baltimore County

Abstract

The NP *machine hypothesis posits the existence of an* $\epsilon > 0$ *and a nondeterministic polynomial-time Turing machine* M *which accepts the language* 0^* *but for which no deterministic Turing machine running in* 2^{n^ϵ} *time can output an accepting path infinitely often. This paper shows two applications of the* NP *machine hypothesis in bounded query complexity. First, if the* NP *machine hypothesis holds, then*

$$\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \implies \mathrm{PH} \subseteq \mathrm{NP}.$$

Without assuming the NP *machine hypothesis, the best known collapse of the Polynomial Hierarchy (*PH*) is to the class* $\mathrm{S}_2^{\mathrm{P}}$ *due to a result of Fortnow, Pavan and Sengupta [9].*

The second application is to bounded query function *classes. If the* NP *machine hypothesis holds then for all constants* $d > 0$*, there exists a constant* $k > d$ *such that for all oracles* X*,*

$$\mathrm{PF}^{\mathrm{SAT}[n^k]} \not\subseteq \mathrm{PF}^{X[n^d]}.$$

In particular, $\mathrm{PF}^{\mathrm{SAT}[n^d]} \subsetneq \mathrm{PF}^{\mathrm{SAT}[n^k]}$*. Without the* NP *machine hypothesis, there are currently no known consequences even if for all* $k > 1$, $\mathrm{PF}^{\mathrm{SAT}[n^k]} \subseteq \mathrm{PF}^{\mathrm{SAT}[n]}$.

1 Introduction

In the study of bounded query complexity classes, we would like to show that each additional query to a hard oracle (e.g., SAT) provides the polynomial-time base machine with additional computational power — either to recognize new languages or to compute new functions. Of course, we cannot prove such results without also showing that $\mathrm{P} \neq \mathrm{NP}$, so the theorems in the area tend to show that if additional queries do not provide additional computational power, then some intractability assumption would be violated (e.g., $\mathrm{P} = \mathrm{NP}$ or PH collapses).

In the case of bounded query language classes, Kadin [13] pioneered the hard/easy argument and showed that for all constants k, $\mathrm{P}_{\mathrm{tt}}^{\mathrm{SAT}[k]} = \mathrm{P}_{\mathrm{tt}}^{\mathrm{SAT}[k+1]} \implies \mathrm{PH} \subseteq \Sigma_3^{\mathrm{P}}$. Subsequent improvements concentrated on collapsing PH to a lower level [8, 5] through efficient searches for "hard strings." For technical reasons that we will not go into here, the case of one query versus two queries is a special case. By considering whether the input string itself might be a hard string, *downward collapses* can be achieved for queries to Σ_i^{P} for $i \geq 2$ [10, 6]:

$$\mathrm{P}^{\Sigma_i^{\mathrm{P}}[1]} = \mathrm{P}^{\Sigma_i^{\mathrm{P}}[2]} \iff \mathrm{PH} \subseteq \Sigma_i^{\mathrm{P}}.$$

These are downward collapses because PH collapses to a level that is lower than that assumed in the premise. Such downward collapses have not been shown either for larger number of queries nor for the $\Sigma_1^{\mathrm{P}} = \mathrm{NP}$ case.

More recently Fortnow, Pavan and Sengupta [9] showed that if $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]}$ then $\mathrm{PH} \subseteq \mathrm{S}_2^{\mathrm{P}}$. Combined with the results of Cai [7], showing that $\mathrm{S}_2^{\mathrm{P}} \subseteq \mathrm{ZPP}^{\mathrm{NP}}$, and of Buhrman and Fortnow [6], showing that $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \implies \mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}}$, we get a very unsatisfying collapse of PH:[1]

$$\begin{aligned}\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} &\implies \mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}} \subseteq \mathrm{ZPP}^{\mathrm{SAT}[1]}\\ &\implies \mathrm{P}^{\mathrm{SAT}} \subseteq \mathrm{ZPP}^{\mathrm{SAT}[2]} = \mathrm{ZPP}^{\mathrm{NP}}\\ &\implies \mathrm{P}^{\mathrm{SAT}} \subseteq \mathrm{ZPP}^{\mathrm{NP}} = \mathrm{PH}\end{aligned}$$

[†]Address: Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250, USA.

[1]Here $\mathrm{ZPP}^{\mathrm{NP}}$ does not collapse to $\mathrm{ZPP}^{\mathrm{NP}[1]}$ because a ZPP computation path can end in accept, reject or "don't know." It takes two queries to narrow the 3 choices down to 1.

The situation cries out for a complete collapse of PH all the way down $\mathrm{P}^{\mathrm{SAT}[1]}$. However, our attempts to show that $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \implies \mathrm{ZPP}^{\mathrm{SAT}[1]} = \mathrm{ZPP}^{\mathrm{SAT}[2]}$ have not been fruitful. Nor were forays into proving $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \implies \mathrm{ZPP}^{\mathrm{SAT}[1]} \subseteq \mathrm{P}^{\mathrm{SAT}}$. It appears to be time to consider additional assumptions.

One approach is to look at assumptions that derandomize $\mathrm{BPP}^{\mathrm{NP}}$ [21, 17, 14, 3, 22]. Then $\mathrm{P}^{\mathrm{NP}} = \mathrm{ZPP}^{\mathrm{NP}} = \mathrm{BPP}^{\mathrm{NP}}$ would give us a complete collapse:

$$\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \implies \mathrm{PH} \subseteq \mathrm{P}^{\mathrm{SAT}[1]}.$$

For the results in this paper, we make the stronger assumption that NP does not have p-measure zero. We show that under this measure hypothesis, $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]}$ leads to a collapse of PH all the way down to NP. This is a *downward collapse* of PH below the original assumption that $\mathrm{P}^{\mathrm{SAT}[2]}$ collapses to $\mathrm{P}^{\mathrm{SAT}[1]}$. Thus, under the measure hypothesis, we are able to achieve the analogous downward collapse previously achieved for Σ_i^{P} where $i \geq 2$.

Theorem 1 If NP does not have p-measure zero, then $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \iff \mathrm{PH} \subseteq \mathrm{NP}$.

We note that Buhrman and Fortnow [6] constructed a relativized world where $\mathrm{NP} \neq \mathrm{coNP}$ but $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} = \mathrm{PSPACE}$. Thus, we can add the two queries problem to a long list of problems that have not been resolved with "traditional" intractability assumptions (e.g., $\mathrm{P} \neq \mathrm{NP}$, PH does not collapse), but have been resolved under the measure hypothesis [16, 19, 18, 20].

In our proofs, we do not work with p-measure or with martingales directly. Instead we make use of the recent work of Hitchcock and Pavan [11] who showed that the measure hypothesis implies the NP machine hypothesis. This hypothesis posits the existence of an $\epsilon > 0$ and a nondeterministic polynomial-time Turing machine M which accepts the language 0^* but for which no deterministic Turing machine running in 2^{n^ϵ} time can output an accepting path infinitely often. We modify Buhrman and Fortnow's proof that $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]}$ implies that "locally" either $\mathrm{NP} = \mathrm{coNP}$ or $\mathrm{SAT} \in \mathrm{P/poly}$ [6]. By carefully redefining their hard and easy strings, we can show that the advice used in the P/poly case can be made polynomially shorter than the input string. This allows us to compute satisfiability in subexponential time in such a way that violates the NP machine hypothesis. Thus, the P/poly case can only occur finitely often, which gives us $\mathrm{NP} = \mathrm{coNP}$ almost everywhere and thus $\mathrm{NP} = \mathrm{coNP}$ by finite patching.

Downward collapse is not as difficult to obtain for bounded query *function* classes. Some of the first results [15, 4] showed that for $q(n) \leq c \log n$, where $c < 1$,

$$\mathrm{PF}^{\mathrm{SAT}[q(n)]} \subseteq \mathrm{PF}^{\mathrm{SAT}[q(n)-1]} \implies \mathrm{P} = \mathrm{NP}.$$

For $q(n) = O(\log n)$, the results use the assumption that PH does not collapse [2, 1]:

$$\mathrm{PF}^{\mathrm{SAT}[q(n)]} \subseteq \mathrm{PF}^{\mathrm{SAT}[q(n)-1]} \implies \mathrm{PH} \subseteq \Sigma_3^{\mathrm{P}}.$$

Not much is known when the number of queries is larger than $O(\log n)$. Thus, we are left in a strange situation where we "know" that 2 queries is more powerful than 1 query, but it is possible that n^2 queries is not any more powerful than n queries. Perhaps after n queries to SAT, a polynomial-time machine can learn everything it needs to know about satisfiability, making additional queries useless. Our second result shows that under the measure hypothesis, this cannot happen:

Theorem 2 If NP does not have p-measure zero, then for any constant $d > 0$, there exists a constant $k > d$ such that

$$\mathrm{PF}^{\mathrm{SAT}[n^d]} \subsetneq \mathrm{PF}^{\mathrm{SAT}[n^k]}.$$

Theorem 2 shows that under the measure hypothesis, even with polynomially many queries to SAT at your disposal, more queries will eventually let you compute more functions. Again, the proof uses the NP machine hypothesis. This time we modify one of Krentel's proof to achieve the desired results.

In the next section of the paper, we give formal definitions to the notions mentioned above. Readers familiar with the definition of the NP machine hypothesis and of the bounded query classes can safely skip over this section since we do not introduce any new terminology. In Sections 4 and 5, we prove the two main theorems of the paper.

2 Preliminaries

Definition 3 [11] **NP machine hypothesis:** There exist an $\epsilon > 0$ and an NP machine M such that $L(M) = 0^*$ and for any 2^{n^ϵ}-time bounded deterministic Turing machine D, the number of lengths n where $D(0^n)$ outputs an accepting path of $M(0^n)$ is finite.

By a simple padding argument, the NP machine hypothesis can be put in a more useful form:

Fact 4 [12] Assuming that the NP machine hypothesis holds, then for any polynomial n^d, there exists an NP machine N such that $L(N) = 0^*$ and for any 2^{n^d}-time bounded deterministic Turing machine D, the number of lengths n where $D(0^n)$ outputs an accepting path of $N(0^n)$ is finite.

Hitchcock and Pavan [11] showed a strong connection between resourced-bounded measure and the NP machine hypothesis. Lutz [16] defined a resource-bounded analog of

Lebesgue measure (from real analysis) to "investigate the distribution of nonuniform complexities in uniform complexity classes." Since then the assumption that NP does not have p-measure zero (often called the measure hypothesis), has been used to obtain results where more traditional intractability assumptions have failed.

Definition 5 A *martingale* is a function $d : \Sigma^* \to [0, \infty)$ with the property that for all $w \in \Sigma^*$, $d(w) = (d(w0) + d(w1))/2$. A martingale *succeeds* on a language $A \subseteq \Sigma^*$ if

$$\limsup_{n\to\infty} d(A \upharpoonright n) = \infty,$$

where $A \upharpoonright n$ is the length n prefix of A's characteristic sequence. A class $\mathcal{C}$ of languages has p-measure 0, if there exists a polynomial-time computable martingale that succeeds on every $A \in \mathcal{C}$.

Fact 6 [11] If NP does not have p-measure zero, then the NP machine hypothesis holds.

Definition 7 Let $q(n)$ be a polynomial-time computable function and X be any language. We use $\mathrm{P}^{X[q(n)]}$ to denote the class of languages recognized by deterministic polynomial-time Turing machines which make at most $q(n)$ serial queries (a.k.a. adaptive queries) to the oracle X on inputs of length n. When the queries are made in parallel (non-adaptively), we use the notation $\mathrm{P}_{\mathrm{tt}}^{X[q(n)]}$. We use $\mathrm{PF}^{X[q(n)]}$, $\mathrm{PF}_{\mathrm{tt}}^{X[q(n)]}$ for the analogous classes of *functions*. Also, when the machines are allowed any polynomial number of queries, we drop the $q(n)$ and use $\mathrm{P}^{\mathrm{SAT}}$, $\mathrm{P}_{\mathrm{tt}}^{\mathrm{SAT}}$, $\mathrm{PF}^{\mathrm{SAT}}$and $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}}$.

3 Two Queries Revisited

In this section we redefine the Easy-I, Easy-II, Easy-III and Easy-IV sets of Buhrman and Fortnow [6]. The main difference here is that we pay special attention to the polynomial bounds of the existential quantifiers. Where a Boolean formula ϕ of length n is concerned, we will quantify over formulas ψ of length n^k (for some constant k to be chosen later). This will allow us to compute satisfiability for formulas of length $\leq n^k$ using advice that has length $O(n)$ in Lemma 17.

Consider the following languages:

$$\begin{aligned}\textsc{Parity} \quad = \quad & \{\, (\phi_1, \phi_2) \mid (\phi_1 \in \mathrm{SAT} \wedge \phi_2 \in \overline{\mathrm{SAT}}) \\ & \vee (\phi_1 \in \overline{\mathrm{SAT}} \wedge \phi_2 \in \mathrm{SAT}) \,\}\end{aligned}$$

$$\overline{\mathrm{SAT}} \wedge \mathrm{SAT} = \{\, (\phi_1, \phi_2) \mid \phi_1 \in \overline{\mathrm{SAT}} \wedge \phi_2 \in \mathrm{SAT}\}$$

$$\mathrm{SAT} \oplus \overline{\mathrm{SAT}} = \{\, (\phi, +) \mid \phi \in \mathrm{SAT} \,\} \cup \{\, (\phi, -) \mid \phi \in \overline{\mathrm{SAT}} \,\}$$

Under the assumption that $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]}$, there exist a $\leq_{\mathrm{m}}^{\mathrm{P}}$-reduction g from $\textsc{Parity}$ to $\mathrm{SAT} \oplus \overline{\mathrm{SAT}}$ and a $\leq_{\mathrm{m}}^{\mathrm{P}}$-reduction h from $\overline{\mathrm{SAT}} \wedge \mathrm{SAT}$ to $\mathrm{SAT} \oplus \overline{\mathrm{SAT}}$. This is because $\mathrm{SAT} \oplus \overline{\mathrm{SAT}}$ is a $\leq_{\mathrm{m}}^{\mathrm{P}}$-complete language for $\mathrm{P}^{\mathrm{SAT}[1]}$.

We now define Easy-I, Easy-II, Easy-III and Easy-IV formulas.

Definition 8 Let ϕ be a Boolean formula of length n and k be any constant.

1. ϕ is called Easy-I if $\exists \psi, |\psi| \leq n^k$, and $h(\phi, \psi) = (\tau, +)$ and $\tau \in \mathrm{SAT}$.
2. ϕ is called Easy-II if

   i. ϕ is Easy-I or

   ii. ϕ is the leaf of a self-reduction tree and evaluates to *false* or

   iii. The immediate children of ϕ in its self-reduction tree are Easy-I.
3. ϕ is called Easy-III if

   i. ϕ is Easy-II or

   ii. There is an Easy-II formula ψ, $|\psi| \leq n^k$, such that $h(\psi, \phi) = (\tau, -)$ and $\tau \in \mathrm{SAT}$.
4. ϕ is called Easy-IV if

   i. ϕ is Easy-III or

   ii. $\exists \psi \in \mathrm{SAT}$, $|\psi| \leq n^k$ and $g(\phi, \psi) = (\tau, +)$ and $\tau \in \mathrm{SAT}$ or

   iii. There is an Easy-III formula ψ, $|\psi| \leq n^k$, and $g(\phi, \psi) = (\tau, -)$ and $\tau \in \mathrm{SAT}$.

Definition 9

1. Hard-I $= \overline{\mathrm{SAT}} -$ Easy-I
2. Hard-II $= \overline{\mathrm{SAT}} -$ Easy-II
3. Hard-III $= \overline{\mathrm{SAT}} -$ Easy-III
4. Hard-IV $= \overline{\mathrm{SAT}} -$ Easy-IV

The lemmas in the rest of this section follow the original proof of Buhrman and Fortnow [6] quite closely. Our intention here is to verify that the new definitions do not introduce any problems.

Lemma 10 There exist NP machines that recognize the languages Easy-I, Easy-II, Easy-III and Easy-IV.

Proof: It suffices to note that the definition of Easy-I, Easy-II, Easy-III and Easy-IV only involve existential quantifiers. The longest string involved in a quantifier is in the case of Easy-IV. Here we need to verify that a string ψ with length up to n^k is Easy-III. Checking ψ is Easy-III can involve strings as long as n^{3k} which is still polynomial in the length of the original input. $\square$

Lemma 11 Either there exists a Hard-IV formula of length n or there exist polynomial-time verifiable witnesses for all Boolean formulas $\phi \in \overline{\mathrm{SAT}}^{=n}$.

Proof: If there exists no Hard-IV formula of length n, then every Boolean formula $\phi \in \overline{\mathrm{SAT}}^{=n}$ is an Easy-IV formula, thus ensuring a polynomial-time verifiable witness for its unsatisfiability. $\square$

Lemma 12 If ϕ is a Hard-IV formula of length n, and $\psi \in \mathrm{SAT}$, $|\psi| \leq n^k$, then $g(\phi, \psi) = (\tau, -)$ for some τ.

Proof: Suppose by contradiction that $g(\phi, \psi) = (\tau, +)$. Since $\psi \in \mathrm{SAT}$ and $\phi \in \text{Hard-IV} \subseteq \overline{\mathrm{SAT}}$, we have $(\phi, \psi) \in \textsc{Parity}$. Furthermore, since g is a $\leq_{\mathrm{m}}^{\mathrm{P}}$-reduction from $\textsc{Parity}$ to $\mathrm{SAT} \oplus \overline{\mathrm{SAT}}$, we must have $(\tau, +) \in \mathrm{SAT} \oplus \overline{\mathrm{SAT}}$. Thus, $\tau \in \mathrm{SAT}$ which implies that ϕ is an Easy-IV formula, which is a contradiction. $\square$

Lemma 13 If ϕ is a Hard-IV formula of length n, and ψ is an Easy-III formula, $|\psi| \leq n^k$, then $g(\phi, \psi) = (\tau, +)$.

Proof: As in the proof of Lemma 12, suppose by contradiction that $g(\phi, \psi) = (\tau, -)$. Then $\tau \in \mathrm{SAT}$. (This time $(\phi, \psi) \notin \textsc{Parity}$.) Thus, ϕ is an Easy-IV formula, a contradiction. $\square$

Lemma 14 If there is a Hard-I formula ϕ of length n, then there is a Hard-I Easy-II formula α of length n.

Proof: Consider the self-reduction tree of ϕ. Since $\phi \in \overline{\mathrm{SAT}}$, every formula in the tree is unsatisfiable and must be either Easy-I or Hard-I. Thus, there must exist a Hard-I formula α that is the lowest Hard-I formula in the tree. Then α is either a leaf or has two Easy-I children. In either case, α is Easy-II which gives us a Hard-I Easy-II formula. Without loss of generality, formulas in the self-reduction tree have length $n = |\phi|$, thus α is a Hard-I Easy-II formula of length n. $\square$

Lemma 15 If α is a Hard-I Easy-II formula of length n and $\psi \in \mathrm{SAT}$, where $|\psi| \leq n^k$, then $h(\alpha, \psi) = (\tau, -)$.

Proof: Suppose by contradiction that $h(\alpha, \psi) = (\tau, +)$ for some τ. Since we are given that $\alpha \in \overline{\mathrm{SAT}}$ and $\psi \in \mathrm{SAT}$, $(\alpha, \psi) \in \overline{\mathrm{SAT}} \wedge \mathrm{SAT}$. Since h $\leq_{\mathrm{m}}^{\mathrm{P}}$-reduces $\overline{\mathrm{SAT}} \wedge \mathrm{SAT}$ to $\mathrm{SAT} \oplus \overline{\mathrm{SAT}}$, we must have $(\tau, +) \in \mathrm{SAT} \oplus \overline{\mathrm{SAT}}$. Thus, $\tau \in \mathrm{SAT}$, which implies that α is an Easy-I formula, a contradiction. $\square$

Lemma 16 If α is a Hard-I Easy-II formula of length n and ψ is a Hard-III formula such that $n^{1/k} \leq |\psi| \leq n^k$, then $h(\alpha, \psi) = (\tau, +)$.

Proof: Suppose that $h(\alpha, \psi) = (\tau, -)$. Then, $(\alpha, \psi) \notin \overline{\mathrm{SAT}} \wedge \mathrm{SAT}$ implies that $\tau \in \mathrm{SAT}$. Since $|\psi| \geq n^{1/k}$ and $|\alpha| = n$, $|\alpha| \leq |\psi|^k$. Thus, α and τ constitute a witness that ψ is an Easy-III formula and we arrive at a contradiction. $\square$

Lemma 17 There exists a polynomial-time machine D which takes input of the form (ψ, ϕ, α) such that if ϕ is a Hard-IV formula, with $|\phi| = n$, and α is a Hard-I Easy-II formula, with $|\alpha| = n$, then for all ψ, $|\psi| \leq n^k$, $\psi \in \mathrm{SAT}$ if and only if $D(\psi, \phi, \alpha)$ accepts.

Proof: Consider the algorithm for the machine D given below. Assume without loss of generality that $|\psi| \geq n^{1/k}$. If ψ were too short, we could always pad ψ up to any length greater than $n^{1/k}$ and less than n^k.

1. If $g(\phi, \psi) = (\tau, +)$, then reject ψ.
2. Otherwise, if $h(\alpha, \psi) = (\tau, +)$, reject ψ.
3. Otherwise, accept ψ.

By Lemma 12, if $g(\phi, \psi) = (\tau, +)$ then $\psi \notin \mathrm{SAT}$. Thus, D rejects correctly in Step 1. If D proceeds to Step 2, then $g(\phi, \psi) = (\tau, -)$ and by Lemma 13, $\psi \notin$ Easy-III. Hence, in Step 2, either $\psi \in \mathrm{SAT}$ or $\psi \in$ Hard-III. Now, in Step 2, if $h(\alpha, \psi) = (\tau, +)$, then by Lemma 15, $\psi \notin \mathrm{SAT}$. Thus, D rejects correctly in Step 2 as well. Finally, if D proceeds to Step 3, then by Lemma 16, $h(\alpha, \psi) = (\tau, -)$ implies that $\psi \notin$ Hard-III. (This is where we need $|\psi| \geq n^{1/k}$.) Since we already know that either $\psi \in \mathrm{SAT}$ or $\psi \in$ Hard-III, ψ must be in SAT. Thus, D accepts correctly in Step 3. $\square$

By self-reducibility of SAT we have:

Corollary 18 There exists a polynomial-time machine D' which takes input of the form (ψ, ϕ, α) such that if ϕ is a Hard-IV formula, with $|\phi| = n$, and α is a Hard-I Easy-II formula, with $|\alpha| = n$, then for all Boolean formulas ψ, where $|\psi| \leq n^k$, $D'(\psi, \phi, \alpha)$ outputs a satisfying assignment of ψ.

4 The NP Machine Hypothesis and the Two Queries Problem

Lemma 19 Under the NP machine hypothesis, $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \implies \mathrm{PH} \subseteq \mathrm{NP}$.

Proof: By Fact 4, let N be an NP machine accepting 0^* such that no deterministic 2^{n^2}-time bounded machine can compute an accepting computation path of $N(0^n)$ at infinitely many n's. Let k be a constant such that when we apply Cook's reduction to $N(0^n)$ and obtain a Boolean formula ψ, we have $|\psi| \leq n^k$.

Let us assume $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]}$. Now define Easy-$i$ in Definition 8 using the k obtained from Cook's reduction. Then, we can have either infinitely many Hard-IV formulas or not. If there are finitely many Hard-IV formulas, then there exists a constant n_0, such that there exist no Hard-IV formulas with length greater than or equal to n_0. By Lemma 11, we have polynomial-time verifiable witnesses for all $\phi \in \overline{\mathrm{SAT}}^{\geq n_0}$. Thus we have $\mathrm{NP} = \mathrm{coNP}$ by finite patching.

Now let us assume that there are infinitely many Hard-IV formulas. We show that this contradicts the NP machine hypothesis. We construct a machine 2^{n^2}-time bounded machine M which outputs an accepting computation path of $N(0^n)$, whenever there exists a Hard-IV formula ϕ of length n.

Machine M: on input 0^n.

1. Apply Cook's reduction to $N(0^n)$ and let ψ, $|\psi| \leq n^k$, be the Boolean formula obtained after applying Cook's reduction.
2. For all Boolean formulas ϕ and α such that $|\phi| = |\alpha| = n$:
   (a) Run D' on input (ψ, ϕ, α). (*Remark:* D' is the machine from Corollary 18.)
   (b) When D' outputs a string x, check if x is a satisfying assignment for ψ. If so, recover an accepting computation path π of $N(0^n)$ from x and output the path π.

Fix a length n where there exists a Hard-IV formula ϕ of length n. Recall from Lemma 14 that a Hard-I Easy-II formula exists whenever there is a Hard-I formula. Since Hard-IV formulas are also Hard-I, we are guaranteed to have a Hard-I Easy-II formula α of length n. Observe that if D' is supplied with a Hard-IV formula ϕ and a Hard-I Easy-II formula α in Step 2 of M, then M will output an accepting computation path of $N(0^n)$. Since M tries all possible ϕ and α of length n, during some iteration of the loop in Step 2, a Hard-IV ϕ and a Hard-I Easy-II α will be used. Note that it is possible that M provides an output when ϕ is not Hard-IV or when α is not Hard-I Easy-II. However, M always verifies that the assignment x produced by D' is satisfying, so M never produces an incorrect output.

Finally we note that the running time of M is bounded by $2^{2n}p(n)$, for some polynomial $p(n)$. Step 2 takes 2^{2n} iterations to cycle through all ϕ and α, where $|\phi| = |\alpha| = n$, and each iteration takes some polynomial time bounded by $p(n)$. The $2^{2n}p(n)$ running time grows slower than 2^{n^2}. Thus we have a 2^{n^2}-time bounded machine which outputs accepting computation paths of $N(0^n)$ at infinitely many n's. This contradicts the NP machine hypothesis. Thus the lemma is proved. □

Theorem 1 If NP does not have p-measure zero, then $\mathrm{P}^{\mathrm{SAT}[1]} = \mathrm{P}^{\mathrm{SAT}[2]} \iff \mathrm{PH} \subseteq \mathrm{NP}$.

Proof: Since the measure hypothesis implies the NP machine hypothesis [11], we have the forward direction from Lemma 19. The other direction is trivial. □

5 The NP Machine Hypothesis and Bounded Query Functions

Lemma 20 Under the NP machine hypothesis, for any constant $d > 0$, there exists a constant $k > d$ such that for all oracles X, $\mathrm{PF}^{\mathrm{SAT}[n^k]} \not\subseteq \mathrm{PF}^{X[n^d]}$.

Proof: By Fact 4, let N be an NP machine accepting 0^* such that no $2^{n^{d+1}}$-time bounded machine can compute an accepting path of $N(0^n)$ at infinitely many n's. Let ϕ be the Boolean formula obtained by applying Cook's reduction to $N(0^n)$. As before, $|\phi| \leq n^k$ for some constant k. We can see that a $\mathrm{PF}^{\mathrm{SAT}[n^k]}$ machine will be able to compute a satisfying assignment for ϕ if one exists. Using this satisfying assignment we can obtain an accepting computation path for $N(0^n)$. Let us assume that $\mathrm{PF}^{\mathrm{SAT}[n^k]} \subseteq \mathrm{PF}^{X[n^d]}$ for some oracle X. Then there exists an equivalent $\mathrm{PF}^{X[n^d]}$ machine M which on input 0^n outputs an accepting computation path of $N(0^n)$. Now we can construct a deterministic machine M' to traverse the oracle query tree of $M(0^n)$ and at each leaf of the tree, M' checks whether the string π output by M at that leaf is in fact an accepting computation path of $N(0^n)$. If so, M' outputs π as an accepting computation path of $N(0^n)$. Otherwise, M' continues traversing the rest of the oracle query tree. The machine M' terminates in $2^{n^d}p(n)$ time, for some polynomial $p(n)$. Since $2^{n^d}p(n)$ is asymptotically less than $2^{n^{d+1}}$, we arrive at a contradiction to the NP machine hypothesis.

Finally note that if $k \leq d$, then we already have a $\mathrm{PF}^{\mathrm{SAT}[n^d]}$ machine that outputs accepting computation paths of $N(0^n)$ without assuming $\mathrm{PF}^{\mathrm{SAT}[n^k]} \subseteq \mathrm{PF}^{X[n^d]}$. The machine M' described above will contradict the NP machine hypothesis. Thus, k must be greater than d. □

Theorem 2 If NP does not have p-measure zero, then for any constant $d > 0$, there exists a constant $k > d$ such that $\mathrm{PF}^{\mathrm{SAT}[n^d]} \subsetneq \mathrm{PF}^{\mathrm{SAT}[n^k]}$.

We now strengthen Theorem 2 for bounded query function classes of the form $\mathrm{PF}^{\mathrm{SAT}[n^\delta]}$ for certain values of $\delta < 1$.

Theorem 21 Suppose the NP machine hypothesis holds. Then, there exist constants d and k such that $d < k$ and

(a) for all δ, $0 \leq \delta < d/k$, for all ϵ, $\epsilon - \delta > 1 - d/k$, and for all oracles X, $\mathrm{PF}^{\mathrm{SAT}[n^\epsilon]} \not\subseteq \mathrm{PF}^{X[n^\delta]}$.

(b) for all $\epsilon > 1 - d/k$, for all oracles X, for all $i \geq 0$, $\mathrm{PF}^{\mathrm{SAT}[n^\epsilon]} \not\subseteq \mathrm{PF}^{X[\log^i n]}$.

Proof: By Fact 4 and the NP machine hypothesis, we have a nondeterministic polynomial-time Turing machine N accepting 0^*, such that no deterministic Turing machine running in 2^{n^d} time can compute an accepting computation path of $N(0^n)$ for infinitely many n's. Let n^k be an upper bound on the length of the Boolean formula obtained by running Cook's reduction on $N(0^n)$.

Let us assume $\mathrm{PF}^{\mathrm{SAT}[n^\epsilon]} \subseteq \mathrm{PF}^{X[n^\delta]}$ for some oracle X and some δ and ϵ such that $0 \leq \delta < \epsilon \leq 1$. Consider a $\mathrm{PF}^{\mathrm{SAT}[n^\epsilon]}$ Turing machine M_1 which takes as input $\langle\phi, w\rangle$, where ϕ is a Boolean formula and w is a bit string such that $|w| \leq |\phi|$. The bit string w contains an assignment to some or all of the variables in ϕ. So it is valid to assume that $|w| \leq |\phi|$. The machine M_1 computes a bit string x, $|x| = n^\epsilon$, where $n = |\langle\phi, w\rangle|$ such that wx is the prefix of the lexicographically maximum satisfying assignment among all satisfying assignments of ϕ with w as prefix. If no satisfying assignment of ϕ has w as prefix, the machine M_1 can output arbitrary x. Since $\mathrm{PF}^{\mathrm{SAT}[n^\epsilon]} \subseteq \mathrm{PF}^{X[n^\delta]}$, there exists a $\mathrm{PF}^{X[n^\delta]}$ machine M_2 which computes the same function as M_1.

Now consider the following deterministic Turing machine M_3 which computes an accepting computation path of $N(0^n)$. We will analyze the running time of M_3 later.

```
$M_3(0^n)$ :

1. Apply Cook's reduction to $N(0^n)$. Let $\phi$, $|\phi| = m \leq n^k$, be the Boolean formula obtained after Cook's reduction.

2. Call $M_3\_SUB(\phi, \lambda)$, where $\lambda$ is the null string. Output an accepting computation path of $N(0^n)$ by decoding the satisfying assignment returned by the function call to $M_3\_SUB$.

$M_3\_SUB(\langle\phi, w\rangle)$ :

1. If $|w| \geq$ #variables in $\phi$
    (a) Substitute the bit values in $w$ (starting from left) for the variables in $\phi$. If $\phi$ evaluates to TRUE return $w$. Otherwise return FALSE.

2. For all $x \in \{0,1\}^{|\langle\phi,w\rangle|^\delta}$ do
    (a) Traverse the query tree of $M_2^X(\langle\phi, w\rangle)$ using the bit string $x$. Let $x'$, $|x'| = |\langle\phi, w\rangle|^\epsilon$, be the bit string at the corresponding leaf.
    (b) Call $M_3\_SUB(\langle\phi, wx'\rangle)$.
```

Let us upper bound the running time of M_3_SUB on input $\langle\phi, \epsilon\rangle$, where $|\phi| = m \leq n^k$. Each recursive call to M_3_SUB determines at least m^ϵ variables of ϕ. Since the recursion terminates when all the variables of ϕ are determined, the depth of the recursion stack is at most $m^{1-\epsilon}$. Furthermore, in each procedure call to M_3_SUB, we try at most $2^{(2m)^\delta}$ possible bit strings. Essentially, we have a tree of depth at most $m^{1-\epsilon}$ and the degree of each non-leaf node is bounded by $2^{(2m)^\delta}$. Thus, we have the overall running time of M_3 bounded by $cm2^{(2m)^\delta m^{1-\epsilon}}$, for some constant c. The linear factor is the time taken to verify that an assignment to the variables of ϕ is satisfying.

Substituting n^k for m, we have:

$$cm2^{(2m)^\delta m^{1-\epsilon}} = cn^k 2^{(2n^k)^\delta n^{k(1-\epsilon)}}.$$

Now to arrive at a contradiction to the NP machine hypothesis, we need the following condition to hold asymptotically.

$$\begin{array}{rrcl} & cn^k 2^{(2n^k)^\delta n^{k(1-\epsilon)}} & \leq & 2^{n^d} \\ \Longleftarrow & n^k 2^{(2n^k)^\delta n^{k(1-\epsilon)}} & \leq & 2^{n^d} \\ \Longleftarrow & 2n^{k\delta+k-k\epsilon} & < & n^d \\ \Longleftarrow & k\delta + k - k\epsilon & < & d \\ \Longleftarrow & \epsilon & > & \delta + 1 - d/k \end{array}$$

So whenever $\epsilon - \delta > 1 - d/k$, we have that $\mathrm{PF}^{\mathrm{SAT}[n^\epsilon]} \subseteq \mathrm{PF}^{X[n^\delta]}$ implies a contradiction to the NP machine hypothesis. This proves the part (a) of the theorem.

We can use the same technique as used to prove part (a) of the theorem to analyze the case of $\mathrm{PF}^{\mathrm{SAT}[n^\epsilon]} \subseteq \mathrm{PF}^{X[\log^i n]}$. We can see that to arrive at a contradiction to the NP machine hypothesis we need the following condition to hold asymptotically.

$$\begin{array}{rrcl} & m2^{m^{1-\epsilon}\log^i(2m)} & \leq & 2^{n^d} \\ \Longleftarrow & m^{1-\epsilon}\log^i(2m) & < & n^d \\ \Longleftarrow & n^{k(1-\epsilon)}((k+1)\log n)^i & < & n^d \\ \Longleftarrow & n^{k(1-\epsilon)} & < & n^d \\ \Longleftarrow & k(1-\epsilon) & < & d \\ \Longleftarrow & \epsilon & > & 1 - d/k \end{array}$$

Thus we have the proof for the part (b) of the theorem. □

It is interesting to note from part (a) of Theorem 21 that if the value of k is close to d, then the difference between ϵ and δ that is required to contradict NP machine hypothesis also decreases. Intuitively it indicates that the harder the languages in NP, the tighter the bounded query function hierarchy. Also, observe that whenever $\delta \leq d/k$, we can choose an $\epsilon \leq 1$. Similarly we can observe from part (b) of Theorem 21, the closer the value of k to d, the smaller the ϵ required to contradict the NP machine hypothesis.

Corollary 22 If NP does not have p-measure zero, then there exist constants d and k such that $d < k$ and

(a) for all δ, $0 \leq \delta < d/k$, and ϵ, $\epsilon - \delta > 1 - d/k$, $\mathrm{PF}^{\mathrm{SAT}[n^\delta]} \subsetneq \mathrm{PF}^{\mathrm{SAT}[n^\epsilon]}$.

(b) for all $\epsilon > 1 - d/k$ and all i, $\mathrm{PF}^{\mathrm{SAT}[\log^i n]} \subsetneq \mathrm{PF}^{\mathrm{SAT}[n^\epsilon]}$.

At this point, one might ask if similar theorems can be proven for *parallel* queries to SAT. In the proofs above, we rely on the fact that $\mathrm{PF}^{\mathrm{SAT}[q(n)]}$ machines can use serial queries to find the lexicographically maximum satisfying assignment. However, it is only necessary for the machine to find some satisfying assignment — it does not have to be the largest one. If a $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}}$ machine can find a satisfying assignment for a Boolean formula, then we would get theorems analogous to Theorems 2 and 21 for the parallel bounded query hierarchy of SAT.

Furthermore, note that Theorems 2 and 21 would hold for parallel queries to any oracle A (instead of SAT), as long as the $\mathrm{PF}_{\mathrm{tt}}^{A[n^\epsilon]}$ machine can find some satisfying assignment for a given Boolean formula. For example, if A is a $\leq_{\mathrm{m}}^{\mathrm{P}}$-complete language for a class $\mathcal{C}$ that contains P^{NP}, then we could obtain the analogs of Theorems 2 and 21 for parallel queries to A. (This is because a P^{NP} machine can find the lexicographically largest satisfying assignment. Then n^ϵ *parallel* queries to A can find the first n^ϵ bits of this assignment).

It is not known without making any additional assumptions whether a $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}}$ can find a satisfying assignment for a Boolean formula. However under plausible hardness assumptions, Klivans and Van Melkebeek [14] showed that a $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}}$ machine can find a satisfying assignment for a Boolean formula by derandomizing Valiant and Vazirani's random reduction from SAT to USAT [23].

Although the NP machine hypothesis is known to derandomize $\mathrm{BPP}^{\mathrm{SAT}}$ to $\mathrm{P}^{\mathrm{SAT}}$ [11], it is not known whether the NP machine hypothesis also implies the derandomization Valiant and Vazirani's reduction. However, observe that a $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[q(n)]}$ machine can find a satisfying assignment for formulas that have unique satisfying assignments. So we next make use of the UP machine hypothesis to prove the analogs of Theorems 24 and 25 for parallel queries to SAT.

Definition 23 [11] **UP machine hypothesis:** There exist an $\epsilon > 0$ and a UP machine M such that $L(M) = 0^*$ and for any 2^{n^ϵ}-time bounded deterministic Turing machine D, the number of lengths n where $D(0^n)$ outputs the accepting path of $M(0^n)$ is finite.

No connection between the measure hypothesis and the UP machine hypothesis is known at this time. Nevertheless, we have following theorems:

Theorem 24 Under the UP machine hypothesis, for any constant $d > 0$, there exists a constant $k > d$ such that for all oracles X, $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^k]} \not\subseteq \mathrm{PF}^{X[n^d]}$ (in particular, $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^d]} \subsetneq \mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^k]}$).

Theorem 25 Suppose the UP machine hypothesis holds. Then, there exist constants d and k such that $d < k$ and

(a) for all δ, $0 \leq \delta < d/k$, for all ϵ, $\epsilon - \delta > 1 - d/k$, and for all oracles X, $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^\epsilon]} \not\subseteq \mathrm{PF}^{X[n^\delta]}$ (in particular, $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^\delta]} \subsetneq \mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^\epsilon]}$).

(b) for all $\epsilon > 1 - d/k$, for all oracles X, for all $i \geq 0$, $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^\epsilon]} \not\subseteq \mathrm{PF}^{X[\log^i n]}$ (in particular, $\mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[\log^i n]} \subsetneq \mathrm{PF}_{\mathrm{tt}}^{\mathrm{SAT}[n^\epsilon]}$).

6 Conclusion

We have observed a nice connection between resource-bounded measure and bounded query complexity — two established areas of computational complexity theory which, as far as we know, have not crossed paths. Having the NP machine hypothesis as an intermediary was very helpful. We note in retrospect that the Buhrman-Fortnow proof and the NP machine hypothesis are almost a perfect match. We needed a proof of SAT $\in$ P/poly where the advice is constructed explicitly and where we have enough control to make the advice polynomially shorter. This is not the case for other proofs in bounded query complexity. For example, the "advise trick" of Amir, Beigel and Gasarch [2, 1] produces advice strings that are too long for the NP machine hypothesis and there are no obvious ways to make them shorter. Another point is that in the Buhrman-Fortnow proof, we really needed the infinitely often versus almost everywhere aspect of the NP machine hypothesis to limit the SAT $\in$ P/poly case to a finite number of lengths. In contrast, the proofs in Section 5 actually violate the NP machine hypothesis at every length n.

References

[1] A. Amir, R. Beigel, and W. Gasarch. Some connections between bounded query classes and non-uniform complexity. *Information and Control*, 186(1):104–139, October 2003.

[2] A. Amir, R. Beigel, and W. I. Gasarch. Some connections between bounded query classes and non-uniform complexity. In *Proceedings of the 5th Structure in Complexity Theory Conference*, pages 232–243, 1990.

[3] V. ErvIn and J. Köbler. On pseudorandomness and resource-bounded measure. *Theoretical Computer Science*, 255(1–2):205–221, 2001.

[4] R. Beigel. NP-hard sets are p-superterse unless R = NP. Technical Report 4, Department of Computer Science, The Johns Hopkins University, 1988.

[5] R. Beigel, R. Chang, and M. Ogiwara. A relationship between difference hierarchies and relativized polynomial hierarchies. *Mathematical Systems Theory*, 26(3):293–310, July 1993.

[6] H. Buhrman and L. Fortnow. Two queries. *J. Comput. Syst. Sci.*, 59(2):182–194, 1999.

[7] J.-Y. Cai. $\mathrm{S}_2^{\mathrm{P}} \subseteq \mathrm{ZPP}^{\mathrm{NP}}$. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 620–629. IEEE Computer Society, Oct. 2001.

[8] R. Chang and J. Kadin. The Boolean hierarchy and the polynomial hierarchy: A closer connection. In *Proceedings of the 5th Structure in Complexity Theory Conference*, pages 169–178, July 1990. To appear in *SIAM Journal on Computing*.

[9] L. Fortnow, A. Pavan, and S. Sengupta. Proving SAT does not have small circuits with an application to the two queries problem. In *Proceedings of the 18th Annual IEEE Conference on Computational Complexity*, pages 347–350, July 2003. To appear in *Journal of Computer and System Sciences*.

[10] E. Hemaspaandra, L. A. Hemaspaandra, and H. Hempel. Downward collapse within the polynomial hierarchy. *SIAM J. Comput.*, 28(2):383–393, April 1999.

[11] J. M. Hitchcock and A. Pavan. Hardness hypotheses, derandomization, and circuit complexity. In *Proceedings of the 24th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 336–347. Springer-Verlag, 2004.

[12] J. M. Hitchcock and A. Pavan. Comparing reductions to NP-complete sets. In *Automata, Languages, and Programming (ICALP)*, pages 465–476, 2006.

[13] J. Kadin. The polynomial time hierarchy collapses if the Boolean hierarchy collapses. *SIAM J. Comput.*, 17(6):1263–1282, December 1988.

[14] A. R. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. In *ACM Symposium on Theory of Computing*, pages 659–667, 1999.

[15] M. W. Krentel. The complexity of optimization problems. *J. Comput. Syst. Sci.*, 36(3):490–509, 1988.

[16] J. H. Lutz. Almost everywhere high nonuniform complexity. *J. Comput. Syst. Sci.*, 44(2):220–258, April 1992.

[17] J. H. Lutz. Observations on measure and lowness for Δ_2^{P}. *Theory of Computing Systems*, 30(4):429–442, July 1997.

[18] J. H. Lutz. The quantitative structure of exponential time. In L. A. Hemaspaandra and A. L. Selman, editors, *Complexity Theory Retrospective II*, pages 225–254. Springer-Verlag, 1997.

[19] J. H. Lutz and E. Mayordomo. Cook versus Karp-Levin: Separating completeness notions if NP is not small. *Theoretical Computer Science*, 164(1–2):141–163, 1996.

[20] J. H. Lutz and E. Mayordomo. Twelve problems in resource-bounded measure. *Bulletin of the European Association for Theoretical Computer Science*, 68:64–80, 1999. Also in *Current Trends in Theoretical Computer Science: Entering the 21st Century,* pp. 83–101, World Scientific, 2001.

[21] N. Nisan and A. Wigderson. Hardness vs. randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[22] R. Shaltiel and C. Umans. Pseudorandomness for approximate counting and sampling. In *Proceedings of the 20th Annual IEEE Conference on Computational Complexity*, pages 212–226, 2005.

[23] L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47(1):85–93, 1986.