

Approximate Degree in Classical and Quantum Computing

Other titles in Foundations and Trends® in Theoretical Computer Science

Multi-Valued Reasoning about Reactive Systems

Orna Kupferman

ISBN: 978-1-63828-138-2

Quantified Derandomization: How to Find Water in the Ocean

Roei Tell

ISBN: 978-1-63828-092-7

Complexity Theory, Game Theory, and Economics: The Barbados Lectures

Tim Roughgarden

ISBN: 978-1-68083-654-7

Semialgebraic Proofs and Efficient Algorithm Design

Noah Fleming, Pravesh Kothari and Toniann Pitassi

ISBN: 978-1-68083-636-3

Higher-order Fourier Analysis and Applications

Hamed Hatami, Pooya Hatami and Shachar Lovett

ISBN: 978-1-68083-592-2

Approximate Degree in Classical and Quantum Computing

Mark Bun

Boston University
mbun@bu.edu

Justin Thaler

Georgetown University
justin.thaler@georgetown.edu

now

the essence of knowledge

Boston — Delft



Documents

Document Sharing Hub

Foundations and Trends® in Theoretical Computer Science

Published, sold and distributed by:

now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:

now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

M. Bun and J. Thaler. *Approximate Degree in Classical and Quantum Computing*. Foundations and Trends® in Theoretical Computer Science, vol. 15, no. 3-4, pp. 229–423, 2022.

ISBN: 978-1-63828-141-2

© 2023 M. Bun and J. Thaler

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publishers.

Photocopying. In the USA: This journal is registered at the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by now Publishers Inc for users registered with the Copyright Clearance Center (CCC). The ‘services’ for users can be found on the internet at: www.copyright.com

For those organizations that have been granted a photocopy license, a separate system of payment has been arranged. Authorization does not extend to other kinds of copying, such as that for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale. In the rest of the world: Permission to photocopy must be obtained from the copyright owner. Please apply to now Publishers Inc., PO Box 1024, Hanover, MA 02339, USA; Tel. +1 781 871 0245; www.nowpublishers.com; sales@nowpublishers.com

now Publishers Inc. has an exclusive license to publish this material worldwide. Permission to use this content must be obtained from the copyright license holder. Please apply to now Publishers, PO Box 179, 2600 AD Delft, The Netherlands, www.nowpublishers.com; e-mail: sales@nowpublishers.com

Contents

1	Introduction	2
2	Preliminaries	6
2.1	Terminology and Notation	6
2.2	The Cast of Characters	10
3	General Upper Bound Techniques	14
3.1	Interpolation	14
3.2	Chebyshev Approximations	16
3.3	Rational Approximation and Threshold Degree Upper Bounds	19
3.4	Error Reduction for Approximating Polynomials	20
3.5	Robust Composition	21
4	Polynomials from Query Algorithms	24
4.1	A (Very) Brief Introduction to Query Complexity	24
4.2	Upper Bounds from Quantum Algorithms	25
4.3	Consequences of the Vanishing-Error Upper Bound for OR	29
4.4	More Algorithmically Inspired Polynomials	30
4.5	Algorithmically-Inspired Upper Bound for Composed Functions	36

5	Lower Bounds by Symmetrization	39
5.1	Symmetrization Lower Bound for OR	41
5.2	Arbitrary Symmetric Functions	43
5.3	Threshold Degree Lower Bound for the Minsky-Papert CNF	44
6	The Method of Dual Polynomials	48
6.1	A Dual Polynomial for OR_n	52
7	Dual Lower Bounds for Block-Composed Functions	61
7.1	The Approximate Degree of $\text{AND}_m \circ \text{OR}_b$ is $\Omega(\sqrt{m \cdot b})$. .	64
7.2	Hardness Amplification via Dual Block Composition	65
7.3	Some Unexpected Applications of Dual Block Composition	75
8	Beyond Block-Composed Functions	79
8.1	Surjectivity: A Case Study	80
8.2	Other Functions and Applications to Quantum Query Complexity	85
8.3	Approximate Degree of AC^0	86
8.4	Proof of Theorem 8.3	87
8.5	Collision and PTP Lower Bound	95
8.6	Element Distinctness Lower Bound	105
9	Spectral Sensitivity	107
10	Approximate Rank Lower Bounds from Approximate Degree	115
10.1	A Query Complexity Zoo	116
10.2	Communication Complexity	119
10.3	Lifting Theorems: Communication Lower Bounds from Query Lower Bounds	121
10.4	Communication Lower Bounds via Approximate Rank . . .	122
10.5	Sign-Rank Lower Bounds	146
10.6	Extensions to Multiparty Communication Complexity . . .	160
11	Assorted Applications	161
11.1	Secret Sharing Schemes	161
11.2	Learning Algorithms	163

11.3 Circuit Lower Bounds from Approximate Degree Upper	
Bounds	172
11.4 Parity is not in $LTF \circ AC^0$	177
Acknowledgements	182
References	183

Approximate Degree in Classical and Quantum Computing

Mark Bun¹ and Justin Thaler²

¹*Boston University, USA; mbun@bu.edu*

²*Georgetown University, USA; justin.thaler@georgetown.edu*

ABSTRACT

The approximate degree of a Boolean function f captures how well f can be approximated pointwise by low-degree polynomials. This monograph surveys what is known about approximate degree and illustrates its applications in theoretical computer science.

A particular focus of the survey is a method of proving lower bounds via objects called *dual polynomials*. These represent a reformulation of approximate degree using linear programming duality. We discuss in detail a recent, powerful technique for constructing dual polynomials, called “dual block composition”.

Mark Bun and Justin Thaler (2022), “Approximate Degree in Classical and Quantum Computing”, Foundations and Trends® in Theoretical Computer Science: Vol. 15, No. 3-4, pp 229–423. DOI: 10.1561/0400000107.

©2023 M. Bun and J. Thaler

1

Introduction

The ability (or inability) to represent or approximate Boolean functions by polynomials is a central concept in complexity theory, underlying interactive and probabilistically checkable proof systems, circuit lower bounds, quantum complexity theory, and more. In this monograph, we survey what is known about a particularly natural notion of approximation by polynomials, capturing pointwise approximation over the real numbers. The ε -*approximate degree* of a Boolean function $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$, denoted $\widetilde{\deg}_\varepsilon(f)$, is the least total degree of a real polynomial $p: \{-1, 1\}^n \rightarrow \mathbb{R}$ such that

$$|f(x) - p(x)| \leq \varepsilon \text{ for all } x \in \{-1, 1\}^n. \quad (1.1)$$

By total degree of p , we refer to the maximum sum of the degrees of all variables appearing in any monomial. For example, $p(x_1, x_2, x_3) = x_1^2 x_2 x_3^2 + x_1 x_2^3$ has total degree 5.

Every Boolean function is approximated to error $\varepsilon = 1$ by the constant 0 function, implying that $\widetilde{\deg}_1(f) = 0$ for all such f . However, whenever ε is strictly less than 1, $\widetilde{\deg}_\varepsilon(f)$ is a fascinating notion with a rich theory and applications throughout theoretical computer science.

Applications of approximate degree lower bounds. The study of approximate degree is itself a “proto-complexity theory” [2], with point-wise approximation by real polynomials serving as a rudimentary model of computation, and degree acting as a measure of complexity. Moreover, when f has large (say, $n^{\Omega(1)}$) approximate degree, it is also hard to compute in a variety of other computational models. Different models correspond to different settings of the error parameter ε with two regimes of particular interest. First, if $\widetilde{\deg}_{1/3}(f)$ is large, then f cannot be efficiently evaluated by *bounded-error* quantum query algorithms [16].¹ This connection is often referred to as the “polynomial method in quantum computing.”

Second, if $\deg_\varepsilon(f)$ is large *for every* $\varepsilon < 1$, then f is difficult to compute by *unbounded-error* randomized (or quantum) query algorithms (see, e.g., [56, Lemma 6]). These are randomized algorithms that are only required to do slightly better than random guessing, and correspond to the complexity class **PP** (short for probabilistic polynomial time) defined by Gill [63]. This connection has been used to answer long-standing questions in relativized complexity, e.g., in studying the power of statistical zero-knowledge proofs (Section 7.2), and in communication complexity (Section 10). Approximability of f in this error regime, wherein the error ε is allowed to be arbitrarily close to (but strictly less than) 1,² is captured by a notion termed *threshold degree* and denoted $\deg_\pm(f)$.

Applications of approximate degree upper bounds. As just discussed, lower bounds on $\widetilde{\deg}_\varepsilon(f)$ imply hardness results for computing f . There are also many applications of upper bounds on $\widetilde{\deg}_\varepsilon(f)$, typically in the design of fast algorithms in areas such as learning theory [71], [75] (see Section 11.2) and differential privacy [51], [127].

¹The choice of constant $1/3$ is made for aesthetic reasons. Replacing $\varepsilon = 1/3$ with any other constant in $(0, 1)$ changes the ε -approximate degree of f by at most a constant factor.

²Approximate degree is a meaningful notion even for error parameters ε that are *doubly-exponentially* close to 1. In particular, for any degree bound d , there are known Boolean functions that can be approximated by degree- d polynomials to error $1 - 2^{-n^{\Theta(d)}}$ but not to smaller error [48], [98], [99].

In addition to algorithmic applications, approximate degree upper bounds have also been used to prove complexity *lower bounds*. Here is an illustrative example. Suppose one shows that every circuit over n -bit inputs in a class $\mathcal{C}$ can be approximated to error $\varepsilon < 1$ by a polynomial of degree $o(n)$. We know that simple functions f such as Majority and Parity require approximate degree $\Omega(n)$, and therefore cannot be computed by circuits in $\mathcal{C}$. In fact, if $\varepsilon = 1/3$, then one can even conclude that $\mathcal{C}$ is not powerful enough to compute these functions *on average*, meaning that for every circuit $C \in \mathcal{C}$, we have $\Pr_{x \sim \{-1,1\}^n}[C(x) = f(x)] \leq 1/2 + \frac{1}{n^{\omega(1)}}$ [43], [125]. This principle underlies several state-of-the-art lower bounds for frontier problems in circuit complexity (Section 11.3).

Goals of this survey. This survey covers recent progress on proving approximate degree lower and upper bounds and describes some applications of the new bounds to oracle separations, quantum query and communication complexity, and circuit complexity. On the lower bounds side, progress has followed from an approach called the *method of dual polynomials*, which seeks to prove approximate degree lower bounds by constructing solutions to (the dual of) a certain linear program that captures the approximate degree of any function. This survey explains how several of these advances have been unlocked by a particularly simple and elegant technique—called *dual block composition*—for constructing solutions to this dual linear program. We also provide concise coverage of even more recent lower bound technique based on a new complexity measure called *spectral sensitivity*.

On the upper bounds side, recent explicit constructions of approximating polynomials have been inspired by quantum query algorithms. These constructions also involve new techniques that first express the approximations as sums of exponentially many high-degree terms, and then replace each term with a low-degree approximation that is accurate to exponentially small error.

Roadmap and suggestions for reading the survey. After covering preliminaries (Section 2), we begin in Sections 3 and 4 by covering

approximate degree upper bounds, i.e., techniques for constructing low-degree approximations to Boolean functions. We then turn to lower bound techniques, starting with the simpler and older technique of symmetrization (Section 5) before turning to the method of dual polynomials (Section 6). The next two sections provide progressively more sophisticated developments of this technique, with Section 7 introducing dual block composition as a technique for lower bounding the approximate degree of block-composed functions, and Section 8 moving beyond block-composed functions. Section 9 covers approximate degree lower bounds via spectral sensitivity.

The survey then turns to applications of approximate degree upper and lower bounds. Section 10 covers (a variant of) the so-called pattern matrix method for translating approximate degree lower bounds into approximate-rank and communication lower bounds. Section 11 covers assorted additional applications of both upper and lower bounds on approximate degree.

We have primarily organized the survey by technique. For example, all upper bounds that we cover appear in Sections 3 and 4, with the exception of the approximate degree upper bound for a function called Surjectivity that appears in Section 8.1. This organization maximizes technical and conceptual continuity, but does have some downsides. The results are not covered in increasing order of difficulty, e.g., the easiest lower bounds come after the most challenging upper bounds. It also means that for any specific function or class of functions, the tight upper and lower bounds appear in different parts of the survey.

Readers may wish to skip some of the more technical results that we cover on a first reading. Prominent examples include the upper bound for a function called Element Distinctness in Section 4.4, the proof of Theorem 7.7 in Section 7.2 on a state-of-the-art lower bound for block-composed functions, the entirety of Section 8.5 on lower bounds for problems called Collision and Permutation Testing, and the proof of Theorem 10.23 in Section 10.5, which constructs a dual witness for the high threshold-degree of an AC^0 function with certain “smoothness” properties that are important for applications in communication- and circuit-complexity.

2

Preliminaries

2.1 Terminology and Notation

Boolean functions, Hamming weight, etc. In this monograph, we primarily model Boolean functions as mapping the domain $\{-1, 1\}^n$ to the range $\{-1, 1\}$, with -1 interpreted as logical TRUE and $+1$ is interpreted as logical FALSE.

Many authors instead use the domain $\{0, 1\}^n$ or the range $\{0, 1\}$, with 1 interpreted as TRUE and 0 as FALSE. The domain $\{-1, 1\}^n$ and range $\{-1, 1\}$ turn out to be the most convenient choice for proving approximate degree lower bounds. and hence we use this convention throughout the vast majority of this survey. The domain $\{0, 1\}^n$ and the range $\{0, 1\}$ can be more convenient when establishing approximate degree upper bounds; accordingly we do use this domain and range sparingly (see Section 4.5).

We often use a subscript after a function to clarify the number of variables over which it is defined. For example, OR_n denotes the function over domain $\{-1, 1\}^n$ that evaluates to -1 if at least one of its inputs equals -1 , and otherwise evaluates to 1.

For any input $x \in \{-1, 1\}^n$, we let $|x| = \sum_{i=1}^n (1 - x_i)/2$ denote the number of coordinates of x equal to -1 and refer to $|x|$ as the *Hamming weight* of x . We let

$$\mathcal{A}(x) = \sum_{i=1}^n x_i.$$

Note that $|x|$ and $\mathcal{A}(x)$ are both degree-1 polynomials in x .

We use $[n]$ to denote the set $\{1, 2, \dots, n\}$, $[n]^*$ to denote $\{0, 1, \dots, n\}$, and $\mathbf{1}_n$ to denote the input in $\{-1, 1\}^n$ in which all entries are 1.

For a real number t , we let $\text{sgn}(t)$ equal 1 if t is nonnegative and equal -1 if t is negative. All logarithms in this survey have base 2 unless specified otherwise.

Binomial coefficients. For integers $0 \leq k \leq n$, $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ denotes the binomial coefficient, and $\binom{n}{\leq k}$ denotes the partial sum $\sum_{i=0}^k \binom{n}{i}$. Two standard upper bounds on these quantities are

$$\binom{n}{k} \leq \binom{n}{\leq k} \leq (ne/k)^k$$

and for $n, k > 1$,

$$\binom{n}{\leq k} \leq n^k.$$

Function composition and product distributions. For two functions f_m, g_b , we denote by $f_m \circ g_b$ the block-composed function over domain $(\{-1, 1\}^b)^m$, i.e.,

$$(f_m \circ g_b)(x_1, \dots, x_m) := f(g(x_1), \dots, g(x_m)).$$

Given probability distributions $\mu_1, \dots, \mu_m$ over $\{-1, 1\}^b$, we let $\otimes_{i=1}^m \mu_i$ denote the product distribution over $(x_1, \dots, x_m) \in (\{-1, 1\}^b)^m$ in which x_i is drawn from distribution μ_i . Given a single probability distribution μ , the distribution $\mu^{\otimes n}$ is the product distribution over $(x_1, \dots, x_n)$ that chooses each x_i independently according to distribution μ . Given a finite set Y , we use $y \sim Y$ to indicate that y is drawn uniformly at random from Y .

Polynomials and their degree. We assume that all polynomials with domain $\{-1, 1\}^n$ are multilinear. This means that they have degree at most one in each variable, e.g., $p(x_1, x_2, x_3) = x_1x_2x_3$ is multilinear, but $p(x_1, x_2, x_3) = x_1^2x_2x_3$ is not. This is without loss of generality because $x_i^2 = 1$ whenever $x_i \in \{-1, 1\}$. We denote the degree of a univariate polynomial q by $\deg(q)$. For a multivariate polynomial p , we denote by $\deg(p)$ the total degree of p , i.e., the maximum sum of variable degrees over all monomials of p with nonzero coefficients. For example, the polynomial $p(x_1, x_2, x_3) = x_1x_2x_3 + x_1x_3$ has total degree three.

Approximate degree and threshold degree. Recall that in applications of approximate degree, two regimes for the error parameter ε are of particular relevance. The first is $\varepsilon = 1/3$. For brevity, we use $\widetilde{\deg}(f)$ as a shorthand for $\widetilde{\deg}_{1/3}(f)$, and refer to this quantity without qualification as the *approximate degree* of f . The second regime of special interest considers all ε arbitrarily close to, but strictly less than, 1. This regime is equivalent to a notion called the *threshold degree* of f , denoted $\deg_{\pm}(f)$, which is the least degree of a polynomial p such that

$$p(x) \cdot f(x) > 0 \text{ for all } x \in \{-1, 1\}^n. \quad (2.1)$$

It is not hard to see that the threshold degree of f is greater than d if and only if for *every* $\varepsilon < 1$, f cannot be approximated to error ε by any degree- d polynomial. Any function p that satisfies Condition (2.1) is said to *sign-represent* f , and p is called a *polynomial threshold function* (PTF) for f . If p has degree 1, then it is called a *linear threshold function* (LTF) for f . Another term for an LTF is a *halfspace*. If a nonzero polynomial p satisfies Condition (2.1) with weak rather than strict inequality, p is said to *weakly sign-represent* f .

Basics of Fourier analysis. For $S \subseteq [n]$, let $\chi_S(x) = \prod_{i \in S} x_i$; we refer to χ_S as the *parity function* over S , or sometimes as the S 'th parity function. We will occasionally refer to any such parity function as a *Fourier basis function* or simply as a *monomial*.

For any function $f: \{-1, 1\}^n \rightarrow \mathbb{R}$, there is a unique multilinear polynomial

$$p(x) = \sum_{x \in \{-1, 1\}^n} \hat{f}(S) \cdot \chi_S(x)$$

such that $p(x) = f(x)$ for all $x \in \{-1, 1\}^n$. The quantity $\hat{f}(S)$ is called the S 'th Fourier coefficient of f . The *degree* of f (sometimes referred to as the Fourier degree of f) is the total degree of p , i.e., the size of the largest set S such that $\hat{f}(S) \neq 0$. Sometimes p is referred to as the *multilinear extension* of f , because, unlike f , it makes sense to evaluate p even at inputs that are not in $\{-1, 1\}^n$.

For a real-valued function $f: \{-1, 1\}^n \rightarrow \mathbb{R}$ given by $f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \chi_S(x)$, the *Fourier weight* of f , denoted $\text{weight}(f)$, is $\sum_{S \subseteq [n]} |\hat{f}(S)|$. In other words, $\text{weight}(f)$ is the ℓ_1 -norm of the Fourier coefficients of f .¹ Likewise, the *Fourier sparsity* of f , denoted $\text{sparsity}(f)$, is the number of non-zero Fourier coefficients of f .

The following basic fact offers a crude upper bound on the Fourier sparsity and Fourier weight of any low-degree function.

Fact 2.1. Let $f: \{-1, 1\}^n \rightarrow \mathbb{R}$ satisfy $|f(x)| \leq A$ for all $x \in \{-1, 1\}^n$. If f has Fourier degree at most d , then the Fourier sparsity of f is at most $\binom{n}{\leq d}$ and the Fourier weight of f is at most $A \cdot \binom{n}{\leq d} \leq A \cdot n^d$.

Proof. The set $\{\chi_S: S \subseteq [n]\}$ forms an orthonormal basis for the vector space of all functions mapping $\{-1, 1\}^n \rightarrow \mathbb{R}$ under the inner product relation

$$\langle p, q \rangle = 2^{-n} \sum_{x \in \{-1, 1\}^n} p(x) \cdot q(x).$$

Accordingly, we can express the Fourier coefficients of f via:

$$\hat{f}(S) = \langle f, \chi_S \rangle = 2^{-n} \sum_{x \in \{-1, 1\}^n} f(x) \cdot \chi_S(x).$$

Hence, $|\hat{f}(S)| \leq A$ for each $S \subseteq [n]$. If f has Fourier degree at most d , then $\hat{f}(S) \neq 0$ for at most $\binom{n}{\leq d}$ subsets S . The fact follows. $\square$

¹We remark that many authors use the term “Fourier weight” to refer to the (squared) ℓ_2 -norm of the Fourier coefficients of f – this is not the convention used here.

We use a special notation for the parity function on all n bits, $\chi_{[n]}$, denoting it by $\oplus_n$. The bit-wise parity of two vectors $x, y \in \{-1, 1\}^n$ is denoted $x \oplus y$.

A useful inequality. Several times in this survey, we use the following fact:

Fact 2.2. For any real number $t > 1$,

$$(1 - 1/t)^t \leq 1/e,$$

and

$$(1 - 1/t)^{t-1} \geq 1/e,$$

where $e \approx 2.718$ is Euler's constant.

2.2 The Cast of Characters

We briefly introduce the most important functions that appear throughout this survey. The approximate degree of each of these functions is now mostly or completely understood, and we state what is known for each (see Table 2.1 for a summary). Implications of these degree bounds are detailed later in the survey. Beyond their direct consequences, the efforts that led to these results yielded new and broadly applicable techniques for bounding approximate degree.

Symmetric functions. A *symmetric* function $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ is any function that is invariant under permutations of its input bits. Equivalently, a function is symmetric if its value on any input x depends only on $|x|$, the number of -1 s, also known as the Hamming weight of x . Examples of symmetric functions include the OR, AND, Parity, and Majority functions. These ask whether their inputs have Hamming weight that is greater than 0, exactly n , odd, or at least $n/2$, respectively.

The approximate degree and threshold degree of symmetric functions are completely understood, with several proofs of the upper and lower bounds now known. For approximate degree, the upper and lower bounds are covered in Sections 4.3 and 5.2 (see also Section 7.3), while for threshold degree they are covered in Sections 3.1 and 5.2.

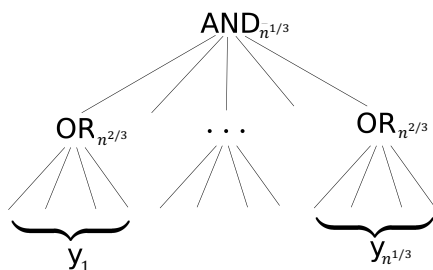


Figure 2.1: The Minsky-Papert CNF on n variables. The top AND gate has fan-in $n^{1/3}$, while each of the bottom OR gates have fan-in $n^{2/3}$. For each $i = 1, 2, \dots, n^{1/3}$, $y_i \in \{-1, 1\}^{n^{2/3}}$ denotes the input to the i th OR gate.

Theorem 2.3. Let $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ be a symmetric function and t be the smallest number such that f is constant on all inputs of Hamming weight between t and $n - t$. Then for $\varepsilon \in (2^{-n}, 1/3)$, we have $\widetilde{\deg}_\varepsilon(f) = \Theta\left(\sqrt{nt} + \sqrt{n \log(1/\varepsilon)}\right)$.

Theorem 2.4. Let $[n]^* = \{0, 1, \dots, n\}$. Let $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ be a symmetric function where $f(x) = F(|x|)$ for a function $F : [n]^* \rightarrow \{-1, 1\}$. Then $\deg_\pm(f) = |\{i : F(i) \neq F(i+1)\}|$.

DNF and CNF formulas. A function that has served a central historical role in the study of approximate degree and threshold degree is the so-called *Minsky-Papert* DNF and CNF.² These are, respectively, the read-once DNF and CNF over n variables, with top fan-in $n^{1/3}$ and bottom fan-in $n^{2/3}$. Here, the top fan-in refers to the number of terms or clauses in the DNF or CNF, respectively, while the bottom fan-in refers to the width of each term or clause. See Figure 2.1 for a depiction. It is known that the Minsky-Papert DNF and CNF have approximate degree $\Theta(n^{1/2})$ (Sections 3.5 and 7.1) and threshold degree $\tilde{\Theta}(n^{1/3})$ (Sections 3.3 and 5.3).³

²A DNF formula (short for *disjunctive normal form*) of size s is an OR of at most s ANDs, where each AND is evaluated over a set of literals. Here, a literal is an input variable or its negation. So, for example, $\text{OR}(x_1 \wedge \bar{x}_2, x_1 \wedge x_3, x_2 \wedge \bar{x}_3)$ is a DNF of size 3. A CNF is defined analogously with the role of OR and AND reversed.

³The tilde notations $\tilde{\Theta}$, $\tilde{\Omega}$, or $\tilde{O}$ hide polylogarithmic, rather than merely constant, factors.

Properties of lists. Several important functions, mainly arising from the quantum computing literature, are best thought of as “properties” of lists of numbers. Let $N \geq R$ with R a power of 2. Each of the following functions takes as input a vector in $x \in \{-1, 1\}^n$ with $n = N \log_2 R$. It interprets the vector as a list of (the binary representations of) N numbers $(k_1, \dots, k_N)$ from range $[R] = \{1, \dots, R\}$, and it outputs -1 if and only if that list has the stated property. For example, suppose $N = R = 4$, and the range elements $1, \dots, 4$ are represented in binary by $(1, 1)$, $(1, -1)$, $(-1, 1)$, and $(-1, -1)$. Then $x = ((1, 1), (-1, 1), (1, -1), (-1, -1))$ would be interpreted as the following list of four numbers: $(1, 3, 2, 3)$.

Two properties of such a list that one might be interested in are: Does every range element appear at least once in the list? Does any range element appear more than once? As we discuss next, the first property is captured by the so-called Surjectivity function, and the second by the Element Distinctness function.

Surjectivity (SURJ): For every $i \in [R]$, does there exist at least one index j such that $k_j = i$? Equivalently, if the input list $(k_1, \dots, k_N)$ is thought of as the evaluation table of some function, then SURJ evaluates to -1 if the function is surjective.

It is known that if $R = N/2$, then the approximate degree of SURJ is $\tilde{\Theta}(n^{3/4})$ (Section 8.1) and the threshold degree is $\tilde{\Theta}(n^{1/2})$ (Section 8.1).

For a positive integer ℓ , call a list of N numbers ℓ -to-1 if ℓ divides N , and N/ℓ range items each appears exactly ℓ times in the list. So for example, if $N = 4$ and $R = 4$, then the list $(4, 3, 2, 1)$ is 1-to-1, and the list $(4, 4, 2, 2)$ is 2-to-1. If $x \in \{-1, 1\}^n$ is interpreted as a list $(k_1, \dots, k_N)$ that is ℓ -to-1, we also refer to x itself as ℓ -to-1. If $N = R$, we refer to 1-to-1 inputs as *permutations*.

Element Distinctness ED: Is the list 1-to-1? That is, for every $i \in [R]$, is it the case that there is at most one index j such that $k_j = i$? Or put yet another way, are all numbers in the list distinct?

Element Distinctness generalizes to the k -distinctness function k -ED for integers $k > 2$. This function interprets its input as a list of N numbers from a range of size R and outputs 1 if and only if there is some range item that appears at least k times in the list.

Table 2.1: Approximate degree and threshold degree of the “cast of characters”—specific functions on n bits playing a prominent role in the study of approximate degree. The symmetric t -threshold function evaluates to -1 if and only if the Hamming of the input is at least t .

Function	Approximate Degree	Threshold Degree
OR, AND	$\Theta(n^{1/2})$	1
t -threshold function ($t \leq n/2$)	$\Theta(\sqrt{nt})$	1
Minsky-Papert DNF, CNF	$\Theta(n^{1/2})$	$\tilde{\Theta}(n^{1/3})$
Surjectivity	$\tilde{\Theta}(n^{3/4})$	$\tilde{\Theta}(n^{1/2})$
Element Distinctness	$\tilde{\Theta}(n^{2/3})$	$\tilde{\Theta}(1)$
Collision, Permutation Testing	$\tilde{\Theta}(n^{1/3})$	$\tilde{\Theta}(1)$

The Collision Problem: Is the list 1-to-1, under the promise that it is either 1-to-1 or 2-to-1?

Permutation Testing (PTP): Is the list a permutation, under the promise that it is either a permutation or *far* from any permutation? By far from any permutation, we mean that at least, say, 10% of range items do not appear even once in the list.

The above three functions all have threshold degree $\tilde{O}(1)$,⁴ but their approximate degrees are much more interesting. For $N = R$, ED has $(1/3)$ -approximate degree $\tilde{\Theta}(n^{2/3})$ (Sections 4.4 and 8.6). Meanwhile, the Collision and Permutation Testing problems have $(1/3)$ -approximate degree $\tilde{\Theta}(n^{1/3})$ (Section 8.5).⁵ k -ED is known to have approximate degree approaching $n^{3/4}$ from below for large values of k . Exactly how quickly $\widetilde{\deg}_{1/3}(k\text{-ED})$ approaches $n^{3/4}$ as k grows remains open, i.e., for every fixed $k > 2$, there remains a polynomial gap between the known upper and lower bounds on $\widetilde{\deg}_{1/3}(k\text{-ED})$.

⁴ED, the Collision Problem, and PTP each evaluates to FALSE if and only if there is one or more *collisions* in the input list, meaning a pair of list elements $k_j = k_{j'}$ with $j \neq j'$. One can count the number of collisions in the input list by summing over each pair of list elements and checking if the pair collides. This yields a polynomial of degree $\tilde{O}(1)$. An appropriate affine transformation of the polynomial approximates each of the three functions to error $1 - \Theta(1/n^2)$.

⁵See Section 7.3 for discussion of how to define the approximate degree of promise problems such as these.

3

General Upper Bound Techniques

In this section, we discuss several fundamental techniques for proving approximate degree and threshold degree upper bounds. Notable examples include polynomial interpolation, approximation via Chebyshev polynomials, and sign-representation via rational approximations.

3.1 Interpolation

A simple place to begin is sign representation of symmetric functions. Recall that a symmetric function f mapping $\{-1, 1\}^n$ to $\{-1, 1\}$ is one that depends only on the Hamming weight of its input, so it is characterized by $f(x) = F(|x|)$ for some univariate function $F: [n]^* \rightarrow \{-1, 1\}$. One example is the OR_n function, which takes the value 1 when $|x| = 0$ and the value -1 otherwise.

It is easy to show that, for $\varepsilon \geq 1 - 1/n$, $\widetilde{\deg}_\varepsilon(\text{OR}_n) = 1$. To see this, let

$$p(x) = \frac{1}{n}(1 - 2|x|). \quad (3.1)$$

Then $\deg(p) = 1$ since $|x|$ is a linear function of x , and moreover

$$|p(x) - \text{OR}_n(x)| \leq 1 - 1/n \text{ for all } x \in \{-1, 1\}^n. \quad (3.2)$$

We can think of the above polynomial p as arising from the following process. Rather than considering F , which is defined over the discrete domain $[n]^*$, consider the piecewise-linear function G that “extends” the domain of F to the continuous interval $[0, n] \subset \mathbb{R}$, i.e., such that

$$G(i) = F(i) \text{ for all } i \in [n]^*. \quad (3.3)$$

In the case of OR_n , G has a single root, at input $t = 1/2$, and Equation (3.1) defines p to be the degree-1 function with this root (p is then scaled appropriately to minimize its approximation error, ensuring that Equation (3.2) holds).

In general, let f be a symmetric function that changes sign k times as the Hamming weight of its input increases from 0 to n . Then f can be sign-represented by a polynomial p of degree k , i.e., $\text{sgn}(p(x)) = f(x)$ for all $x \in \{-1, 1\}^n$.

More explicitly, let $f(x) = F(|x|)$ where $t_1, \dots, t_k$ are the Hamming weights where $F(t_i) \neq F(t_i + 1)$. Then either the following degree- k polynomial sign-represents f , or its negation does:

$$p(x) = (|x| - (t_1 + 1/2))(|x| - (t_2 + 1/2)) \dots (|x| - (t_k + 1/2)). \quad (3.4)$$

Indeed, one can check that $p(x) > 0$ if $|x| \geq t_k + 1$ since in this case all terms in the product on the right hand side of Equation (3.4) are strictly positive. Similarly, $p(x) < 0$ if $|x| \in [t_{k-1} + 1, t_k]$, since in this case the final term in the product is negative and all other terms are positive. And $p(x) > 0$ if $|x| \in [t_{k-2} + 1, t_k]$, since the final two terms in the product are negative and all other terms are positive. And so on. See Figure 3.1 for an example.

The polynomial p in Equation (3.4) is precisely the degree- k polynomial that has the same roots as the natural piecewise-linear function G satisfying Equation (3.3). We refer to this construction as performing interpolation, since it is defining p via its evaluations at specific points, namely at G ’s roots.

The threshold degree upper bound given above is in fact tight for any symmetric function, as we show in Section 5.2. Unfortunately, while the behavior of the constructed sign-representing polynomial p is precisely controlled at the interpolation points, p can grow rapidly as one moves away from those points. In particular, $|p(x)|$ can be as large as $n^{\Theta(k)}$ for

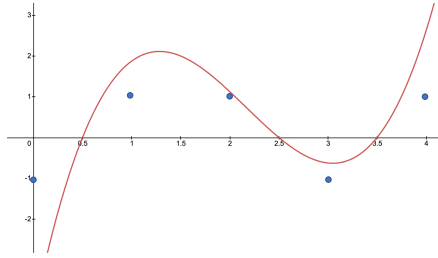


Figure 3.1: A degree-3 polynomial p defined via Equation (3.4) to sign-represent the following function $f(x) : \{-1, 1\}^n \rightarrow \{-1, 1\}$ with three “sign changes”. Define $f(x) = F(|x|)$ where $F(0) = -1$, $F(1) = F(2) = 1$, $F(3) = -1$ and $F(k) = 1$ for all $k \in \{4, \dots, n\}$. The behavior of F at inputs in $\{0, 1, 2, 3, 4\}$ are depicted with blue dots.

some inputs $x \in \{-1, 1\}^n$ (and it is easily checked that $|p(x)|$ is never less than $1/2$ for any $x \in \{-1, 1\}^n$). This means that interpolation-based constructions do not tell us much about the ε -approximate degree of symmetric functions when ε is less than $1 - n^{-O(k)}$. For that, we need more sophisticated tools. The next section describes perhaps the single most important such tool: the Chebyshev polynomials.

3.2 Chebyshev Approximations

Approximate degree considers low-degree point-wise approximations of functions over the discrete, n -dimensional domain $\{-1, 1\}^n$. For centuries, the field of approximation theory has studied a related notion: low-degree point-wise approximations of functions defined over the continuous, unidimensional domain $[-1, 1]$. A typical question in this area is as follows: Given a function $F : [-1, 1] \rightarrow [-1, 1]$, what is the least degree of a real polynomial $P : [-1, 1] \rightarrow \mathbb{R}$ such that $|P(x) - F(x)| \leq \varepsilon$ for all $x \in [-1, 1]$? So-called “Jackson-type” theorems give general answers to this question in terms of the continuity and smoothness properties of F . The *Chebyshev polynomials* are central tools for proving results of this type.

Definition 3.1. For $d \geq 0$, the d ’th Chebyshev polynomial of the first kind is the unique polynomial $T_d : \mathbb{R} \rightarrow \mathbb{R}$ of degree d such that $T_d(\cos \theta) = \cos(d\theta)$ for every $\theta \in \mathbb{R}$.

One can show using the double-angle formula in trigonometry that the function T_d in Definition 3.1 is indeed a polynomial of degree at most d . The Chebyshev polynomials have several alternative characterizations, including one given by the recurrence $T_0(x) = 1$, $T_1(x) = x$, and $T_d(x) = 2xT_{d-1}(x) - T_{d-2}(x)$.

Their importance to classical approximation theorem stems from a number of “extremal” properties. For instance, $x^d - 2^{-d+1}T_d(x)$ is the polynomial of degree $d - 1$ that provides the best pointwise approximation to x^d . The Chebyshev polynomials are also extremal for a classical result called *Markov’s inequality* [89].¹ This inequality states that if G is degree- d polynomial that is bounded over the interval $[-1, 1]$, then its derivative $G'(t)$ cannot be too large at any point within the interval.

Theorem 3.2. (Markov’s inequality) Let $G: [-1, 1] \rightarrow [-1, 1]$ be a real polynomial of degree at most d . Then $\max_{t \in [-1, 1]} |G'(t)| \leq d^2$.

The Chebyshev polynomials are exactly extremal for this result: For any integer $d > 0$, the degree- d Chebyshev polynomial T_d satisfies $|T_d(t)| \leq 1$ for all $t \in [-1, 1]$, while $T'_d(t) \geq d^2$ for all $t \in [1, \infty]$, with equality at $t = 1$. In particular, for $d = \lfloor \sqrt{2n} \rfloor$, $T'_d(1) \approx 2n$.

It should come as little surprise, then, that Chebyshev polynomials are useful tools for the discrete approximation problem captured by approximate degree. Their simplest application is to a tight upper bound on the approximate degree of the OR function [94].

Lemma 3.3. $\widetilde{\deg}_{1/3}(\text{OR}_n) = O(\sqrt{n})$.

Proof. Our goal is to construct a polynomial $p(x)$ such that $p(1^n) = 1$ and $p(x) = -1$ for all other x . Recalling that $\mathcal{A}(x) = \sum_{i=1}^n x_i$, we may do so by defining $p(x) = q(\mathcal{A}(x)/n)$ where q is a univariate polynomial of degree $O(\sqrt{n})$ such that $q(1) = 1$ and $q(t) \leq -2/3$ for all $t \in [-1, 1 - 2/n]$. That is, as its input t decreases from 1, the polynomial q “jumps” very quickly from 1 down toward -1 , and stays near -1 until t leaves the interval $[-1, 1 - 2/n]$.

¹Not to be confused with Markov’s inequality from probability theory, but rather a special case of the “Markov brothers’ inequality” attributed jointly to A.A. Markov and V.A. Markov.

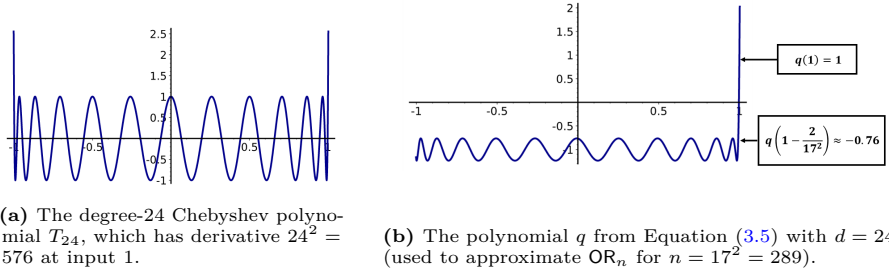


Figure 3.2: Approximating the OR function over 289 variables with a degree-24 polynomial.

By shifting and scaling T_d without increasing its degree (i.e., by performing an affine transformation), we can obtain a univariate polynomial q with these properties. En route to constructing this polynomial q , consider the polynomial $T_d(t + 4/d^2)$ where $d = \lfloor \sqrt{2n} \rfloor$. While $T_d(t + 4/d^2)$ is bounded by 1 for all $t \in [-1, 1 - 2/n]$, the fact that $T'_d(t) \geq d^2$ for all $t \geq 1$ ensures that it jumps up to at least 4 for $t = 1$. An additional affine shift produces the following polynomial q with the desired behavior:

$$q(t) = 2 \cdot \frac{T_d(t + 4/d^2)}{T_d(1 + 4/d^2)} - 1. \quad (3.5)$$

See Figures 3.2 (a) and (b) for illustrations of $T_d(t)$ and $q(t)$ when $n = 17^2 = 289$ and $d = \lfloor \sqrt{2n} \rfloor = 24$.

□

A slight generalization of this proof shows that even for small values of ε , we have $\widetilde{\deg}_\varepsilon(\text{OR}_n) = O(\sqrt{n} \log(1/\varepsilon))$ for vanishing ε (one can also apply generic error reduction as discussed in Section 3.4). This turns out not to be optimal, as OR_n can be approximated to error ε using degree $O(\sqrt{n} \log(1/\varepsilon))$ by taking advantage of the discrete nature of the problem. We describe this upper bound in detail in Section 4.2.

Combining what we know about sign representation and Chebyshev approximation already allows us to construct interesting sign representations for CNF formulas. Specifically, consider the block-composed function $\text{AND}_m \circ \text{OR}_b$. Let $x = (x_1, \dots, x_m) \in (\{-1, 1\}^b)^m$ denote an arbitrary input to this function.

Lemma 3.4. $\deg_{\pm}(\text{AND}_m \circ \text{OR}_b) = O(\sqrt{b \log m})$.

Proof. Let p be a polynomial of degree $O(\sqrt{b \log m})$ that approximates OR_b to error $1/(3m)$. Letting $x_i \in \{-1, 1\}^b$ denote the input to the i th OR gate, the following polynomial of degree $O(\sqrt{b \log m})$ sign-represents $(\text{AND}_m \circ \text{OR}_b)(x_1, \dots, x_m)$:

$$q(x_1, \dots, x_m) := -1 + \sum_{i=1}^m (1 + p(x_i)). \quad (3.6)$$

Indeed, if $\text{OR}(x_i) = -1$ for all i , then $|1 + p(x_i)| \leq 1/(3m)$ for all i , and hence $q(x) \leq -2/3 < 0$. Meanwhile, if $\text{OR}(x_i) = 1$ for even a single i , then $(1 + p(x_i)) \geq 2 - 1/(3m)$ and hence $q(x) > 0$. $\square$

Note that the sign representing polynomial q constructed above is *not* a bounded-error approximation, since it can grow to as large as $\Omega(m)$ on inputs for which $\text{OR}(x_i) = 1$ for $\Omega(m)$ values of i .

3.3 Rational Approximation and Threshold Degree Upper Bounds

Another way to sign-represent a CNF formula is to approximate each OR_b using a *ratio* of low-degree polynomials, combine these as before using a linear sign-representation of AND_m , and then “clear the denominator” to obtain a polynomial. This is a key idea underlying Beigel, Reingold, and Spielman’s famous result that $\mathbf{PP}$ is closed under intersection [20]. Specifically, there are degree-1 polynomials $p(x), q(x)$ over domain $\{-1, 1\}^b$ such that the ratio p/q approximates OR_b to error $1/(3m)$ as follows. For $M \geq 6m$, let $p(x) = 1 - M \cdot |x|$ and $q(x) = 1 + M \cdot |x|$. Then if $x = \mathbf{1}_b$, we have $p(x)/q(x) = \frac{1}{1} = 1$, while if $x \neq \mathbf{1}_b$, we have $\frac{p(x)}{q(x)} \in \left[-1, \frac{1-M}{1+M}\right] \subseteq \left[-1, -1 + \frac{1}{3m}\right]$.

Since $p(x)/q(x)$ approximates OR_b to error $1/(3m)$, by analogy with Equation (3.6), the following quantity sign-represents $(\text{AND}_m \circ \text{OR}_b)(x_1, \dots, x_m)$:

$$-1 + \sum_{i=1}^m \left(1 + \frac{p(x_i)}{q(x_i)}\right). \quad (3.7)$$

Unfortunately, Expression (3.7) is not itself a low-degree polynomial; rather, it is a sum of ratios of degree-1 polynomials. To get a polynomial

that sign-represents $\text{AND}_m \circ \text{OR}_b$, we place all terms in the sum of Expression (3.7) over the common denominator $r(x) = \prod_{j=1}^m q(x_j)$. That is, for $i = 1, \dots, m$, let $s_i(x) := p(x_i) \cdot \prod_{j=1, \dots, m: j \neq i} q(x_j)$. Then Expression (3.7) becomes

$$\frac{1}{r(x)} \cdot \left((m-1)r(x) + \sum_{i=1}^m s_i(x) \right).$$

Finally, observe that $r(x) > 0$ for all $x \in \{-1, 1\}^n$, as it is a product of polynomials $q(x_j)$ which are all positive. Hence, multiplication by the denominator $r(x)$ does not alter the sign of the expression. This means that $p^*(x) := (m-1)r(x) + \sum_{i=1}^m s_i(x)$ sign-represents $\text{AND}_m \circ \text{OR}_b$ and is clearly a polynomial of degree at most m .

Lemma 3.5. $\deg_{\pm}(\text{AND}_m \circ \text{OR}_b) = O(m)$.

Lemmas 3.4 and 3.5 together imply that $\text{AND}_m \circ \text{OR}_b$ has threshold degree at most

$$O(\min\{\sqrt{b \log m}, m\}).$$

Minsky and Papert [91] famously showed a (nearly) matching lower bound of $\Omega(\min\{\sqrt{b}, m\})$. Later, we describe multiple proofs of Minsky and Papert's lower bound; see Theorem 5.6 for one based on symmetrization and Theorem 7.11 for a more general result based on the method of dual polynomials.

3.4 Error Reduction for Approximating Polynomials

Theorem 3.6. Let $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$ be any function. Then for any $\varepsilon > 0$, $\widetilde{\deg}_{\varepsilon}(f) \leq O(\widetilde{\deg}(f) \cdot \log(1/\varepsilon))$.

Proof. We follow the exposition of [57, Claim 4.3], which uses a construction called amplifying polynomials. For even integers $\ell > 0$, define the degree- ℓ univariate polynomial

$$A_{\ell}(u) := \sum_{k \geq \ell/2} \binom{\ell}{k} ((1+u)/2)^k ((1-u)/2)^{\ell-k}.$$

On any input $u \in [-1, 1]$, $A_\ell(u)$ equals the probability that the majority of tosses of a coin come up Heads, when the probability of seeing Heads is $(1+u)/2$. This polynomial has the following properties (easily proved via the Chernoff bound):

- If $u \in [3/5, 1]$, then $2A_\ell(u) - 1 \in [1 - 2e^{-\ell/6}, 1]$.
- If $u \in [-1, -3/5]$, then $2A_\ell(u) - 1 \in [-1, -1 + 2e^{-\ell/6}]$.

Let p be a $1/3$ -approximating polynomial for f , and $\ell = O(\log(1/\varepsilon))$ be a large enough even integer such that $2e^{-\ell/6} \leq \varepsilon$. Then the composition $2A_\ell(p) - 1$ is an ε -approximation for f . $\square$

There is a natural algorithmic interpretation of the construction of Theorem 3.6. Later, in Section 4.1, we will see that for a polynomial p approximating a Boolean function f , $(1 - p(x))/2$ can often be thought of as outputting the acceptance probability of some (randomized or even quantum) algorithm $\mathcal{A}$ computing f , when $\mathcal{A}$ is run on input x . In this context, $A_\ell(p(x))$ is the acceptance probability of the algorithm that runs ℓ independent copies of $\mathcal{A}$ on x and outputs the majority answer.

3.5 Robust Composition

A beautiful and important result of Sherstov (refining earlier work of Buhrman *et al.* [39]) shows that ε -approximate degree can increase at most multiplicatively under block composition.

Theorem 3.7 ([108]). Let $f: \{-1, 1\}^m \rightarrow \{-1, 1\}$ and $g: \{-1, 1\}^n \rightarrow \{-1, 1\}$ be arbitrary Boolean functions. Then $\widetilde{\deg}(f \circ g) \leq O(\deg(f) \cdot \deg(g))$.

Sketch. A natural approach is to attempt to construct a low-degree approximation for $f \circ g$ by simply taking low-degree approximations p and q to f and g respectively, and hoping that the composition $p \circ q$ approximates $f \circ g$. Unfortunately, this does not work directly. The issue is that while $|p(x) - f(x)| \leq 1/3$ for all $x \in \{-1, 1\}^m$, this does not grant us any information about p 's behavior at *non-Boolean* inputs,

while q 's outputs may indeed be non-Boolean, because q does not *exactly* compute g , but rather only approximates it at each input in $\{-1, 1\}^n$. Hence, understanding the behavior of $p \circ q$ requires us to control p 's behavior not only at inputs in $\{-1, 1\}^m$, but in fact at real-valued inputs.

Fortunately, q 's outputs are not totally arbitrary real numbers: since q approximates a Boolean function g to error at most $1/3$, we know that $q(x) \in [-4/3, -2/3] \cup [2/3, 4/3]$ for all $x \in \{-1, 1\}^n$ (in particular, q never outputs values in the interval $(-2/3, 2/3)$). What we need is a way to make p *robust* to q 's errors, in the sense that, for $i = 1, \dots, m$, feeding $q(x_i)$ as the i 'th input to p rather than $g(x_i)$ does not substantially alter the output of p .

Sherstov [108], building on earlier work of [39], showed how to accomplish this with only a constant-factor increase in the degree of q . Specifically, he proved the following lemma, whose proof we omit. However, in the next section (Theorem 4.6), we do prove a weaker result with a far simpler (and algorithmically-inspired) proof, due to Buhrman *et al.* [39].

Lemma 3.8 (Sherstov [108]). Let $\delta > 0$ and $p: \{-1, 1\}^m \rightarrow [-4/3, 4/3]$. Then there is a corresponding polynomial $p_{\text{robust}}: \mathbb{R}^m \rightarrow \mathbb{R}$ of degree $O(\deg p + \log 1/\delta)$ that is robust to noise in the inputs: $|p(x) - p_{\text{robust}}(x + \varepsilon)| < \delta$ for all $x \in \{-1, 1\}^m$ and all $\varepsilon \in [-1/3, 1/3]^n$.²

Accordingly, one can construct an approximating polynomial for $f \circ g$ by taking any $(1/3)$ -approximating polynomial q for g and any $(1/4)$ -approximating polynomial p for f , applying Theorem 3.8 to p (with $\delta = 1/12$) to obtain a polynomial p_{robust} with only a constant-factor blowup in degree. Theorem 3.8 implies that for all inputs $(x_1, \dots, x_m) \in \{-1, 1\}^{n \cdot m}$ to $f \circ g$, $|(p_{\text{robust}} \circ q)(x_1, \dots, x_m) - (f \circ g)(x_1, \dots, x_m)| < (1/4) + (1/12) = 1/3$.

□

²Note that p_{robust} may not be multilinear, as Theorem 3.8 requires control of its behavior at inputs outside of $\{-1, 1\}^m$.

In contrast to the situation for related measures such as quantum query complexity, it is still open whether the bound in Theorem 3.7 is tight for every pair of functions f, g .

Open Problem 3.9. For every pair of total Boolean functions f, g , is it the case that $\widetilde{\deg}(f \circ g) \geq \Omega(\widetilde{\deg}(f) \cdot \widetilde{\deg}(g))$?

4

Polynomials from Query Algorithms

4.1 A (Very) Brief Introduction to Query Complexity

In (deterministic) query complexity, an algorithm is given oracle access to the bits of an unknown input $x \in \{-1, 1\}^n$. Its goal is to evaluate a known function f on x by making as few queries to the oracle as possible. Quantum query complexity is a generalization of this model wherein the algorithm is allowed to make queries in superposition, and must output $f(x)$ with probability at least $2/3$. Here, making queries in superposition roughly means that at each step, the algorithm assigns an amplitude (the quantum analog of a probability) to each possible query that it might make, in the same way a randomized algorithm at each step assigns a probability to each bit to determine which one to query at random. We refer the reader to [12] for details of the model and a recent survey of results.

While quantum query complexity is an information-theoretic model (i.e., the query algorithm is allowed to spend as long as it wants to decide which bits of x to query and to process the oracle's responses to the queries), it turns out to capture much of the power of quantum computing: Most query-efficient quantum algorithms can be realized as time-efficient algorithms and vice versa. While we draw heavily on

intuition from quantum query complexity in this survey, we do not require any details of the quantum query model beyond the (limited and vague) description given above.

Beals *et al.* [16] proved a result that is central to our understanding of quantum query complexity. They showed that for any quantum query algorithm for f making at most T queries on every input, there is a polynomial $p: \{-1, 1\}^n \rightarrow \mathbb{R}$ of total degree at most $2T$ such that for all $x \in \{-1, 1\}^n$, $p(x)$ equals the probability that the algorithm outputs -1 when run on input x .

Since this probability is at least $2/3$ when $f(x) = -1$ and at most $1/3$ when $f(x) = 1$, it follows that $1 - 2p(x)$ approximates f to error $2/3$. In other words, if f is computed by a quantum query algorithm of cost at most T , then the $(2/3)$ -approximate degree of f is at most $2T$.

This has two (equivalent) implications. The first is relevant to deriving approximate degree upper bounds: if one can give a T -query quantum algorithm for f , then one can conclude that the approximate degree of f is *at most* $O(T)$. The second is relevant to deriving quantum query lower bounds: if one can show that $\widetilde{\deg}(f) \geq d$, then the quantum query complexity of f is *at least* $\Omega(d)$ (see Theorem 10.1 in Section 10.1).

4.2 Upper Bounds from Quantum Algorithms

The vanishing-error approximate degree of OR.

Theorem 3.6 takes any $(1/3)$ -approximation to f and turns it into an ε -approximation with an $O(\log(1/\varepsilon))$ -factor blowup in degree. For many functions f , this is not the most efficient way to obtain an ε -approximation. For example, recall from Theorem 3.3 that the approximate degree of OR is $O(\sqrt{n})$. Applying Theorem 3.6 to $f = \text{OR}$ would give an upper bound of $O(\sqrt{n} \cdot \log(1/\varepsilon))$. The following result shows that the dependence on $\log(1/\varepsilon)$ can be improved quadratically. We show a matching lower bound in Section 7.3.

Theorem 4.1 (Buhrman, Cleve, de Wolf, Zalka ([37])). For any $\varepsilon \in [3^{-n}, 1)$, $\widetilde{\deg}_\varepsilon(\text{OR}_n) = O\left(\sqrt{n \log(1/\varepsilon)}\right)$.

Our preferred construction of an ε -approximating polynomial for OR derives the polynomial from an ε -error quantum query algorithm for

OR due to [37]. To understand this algorithm, one need not know any details of the quantum query complexity model. All one needs to know is the following fact about (a variant of) Grover’s search algorithm.

Fact 4.2. For any integer $\ell > 0$, there is a quantum query algorithm Grover_ℓ making $O(\sqrt{n/\ell})$ queries to x such that

- If x has Hamming weight 0, the algorithm rejects with probability 1.
- If x has Hamming weight exactly ℓ , the algorithm accepts with probability 1.
- If $\ell < |x| \leq n$, the algorithm accepts with probability at least $2/3$.

That is, Grover_ℓ computes $\text{OR}(x)$ with error probability at most $1/3$ under the promise that its input has Hamming weight equal to 0 or at least ℓ . And moreover, its error probability is *zero* on the input of Hamming weight 0 or any input of Hamming weight exactly ℓ .

For the reader interested in the underlying quantum algorithms, Fact 4.2 follows by combining two standard variants of Grover search from [32], [34] that each individually makes $O(\sqrt{n/\ell})$ queries. The first is an “exact” version of search which identifies a marked item in an unsorted list of n elements with probability 1 under the promise that exactly ℓ items are marked. The second is a “promise” version of search that finds such a marked element with probability $2/3$ under the promise that at least ℓ items are marked. The guarantee of Fact 4.2 follows by running the exact version of search, and if it fails, running the promise version of search.

With these facts in hand, we now describe the ε -error quantum algorithm for OR of [37].

Let $t = \log(1/\varepsilon)$. For $i = 1, 2, \dots, t$, run Grover_i on x and halt and accept if Grover_i accepts. Accept if any run accepts; otherwise reject.

Error Analysis. The above algorithm rejects with probability 1 when run on the input of Hamming weight 0 because each call to Grover_i for $i = 1, \dots, t$ rejects with probability 1 when run on this input. If run on an input x of Hamming weight between 1 and t , the algorithm

accepts with probability 1, because $\text{Grover}_{|x|}$ will be called on x and will accept with probability 1. Finally, if run on an input x of Hamming weight more than t , the algorithm will accept with probability at least $1 - (1/3)^t \geq 1 - \varepsilon$, because each call to Grover_i will independently accept with probability at least $2/3$.

Query Cost. The number of queries the algorithm makes to the input is

$$\sum_{i=1}^t O\left(\sqrt{n/i}\right) \leq O\left(\sqrt{n \log(1/\varepsilon)}\right).$$

The inequality here follows from the fact that $\sum_{i=1}^t \frac{1}{\sqrt{i}} \leq 1 + \int_1^t \frac{1}{\sqrt{x}} dx = 1 + 2t^{1/2} - 1 = O(t^{1/2})$.

By the seminal result of Beals *et al.* [16] that the ε -error quantum query complexity of any function f is, up to a constant factor, at least its ε -approximate degree, we may immediately conclude from the above algorithm that $\widehat{\deg}_\varepsilon(\text{OR}_n) \leq O(\sqrt{n \log(1/\varepsilon)})$. However, we find it instructive to explicitly construct the ε -approximating polynomial, below.

Turning the algorithm into a polynomial. Let p_i be the univariate polynomial of degree $O(\sqrt{n/i})$ corresponding to the acceptance probability of Grover_i , i.e., p_i has the following four properties.

- $p_i(0) = 0$.
- $p_i(i) = 1$.
- $p_i(r) \in [2/3, 1]$ for $r \in \{i, i+1, \dots, n\}$,
- $p_i(r) \in [0, 1]$ for all $r \in \{0, 1, \dots, n\}$.

The polynomial p_i can be constructed explicitly as follows. The construction of Section 3.2 (after a suitable affine transformation) yields a univariate polynomial Q_n of degree $O(\sqrt{n})$ such that $Q_n(0) = 0$, $Q_n(1) = 1$, $Q_n(k) \in [2/3, 1]$ for all $k \in [1, n]$, and $|Q_n(k)| \leq 1$ for all $k \in [0, n]$. Defining

$$p_i(k) := Q_{n/i}(k/i) \tag{4.1}$$

ensures that p_i has degree $O(\sqrt{n/i})$ and has the four properties above.

Consider

$$p := p_1 + p_2 \cdot (1 - p_1) + p_3 \cdot (1 - p_2) \cdot (1 - p_1) + \dots + p_t \cdot (1 - p_{t-1}) \cdot (1 - p_{t-2}) \cdot \dots \cdot (1 - p_2) \cdot (1 - p_1)$$

where $t = \log(2/\varepsilon)$.

Intuitively, this captures the acceptance probability of the algorithm that, like [37], first runs Grover_1 , and if it returns -1 then it halts and outputs -1 , and if not it runs Grover_2 and it halts and outputs -1 , and if not it runs Grover_3 , and so forth up to Grover_t .

Another way of expressing p is as $p = 1 - \prod_{i \in [t]} (1 - p_i)$. Using this expression, it is easy to check that:

- $p(0) = 0$.
- $p(r) = 1$ for $r \in \{1, \dots, t\}$. Indeed, $p_r(r) = 1$ implies that $\prod_{i \in [t]} (1 - p_i(r)) = 0$, so $p(r) = 1$.
- $p(r) \in [1 - \varepsilon/2, 1]$ for $r \in \{t + 1, \dots, n\}$. That $p(r) \geq 1 - \varepsilon/2$ follows from the fact that $p_i(r) \geq 2/3$ for every $i \in \{1, \dots, t\}$, and hence $p(r) \geq 1 - \left(\frac{1}{3}\right)^t \geq 1 - \varepsilon/2$.

That $p(r) \leq 1$ follows from the fact that $p_i(r) \in [0, 1]$ for every i , and hence $\prod_{i \in [t]} (1 - p_i(r)) \in [0, 1]$.

The degree of p is $\sum_{i=1}^t O(\sqrt{n/i}) = O(\sqrt{n \cdot t}) = O(\sqrt{n \log(1/\varepsilon)})$. Hence, $-1 + 2p(|x|)$ is an ε -approximation to OR_n of degree $O(\sqrt{n \log(1/\varepsilon)})$.

Additional discussion of error reduction. As previously discussed, Theorem 4.1 shows that the generic error reduction result of Theorem 3.6 is not tight for OR , in that it yields a quadratically suboptimal dependence on $\log(1/\varepsilon)$. In fact, it is not known whether Theorem 3.6 is tight for *any* total Boolean function. As far as we know, it could even be the case that for every total function f of approximate degree at most $O(n^c)$ for constant $c \in (0, 1)$, the ε -approximate degree of f is at most $O(n^c \cdot \log(1/\varepsilon)^{1-c})$.

Open Problem 4.3. For some constant $c \in (0, 1)$ exhibit a total Boolean function f with ε -approximate degree $\Theta(\min\{n^c \log(1/\varepsilon), n\})$ for all $\varepsilon \in [2^{-n}, 1/3]$, or show that no such function exists.

4.3 Consequences of the Vanishing-Error Upper Bound for OR

Theorem 4.4. Let $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$ be any Boolean function with $|f^{-1}(-1)| \leq T$, where $f^{-1}(-1) = \{x: f(x) = -1\}$. Then $\widetilde{\deg}_\varepsilon(f) \leq O\left(\sqrt{n \log T} + \sqrt{n \log(1/\varepsilon)}\right)$.

Proof. First, we represent f as a sum of T conjunctions as follows. For each input $y \in \{-1, 1\}^n$, let $\text{EQ}_y(x): \{-1, 1\}^n \rightarrow \{0, 1\}$ denote the function that evaluates to 1 if and only if $x = y$. Then

$$f(x) = 1 - 2 \sum_{y \in T} \text{EQ}_y(x).$$

Note that $\text{EQ}_y(x)$ can be computed by composing (the negation of) OR with the degree-1 function $x_i \mapsto y_i x_i$.

Next, for each $y \in T$, let p_y denote a polynomial that approximates $\text{EQ}_y(x)$ to error ε/T (this can be done with degree $O(\sqrt{n \log(T/\varepsilon)})$ by Theorem 4.1), and define $p(x) = 1 - 2 \sum_{y \in T} p_y(x)$. By the triangle inequality, p approximates f to error at most $T \cdot (\varepsilon/T) = \varepsilon$. $\square$

Consequences for symmetric functions. Suppose $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$ is a symmetric function that is constant at all inputs of Hamming weight between t and $n - t$. Then either f satisfies

$$|f^{-1}(-1)| \leq 2 \sum_{i=0}^t \binom{n}{i} \leq 2n^t \quad (4.2)$$

or the negation of f does. If f satisfies Equation (4.2) then Theorem 4.4 implies that $\widetilde{\deg}_\varepsilon(f) \leq O(\sqrt{nt \log n} + \sqrt{n \log(1/\varepsilon)})$. This result is tight up to a $\sqrt{\log n}$ factor (the tight bound is $O(\sqrt{nt} + \sqrt{n \log(1/\varepsilon)})$ [131]). If the negation of f satisfies Equation (4.2) then we can draw the same conclusion, as approximate degree is preserved under negation.

The results above approximate f by expressing f as a sum of conjunctions, approximates each term of the sum to exponentially small

error, and sums the approximations. The same technique can be used to prove that the approximate degree of the Surjectivity function, SURJ, is $\tilde{O}(n^{3/4})$. We defer coverage of this result to Theorem 8.2 in Section 8.1. The technique arises yet again in the $\tilde{O}(n^{2/3})$ upper bound for Element Distinctness that we prove shortly, in Section 4.4.

4.4 More Algorithmically Inspired Polynomials

Collision and PTP.

Recall that the Collision and Permutation Testing (PTP) problems for domain size N and range size R were defined in Section 2.2. For example, in the PTP problem, the goal is to distinguish the case that the input list is a permutation, from the case that the list is far from any permutation.

The following quantum algorithm for the Collision and PTP problems, due to Brassard, Hoyer, and Tapp [33], solves both problems with query cost $O(N^{1/3})$. For an input list $k = (k_1, \dots, k_N) \in [R]^N$, we describe the algorithm assuming it can learn any k_i with a single query; if the list is actually specified via a bit-vector $x \in (\{-1, 1\}^{\log R})^N$ as per Section 2.2, the query cost would increase by a $\log R$ factor, as each k_i can be learned with $\log R$ queries to bits of x .

For an input list $k = (k_1, \dots, k_N) \in [R]^N$, randomly sample a set S of $N^{1/3}$ list elements. Let $k|_S$ denote the sampled list elements and $k|_{\bar{S}}$ denote the unsampled list elements. If $k|_S$ contains a collision, i.e., $k_i = k_j$ for distinct $i, j \in S$, then halt and accept. Otherwise, let $\mathcal{R} \subseteq [R]$ denote the range elements appearing in the sample, noting that $\mathcal{R}$ is now known to the algorithm. Apply Grover_ℓ (see Fact 4.2) to $k|_{\bar{S}}$ with $\ell = |S|/100$ to search for an element of $\mathcal{R}$ appearing in $k|_{\bar{S}}$, accepting if such an element is found, and rejecting otherwise.

Query cost. The number of queries made by the algorithm is $|S|$, plus the number of queries made by Grover_ℓ . In total, this is $O(|S| + \sqrt{N/|S|}) = O(N^{1/3})$ queries.

Correctness analysis. For simplicity, we analyze the algorithm’s correctness for the Collision problem, though a nearly identical analysis applies to PTP. If the input list k is 1-to-1, then the algorithm will never find a collision and so it rejects with probability 1. If the input list k is 2-to-1, then the algorithm accepts with probability at least $2/3$. This is because, if k is a 2-to-1 input, then either $k|_S$ contains a collision or else there are $|S| > |S|/100$ “cross-collisions” (i.e., $j \notin S$ such that $k_j = k_i$ for at least one $k_i \in S$).

Explicit construction of the resulting polynomials. It is straightforward to translate this quantum query algorithm into an explicit polynomial of degree $O(N^{1/3} \cdot \log R)$ approximating the Collision and PTP problems. For an input $x \in \{-1, 1\}^N$, write $x = (x_1, \dots, x_N) \in (\{-1, 1\}^{\log R})^N$. For a set $S \subseteq [N]$, let $x|_S$ denote the subvector $(x_i : i \in S) \in (\{-1, 1\}^{\log R})^{|S|}$. For a vector of range elements $r = (r_1, \dots, r_{|S|}) \in [R]^{|S|}$, let $\text{EQ}_r(x|_S)$ denote that function that equals 1 if $x|_S$ is the binary representation of $r_1, \dots, r_{N^{1/3}}$. Note that $\text{EQ}_r(x|_S)$ is computed exactly by a polynomial of degree at most $|S| \log R$. Finally, let $\text{inr}_r(x|_{\bar{S}}) \in \{-1, 1\}^{N-|S|}$ be the bit-vector with entries indexed by $[N] \setminus S$ such that $(\text{inr}_r(x|_{\bar{S}}))_i$ equals -1 if and only if there is some $j \in [|S|]$ with $x_i = r_j$. Note that each entry of $\text{inr}_r(x|_{\bar{S}})$ depends on only $\log R$ bits of $x|_{\bar{S}}$, and hence is exactly computed by a polynomial of degree $\log R$.

Then consider the polynomial:

$$\frac{1}{\binom{N}{N^{1/3}}} \sum_{S \subseteq [N] : |S|=N^{1/3}} p_S(x) + q_S(x), \quad (4.3)$$

where

$$p_S(x) = \sum_{r=(r_1, \dots, r_{N^{1/3}}) \in [R]^{N^{1/3}} : r_i = r_j \text{ for some } i \neq j} \text{EQ}_r(x|_S)$$

and

$$q_S(x) = \sum_{r=(r_1, \dots, r_{N^{1/3}}) \in [R]^{N^{1/3}} : \text{all } r_i \text{ are distinct}} \text{EQ}_r(x|_S) \cdot p_\ell(\text{inr}_r(x|_{\bar{S}})).$$

Here, p_ℓ is the polynomial capturing the acceptance probability of Grover_ℓ given in Equation (4.1) of Section 4.2.

The polynomial in Equation (4.3) exactly computes the acceptance probability of the quantum algorithm above. Intuitively, p_S outputs 1 if the sample $x|_S$ contains a collision and 0 otherwise, while $q_S(x)$ outputs the probability that the sample $x|_S$ does not contain a collision, yet Grover search finds a “cross-collision”, i.e., an item outside of the sample that collides with an item in the sample. Equation (4.3) outputs the average of these acceptance probabilities over the random choice of S .

Element Distinctness.

Recall from Section 2.2 that the k -distinctness function k -ED (for constant k) interprets its input as a list of N numbers $(k_1, \dots, k_N)$ from a range of size R and outputs -1 if and only if there is some range item that appears at least k times in the list. The special case $k = 2$ is a particularly natural and is referred to as Element Distinctness, or ED for short. Much later in the survey, Section 8.6 establishes a lower bound of $\Omega(N^{2/3})$ on the approximate degree of ED when $R \geq N$. In this section, we sketch a matching upper bound.

The upper bound has been known since the discovery of an $O(N^{2/3})$ -query quantum algorithm due to Ambainis [11] (recall that Theorem 10.1 translates any quantum query algorithm for a function f into an approximating polynomial for f). But an explicit description of an approximating polynomial was not given until work of Sherstov [114]. We cover Sherstov’s construction, though our presentation is quite different than the treatment in [114].

For illustration, we start by giving a simple quantum algorithm for the negation of ED that makes $O(N^{5/6})$ queries. This implies the existence of an approximating polynomial for ED of degree $O(N^{5/6})$. We then explain how to lower the degree bound to the optimal $O(N^{2/3})$.

The $O(N^{5/6})$ -query algorithm utilizes a subroutine with the following property. Say that a list item k_i is involved in a collision if there is some $j \neq i$ such that $k_j = k_i$. If there is any list item involved in a collision, the subroutine will find the collision with probability at least $1/N^{2/3}$.

SUBROUTINE 1: Randomly sample $b = N^{1/3}$ inputs, then determine if any sampled item is involved in a collision. That is, randomly pick a set $S \subseteq [N]$ of size b and query $\{k_i: i \in S\}$. Let $\{r_1, \dots, r_b\}$ denote the multiset of sampled range items. If there exist distinct $i, j \in [b]$ such that $r_i = r_j$, halt and accept. Otherwise, Grover-search over the un-sampled items $\{k_i: i \notin S\}$ for an appearance of one of $r_1, \dots, r_b$, accepting if Grover search finds such an appearance, and rejecting otherwise. This costs $O(b + \sqrt{N})$ queries (b queries for the sample, $O(\sqrt{N})$ for the Grover search).

If there is one or more k_i 's involved in a collision, the probability that at least one such k_i is sampled by SUBROUTINE 1 is at least $b/N \geq \Omega(1/N^{2/3})$, and conditioned on this event, the probability that the subroutine accepts is at least $2/3$. Meanwhile, if there is no k_i involved in a collision, then obviously the probability that SUBROUTINE 1 accepts is 0. Hence, by Theorem 10.1, SUBROUTINE 1 can be transformed into a polynomial p of degree at most $O(b + \sqrt{N})$ such that $p(x) = 0$ for all $x \in (\text{ED})^{-1}(-1)$ while $p(x) \geq b/N$ for all $x \in (\text{ED})^{-1}(1)$.

Using a technique in quantum algorithm design called amplitude amplification, one can boost the success probability from $\Omega(b/N)$ to $2/3$ while increasing the degree by a factor of $O(\sqrt{N/b}) = O(N^{1/3})$. This yields a query upper bound of $O(N^{1/2} \cdot N^{1/3}) = O(N^{5/6})$. The following theorem captures the analog of amplitude amplification in the context of approximate degree, yielding the claimed $O(N^{5/6})$ approximate degree upper bound for ED.

Theorem 4.5. Let $f: \{-1, 1\}^n \rightarrow \{-1, 1\}$. Suppose that there is a polynomial p of degree at most d , such that $p(x) = 0$ for all $x \in f^{-1}(-1)$ and $\delta \leq p(x) \leq 1$ for all $x \in f^{-1}(1)$. Then the $(1/3)$ -approximate degree of f is at most $O(d \cdot \sqrt{1/\delta})$.

Proof. Using Chebyshev polynomials (cf. Section 3.2), we can construct a univariate polynomial A of degree $O(\sqrt{1/\delta})$ such that $A(0) = 1$ and $A(x) \in [-4/3, -2/3]$ for all $x \in [-1+\delta, 1]$. Then $-A(p(x))$ approximates f to error $1/3$ and has degree $d \cdot \deg(A) \leq O(d \cdot \sqrt{1/\delta})$. $\square$

In summary, we have identified a polynomial approximating ED of the form $-A \circ p$, where A is the polynomial of degree $O(N^{1/3})$ appearing in the proof of Theorem 4.5 and $p(x)$ is the acceptance probability of SUBROUTINE 1 on input x .

A first attempt at improvement. How can one improve the above approximate degree upper bound of $O(N^{5/6})$? First, let us replace the invocation of Grover search in SUBROUTINE 1, which has query cost only $O(\sqrt{N})$ but has non-zero error probability, with an *exact* search (exact search has trivial query cost $\Omega(N)$, but failure probability zero). Let q denote the acceptance probability of this algorithm. Then just as for $-A \circ p$, $-A \circ q$ is a $(1/3)$ -approximation to ED, though its degree is much larger than that of $-A \circ p$.

What actually works. The key idea is that, owing to its composed structure, $A \circ q$ itself can be approximated pointwise by a polynomial of degree just $O(N^{2/3})$, despite the fact that q itself has degree $\Omega(N)$. Since $-A \circ q$ approximates ED, any sufficiently accurate pointwise approximation to $-A \circ q$ will itself approximate ED.

To approximate $A \circ q$, the idea is to express $A \circ q$ as a weighted sum of $2^{O(\deg(A))}$ terms, with each term equal to a power of $(1 - q)$, i.e., of the form $c_i \cdot (1 - q(x))^i$ for some $c_i \in \mathbb{R}$ and integer $i \leq \deg(A)$. We will then approximate each term to exponentially small error, ensuring that the sum of the term-by-term approximations is a good approximation to $A \circ q$.

For example, if $A(t) = 2t^3 - 4t^2 + 3t - 1$, then

$$\begin{aligned} (A \circ q)(x) &= 2q(x)^3 - 4q(x)^2 + 3q(x) - 1 \\ &= -2(1 - q(x))^3 + 2(1 - q(x))^2 - (1 - q(x)). \end{aligned}$$

If we approximate the three terms in this expression, namely $-2(1 - q(x))^3$, $2(1 - q(x))^2$, and $-(1 - q(x))$, individually to error $1/12$ and sum the approximations, we obtain a polynomial Q such that $|Q(x) - (A \circ q)(x)| \leq 3 \cdot 1/12 = 1/4$ for all x in the domain of ED. Note that this technique is identical to that used to approximate any symmetric function (Theorem 4.4) and to approximate SURJ (Theorem 8.2). Details follow.

As the polynomial A from Theorem 4.5 is derived from the Chebyshev polynomial of degree $\deg(A)$, it can be shown that the ℓ_1 -norm of the coefficients of A is $2^{O(\deg(A))}$. That is,

$$A(x) = \sum_{i=0}^{\deg(A)} c_i \cdot (1 - q(x))^i$$

for some real numbers $c_0, \dots, c_{\deg(A)}$ such that

$$\sum_{i=0}^{\deg(A)} |c_i| \leq 2^{O(\deg(A))}.$$

Hence, it suffices to approximate each power $(1 - q(x))^i$ of $q(x)$ to error $\varepsilon = 2^{-\Theta(\deg(A))}$. Summing the approximations yields a polynomial Q such that $|Q(x) - (A \circ q)(x)| \leq \varepsilon \cdot 2^{O(\deg(A))} \leq o(1)$ for all inputs x to ED.

What degree suffices to approximate $(1 - q(x))^i$ to error ε ? Since q specifies the acceptance probability of SUBROUTINE 1 (modified to use exact search rather than Grover search), the quantity

$$(1 - q(x))^i$$

captures the probability that i independent runs of SUBROUTINE 1 all output reject. Here, running $i \leq \deg(A)$ independent instances of SUBROUTINE 1 corresponds to taking i independent subsamples of the input list, with each subsample having size b , and then for each subsample (if there is no collision within the subsample), determining whether any range item from the sample appears elsewhere in the input list.

The key insight is that the above probability is unchanged if we do not run $\deg(A)$ independent searches, one for each subsample of size b , but rather run a *single* search, looking for a second occurrence of any range item in the “combined” subsample of size $b \cdot \deg(A)$. If the single search procedure fails to find a second occurrence of a range item from the combined sample, this is equivalent to the failure of all $\deg(A)$ independent searches, one for each subsample.

Performing the single search procedure using ε -error Grover search (Section 4.2), the acceptance probability of this algorithm

can be approximated to error $\varepsilon = 2^{-\Theta(\deg(A))}$ with degree $O\left(b \cdot \deg(A) + \sqrt{n \cdot \log(1/\varepsilon)}\right) = O(N^{2/3})$ as claimed. Here, $b \cdot \deg(A)$ comes from computing $\deg(A)$ subsamples each of size b , and $\sqrt{n \cdot \log(1/\varepsilon)}$ comes from running an ε -error Grover search procedure to determine if any range item from any of the $\deg(A)$ subsamples appears amongst the $N - b \cdot \deg(A)$ un-sampled items in the input list.

4.5 Algorithmically-Inspired Upper Bound for Composed Functions

We now prove a weaker version of Theorem 3.7, which upper bounds the approximate degree of a composed function $f \circ g$ by the product of the approximate degree of f and the δ' -approximate degree of g where δ' is much lower than in Theorem 3.7, namely $O(1/m)$ rather than a positive constant. The reason we include this theorem despite its sub-optimality is that it has a clean, algorithmically-inspired proof.

Theorem 4.6. Let $\varepsilon, \delta > 0$ and suppose that $f: \{-1, 1\}^m \rightarrow \{-1, 1\}$ satisfies $\widetilde{\deg}_\varepsilon(f) \leq d$ and $g: \{-1, 1\}^n \rightarrow \{-1, 1\}$ satisfies $\deg_{\delta/m}(g) \leq d'$. Then $\widetilde{\deg}_{\varepsilon+\delta}(f \circ g) \leq d \cdot d'$.

Proof. For purposes of this proof, it will be convenient to treat f as a function mapping $\{0, 1\}^m \rightarrow \{0, 1\}$ rather than mapping $\{-1, 1\}^m \rightarrow \{-1, 1\}$, and accordingly to treat g as function from $\{0, 1\}^n$ to $\{0, 1\}$.

Let p be a polynomial of degree that approximates f to error ε , i.e., $|p(x) - f(x)| \leq \varepsilon$ for all $x \in \{0, 1\}^m$. Similarly, let q be a polynomial that approximates g to error at most δ/m . We claim that $p \circ q$ approximates $f \circ g$ to error at most $\varepsilon + \delta$.

The intuition for why this is true is as follows. Suppose that the polynomials p and q , when evaluated on a Boolean input, output the acceptance probability of randomized query algorithms $\mathcal{A}_f$ and $\mathcal{A}_g$ when run on that input. That is, one can think of $p(x)$ as the probability that $\mathcal{A}_f$ accepts when run on input x , and similarly $q(y)$ as the probability that $\mathcal{A}_g$ accepts when run on input y .¹ The fact that p approximates f to error ε means that, on each input y in $\{0, 1\}^m$, $\mathcal{A}_f$ errs when run on

¹In general, p and q may not actually correspond to the acceptance probability of any low-query algorithm, even if they output values in $[0, 1]$ when evaluated at any Boolean input. That is, although for any T -query randomized algorithm, there

y with probability at most ε (i.e., accepts inputs in $f^{-1}(0)$ or rejects inputs in $f^{-1}(1)$), and similarly for $\mathcal{A}_g$.

Let $(x_1, \dots, x_m) \in (\{0, 1\}^n)^m$ denote an input to $f \circ g$, and let $y = (g(x_1), \dots, g(x_m)) \in \{0, 1\}^m$. Then it turns out that $p \circ q$ is the acceptance probability of the natural “composed algorithm” $\mathcal{A}_{f \circ g}$ that runs $\mathcal{A}_f$ on y , and every time $\mathcal{A}_f$ queries a bit y_i of y , it answers the query by running $\mathcal{A}_g$ on x_i . Since $\mathcal{A}_g$ errs with probability at most δ/m , then by a union bound over all (at most m) calls to $\mathcal{A}_g$, with probability at least $1 - \delta$, $\mathcal{A}_g$ returns $g(x_i)$ for every i on which it is called. In this case, the composed algorithm returns the same answer that $\mathcal{A}_f$ returns when run on input $y = (g(x_1), \dots, g(x_m))$. Since $\mathcal{A}_f$ itself errs with probability at most ε , the error probability of $\mathcal{A}_{f \circ g}$ is at most $\varepsilon + \delta$. Since $(p \circ q)(x_1, \dots, x_m)$ is the acceptance probability of $\mathcal{A}_{f \circ g}$ on input $(x_1, \dots, x_m)$, this means that $p \circ q$ approximates $f \circ g$ to error at most $\varepsilon + \delta$.

Here is the formal analysis. Let us assume that $q(x) \in [0, 1]$ for all $x \in \{0, 1\}^n$ and $p(y) \in [0, 1]$ for all $y \in \{0, 1\}^m$.² Recall that p is a multilinear polynomial. Since for any $(x_1, \dots, x_m) \in (\{0, 1\}^n)^m$, $(q(x_1), \dots, q(x_m)) \in [0, 1]^m$, Fact 4.7 below implies that $p(x_1, \dots, x_m)$ equals the expectation of $p(y)$ under the product distribution over $y \in \{0, 1\}^m$ in which $y_i = 1$ with probability $q(x_i)$. Since q approximates g to error at most δ/m , the i th coordinate in this product distribution equals $g(x_i)$ with probability at least $1 - \delta/m$. By a union bound of all m coordinates, this product distribution places mass at least $1 - \delta$ on the point $(g(x_1), \dots, g(x_m))$. Hence, $p(q(x_1), \dots, q(x_m))$ lies in the interval

$$[p(g(x_1), \dots, g(x_m)) - \delta, p(g(x_1), \dots, g(x_m)) + \delta].$$

exists a degree- T polynomial computing its acceptance probability on each input, the converse is not true [10], [114] (see Section 8.2 for details). Nonetheless, thinking of p and q as if they do compute acceptance probabilities of query algorithms can be a powerful source of intuition regarding their behavior.

²In general, a (δ/m) -approximation q to g can take values in $[-\delta/m, 1 + \delta/m]$, but the assumption can be ensured by replacing q with $(q(x) + \delta/m)/(1 + 2\delta/m)$. This maintains the degree of q while increasing the error to at most $2\delta/m$. Similarly, p can be assumed to only take values in $[0, 1]$ with at most a doubling of its error.

Since $|p(y) - f(y)| \leq \varepsilon$ for all $y \in \{0, 1\}^m$, this implies that

$$|p(q(x_1), \dots, q(x_m)) - f(g(x_1), \dots, g(x_m))| \leq \varepsilon + \delta.$$

□

Fact 4.7. For any multilinear polynomial $q: \mathbb{R}^n \rightarrow \mathbb{R}$ and point $(u_1, \dots, u_n) \in [0, 1]^n$, $q(u_1, \dots, u_n)$ equals the expected value of q under the product distribution in which the i th coordinate is chosen to equal 1 with probability u_i and 0 with probability $(1 - u_i)$.

Proof. By linearity of expectation, it suffices to consider a multilinear polynomial q consisting of a single monomial, say, $q(x) = x_1 \cdot x_2 \dots x_i$. In this case, the statement is immediate from the fact that the expectation of the product of independent random variables equals the product of their expectations. □

5

Lower Bounds by Symmetrization

Prior to the development of the method of dual polynomials (Chapter 6), approximate degree lower bounds were typically proved via a technique called symmetrization. The ethos of this technique is that univariate polynomials are generally easier to understand than multivariate polynomials. Hence, symmetrization seeks to reduce the task of lower bounding the ε -approximate degree of a multivariate function f to a question about univariate polynomials. This is usually done by generically transforming an n -variate polynomial p into a univariate polynomial q without increasing its degree. One then argues that if p satisfies Condition (1.1), then q exhibits some behavior that forces it to have large degree. Since $\deg(q)$ lower bounds $\deg(p)$, one concludes that p must have large degree as well.

The transformation giving q is often built from a sequence of probability distributions D_t over $\{-1, 1\}^n$, where t ranges over a (finite or infinite) subset S of $\mathbb{R}$. One then shows that for any n -variate polynomial p , its *symmetrization* $q(t) = \mathbb{E}_{x \sim D_t}[p(x)]$ is a univariate polynomial of degree at most $\deg(p)$ over $t \in S$. Here are two classic examples.

- (t -biased symmetrization): Let S be the interval $[-1, 1]$. For $t \in S$, let μ_t be the distribution over $\{-1, 1\}$ with expected value t . Let B_t be the product distribution $\mu_t^{\otimes n}$ on $\{-1, 1\}^n$.

- (Minsky–Papert symmetrization): Let $S = [n]^*$. For each $t \in S$, define H_t to be the uniform distribution over $x \in \{-1, 1\}^n$ with Hamming weight t (i.e., exactly t entries equal to -1).

The next two lemmas show that both of these classic symmetrization techniques are indeed degree non-increasing maps from n -variate to univariate polynomials.

Lemma 5.1. For any polynomial $p: \{-1, 1\}^n \rightarrow \mathbb{R}$ of total degree at most d , the univariate function $q(t) = \mathbb{E}_{x \sim B_t}[p(x)]$ is a polynomial of degree at most d over $[-1, 1]$.

Proof. By linearity of expectation, it is without loss of generality to consider a polynomial p consisting of a single monomial, e.g., $p(x) = x_1 x_2 \dots x_d$. Then since B_t is a product distribution,

$$\mathbb{E}_{x \sim B_t}[p(x)] = \mathbb{E}_{x_1 \sim \mu_t}[x_1] \cdot \mathbb{E}_{x_2 \sim \mu_t}[x_2] \cdot \dots \cdot \mathbb{E}_{x_d \sim \mu_t}[x_d] = t^d.$$

□

Lemma 5.2. For any polynomial $p: \{-1, 1\}^n \rightarrow \mathbb{R}$ of total degree at most d , the univariate function $q(t) = \mathbb{E}_{x \sim H_t}[p(x)]$ is a polynomial of degree at most d over $[n]^*$.

Proof. Again, it is without loss of generality to assume p is a single monomial, e.g., $p(x) = x_1 x_2 \dots x_d$. Applying the variable transformation $x_i \mapsto (1 - x_i)$ does not alter the degree of p , and hence it is also without loss of generality to assume that $p(x) = (1 - x_1) \cdot (1 - x_2) \cdots (1 - x_d)$. In this case, for $x \in \{-1, 1\}^n$ we have $p(x) = 2^d$ if $x_1 = x_2 = \dots = x_d = -1$ and $p(x) = 0$ otherwise.

For any $t \in [n]^*$, the number of n -bit inputs with Hamming weight exactly t is $\binom{n}{t}$, while the number of such inputs that additionally satisfy $x_1 = x_2 = \dots = x_d = -1$ is $\binom{n-d}{t-d}$. (We are using the convention that if $t - d$ is negative, then $\binom{n-d}{t-d}$ is 0.) It follows that, for any $t \in [n]^*$,

$$\mathbb{E}_{x \sim H_t}[p(x)] = 2^d \cdot \frac{\binom{n-d}{t-d}}{\binom{n}{t}} = \left(2^d \cdot \frac{(n-d)!}{n!} \right) \cdot t(t-1)(t-2) \cdots (t-d+1)$$

or the zero polynomial. Either way, this is a polynomial in t of degree at most d . □

5.1 Symmetrization Lower Bound for OR

The tight $\Omega(\sqrt{n})$ lower bound for the approximate degree of OR_n follows easily from Theorem 5.1 and Markov's inequality (Theorem 3.2). In short, the proof applies Theorem 5.1 to any polynomial p approximating OR_n to derive a univariate polynomial $q(t)$ that is bounded on the whole interval $[-1, 1]$ but has a large “jump” in the vicinity of $t = 1$ (quantitatively, a derivative of $\Omega(n)$). Markov's inequality then implies that q has degree $\Omega(\sqrt{n})$.

Theorem 5.3. The approximate degree of OR_n is $\Omega(\sqrt{n})$.

Proof. Let p approximate OR_n to error at most $1/3$, and let q be the univariate polynomial whose existence is guaranteed by Theorem 5.1. Since the distribution B_1 assigns probability 1 to the input $\mathbf{1}_n$, we may conclude that $q(1) = p(\mathbf{1}_n) \in [2/3, 4/3]$.

Now let $t = 1 - 4/n$. Then B_t assigns probability mass at most $(1 - 2/n)^n < 1/e^2$ to $\mathbf{1}_n$, where the inequality holds by Fact 2.2. Hence,

$$q(1 - 4/n) = \mathbb{E}_{x \sim B_t}[p(x)] \leq (4/3) \cdot 1/e^2 + (-2/3) \cdot (1 - 1/e^2) \leq -1/3.$$

The Mean Value Theorem now implies that there is some $t^* \in (1 - 4/n, 1)$ such that $q'(t^*) \geq n/4$.

Finally, since it approximates OR_n , the polynomial p has magnitude at most $4/3$ over the entire Boolean hypercube $\{-1, 1\}^n$. Hence $q(t) \in [-4/3, 4/3]$ for all $t \in [-1, 1]$ as well. Applying Markov's inequality to $\frac{3}{4}q$ now implies that $\deg(p) \geq \deg(q) \geq \sqrt{3n/16}$. $\square$

An alternative proof using Minsky-Papert Symmetrization. Our proof of Theorem 5.3 derived the $\Omega(\sqrt{n})$ lower bound for OR_n from t -biased symmetrization (Theorem 5.1). The bound can also be derived from Minsky-Papert symmetrization (Theorem 5.2), but the derivation is complicated by the fact that the univariate polynomial $q(t) = \mathbb{E}_{x \sim H_t}[p(x)]$ in Theorem 5.2 is *not* a priori guaranteed to be bounded in magnitude at inputs non-integer values between 0 and n . This prevents a direct application of Markov's inequality even to affine transformations of q , as Markov's inequality applies only to polynomials bounded over the real interval $[-1, 1]$.

Our preferred way to address this issue is to invoke a result of Coppersmith and Rivlin [55], which states that any degree- d polynomial that is bounded at integer inputs in $[0, n]$ cannot be larger than $2^{O(d^2/n)}$ even at non-integer inputs within the same interval (we do not prove this result in this survey).

Theorem 5.4 (Coppersmith and Rivlin [55]). For every univariate polynomial q of degree d such that $|q(x)| \leq 1$ for all integers $x \in [0, n]$, we have $|q(x)| < a \cdot 2^{bd^2/n}$ for all real $x \in [0, n]$, where $a, b > 0$ are universal constants.

Combined with Theorem 5.2, this rules out an approximating polynomial for OR_n of degree $o(\sqrt{n})$. If such a polynomial existed, then Theorem 5.2 would turn it into a univariate polynomial q of degree $\ll \sqrt{n}$ such that $q(0) \in [2/3, 4/3]$, $q(1) \in [-4/3, -2/3]$, and $|q(t)| \leq 4/3$ for all $t \in [n]^*$. Moreover, since $d^2/n < 1$, Theorem 5.4 guarantees that there is a universal constant c such that $|q(x)| \leq c$ for all real numbers $x \in [0, n]$. That is, for degrees smaller than $\sqrt{n}$, boundedness at integer inputs implies boundness even at non-integer inputs in the same interval. Hence, letting $C = \max\{c, 4/3\}$, $Q(x) := (1/C) \cdot q((1-x)/2)$ is a univariate polynomial of degree $\ll \sqrt{n}$ such that $Q'(x) \geq \Omega(n)$ for some $x \in [1 - 2/n, 1]$ (here, we are invoking the Mean Value Theorem just as in the proof of Theorem 5.3). Moreover, $|Q(x)| \leq 1$ for all real numbers $x \in [-1, 1]$. But this contradicts Markov's inequality.

Application: Optimality of Grover's search algorithm. Recall from Section 4.1 that an approximate degree lower bound for f implies a lower bound on the quantum query complexity of f . Theorem 5.3 thus implies that the quantum query complexity of OR_n is $\Omega(\sqrt{n})$, matching the upper bound achievable via Grover's search algorithm (Section 4.2). Equivalently, to quote the most recent tagline of Scott Aaronson's blog, "quantum computers need $\sim \sqrt{n}$ queries to search a list of size n ." While this did not give the first tight quantum query lower bound proof for OR_n —it was first proved by Bennett *et al.* [26] using different techniques—the proof via approximate degree has other consequences in quantum complexity. We will see later (Section 10) that approximate degree lower bounds extend in a black-box manner from quantum query

to quantum communication lower bounds [107], [123]. In particular, the lower bound for OR_n transfers to a tight $\Omega(\sqrt{n})$ lower bound on the quantum communication complexity of the Disjointness function, DISJ , an important result (first proved by Razborov [101]) that is not known to follow from other techniques for lower bounding quantum query complexity. We cover this result in this survey (Theorem 10.13 in Section 10.4).

5.2 Arbitrary Symmetric Functions

Threshold degree lower bound via Minsky-Papert symmetrization.

Let $f(x): \{-1, 1\}^n \rightarrow \{-1, 1\}$ be a symmetric Boolean function such that $f(x) = F(|x|)$ for all $x \in \{-1, 1\}^n$. Suppose that $F(i-1) \neq F(i)$ for k values of $i \in [n]$. We saw in Section 3.1 that $\deg_{\pm}(F) \leq k$. Here we prove a matching lower bound. Let p be a polynomial that sign-represents f and let q be the Minsky-Papert symmetrization of p as per Theorem 5.2. Then q is a univariate polynomial of degree at most $\deg(p)$, and $q(i-1) \neq q(i)$ for k values of $i \in [n]$. If $k > 0$, this means that q is a non-zero polynomial with at least k roots, and hence $\deg(q) \geq k$. Since $\deg(q) \leq \deg(p)$, we conclude that the sign-representation p for f has degree at least k .

Approximate degree lower bound via Minsky-Papert symmetrization.

Recall from Section 4.3 that if f is a symmetric function that is constant on all inputs of Hamming weight between t and $n-t$ with $t \leq n/2$, then $\widetilde{\deg}(f) = O(\sqrt{nt})$. Here, we sketch a proof of a matching lower bound. For simplicity, let us focus on the symmetric t -threshold function on n bits, $\text{THR}_n^t: \{-1, 1\}^n \rightarrow \{-1, 1\}$, for which $\text{THR}_n^t(x) = -1$ if and only if $|x| \geq t$. The lower bound $\widetilde{\deg}(\text{THR}_n^t) \geq \Omega(\sqrt{tn})$ was originally proved by Paturi [96] using symmetrization, and a variant of Markov's inequality (Theorem 3.2) called the Markov-Bernstein inequality. Our sketch of Paturi's analysis in this section omits some technical details. We give a different proof of the result via the method of dual polynomials in Section 7.3.