Amplification with One NP Oracle Query

Thomas Watson
University of Memphis, Memphis, TN, USA
Thomas.Watson@memphis.edu

Abstract

We provide a complete picture of the extent to which amplification of success probability is possible for randomized algorithms having access to one NP oracle query, in the settings of two-sided, one-sided, and zero-sided error. We generalize this picture to amplifying one-query algorithms with q-query algorithms, and we show our inclusions are tight for relativizing techniques.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Amplification, NP, oracle, query

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.96

Category Track A: Algorithms, Complexity and Games

Related Version The full version of the paper is available at `https://eccc.weizmann.ac.il/report/2018/058/`.

Funding Supported by NSF grant CCF-1657377.

1 Introduction

Amplification of the success probability of randomized *algorithms* is a ubiquitous tool in complexity theory. We investigate amplification for randomized *reductions* to NP-complete problems, which can be modeled as randomized algorithms with the ability to make queries to an NP oracle. The usual amplification strategy involves running multiple independent trials, which would also increase the number of NP oracle queries, so this does not generally work if we restrict the number of queries. We study, and essentially completely answer, the following question:

> *If a language is solvable with success probability p by a randomized polynomial-time algorithm with access to one* NP *oracle query, what is the highest success probability achievable with one query (or $q > 1$ many queries) to an* NP *oracle?*

The question makes sense for two-sided error ($\mathsf{BPP}^{\mathsf{NP}[1]}$), one-sided error ($\mathsf{RP}^{\mathsf{NP}[1]}$), and zero-sided error ($\mathsf{ZPP}^{\mathsf{NP}[1]}$), and it was mentioned in [2] as "an interesting problem worthy of further investigation." Partial results for zero-sided error were shown in [3]. The question is also relevant to the extensive literature on bounded NP queries (the boolean hierarchy); e.g., $\mathsf{ZPP}^{\mathsf{NP}[1]}$ shows up frequently in the context of the "two queries problem" [4], which was the main application area of the results from [3].

Our first contribution characterizes the best amplification achievable by relativizing techniques in the two-sided error setting. In general, the best strategy for amplifying plain randomized algorithms is to take the majority vote of q independent trials, which in our setting would naively involve q NP oracle queries. One may suspect this majority vote strategy is optimal for us. We show this intuition is a red herring; it is possible to do better by "combining" NP oracle queries across different trials. As an extreme example, consider the special case of randomized *mapping* reductions to NP problems. These are equivalent to Arthur–Merlin games (AM), for which amplification is possible by running independent

46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).
Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;
Article No. 96; pp. 96:1–96:13


Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

trials and simply having Merlin's message consist of certificates for a majority of the trials. However, if we allow one NP oracle query, but do not necessarily output the same bit the oracle returns, then combining queries is less straightforward, and it turns out amplification is only possible to a limited extent.

Our main take-home message is that starting with success probability greater than $\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{k+1}$, where k is an integer, we can get arbitrarily close to $\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{k}$ success probability while still using one NP query; using q nonadaptive queries, roughly a factor q improvement over this is possible.

We give precise definitions in Section 2, but we now clarify our notation before stating the theorem. For $\epsilon \in (0, 1]$ (the *advantage*), $\mathsf{BPP}_{\epsilon}^{\mathsf{NP}[1]}$ is the set of all languages solvable by a randomized polynomial-time algorithm that may make one query to an NP oracle and produces the correct output with probability $\geq \frac{1}{2} + \frac{1}{2}\epsilon$ on each input. For convenience, we define $\mathsf{BPP}_{>\epsilon}^{\mathsf{NP}[1]}$ by requiring that for some constant c there exists such an algorithm with advantage $\geq \epsilon + n^{-c}$, and we define $\mathsf{BPP}_{\epsilon>}^{\mathsf{NP}[1]}$ by requiring that for every constant d there exists such an algorithm with advantage $\geq \epsilon - 2^{-n^d}$; the reason for these conventions is just that they naturally arise in the proofs (e.g., standard majority amplification implies $\mathsf{BPP}_{>0} = \mathsf{BPP}_{1>}$). We make similar definitions for $\mathsf{BPP}^{\mathsf{NP}\|[q]}$ but allowing q nonadaptive NP oracle queries. Allowing q adaptive NP queries is equivalent to allowing $2^q - 1$ nonadaptive NP queries [1].

▶ **Theorem 1** (Two-sided error). *For integers $1 \leq q \leq k$:*

- *If q is odd:*

$$\mathsf{BPP}_{>1/(k+1)}^{\mathsf{NP}[1]} \subseteq \mathsf{BPP}_{q/k>}^{\mathsf{NP}\|[q]} \quad \textit{and} \quad \mathsf{BPP}_{1/k}^{\mathsf{NP}[1]} \not\subseteq \mathsf{BPP}_{>q/k}^{\mathsf{NP}\|[q]} \textit{ relative to an oracle.}$$

- *If q, k are even:*

$$\mathsf{BPP}_{>1/(k+1)}^{\mathsf{NP}[1]} \subseteq \mathsf{BPP}_{q/k>}^{\mathsf{NP}\|[q]} \quad \textit{and} \quad \mathsf{BPP}_{1/(k-1)}^{\mathsf{NP}[1]} \not\subseteq \mathsf{BPP}_{>q/k}^{\mathsf{NP}\|[q]} \textit{ relative to an oracle.}$$

The word "oracle" has two meanings here. Besides the bounded NP oracle queries of central interest, "relative to an oracle" means there exists a language such that the separation holds when all computations (the randomized algorithm and the NP verifier) can make polynomially many adaptive queries to an oracle for that language. In particular, in the context of our relativized separations, randomized algorithms have access to two oracles. The separations in Theorem 1 are tight since the inclusions relativize. This implies that using "black-box simulation" techniques, it is not possible to significantly improve any of our inclusions.

If we start with advantage $> \frac{1}{k+1}$ where k is an integer, Theorem 1 tells us the best advantage achievable with q nonadaptive NP queries using relativizing techniques: if k is even we can amplify to essentially $\frac{q}{k}$; if k is odd we can amplify to essentially $\frac{q}{k}$ if q is odd, and $\frac{q}{k+1}$ if q is even. (Theorem 1 does not explicitly mention the case where q is even and k is odd, but in this case the best inclusion and separation are obtained by applying the theorem to the even integer $k + 1$.)

A subtle issue is whether "$q/k>$" in the inclusion subscripts can be improved to "q/k"; e.g., it remains open to show that $\mathsf{BPP}_{>1/3}^{\mathsf{NP}[1]} \subseteq \mathsf{BPP}_{1/2}^{\mathsf{NP}[1]}$ or that $\mathsf{BPP}_{>1/3}^{\mathsf{NP}[1]} \not\subseteq \mathsf{BPP}_{1/2}^{\mathsf{NP}[1]}$ relative to an oracle.

The proof of Theorem 1 appears in Section 3. No such nontrivial inclusion was known before; for relativized separations, the case $q = 1$, $k = 2$ was shown in [7].

Zero-sided error algorithms must output the correct bit with probability at least some $\epsilon \in (0, 1]$ and output $\perp$ (plead ignorance) with the remaining probability. We define the advantage (the subscript of $\mathsf{ZPP}^{\mathsf{NP}\|[q]}$) to be this ϵ.

[3] proved that $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{1/4}$ and $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/2} \subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{1>}$,[1] and left it unresolved what happens between advantages $\frac{1}{4}$ and $\frac{1}{2}$. We settle this decade-old open problem: amplification is possible between $\frac{1}{4}$ and $\frac{1}{3}$ and between $\frac{1}{3}$ and $\frac{1}{2}$.

▶ **Theorem 2** (Zero-sided error). *For integers $1 \leq q \leq k \leq 4$:*

- *If $k = 4$:* $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{q/k>}$.
- *If $k \leq 3$:* $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/(k+1)} \subseteq \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{q/k>}$.
- *If $q = 1$:* $\mathsf{ZPP}^{\mathsf{NP}[1]}_{1/k} \not\subseteq \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{>q/k}$ *relative to an oracle.*

Moreover, the "$q/k>$" in the inclusion subscripts can be improved to "q/k" if $q < k \geq 3$.

The proof of Theorem 2 appears in Section 4. The "moreover" part uses a trick described in [3] for getting a tiny boost in the advantage. Like the situation with $\mathsf{BPP}^{\mathsf{NP}[1]}$, it remains open to show that $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/3} \subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{1/2}$ or that $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/3} \not\subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{1/2}$ relative to an oracle. There is no reason to consider $k > 4$ in Theorem 2, since then $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/(k+1)} \subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{q/4>}$.

We conjecture that the third bullet in Theorem 2 also holds for $q > 1$ (i.e., the relativized separations $\mathsf{ZPP}^{\mathsf{NP}[1]}_{1/4} \not\subseteq \mathsf{ZPP}^{\mathsf{NP}\|[2]}_{>2/4}$ and $\mathsf{ZPP}^{\mathsf{NP}[1]}_{1/4} \not\subseteq \mathsf{ZPP}^{\mathsf{NP}\|[3]}_{>3/4}$ and $\mathsf{ZPP}^{\mathsf{NP}[1]}_{1/3} \not\subseteq \mathsf{ZPP}^{\mathsf{NP}\|[2]}_{>2/3}$). This remains open, though we are aware of how to prove that $\mathsf{ZPP}^{\mathsf{NP}[1]}_{1/4} \not\subseteq \mathsf{ZPP}^{\mathsf{NP}\|[2]}_{>3/4}$. Anyway, $q = 1$ is the most natural case, and we provide a complete proof for it.

One-sided error algorithms must always output 0 if the answer is 0, and must output 1 with probability at least some $\epsilon \in (0, 1]$ if the answer is 1. We define the advantage (the subscript of $\mathsf{RP}^{\mathsf{NP}\|[q]}$) to be this ϵ. The proof of Theorem 3 appears in the full version of this paper [6] and is relatively straightforward.

▶ **Theorem 3** (One-sided error).

- $\mathsf{RP}^{\mathsf{NP}[1]}_{>1/2} \subseteq \mathsf{RP}^{\mathsf{NP}[1]}_{1>}$.
- $\mathsf{RP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{RP}^{\mathsf{NP}[1]}_{1/2} \cap \mathsf{RP}^{\mathsf{NP}\|[2]}_{1>}$ *and* $\mathsf{RP}^{\mathsf{NP}[1]}_{1/2} \not\subseteq \mathsf{RP}^{\mathsf{NP}[1]}_{>1/2}$ *relative to an oracle.*

Finally, we point out that none of the inclusions in this paper can be strengthened to yield advantage exactly 1 via relativizing techniques, since $\mathsf{BPP} \subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/2}$ relativizes [2] but $\mathsf{BPP} \not\subseteq \mathsf{P}^{\mathsf{NP}}$ relative to an oracle [folklore].

2 Definitions

We formally define the relevant complexity classes in Section 2.1 and their decision tree analogues (which are used for relativized separations) in Section 2.2.

2.1 Time complexity

We think of a randomized algorithm M as taking a uniformly random string $s \in \{0,1\}^r$ (for some number of coins r that depends on the input length); we let $M_s(x)$ denote M running on input x with outcome s.

For $\epsilon \in (0, 1]$ (the *advantage*) and integer $q \geq 1$, language L is in $\mathsf{BPP}^{\mathsf{NP}\|[q]}_{\epsilon}$ iff there is a polynomial-time randomized algorithm M (taking input x and coin tosses $s \in \{0,1\}^r$) and a language $L' \in \mathsf{NP}$ such that the following hold.

[1] [7] gave an alternative proof of the latter but with only $1 - \frac{1}{\text{poly}}$, rather than $1 - \frac{1}{\exp}$, success probability.

Syntax: The computation of $M_s(x)$ produces a tuple of query strings $(z_1, \ldots, z_q)$ and a truth table $out\colon \{0,1\}^q \to \{0,1\}$; the output is then $out(L'(z_1), \ldots, L'(z_q))$.
Correctness: The output is $L(x)$ with probability $\geq \frac{1}{2} + \frac{1}{2}\epsilon$.

$\mathsf{RP}^{\mathsf{NP}\|[q]}_{\epsilon}$ is defined similarly except for correctness, we require the output is always 0 if $L(x) = 0$, and is 1 with probability $\geq \epsilon$ if $L(x) = 1$. $\mathsf{ZPP}^{\mathsf{NP}\|[q]}_{\epsilon}$ is defined similarly except $out\colon \{0,1\}^q \to \{0,1,\perp\}$ and for correctness, we require the output is always $L(x)$ or $\perp$, and is $L(x)$ with probability $\geq \epsilon$.

For $\mathcal{C} \in \left\{\mathsf{BPP}^{\mathsf{NP}\|[q]}, \mathsf{RP}^{\mathsf{NP}\|[q]}, \mathsf{ZPP}^{\mathsf{NP}\|[q]}\right\}$, we define

$$\mathcal{C}_{>\epsilon} = \bigcup_{\text{constants } c} \mathcal{C}_{\epsilon+n^{-c}} \quad \text{and} \quad \mathcal{C}_{\epsilon>} = \bigcap_{\text{constants } d} \mathcal{C}_{\epsilon-2^{-n^d}}.$$

When $q = 1$ we may drop the $\|$ from the superscripts.

2.2 Decision tree complexity

We think of a randomized decision tree T as the uniform distribution over a multiset of corresponding deterministic decision trees T_s indexed by $s \in \{0,1\}^r$; we denote this as $T \sim \left\{T_s : s \in \{0,1\}^r\right\}$. In this setting, "query" actually has two meanings for us: a decision tree makes queries to individual input bits, then it forms an NP-type (DNF) oracle query.

We define a $\mathsf{BPP}^{\mathsf{NP}\|[q]}_{\epsilon}$-type decision tree T for $f\colon \{0,1\}^n \to \{0,1\}$ on input x as follows.

Syntax: $T \sim \left\{T_s : s \in \{0,1\}^r\right\}$ where each T_s makes queries to the bits of x until it reaches a leaf, which is labeled with a tuple of DNFs $(\varphi_1, \ldots, \varphi_q)$ and a function $out\colon \{0,1\}^q \to \{0,1\}$; the output is then $out(\varphi_1(x), \ldots, \varphi_q(x))$.
Correctness: The output is $f(x)$ with probability $\geq \frac{1}{2} + \frac{1}{2}\epsilon$.
Cost: The maximum height of any T_s, plus the maximum width of any DNF appearing at a leaf.

An $\mathsf{RP}^{\mathsf{NP}\|[q]}_{\epsilon}$-type decision tree is defined similarly except for correctness we require the output is always 0 if $f(x) = 0$, and is 1 with probability $\geq \epsilon$ if $f(x) = 1$. A $\mathsf{ZPP}^{\mathsf{NP}\|[q]}_{\epsilon}$-type decision tree is defined similarly except $out\colon \{0,1\}^q \to \{0,1,\perp\}$ and for correctness, we require the output is always $f(x)$ or $\perp$, and is $f(x)$ with probability $\geq \epsilon$.

We follow the convention of overloading complexity class names as decision tree complexity measures: for $\mathcal{C} \in \left\{\mathsf{BPP}^{\mathsf{NP}\|[q]}, \mathsf{RP}^{\mathsf{NP}\|[q]}, \mathsf{ZPP}^{\mathsf{NP}\|[q]}\right\}$, $\mathcal{C}^{\mathsf{dt}}_{\epsilon}(f)$ denotes the minimum cost of any $\mathcal{C}_{\epsilon}$-type decision tree for a partial function f, and $\mathcal{C}^{\mathsf{dt}}_{\epsilon}$ also denotes the class of all families of f's with $\mathcal{C}^{\mathsf{dt}}_{\epsilon}(f) \leq \mathrm{polylog}(n)$, and we define

$$\mathcal{C}^{\mathsf{dt}}_{>\epsilon} = \bigcup_{\text{constants } c} \mathcal{C}^{\mathsf{dt}}_{\epsilon+\log^{-c} n} \quad \text{and} \quad \mathcal{C}^{\mathsf{dt}}_{\epsilon>} = \bigcap_{\text{constants } d} \mathcal{C}^{\mathsf{dt}}_{\epsilon-n^{-d}}.$$

3 Two-sided error

To prove Theorem 1, we first restate it in a more convenient form.

▶ **Theorem 1** (Two-sided error, restated). *For integers $1 \leq q \leq k$:*

(i) *If k, q are odd:* $\mathsf{BPP}^{\mathsf{NP}[1]}_{>1/(k+1)} \subseteq \mathsf{BPP}^{\mathsf{NP}\|[q]}_{q/k>}$.

(ii) *If k is even:* $\mathsf{BPP}^{\mathsf{NP}[1]}_{>1/(k+1)} \subseteq \mathsf{BPP}^{\mathsf{NP}\|[q]}_{q/k>}$.

(iii) *If q, k are even:* $\mathsf{BPP}^{\mathsf{NP}[1]}_{1/(k-1)} \not\subseteq \mathsf{BPP}^{\mathsf{NP}\|[q]}_{>q/k}$ *relative to an oracle.*

(iv) *If q is odd:* $\mathsf{BPP}^{\mathsf{NP}[1]}_{1/k} \not\subseteq \mathsf{BPP}^{\mathsf{NP}\|[q]}_{>q/k}$ *relative to an oracle.*

We prove the inclusions (i) and (ii) in Section 3.1 and the separations (iii) and (iv) in Section 3.2.

3.1 Inclusions

We prove the $q = 1$ case of (i) in Section 3.1.1 and the $q = 1$ case of (ii) in Section 3.1.2 (together these show that $\mathsf{BPP}^{\mathsf{NP}[1]}_{>1/(k+1)} \subseteq \mathsf{BPP}^{\mathsf{NP}[1]}_{1/k>}$ for all integers $k \geq 1$), then we generalize to the $q > 1$ case of (i) in Section 3.1.3 and the $q > 1$ case of (ii) in the full version [6]. The techniques from [3] for the zero-sided error setting are not particularly helpful for the two-sided error setting, so we develop the ideas from scratch.

We now describe the common setup. For some constant c we have $L \in \mathsf{BPP}^{\mathsf{NP}[1]}_{1/(k+1)+n^{-c}}$, witnessed by a polynomial-time randomized algorithm M (taking input x and coin tosses $s \in \{0,1\}^r$) and a language $L' \in \mathsf{NP}$. For an arbitrary constant d, we wish to show $L \in \mathsf{BPP}^{\mathsf{NP}\|[q]}_{q/k-2^{-n^d}}$.

Fix an input x. The first step is to sample a sequence of $m = O(n^{2c+d})$ many independent strings $s^1, \ldots, s^m \in \{0,1\}^r$, so with probability $\geq 1 - 2^{-n^d-1}$, the sequence is *good* in the sense that on input x, M still has advantage strictly greater than $\frac{1}{k+1}$ when its coin tosses are chosen uniformly from the multiset $\{s^1, \ldots, s^m\}$. Then we design a polynomial-time randomized algorithm which, given a good sequence, outputs $L(x)$ with advantage $\geq \frac{q}{k}$ after making q nonadaptive NP oracle queries. Hence, over the random $s^1, \ldots, s^m$ and the other randomness of our algorithm,

$$\begin{aligned}\mathbb{P}[\text{output is } L(x)] &\geq \mathbb{P}\big[\text{output is } L(x) \,\big|\, s^1, \ldots, s^m \text{ is good}\big] - \mathbb{P}[s^1, \ldots, s^m \text{ is bad}] \\ &\geq \left(\tfrac{1}{2} + \tfrac{1}{2} \cdot \tfrac{q}{k}\right) - 2^{-n^d-1} \;=\; \tfrac{1}{2} + \tfrac{1}{2}\left(\tfrac{q}{k} - 2^{-n^d}\right).\end{aligned}$$

Henceforth fix a good sequence $s^1, \ldots, s^m$, and let z^h and $out^h \colon \{0,1\} \to \{0,1\}$ be the query string and truth table produced by $M_{s^h}(x)$ (so the output is $out^h(L'(z^h))$). We assume w.l.o.g. that each out^h is nonconstant, and is hence either identity or negation. Henceforth assume that identity is at least as common as negation among $out^1, \ldots, out^m$; the proof is completely analogous if negation is more common.

Taking probabilities over a uniformly random $h \in [m]$, we make the following definitions.

$$\begin{aligned}\alpha &= \tfrac{1}{2}\mathbb{P}[out^h = \mathrm{id}] & \beta &= \tfrac{1}{2}\mathbb{P}[out^h = \mathrm{neg}] \\ a &= \mathbb{P}\big[out^h = \mathrm{id},\ L'(z^h) = 1\big] - \alpha & b &= \mathbb{P}\big[out^h = \mathrm{neg},\ L'(z^h) = 1\big] - \beta\end{aligned}$$

The key observation is now

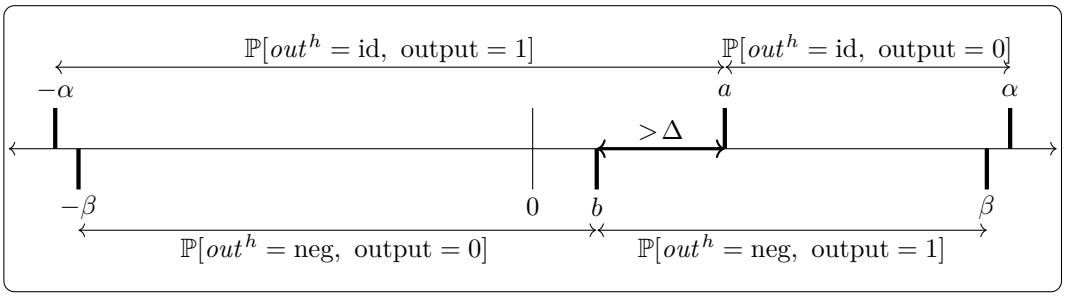
$$\begin{aligned}&(a + \alpha) + (\beta - b) \\ &= \mathbb{P}\big[out^h = \mathrm{id},\ \text{output} = 1\big] + \big(\mathbb{P}[out^h = \mathrm{neg}] - \mathbb{P}\big[out^h = \mathrm{neg},\ \text{output} = 0\big]\big) \\ &= \mathbb{P}\big[out^h = \mathrm{id},\ \text{output} = 1\big] + \mathbb{P}\big[out^h = \mathrm{neg},\ \text{output} = 1\big] \;=\; \mathbb{P}[\text{output} = 1]\end{aligned}$$

and thus, defining $\Delta = \frac{1}{2} \cdot \frac{1}{k+1}$, we have

$$a - b \;=\; (a + \alpha) + (\beta - b) - \tfrac{1}{2} \;=\; \mathbb{P}[\text{output} = 1] - \tfrac{1}{2} \begin{cases} > \Delta & \text{if } L(x) = 1 \\ < -\Delta & \text{if } L(x) = 0 \end{cases}$$

because of M's advantage w.r.t. a good sequence $s^1, \ldots, s^m$.

This figure shows an example of how these values may fall on the number line if $L(x) = 1$:



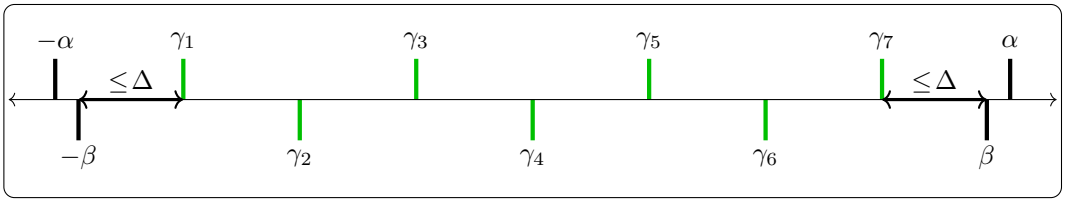
The following summarizes the key properties so far.

$$\alpha \geq \beta \qquad a \in [-\alpha, \alpha] \qquad a - b > \Delta \ \text{ if } L(x) = 1$$
$$\alpha + \beta = \tfrac{1}{2} \qquad b \in [-\beta, \beta] \qquad b - a > \Delta \ \text{ if } L(x) = 0$$

Also, for any real p, testing whether $a \geq p$ can be expressed as an NP oracle query: a witness consists of a list of witnesses for $L'(z^h) = 1$ for at least $(p + \alpha)m$ many h's with $out^h = \text{id}$. Similarly, testing whether $b \geq p$ can be expressed as an NP oracle query.

3.1.1 Proof of (i): $q = 1$

For $i \in [k]$ define $\gamma_i = (i - \frac{k+1}{2})\Delta$. We have $\beta - \gamma_k \leq \Delta$ and $\gamma_1 - (-\beta) \leq \Delta$ since $\beta \leq \frac{1}{4} = \big((k+1) - \frac{k+1}{2}\big)\Delta$. This figure shows an example with $k = 7$:



Our algorithm now picks one of these k possibilities uniformly at random:[2]

- for some odd $i \in [k]$: output 1 iff $a \geq \gamma_i$,
- for some even $i \in [k]$: output 0 iff $b \geq \gamma_i$.

First suppose $L(x) = 1$. We have $a > \gamma_1$ since $a - b > \Delta$ and $b \geq -\beta$ and $\gamma_1 - (-\beta) \leq \Delta$. Consider the greatest odd $j \in [k]$ such that $a \geq \gamma_j$; thus $a \geq \gamma_i$ for $\frac{j+1}{2}$ many odd i's $(1, 3, \ldots, j)$. If $j < k$ then $b < \gamma_{j+1}$ since $a - b > \Delta$ and $a < \gamma_{j+2}$; thus $b < \gamma_i$ for at least $\frac{k-j}{2}$ many even i's $(j+1, j+3, \ldots, k-1)$. Hence the probability of outputting 1 is at least $\frac{1}{k}\big(\frac{j+1}{2} + \frac{k-j}{2}\big) = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{k}$.

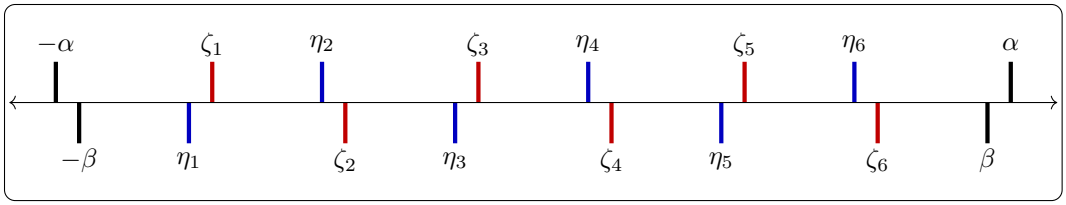
Now suppose $L(x) = 0$. We have $a < \gamma_k$ since $b - a > \Delta$ and $b \leq \beta$ and $\beta - \gamma_k \leq \Delta$. Consider the least odd $j \in [k]$ such that $a < \gamma_j$; thus $a < \gamma_i$ for $\frac{k-j+2}{2}$ many odd i's $(j, j+2, \ldots, k)$. If $j > 1$ then $b > \gamma_{j-1}$ since $b - a > \Delta$ and $a \geq \gamma_{j-2}$; thus $b \geq \gamma_i$ for at least $\frac{j-1}{2}$ many even i's $(2, 4, \ldots, j-1)$. Hence the probability of outputting 0 is at least $\frac{1}{k}\big(\frac{k-j+2}{2} + \frac{j-1}{2}\big) = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{k}$.

[2] Of course, if k is not a power of 2 and we insist on using uniform coin flips as our only source of randomness, then we must incur a tiny error since it is not possible to exactly sample $i \in [k]$ uniformly. We sweep this pedantic issue under the rug throughout the paper.

That concludes the formal proof, but here is an intuitive way to visualize what is happening: Call γ_i for odd i "upper marks," and call γ_i for even i "lower marks," and assume for convenience all lower marks are in $(-\beta, \beta)$. Suppose $L(x) = 1$ and $b = -\beta$ so $a > \gamma_1$; then at least one upper mark is left of a and all $\frac{k-1}{2}$ lower marks are right of b, resulting in $\frac{k+1}{2}$ of the algorithm's possibilities outputting 1. Now as we continuously sweep a and b to the right, keeping $a - b$ fixed, a passes each upper mark before b passes the preceding lower mark, so at all times at least $\frac{k+1}{2}$ of the possibilities output 1. Suppose $L(x) = 0$ and $b = \beta$ so $a < \gamma_k$; then at least one upper mark is right of a and all $\frac{k-1}{2}$ lower marks are left of b, resulting in $\frac{k+1}{2}$ of the algorithm's possibilities outputting 0. Now as we continuously sweep a and b to the left, keeping $b - a$ fixed, a passes each upper mark before b passes the succeeding lower mark, so at all times at least $\frac{k+1}{2}$ of the possibilities output 0.

3.1.2 Proof of (ii): $q = 1$

For $i \in [k]$ define $\zeta_i = -\beta + i\Delta$ and $\eta_i = -\alpha + i\Delta$. Note that $\alpha - \zeta_k = \Delta$ (so $\zeta_1, \ldots, \zeta_k$ divide the interval $[-\beta, \alpha]$ into $k+1$ subintervals each of length Δ) and $\beta - \eta_k = \Delta$ (so $\eta_1, \ldots, \eta_k$ divide the interval $[-\alpha, \beta]$ into $k+1$ subintervals each of length Δ). This figure shows an example with $k = 6$:



Our algorithm now picks one of these $2k$ possibilities uniformly at random:

- for some odd $i \in [k]$: output 1 iff $a \geq \zeta_i$,
- for some even $i \in [k]$: output 0 iff $b \geq \zeta_i$,
- for some even $i \in [k]$: output 1 iff $a \geq \eta_i$,
- for some odd $i \in [k]$: output 0 iff $b \geq \eta_i$.

First suppose $L(x) = 1$. We have $a > \zeta_1$ since $a - b > \Delta$ and $b \geq -\beta$. Consider the greatest odd $j \in [k]$ such that $a \geq \zeta_j$; thus $a \geq \zeta_i$ for $\frac{j+1}{2}$ many odd i's $(1, 3, \ldots, j)$. We have $b < \zeta_{j+1}$ since $a - b > \Delta$ and either $a < \zeta_{j+2}$ (if $j < k-1$) or $a \leq \alpha$ and $\alpha - \zeta_k = \Delta$ (if $j = k-1$); thus $b < \zeta_i$ for at least $\frac{k-j+1}{2}$ many even i's $(j+1, j+3, \ldots, k)$. Consider the greatest even $j' \in [k]$ such that $a \geq \eta_{j'}$, or let $j' = 0$ if it does not exist; thus $a \geq \eta_i$ for $\frac{j'}{2}$ many even i's $(2, 4, \ldots, j')$. If $j' < k$ then $b < \eta_{j'+1}$ since $a - b > \Delta$ and $a < \eta_{j'+2}$; thus $b < \eta_i$ for at least $\frac{k-j'}{2}$ many odd i's $(j'+1, j'+3, \ldots, k-1)$. Hence the probability of outputting 1 is at least $\frac{1}{2k}\left(\frac{j+1}{2} + \frac{k-j+1}{2} + \frac{j'}{2} + \frac{k-j'}{2}\right) = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{k}$.

Now suppose $L(x) = 0$. Consider the least odd $j \in [k]$ such that $a < \zeta_j$, or let $j = k+1$ if it does not exist; thus $a < \zeta_i$ for $\frac{k-j+1}{2}$ many odd i's $(j, j+2, \ldots, k-1)$. If $j > 1$ then $b > \zeta_{j-1}$ since $b - a > \Delta$ and $a \geq \zeta_{j-2}$; thus $b \geq \zeta_i$ for at least $\frac{j-1}{2}$ many even i's $(2, 4, \ldots, j-1)$. We have $a < \eta_k$ since $b - a > \Delta$ and $b \leq \beta$ and $\beta - \eta_k = \Delta$. Consider the least even $j' \in [k]$ such that $a < \eta_{j'}$; thus $a < \eta_i$ for $\frac{k-j'+2}{2}$ many even i's $(j', j'+2, \ldots, k)$. We have $b > \eta_{j'-1}$ since $b - a > \Delta$ and either $a \geq \eta_{j'-2}$ (if $j' > 2$) or $a \geq -\alpha$ (if $j' = 2$); thus $b \geq \eta_i$ for at least $\frac{j'}{2}$ many odd i's $(1, 3, \ldots, j'-1)$. Hence the probability of outputting 0 is at least $\frac{1}{2k}\left(\frac{k-j+1}{2} + \frac{j-1}{2} + \frac{k-j'+2}{2} + \frac{j'}{2}\right) = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{k}$.

That concludes the formal proof, but here is an intuitive way to visualize what is happening: Call ζ_i for odd i and η_i for even i "upper marks," and call ζ_i for even i and η_i for odd i "lower marks," and assume for convenience all lower marks are in $(-\beta, \beta)$. Suppose $L(x) = 1$ and $b = -\beta$ so $a > \zeta_1$; then at least one upper mark is left of a and all k lower marks are right of b, resulting in $k+1$ of the algorithm's possibilities outputting 1. Now as we continuously sweep a and b to the right, keeping $a - b$ fixed, a passes each upper mark (ζ_i or η_i) before b passes the corresponding preceding lower mark (ζ_{i-1} or η_{i-1} respectively), so at all times at least $k+1$ of the possibilities output 1. Suppose $L(x) = 0$ and $b = \beta$ so $a < \eta_k$; then at least one upper mark is right of a and all k lower marks are left of b, resulting in $k+1$ of the algorithm's possibilities outputting 0. Now as we continuously sweep a and b to the left, keeping $b - a$ fixed, a passes each upper mark (ζ_i or η_i) before b passes the corresponding succeeding lower mark (ζ_{i+1} or η_{i+1} respectively), so at all times at least $k+1$ of the possibilities output 0.

3.1.3 Proof of (i): $q > 1$

For $i \in [k]$ define I_i as the set of q successive integers starting with i and wrapping around to 1 when k is exceeded: $I_i = \{i, i+1, \ldots, i+q-1\}$ if $i \leq k-q+1$, and $I_i = \{i, i+1, \ldots, k, 1, 2, \ldots, i+q-1-k\}$ if $i > k-q+1$. Define $i^{\nwarrow} = \min(\text{odd } i' \in I_i) - k - 1$ and $i^{\swarrow} = \min(\text{even } i' \in I_i) - k - 1$; the $-k-1$ is a simple way to ensure $i^{\nwarrow}, i^{\swarrow} < \min(i' \in I_i)$. Since k, q are odd, the sorted order of $I_i \cup \{i^{\nwarrow}, i^{\swarrow}\}$ alternates between odd and even numbers.

Our algorithm picks $i \in [k]$ uniformly at random and for each $i' \in I_i$ does an oracle query to see whether $a \geq \gamma_{i'}$ if i' is odd, or whether $b \geq \gamma_{i'}$ if i' is even. Consider the greatest odd $i^\sharp \in I_i$ such that $a \geq \gamma_{i^\sharp}$, or let $i^\sharp = i^{\nwarrow}$ if it does not exist. Consider the greatest even $i^\flat \in I_i$ such that $b \geq \gamma_{i^\flat}$, or let $i^\flat = i^{\swarrow}$ if it does not exist. Our algorithm outputs 1 if $i^\sharp > i^\flat$, or 0 if $i^\flat > i^\sharp$.

First suppose $L(x) = 1$. Consider the greatest odd $j \in [k]$ such that $a \geq \gamma_j$ (which exists since $a > \gamma_1$). We have $i^\sharp > i^\flat$ if one of the following mutually exclusive events holds:

(1) $j \in I_i$, since then $i^\sharp = j$ and $i^\flat \leq j-1$ (since $b < \gamma_{j+1}$ if $j < k$);
(2) i is odd and $i \leq j-q-1$, since then $i^\sharp = i+q-1$ and trivially $i^\flat \leq i+q-2$;
(3) i is even and $j+1 \leq i \leq j-q-1+k$, since then either:
- $i \leq k-q$, in which case $i^\sharp = i^{\nwarrow} > i^{\swarrow} = i^\flat$, or
- $i = k-q+2$, in which case $i^\sharp = 1$ and $i^\flat = i^{\swarrow} < 1$, or
- $i \geq k-q+4$, in which case $i^\sharp = i+q-1-k$ and $i^\flat \leq i+q-2-k$.

There are q many type-(1) i's. If $j > q$ then there are $\frac{j-q}{2}$ many type-(2) i's $(1, 3, \ldots, j-q-1)$ and $\frac{k-j}{2}$ many type-(3) i's $(j+1, j+3, \ldots, k-1)$. If $j \leq q$ then there are $\frac{k-q}{2}$ many type-(3) i's $(j+1, j+3, \ldots, j-q-1+k)$. Either way, $i^\sharp > i^\flat$ holds for at least $q + \frac{k-q}{2} = \frac{k+q}{2}$ many i's, and hence the probability of outputting 1 is at least $\frac{1}{k} \cdot \frac{k+q}{2} = \frac{1}{2} + \frac{1}{2} \cdot \frac{q}{k}$.

Now suppose $L(x) = 0$. Consider the least odd $j \in [k]$ such that $a < \gamma_j$ (which exists since $a < \gamma_k$). As a special case, if $j = 1$ then $i^\sharp = i^{\nwarrow}$ and so $i^\flat > i^\sharp$ if $i^{\swarrow} > i^{\nwarrow}$, which happens for $\frac{k+q}{2}$ many i's $(1, 3, \ldots, k-q+1$ and $k-q+2, k-q+3, \ldots, k)$. Now assume $j > 1$. We have $i^\flat > i^\sharp$ if one of the following mutually exclusive events holds:

(1) $j-1 \in I_i$, since then $i^\sharp \leq j-2$ and $i^\flat \geq j-1$ (since $b > \gamma_{j-1}$ if $j > 1$);
(2) i is even and $i \leq j-q-2$, since then $i^\sharp = i+q-2$ and $i^\flat = i+q-1$;
(3) i is odd and $j \leq i \leq j-q-2+k$, since then either:
- $i \leq k-q+1$, in which case $i^\sharp = i^{\nwarrow} < i^{\swarrow} \leq i^\flat$, or
- $i \geq k-q+3$, in which case $i^\sharp = i+q-2-k$ and $i^\flat \geq i+q-1-k$.

There are q many type-(1) i's. If $j > q$ then there are $\frac{j-q-2}{2}$ many type-(2) i's $(2, 4, \ldots, j-q-2)$ and $\frac{k-j+2}{2}$ many type-(3) i's $(j, j+2, \ldots, k)$. If $j \leq q$ then there are $\frac{k-q}{2}$ many type-(3) i's $(j, j+2, \ldots, j-q-2+k)$. Either way, $i^\flat > i^\sharp$ holds for at least $q + \frac{k-q}{2} = \frac{k+q}{2}$ many i's, and hence the probability of outputting 0 is at least $\frac{1}{k} \cdot \frac{k+q}{2} = \frac{1}{2} + \frac{1}{2} \cdot \frac{q}{k}$.

3.2 Separations

The relativized separations follow routinely [5] from the corresponding decision tree complexity separations:

(iii) If q, k are even: $\mathsf{BPP}^{\mathsf{NP}[1]\mathsf{dt}}_{1/(k-1)} \not\subseteq \mathsf{BPP}^{\mathsf{NP}\|[q]\mathsf{dt}}_{>q/k}$.

(iv) If q is odd: $\mathsf{BPP}^{\mathsf{NP}[1]\mathsf{dt}}_{1/k} \not\subseteq \mathsf{BPP}^{\mathsf{NP}\|[q]\mathsf{dt}}_{>q/k}$.

We prove (iii) in Section 3.2.1 and (iv) in the full version [6]; the arguments are similar in structure. Our proof of (iv) also works if q is even, but in that case the result is subsumed by (iii). The case $q = 1$, $k = 2$ of (iii) was proven in [7], but our proof is somewhat different even specialized to that case.

Let $\mathrm{wt}(\cdot)$ refer to Hamming weight. Henceforth fix the constants q and k, and assume $q < k$ since otherwise there is nothing to prove.

3.2.1 Proof of (iii)

Define the partial function $f\colon \{0,1\}^n \to \{0,1\}$ that interprets its input as $(x, y) \in \{0,1\}^{n/2} \times \{0,1\}^{n/2}$, such that

$$f(x,y) = \begin{cases} 1 & \text{if } \mathrm{wt}(x) = \mathrm{wt}(y) + 1 \leq \frac{k}{2} \\ 0 & \text{if } \mathrm{wt}(x) = \mathrm{wt}(y) \leq \frac{k}{2} - 1 \end{cases}.$$

▶ **Lemma 4.** $\mathsf{BPP}^{\mathsf{NP}[1]\mathsf{dt}}_{1/(k-1)}(f) \leq \frac{k}{2}$.

▶ **Lemma 5.** $\mathsf{BPP}^{\mathsf{NP}\|[q]\mathsf{dt}}_{q/k+\delta}(f) \geq \Omega(\delta n)$ *for every* $\delta(n)$.

The separation follows by taking $\delta = \log^{-c} n$ for any constant c.

Proof of Lemma 4. Given (x, y), pick one of these $k-1$ possibilities uniformly at random:

- for some $i \in [\frac{k}{2}]$: output 1 iff $\mathrm{wt}(x) \geq i$,
- for some $i \in [\frac{k}{2} - 1]$: output 0 iff $\mathrm{wt}(y) \geq i$.

The decision tree does not directly query any bits of (x, y), and the DNF has width $i \leq \frac{k}{2}$ (it is the OR over all i-subsets of either x's bits or y's bits, of the AND of those bits), so the cost is $\frac{k}{2}$. If $f(x, y) = 1$ with $\mathrm{wt}(x) = j$ and $\mathrm{wt}(y) = j - 1$, then the probability of outputting 1 is $\frac{j+((k/2-1)-(j-1))}{k-1} = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{k-1}$ since conditioned on picking x, the output is 1 iff $i \leq j$, and conditioned on picking y, the output is 1 iff $i \geq j$. Similarly, if $f(x, y) = 0$ with $\mathrm{wt}(x) = \mathrm{wt}(y) = j$, then the probability of outputting 1 is $\frac{j+((k/2-1)-j)}{k-1} = \frac{1}{2} - \frac{1}{2} \cdot \frac{1}{k-1}$. ◀

Proof of Lemma 5. It suffices to show that for some distribution on valid inputs (x, y) to f, every cost-$o(\delta n)$ $\mathsf{P}^{\mathsf{NP}\|[q]}$-type decision tree T has advantage $< \frac{q}{k} + \delta$ over a random input. Let $T(x, y)$ denote the output produced after T receives the answers to its DNF queries. Let u be the leaf reached after seeing only 0's, and say u is labeled with DNFs $(\varphi_1, \ldots, \varphi_q)$ and function $out\colon \{0,1\}^q \to \{0,1\}$ (so if (x, y) leads to u then $T(x, y) = out(\varphi_1(x, y), \ldots, \varphi_q(x, y))$).

We generate the distribution on valid inputs (x, y) as follows. Let $v^0 = w^0 \in \{0,1\}^{n/2}$ be the all-0 string, and for $i = 1, \ldots, \frac{k}{2}$ obtain v^i by flipping a uniformly random 0 of v^{i-1}

to a 1, and for $i = 1, \ldots, \frac{k}{2} - 1$ obtain w^i by flipping a uniformly random 0 of w^{i-1} to a 1. Pick a uniformly random $j \in [\frac{k}{2}]$, and then let (x, y) be either the 1-input (v^j, w^{j-1}) or the 0-input (v^{j-1}, w^{j-1}) with probability $\frac{1}{2}$ each.

Let v denote $(v^0, \ldots, v^{k/2})$ and w denote $(w^0, \ldots, w^{k/2-1})$, and call (v, w) *good* iff:

- for each $j \in [\frac{k}{2}]$: both inputs (v^j, w^{j-1}) and (v^{j-1}, w^{j-1}) lead to u, and
- for each $j \in [\frac{k}{2}]$ and each $i \in [q]$: $\varphi_i(v^j, w^{j-1}) \geq \varphi_i(v^{j-1}, w^{j-1}) \geq \varphi_i(v^{j-1}, w^{j-2})$ (the latter inequality is only required if $j > 1$).

We claim that

(1) $\mathbb{P}[(v, w) \text{ is bad}] < \frac{\delta}{2}$, and

(2) $\mathbb{P}\big[T(x, y) = f(x, y) \,\big|\, (v, w) \text{ is good}\big] \leq \frac{1}{2} + \frac{1}{2} \cdot \frac{q}{k}$,

from which it follows that

$$\mathbb{P}[T(x, y) = f(x, y)] \leq \mathbb{P}\big[T(x, y) = f(x, y) \,\big|\, (v, w) \text{ is good}\big] + \mathbb{P}[(v, w) \text{ is bad}] < \tfrac{1}{2} + \tfrac{1}{2}(\tfrac{q}{k} + \delta).$$

We argue claim (1). Since the path to u queries $o(\delta n)$ locations, with probability $\geq 1 - o(k\delta) > 1 - \frac{\delta}{4}$ each of the 1's placed throughout v and w avoids these locations, in which case the first bullet holds in the definition of good. Fixing j and i in the second bullet, if we condition on $\varphi_i(v^{j-1}, w^{j-1}) = 1$ and choose an arbitrary term of φ_i that accepts (v^{j-1}, w^{j-1}), then since the term has width $o(\delta n)$, with probability $\geq 1 - o(\delta)$ the 1 that is placed to obtain v^j from v^{j-1} avoids this term, in which case the term continues to accept (v^j, w^{j-1}) and so $\varphi_i(v^j, w^{j-1}) = 1$. Thus $\mathbb{P}\big[\varphi_i(v^j, w^{j-1}) \geq \varphi_i(v^{j-1}, w^{j-1})\big] \geq \mathbb{P}\big[\varphi_i(v^j, w^{j-1}) = 1 \,\big|\, \varphi_i(v^{j-1}, w^{j-1}) = 1\big] \geq 1 - o(\delta)$. Similarly, $\mathbb{P}\big[\varphi_i(v^{j-1}, w^{j-1}) \geq \varphi_i(v^{j-1}, w^{j-2})\big] \geq 1 - o(\delta)$. A union bound over j and i shows that the second bullet holds with probability $\geq 1 - o(kq\delta) > 1 - \frac{\delta}{4}$, so finally the two bullets hold simultaneously with probability $> 1 - \frac{\delta}{2}$.

We argue claim (2). Condition on any particular good (v, w). We abbreviate the q-tuple $(\varphi_1(x, y), \ldots, \varphi_q(x, y))$ as $\varphi(x, y) \in \{0, 1\}^q$. Consider the sequence of k inputs (v^0, w^0), $(v^1, w^0), (v^1, w^1), (v^2, w^1), \ldots$ (like climbing a ladder but placing both feet on each rung). Each of these possibilities for (x, y) leads to u and thus $T(x, y) = \mathit{out}(\varphi(x, y))$. Also, the corresponding sequence of $\varphi(x, y)$'s is monotonically nondecreasing in each of the q coordinates. Thus the sequence of inputs can be partitioned into segments of lengths say $\ell_0, \ell_1, \ldots, \ell_q$ (which sum to k) such that for the first ℓ_0 (x, y)'s in the sequence, $\varphi(x, y)$ has weight 0 (hence $T(x, y)$ is the same), and for the next ℓ_1 (x, y)'s in the sequence, $\varphi(x, y)$ is the same weight-1 string (hence $T(x, y)$ is the same), and so on. Since each segment alternates between 0-inputs and 1-inputs of f, we have $T(x, y) = f(x, y)$ for at most $\lceil \frac{\ell_i}{2} \rceil \leq \frac{\ell_i + 1}{2}$ inputs in the i^{th} segment.

Thus, out of the k possibilities for (x, y) given (v, w), at most $\sum_{i=0}^{q} \frac{\ell_i + 1}{2} = \frac{k}{2} + \frac{q+1}{2}$ are such that $T(x, y) = f(x, y)$. This implies that $\mathbb{P}\big[T(x, y) = f(x, y) \,\big|\, (v, w) \text{ is good}\big] \leq \frac{1}{2} + \frac{1}{2} \cdot \frac{q+1}{k}$, which is almost what we want. This issue can be fixed by observing that since k is even and $q + 1$ (the number of segments) is odd, at least one segment must have even length, in which case $\lceil \frac{\ell_i}{2} \rceil = \frac{\ell_i}{2}$. Thus, out of the k possibilities for (x, y) given (v, w), $T(x, y) = f(x, y)$ holds for at most $\frac{k}{2} + \frac{q}{2}$ of them, which gives (2). ◀

4 Zero-sided error

We now prove Theorem 2, restated here for convenience.

▶ **Theorem 2** (Zero-sided error, restated). *For integers $1 \leq q \leq k \leq 4$:*

(i) *If $k = 4$:* $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{q/k>}$.

(ii) *If* $k \leq 3$*:* $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/(k+1)} \subseteq \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{q/k>}$.

(iii) $\mathsf{ZPP}^{\mathsf{NP}[1]}_{1/k} \not\subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/k}$ *relative to an oracle.*

Moreover, the "$q/k>$*" in the inclusion subscripts can be improved to "*q/k*" if* $q < k \geq 3$.

We prove the inclusions (i) and (ii) in Section 4.1 and the separations (iii) in the full version [6].

4.1 Inclusions

Straightforwardly generalizing the proof of $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{ZPP}^{\mathsf{NP}[1]}_{1/4}$ in [3] yields (i), but we take a different tack by showing in Section 4.1.1 that (i) follows directly from Theorem 3. We prove (ii) from first principles in Section 4.1.2; our proof for the case $k = 1$ is equivalent to the one in [3], but we include it for completeness.

4.1.1 Proof of (i)

Let $L \in \mathsf{ZPP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{RP}^{\mathsf{NP}[1]}_{>0}$. By Theorem 3 and closure of $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>0}$ under complement,

$$L \in \mathsf{RP}^{\mathsf{NP}[1]}_{1/2} \text{ by some algorithm } M^1, \qquad L \in \mathsf{RP}^{\mathsf{NP}\|[2]}_{1>} \text{ by some algorithm } M^2,$$
$$\overline{L} \in \mathsf{RP}^{\mathsf{NP}[1]}_{1/2} \text{ by some algorithm } \overline{M}^1, \qquad \overline{L} \in \mathsf{RP}^{\mathsf{NP}\|[2]}_{1>} \text{ by some algorithm } \overline{M}^2.$$

We let each of these four M-algorithms refer to the entire computation, including the NP oracle queries, which we elide for convenience. (Note that $\overline{M}^i$ does not mean "complement of M^i" – it is a different algorithm.) We assume M^2 and $\overline{M}^2$ have advantage $\geq 1 - 2^{-n^d}$ for an arbitrary constant d. Furthermore, we assume all four algorithms have been modified to output $\perp$ instead of 0, and $\overline{M}^1$ and $\overline{M}^2$ have been modified to output 0 instead of 1.

If $q = 1$: $L \in \mathsf{ZPP}^{\mathsf{NP}[1]}_{1/4}$ by running M^1 or $\overline{M}^1$ with probability $\frac{1}{2}$ each.

If $q = 2$: $L \in \mathsf{ZPP}^{\mathsf{NP}\|[2]}_{1/2}$ by running M^1 and $\overline{M}^1$, and if one of them outputs a bit, outputting that bit or $\perp$ otherwise.

If $q = 4$: $L \in \mathsf{ZPP}^{\mathsf{NP}\|[4]}_{1>}$ by running M^2 and $\overline{M}^2$, and if one of them outputs a bit, outputting that bit or $\perp$ otherwise.

If $q = 3$: $L \in \mathsf{ZPP}^{\mathsf{NP}\|[3]}_{3/4>}$ by running M^1 and $\overline{M}^2$ with probability $\frac{1}{2}$, or M^2 and $\overline{M}^1$ with probability $\frac{1}{2}$, and if one of them outputs a bit, outputting that bit or $\perp$ otherwise. This falls slightly short of our promise of showing $L \in \mathsf{ZPP}^{\mathsf{NP}\|[3]}_{3/4}$, but that can be fixed by noting that the proof of Theorem 3 actually shows that M^1 and $\overline{M}^1$ can have advantage $\geq \frac{1}{2} + 2^{-n^e}$ for some constant e depending on L. Then taking $d \geq e$ ensures we get advantage $\geq \frac{1}{2}\left(\frac{1}{2} + 2^{-n^e}\right) + \frac{1}{2}\left(1 - 2^{-n^d}\right) \geq \frac{3}{4}$.

4.1.2 Proof of (ii)

We just prove $\mathsf{ZPP}^{\mathsf{NP}[1]}_{>1/(k+1)} \subseteq \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{q/k>}$; the "moreover" part follows by exactly the same trick (due to [3]) for strengthening $\mathsf{RP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{RP}^{\mathsf{NP}[1]}_{1/2>}$ to $\mathsf{RP}^{\mathsf{NP}[1]}_{>0} \subseteq \mathsf{RP}^{\mathsf{NP}[1]}_{1/2}$, which we describe in the full version [6].

For some constant c we have $L \in \mathsf{ZPP}^{\mathsf{NP}[1]}_{1/(k+1)+n^{-c}}$, witnessed by a polynomial-time randomized algorithm M (taking input x and coin tosses $s \in \{0,1\}^r$) and a language $L' \in \mathsf{NP}$. For an arbitrary constant d, we wish to show $L \in \mathsf{ZPP}^{\mathsf{NP}\|[q]}_{q/k-2^{-n^d}}$.

Fix an input x. The first step is to sample a sequence of $m = O(n^{2c+d})$ many independent strings $s^1, \ldots, s^m \in \{0,1\}^r$, so with probability $\geq 1 - 2^{-n^d}$, the sequence is *good* in the sense that on input x, M still has advantage strictly greater than $\frac{1}{k+1}$ when its coin tosses are chosen uniformly from the multiset $\{s^1, \ldots, s^m\}$. Then we design a polynomial-time randomized algorithm which, given a good sequence, outputs $L(x)$ with probability $\geq \frac{q}{k}$ after making q nonadaptive NP oracle queries, and which has zero-sided error for all sequences (good and bad). Hence, over the random $s^1, \ldots, s^m$ and the other randomness of our algorithm,

$$\mathbb{P}[\text{output is } L(x)] \;\geq\; \mathbb{P}\big[\text{output is } L(x) \,\big|\, s^1, \ldots, s^m \text{ is good}\big] - \mathbb{P}[s^1, \ldots, s^m \text{ is bad}] \;\geq\; \tfrac{q}{k} - 2^{-n^d}.$$

Henceforth fix a good sequence $s^1, \ldots, s^m$, and let z^h and $out^h \colon \{0,1\} \to \{0,1\}$ be the query string and truth table produced by $M_{s^h}(x)$ (so the output is $out^h(L'(z^h))$). We assume w.l.o.g. that out^h is nonconstant. If there is an h such that $out^h \in \{\text{id}, \text{neg}\}$, then our algorithm simply uses the NP oracle to evaluate $L'(z^h)$ and then outputs $out^h(L'(z^h)) = L(x)$. Otherwise, each out^h is one of the four functions out_{ab} (for $ab \in \{0,1\}^2$) that maps a to b and $1-a$ to $\perp$:

	out_{00}	out_{01}	out_{10}	out_{11}
0	0	1	$\perp$	$\perp$
1	$\perp$	$\perp$	0	1

For each $ab \in \{0,1\}^2$ consider the "ab" query:

$\exists\, h :\ out^h = out_{ab}$ and $L'(z^h) = a$?

If $a = 1$ then the "ab" query can be expressed as an NP oracle query: a witness consists of an h with $out^h = out_{ab}$ and a witness for $L'(z^h) = 1$. If $a = 0$ then the "ab" query can be expressed as a coNP oracle query: a nonexistence witness consists of a witness for $L'(z^h) = 1$ for each h such that $out^h = out_{ab}$. We say the "ab" query returns yes iff it indicates the existence of such an h (i.e., the NP oracle returns the bit a). If the "ab" query returns yes, we can safely output b since there exists an h such that $out^h(L'(z^h)) = out_{ab}(a) = b = L(x)$.

Our algorithm is:

1. Identify a set $P \subseteq \{0,1\}^2$ of size k for which there is guaranteed to exist an $ab \in P$ such that the "ab" query would return yes.
2. Pick a uniformly random $Q \subseteq P$ of size q.
3. For each $ab \in Q$ do the "ab" query and output b if it returns yes.
4. Finally output $\perp$ if all queries returned no.

This outputs $L(x)$ with probability $\geq \frac{q}{k}$. We just need to prove that we can indeed find such a P in step 1. Let $H = \{h \in [m] : M_{s^h} \text{ outputs } L(x)\}$ (so by assumption, $|H| > \frac{m}{k+1}$) and $H_{ab} = \{h \in [m] : out^h = out_{ab}\}$. Note that the "$ab$" query would return yes iff $H \cap H_{ab} \neq \emptyset$, and that $H \subseteq H_{0b} \cup H_{1b}$ for $b = L(x)$.

If $k = 3$: Let P contain all ab's except the one with the smallest H_{ab} (which has size $\leq \frac{m}{4}$), breaking ties arbitrarily. Then $H \cap H_{ab} \neq \emptyset$ for at least one $ab \in P$ assuming $|H| > \frac{m}{4}$.

If $k = 2$: If $|H_{00} \cup H_{10}| \leq \frac{m}{3}$ then $L(x) = 1$ assuming $|H| > \frac{m}{3}$, so we can let $P = \{01, 11\}$. Similarly, if $|H_{01} \cup H_{11}| \leq \frac{m}{3}$ then we can let $P = \{00, 10\}$. Otherwise, the smaller of H_{00}, H_{10} has size $\leq \frac{m}{3}$, and the smaller of H_{01}, H_{11} has size $\leq \frac{m}{3}$, so we can let P contain the two ab's corresponding to the larger of H_{00}, H_{10} and the larger of H_{01}, H_{11}, breaking ties arbitrarily.

If $k = 1$: If $|H_{00} \cup H_{10}| \leq \frac{m}{2}$ then $L(x) = 1$ assuming $|H| > \frac{m}{2}$, and furthermore the smaller of H_{01}, H_{11} has size $\leq \frac{m}{2}$, so we can let P contain the ab corresponding to the larger of H_{01}, H_{11}. Similarly, if $|H_{01} \cup H_{11}| < \frac{m}{2}$ then we can let P contain the ab corresponding to the larger of H_{00}, H_{10}.

References

1 Richard Beigel. Bounded Queries to SAT and the Boolean Hierarchy. *Theoretical Computer Science*, 84(2):199–223, 1991. doi:10.1016/0304-3975(91)90160-4.

2 Jin-Yi Cai and Venkatesan Chakaravarthy. On Zero Error Algorithms Having Oracle Access to One Query. *Journal of Combinatorial Optimization*, 11(2):189–202, 2006. doi:10.1007/s10878-006-7130-0.

3 Richard Chang and Suresh Purini. Amplifying $\mathrm{ZPP}^{\mathrm{SAT}[1]}$ and the Two Queries Problem. In *Proceedings of the 23rd Conference on Computational Complexity (CCC)*, pages 41–52. IEEE, 2008. doi:10.1109/CCC.2008.32.

4 Rahul Tripathi. The 1-Versus-2 Queries Problem Revisited. *Theory of Computing Systems*, 46(2):193–221, 2010. doi:10.1007/s00224-008-9126-x.

5 Nikolai Vereshchagin. Relativizability in Complexity Theory. In *Provability, Complexity, Grammars*, volume 192 of *AMS Translations, Series 2*, pages 87–172. American Mathematical Society, 1999.

6 Thomas Watson. Amplification with One NP Oracle Query. Technical Report TR18-058, Electronic Colloquium on Computational Complexity (ECCC), 2018. URL: https://eccc.weizmann.ac.il/report/2018/058/.

7 Thomas Watson. A $\mathrm{ZPP}^{\mathrm{NP}[1]}$ Lifting Theorem. In *Proceedings of the 36th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 59:1–59:16. Schloss Dagstuhl, 2019. doi:10.4230/LIPIcs.STACS.2019.59.