Complexity measures on the symmetric group and beyond

Neta Dafni

Complexity measures on the symmetric group and beyond

Research Thesis

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

Neta Dafni

Submitted to the Senate
of the Technion — Israel Institute of Technology
Kislev 5782 Haifa November 2021

This research was carried out under the supervision of Prof. Yuval Filmus, in the Faculty of Computer Science.

Some results in this thesis have been published as articles by the author and research collaborators in conferences and journals during the course of the author's MSc research period, the most up-to-date versions of which being:

Neta Dafni, Yuval Filmus, Noam Lifshitz, Nathan Lindzey, and Marc Vinyals. Complexity measures on the symmetric group and beyond. In *12th Innovations in Theoretical Computer Science Conference*, volume 185 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 87, 5. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2021.

Acknowledgements

First, I would like to thank my supervisor, Yuval Filmus, for teaching me a drop of his knowledge with utmost patience, and for guiding me through my research.

I thank my family, for their love and support throughout my life, and most of all my dear mother, who during the course of my research period forced me to push through tough times.

I thank my office mates and other friends – Inbar Kaslasi, Victor Kolobov, Ohad Barta and Matan Peled, for insightful conversations and for helping me improve my thesis. I also thank Guy Raveh and Vladimir Kalnizky for their continuous encouragement.

The generous financial help of the Technion is gratefully acknowledged.

Contents

List of Figures

Abstract

The complexity of Boolean functions on the Boolean cube can be measured in several ways, such as circuit complexity and decision tree complexity. Classical results show that many of these measures are polynomially related, such as decision tree complexity, certificate complexity, degree and sensitivity.

Recently, the study of Boolean functions has been extended to functions on domains other than the Boolean cube, such as the slice, high-dimensional expanders and the Grassmann scheme. A natural question is whether the study of different complexity measures can be generalized to some of these domains.

In this work, we begin with surveying recent work that generalizes the definitions of some of the classical complexity measures to domains other than the Boolean cube, including the symmetric group, which is our main focus. We then describe results regarding the relations between the different measures, generalizing the classical results and showing that polynomial relations exist between the considered measures for many other domains.

We then focus on functions with low sensitivity, with the goal of constructing efficient circuits for them. First, we generalize a result regarding the existence of small circuits for functions with low sensitivity on the Boolean cube, constructing a circuit of size $n^{O(s(f))}$, where $s(f)$ is the sensitivity of f. Next, we generalize a result regarding the exisence of low-depth circuits for functions with low sensitivity on the Boolean cube, and construct such circuits for functions over domains satisfying certain properties, including the symmetric group. Our construction results in an unbounded fan-in circuit of depth $O(s(f)\log n)$ in the case of the symmetric group.

The last part of our work concerns intersecting families of permutations. A family of permutations is t-intersecting if any two permutations in the family agree on at least t elements. It is known that for a large enough n, the maximal size of a t-intersecting family of permutations is $(n-t)!$, and a family of that size is a t-star, that is, a family of the form $\{\pi \in S_n \mid \pi(i_1) = j_1, ..., \pi(i_t) = j_t\}$ for some distinct $i_1, ..., i_t \in [n]$ and some distinct $j_1, ..., j_t \in [n]$. We generalize this result to the case of t-setwise-intersection, where any two permutations in the family agree on the image of a set of t elements. We give an alternative proof to a known result stating that for a large enough n, the maximal size of a t-setwise-intersecting family of permutations is $t!(n-t)!$, and a family of that size is a t-setwise-star, that is, a family of the form $\{\pi \in S_n \mid \pi(S) = T\}$ for some distinct $S, T \in \binom{[n]}{t}$. Our proof is much simpler than the known proof (which proves a stronger statement) and works for smaller values of n.

Chapter 1

Introduction

The Boolean cube of dimension n is the domain $\{0,1\}^n$. A Boolean function is a function $f\colon \{0,1\}^n \to \{0,1\}$. Such a function can be the function computed by a Boolean circuit, represent a decision problem, represent a property of objects such as graphs or subsets, or many other examples.

In complexity theory, the main question is how complex some object is, and it is natural to consider Boolean functions and how hard it is to compute them. Since the complexity of a function can be measured in several ways, and there are different models of computation, these questions are formalized in different ways, and several complexity measures appear in the literature. We ultimately want to know about circuit complexity, which is interesting for numerous reasons, one of the main ones being the connection to the classical "P vs. NP" problem: if SAT is not computable by a polynomial sized circuit, then P$\neq$NP.

Research in circuit complexity is yet to yield much, but there exists interest in other complexity measures, on which we can say more. One of the classical measures is decision tree complexity, or query complexity, which considers the depth of decision trees computing a function. Its non-deterministic version is certificate complexity. Decision trees have applications in machine learning among other fields, and are also related to other computational models, such as communication complexity. Other measures include degree and sensitivity.

Since there are different standard complexity measures, the following question arises: What is the "real" way to measure the complexity of a function? It is therefore natural to look for relations between the different measures, as well as gaps between them. The question about which measure is the "correct" measure becomes simplified by existing work that shows that many of the standard complexity measures, such as decision tree complexity, certificate complexity and degree, are polynomially related. It has been conjectured for a while that sensitivity is also polynomially related to the others, but the relation was only known in one direction, until recently, when Huang [Hua19] has shown that sensitivity is indeed polynomially related to the other measures.

Recently, the study of Boolean functions has been extended to functions on domains other than the Boolean cube. Such domains include the slice (the subset of the Boolean cube consisting of all vectors with fixed Hamming weight), which is used to analyze the Erdős-Rényi $G(n,m)$ model in random graphs, and high-dimensional expanders. O'Donnell and

Wimmer [OW09] used Boolean functions on the slice to construct optimal nets for monotone functions. Barak et al. [BGH+11] used Boolean functions on the Reed–Muller code to construct and analyze the influential "short code". Khot, Minzer and Safra [KMS17] used Grassmann graphs for a reduction from the 3-Lin problem to the 2-to-2 Games problem, thus proving Khot's 2-to-2 conjecture [Kho02], a cousin of the celebrated unique games conjecture.

Can we expand the study of different complexity measures to other domains? In [DFL+21], the definitions of some of the classical complexity measures have been generalized to Boolean functions over domains including the symmetric group S_n (the set of permutations on $\{1, ..., n\}$), the perfect matching scheme (the set of perfect matchings in K_{2n}), the slice and the multislice (a multicoloured version of the slice, which generalizes the symmetric group as well). It is then shown that all complexity measures other than sensitivity remain polynomially equivalent for functions over those domains, by generalizing the classical arguments, and sensitivity is added to the mix using representation theory.

Efficient computation of functions with low sensitivity

Sensitivity is a complexity measure for Boolean functions, measuring how much the function is "sensitive" to small changes in the input, and defined as the maximal degree in the graph $(\{0,1\}^n, E)$ where (x, y) is an edge if $d(x, y) = 1$ and $f(x) \neq f(y)$.

For decades, the sensitivity conjecture for functions over the Boolean cube remained open, until it was proved by Huang [Hua19]. The sensitivity conjecture states that low sensitivity functions also have low query complexity. Prior to Huang's work, researchers tried to prove weaker statements in which query complexity is replaced by various measures of circuit complexity. Gopalan et al. [GNS+16] have used the fact that low sensitivity of a function allows local correction to prove the "ball property" for functions over the Boolean cube, showing that functions with low sensitivity can be recovered from their values on a ball of small radius (with respect to the Hamming distance): The value of f at any point outside the ball can be obtained as a majority of some of its neighbors that are closer to the ball, and so f can be computed recursively. The ball property was then used to construct small circuits for functions with low sensitivity. In addition, they constructed low depth circuits for functions with low sensitivity, where the idea is to recursively introduce "one-sided noise", that is, zero some of the ones of the input, at each step computing the value of f with high enough probability, until we reach a point of small enough Hamming distance, whose value we already know.

Since the solution to the sensitivity conjecture has been introduced, the problem of bounding circuit depth in terms of sensitivity could be solved using the relation between sensitivity and decision tree complexity, and transforming decision trees into circuits. However, the direct construction by Gopalan et al. [GNS+16] gives better bounds than the construction that uses decision trees, and so this construction remains interesting.

In [DFL+21], the ball property result was generalized to domains other than the Boolean cube, including the symmetric group. We apply this result to generalize the construction

of small size circuits and low depth circuits for functions of low sensitivity. The symmetric group analog of the operation of moving from a point on the Boolean cube to its neighbor is applying some transposition to the input, and the analog of zeroing some of the ones to create "one-sided noise" is "removing" elements from the cycle representation of the input.

Below are versions of two of our main results, restricted to the symmetric group.

Definition (sensitivity). The *sensitivity* $s(f,x)$ of a function $f\colon S_n \to \{0,1\}$ at a point $x \in S_n$ is the maximum number s of disjoint transpositions $\tau_1, ..., \tau_s$ such that $f(\tau x) \neq f(x)$ for each τ. The *sensitivity* of f is $\max_{x \in S_n} s(f,x)$.

Theorem 1. Let $f\colon S_n \to \{0,1\}$ be a function with sensitivity s. Then f is computable by a De Morgan circuit of size $n^{O(s)}$.

Theorem 2. Let $f\colon S_n \to \{0,1\}$ be a function with sensitivity s. Then f is computable by an unbounded fan-in De Morgan circuit of depth $O(s \log n)$.

We also prove similar results for functions on the perfect matching scheme.

Intersecting families of permutations

In this part of the thesis, we apply the relation between function degree and certificate complexity to the study of intersecting families of permutations.

The study of intersecting families began with Erdős, Ko and Rado [EKR61], who studied intersecting families over the domain $\binom{[n]}{k}$. In this context, an intersecting family is a subset $\mathcal{F} \subseteq \binom{[n]}{k}$ such that every $x, y \in \mathcal{F}$ intersect. Subsequent work extended their results in two different directions, the basic question being how large can an intersecting family be, and what do families of the maximum size look like. One of these directions is considering t-intersecting families – a t-intersecting family is a subset $\mathcal{F} \subseteq \binom{[n]}{k}$ such that every $x, y \in \mathcal{F}$ satisfy $|x \cap y| \geq t$. The Ahlswede-Khachatrian theorem [AK97] generalizes the Erdős-Ko-Rado theorem for t-intersecting families. The other direction is considering other domains. Frankl and Wilson [FW86] extended the theorem to vector spaces, using a spectral technique known as the *weighted Hoffman bound*. This technique was also used by Wilson [Wil84] to prove a special case of the Ahlswede-Khachatrian theorem, and by Fridgut [Fri08], who extended the study of t-intersecting families to the case where the sets in the family are not restricted to be of a fixed size, and derived stability versions of this result as well as previous ones.

These two directions were combined in the work of Ellis, Fridgut and Pilpel [EFP11], who studied t-intersecting families over the symmetric group. A t-intersecting family over the symmetric group is a subset $\mathcal{F} \subseteq S_n$ such that every $x, y \in \mathcal{F}$ agree on at least t elements. In [EFP11] it was shown that for every t, for large enough n (depending on t), the maximum size of a t-intersecting family is $(n-t)!$. Note that this size is achieved by a t-star, that is, a family of the form $\{\pi \in S_n \mid \pi(i_1) = j_1, ..., \pi(i_t) = j_t\}$ for some distinct $i_1, ..., i_t \in [n]$ and some distinct $j_1, ..., j_t \in [n]$. For $t = 1$, the size of the maximal intersecting families can be shown using a generalization of Katona's circle method. Using the spectral method,

[EFP11] showed that the characteristic function of such a family has degree 1, and applied this using the Birkhoff-von Neumann theorem to characterize those families as 1-stars. In general, they showed that the characteristic function of a maximum sized t-intersecting family has degree t. Additionally, they claimed to have characterized all such families as t-stars using a generalization of Birkhoff-von Neumann, but their argument was incorrect for $t > 1$, as pointed out in [Fil17]. The result still holds, as follows from an argument of Ellis [Ell11], who proved a much stronger result using a complex proof. An alternative, combinatorical argument was given in [DFL$^+$21], using the relation between degree and certificate complexity. That proof is simpler than the one given by [Ell11] and gives a better bound for the value of n for which the result holds.

We generalize this work to t-setwise-intersecting families. A set $\mathcal{F} \subseteq S_n$ is t-setwise-intersecting if every $x, y \in \mathcal{F}$ agree on the image of some set in $\binom{[n]}{t}$. Ellis [Ell12] showed that for every t, the maximal size of those families is $t!\,(n-t)!$ for large enough n; showed that the characteristic function of such a family has degree t; and characterized them as t-setwise-stars – families of the form $\{\pi \in S_n \mid \pi(S) = T\}$ for some $S, T \in \binom{[n]}{t}$. We give a simpler, combinatorical proof for the size of the maximal families and their characterization, with a better bound on n, using the relation between degree and certificate complexity.

Chapter 2

Background

2.1 The Boolean cube and Boolean functions

The Boolean cube of dimension $n \in \mathbb{N}$ is the domain $\{0,1\}^n$. Given $x \in \{0,1\}$, we denote the coordinates by $x_1, ..., x_n$, and for each input x in the domain, each x_i is assigned a value in $\{0,1\}$. The domain consists of all such possible assignments. A Boolean function on the cube is a function $f\colon \{0,1\}^n \to \{0,1\}$, assigning every point on the cube a Boolean value.

The study of Boolean functions meets complexity theory with the question of how complex a Boolean function is. A complexity measure of a Boolean function on the cube is a function assigning each Boolean function a value indicating how complex it is. The most classical model of computation is Boolean circuits, and thus the main interest is surrounding circuit complexity. However, other complexity measures are studied as well, and in this work we focus on some of those.

2.1.1 Complexity measures

The complexity measures we now define are standard and have been studied for years. For reference regarding the following definitions and facts, see the survey of Buhrman and deWolf [Bd02].

Degree and approximate degree

A *polynomial* over the Boolean cube is a function $P\colon \{0,1\}^n \to \mathbb{R}$ of the form

$$P(x) = \sum_{S \subseteq [n]} c_S \prod_{i \in S} x_i$$

where $c_S \in \mathbb{R}$. The *degree* of a polynomial P is the maximum size of a set S such that $c_S \neq 0$.

A Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ can be uniquely represented by a polynomial P such that $f(x) = P(x)$ for all $x \in \{0,1\}^n$. The *degree* of f, denoted $\deg f$, is the degree of P.

The ϵ-*approximate degree* of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, denoted $\widetilde{\deg}_\epsilon f$, is the minimum degree of a polynomial P such that $|f(x) - P(x)| \leq \epsilon$ for all $x \in \{0,1\}^n$. The *approximate degree* of a function f, denoted $\widetilde{\deg} f$, is $\widetilde{\deg}_{\frac{1}{3}} f$. It is easy to see that $\widetilde{\deg} f \leq \deg f$ for all f, and It is known that $\widetilde{\deg}_\epsilon f = \Theta\left(\widetilde{\deg} f\right)$ for all $\epsilon \in \left(0, \frac{1}{2}\right)$.

Certificate complexity

Let $f\colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. A *certificate* for a point $x \in \{0,1\}^n$ is a set $C \subseteq [n]$ such that $f(y) = f(x)$ whenever $y_i = x_i$ for all $i \in C$. The certificate complexity of f at x, denoted $C(f,x)$, is the minimum size of a certificate for x. The *certificate complexity* of f, denoted $C(f)$, is $\max_x C(f,x)$.

Decision tree complexity

A *decision tree* is a tree whose internal nodes are labeled by elements of $[n]$ and whose edges and leaves are labeled by elements of $\{0,1\}$. An internal node labeled with $i \in [n]$ is identified with the query "$x_i =?$". The decision tree computes the function that gives every $x \in \{0,1\}^n$ the label of the leaf identified with x in the natural way.

The *decision tree complexity* of a function $f\colon \{0,1\}^n \to \{0,1\}$, denoted $D(f)$, is the minimum depth (measured by edges) of a decision tree computing f.

The ϵ-*error randomized decision tree complexity* of f, denoted $R_\epsilon(f)$, is the minimum R such that there is a probability distribution $\mathcal{D}$ on decision trees of depth at most R such that $\Pr[T(x) = f(x)] \geq 1 - \epsilon$ for all $x \in \{0,1\}^n$. We define $R(f) = R_{\frac{1}{3}}(f)$. It is easy to see that $R(f) \leq D(f)$ for all f, and it is known that $R_\epsilon(f) = \Theta(R(f))$ for all $\epsilon \in \left(0, \frac{1}{2}\right)$.

Sensitivity and block sensitivity

The *sensitivity* of a function $f\colon \{0,1\}^n \to \{0,1\}$ at a point $x \in \{0,1\}^n$, denoted $s(f,x)$, is the number of points y at Hamming distance 1 from x such that $f(y) \neq f(x)$. In other words, $s(f,x) = |\{i \in [n] \mid f(x \oplus i) \neq f(x)\}|$, where $x \oplus i$ is the result of flipping the i'th bit. The *sensitivity* of f, denoted $s(f)$, is $\max_x s(f,x)$.

The *block sensitivity* of a function f at a point x, denoted $bs(f,x)$, is the maximum number s of disjoint subsets $B_1, ..., B_s \subseteq [n]$ such that $f(x \oplus B_i) \neq f(x)$ for all i, where $x \oplus B_i$ is the result of flipping all the bits whose indices belong to B_i. The *block sensitivity* of f is $bs(f) = \max_x bs(f,x)$.

2.1.2 Relations between the measures

It has been known for a while that all the above measures except sensitivity are polynomially related.

Recently, Huang [Hua19] showed that sensitivity is polynomially related to the others.

	D	R	C	s	bs	deg	$\widetilde{\deg}$
D		$2,3$	$2,2$	$3,6$	$2,3$	$2,3$	$4,4$
R	$1,1$		$2,2$	$3,6$	$2,3$	$2,3$	$4,4$
C	$1,1$	$1,2$		$2.22,5$	$2,2$	$1.63,3$	$2,4$
s	$1,1$	$1,1$	$1,1$		$1,1$	$1.63,2$	$2,2$
bs	$1,1$	$1,1$	$1,1$	$2,4$		$1.63,2$	$2,2$
deg	$1,1$	$1.33,2$	$2,2$	$2,2$	$2,2$		$2,2$
$\widetilde{\deg}$	$1,1$	$1,1$	$1,1$	$2,2$	$2,2$	$1,1$	

Table 2.1: Best known separations between complexity measures

Table 2.1 is taken from [ABK+21] and [BGJK21], and shows the separations between the complexity measures; for two complexity measures α and β, an entry a, b in row α and column β means that there exists a function g with $\alpha(g) \geq \beta(g)^{a-o(1)}$, and for all functions f, $\alpha(f) \leq \beta(f)^{b+o(1)}$.

2.2 Generalization to other domains

In [DFL+21], the study of different complexity measures of Boolean functions and the relations between them was extended to functions over domains other than the Boolean cube. The definitions of the complexity measures listed above were generalized to other domains, and it was shown that they are polynomially related for domains that satisfy certain properties.

2.2.1 Examples of domains

First let us define some of the domains we consider.

Definition (Symmetric group). The symmetric group S_n is the set of all permutations on $[n]$.

Definition (Perfect matching scheme). The perfect matching scheme is the set of all perfect matchings on the graph K_{2n}.

Definition (Slice). The slice, $\binom{[n]}{k}$, is the set of all vectors in $\{0,1\}^n$ of Hamming weight k.

Definition (Multislice). Let $\lambda_1, ..., \lambda_m$ be a sequence of positive integers summing to n. The multislice $M(\lambda)$ is the subset of $\{1, ..., m\}^n$ consisting of all vectors having exactly λ_i coordinates labeled i. The multislice generalizes both the slice and the symmetric group.

2.2.2 Domains as collections of sets

Let us now define the framework in which we work. Here we view each point in a domain as a *set* of elements over some universe.

Definition (Domain). Fix a universe $\mathcal{U}$ and $n \in \mathbb{N}$. A *domain* is a collection of subsets of $\mathcal{U}$ of size n.

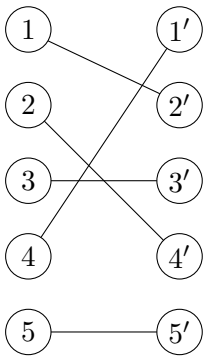

Figure 2.1: Depiction of the permutation (1 2 4) in S_5 as a perfect matching in the bipartite graph $K_{5,5}$

Boolean cube 2.2.1. A Boolean vector $x \in \{0,1\}^n$ can be identified with the set $\{(i, x_i) \mid i \in [n]\} \subseteq [n] \times \{0,1\}$. The Boolean cube can thus be thought of as the product set

$$\prod_{i=1}^{n} \{(i,0),(i,1)\}$$

over the universe $\mathcal{U} = [n] \times \{0,1\}$.

Symmetric group 2.2.2. A permutation $\pi \in S_n$ can be identified with the set $\{(i, \pi(i)) \mid i \in [n]\} \subseteq [n]^2$. Consider the complete bipartite graph $K_{n,n}$ and name the vertices $L = \{1, ..., n\}, R = \{1', ..., n'\}$. A permutation is therefore a perfect matching in the graph, matching every left vertex i to the right vertex $\pi(i)'$ (see Figure 2.1). Under this definition, S_n is simply the set of all perfect matchings in $K_{n,n}$, and is a domain over the set $\mathcal{U}$ of all edges in the graph.

2.2.3 Complexity measures for general domains

We now generalize the definitions of the different complexity measures.

Let $\mathcal{X}$ be a domain.

Degree and approximate degree

A *polynomial* is a function $P\colon \mathcal{X} \to \mathbb{R}$ of the form

$$P(x) = \sum_{S \subseteq \mathcal{U}} c_S [\![x \supseteq S]\!]$$

where $c_S \in \mathbb{R}$ and $[\![x \supseteq S]\!]$ is the Boolean function which equals 1 if $x \supseteq S$. The definition of $\widetilde{\deg_\epsilon} f$ is identical to the definition over the Boolean cube, and we define $\deg f = \widetilde{\deg_0} f$.

Certificate complexity

Let $f\colon \mathcal{X} \to \{0,1\}$ be a Boolean function. A *certificate* for a point $x \in \mathcal{X}$ is a set $c \subseteq x$ such that $f(y) = f(x)$ whenever $y \supseteq c$. The rest of the definition is identical to the definition over the Boolean cube.

Decision tree complexity

In order to handle decision trees over arbitrary domains, we first need to define what a query is.

Definition (query). A *query* Q is a subset of $\mathcal{U}$ which intersects each set in $\mathcal{X}$ in exactly one point. Each domain is associated with a set $\mathcal{Q}$ of allowed queries (this set doesn't necessarily contain all possible subsets intersecting x at exactly one point). To avoid trivialities, we assume that $\mathcal{Q}$ satisfies the following property: Every element of $\mathcal{U}$ is an element of some query in $\mathcal{Q}$. This ensures that any function can be represented as a decision tree.

Boolean cube 2.2.3. Here, the allowed queries are of the form $\{(i,0),(i,1)\}$ for some $i \in [n]$, which corresponds to the question "$x_i =?$".

Symmetric group 2.2.4. Here, the allowed queries are of the form $\{(i,j) \mid j \in [n]\}$ for some $i \in [n]$, which corresponds to the question "$\pi(i) =?$", or $\{(i,j) \mid i \in [n]\}$ for some $j \in [n]$, which corresponds to the question "$\pi^{-1}(j) =?$"

Slice 2.2.5. Here, the allowed queries are of the form $\{(i,0),(i,1)\}$ for some $i \in [n]$, similarly to the Boolean cube.

A *decision tree* is a tree whose internal nodes are labeled by elements of $\mathcal{Q}$, whose edges are labeled by elements of $\mathcal{U}$ (the children of a node labeled Q are labeled by the elements of Q), and whose leaves are labeled by elements of $\{0,1\}$. A decision tree computes a function in the natural way: Given an input x, start at the root. At each query Q, follow the edge corresponding to the unique element of $\mathcal{U}$ in $x \cap Q$. When a leaf is reached, return its label.

The rest of the definition is identical to the definition over the Boolean cube.

Sensitivity and block sensitivity

Recall that over the cube, the block sensitivity of a function $f\colon \{0,1\}^n \to \{0,1\}$ at a point $x \in \{0,1\}^n$, is the maximum number of disjoint subsets $B_1,...,B_s \subseteq [n]$ such that $f(x \oplus B_i) \neq f(x)$ for all i, where $x \oplus B_i$ is the result of flipping all the bits whose indices belong to B_i. Viewing $x = x_1,...,x_n$ as the set $\{(j,x_j) \mid j \in [n]\}$, flipping all the bits whose indices are in B_i corresponds to removing the elements in $\{(j,x_j) \mid j \in B_i\}$ from x and adding the elements in $\{(j,1-x_j) \mid j \in B_i\}$ instead. Requiring that the blocks B_i be disjoint corresponds to requiring that the sets $\{(j,x_j) \mid j \in B_i\}$ be disjoint. Therefore, we generalize this definition in the following way:

The *block sensitivity* of a function $f\colon \mathcal{X} \to \{0,1\}$ at a point $x \in \mathcal{X}$, denoted $bs(f,x)$, is the maximum number of points $y_1, ..., y_s \in \mathcal{X}$ such that $f(y_i) \neq f(x)$ for all i, and the sets $x \backslash y_i$ are disjoints.

The *block sensitivity* of f is $bs(f) = \max_x bs(f,x)$.

Over the cube, sensitivity is defined similarly to block sensitivity with the constraint that the blocks are all of size 1, or in other words, that the blocks are as small as possible without being empty. In order to generalize the definition of sensitivity to other domains, we need to generalize the notion of a block being "as small as possible".

Definition. For a domain $\mathcal{X}$, the *chunk size* $\mathfrak{c}$ is the minimal value of $|x \backslash y|$ for $x, y \in \mathcal{X}$.

For example, in the Boolean cube, for two points $x, y \in \{0,1\}^n$, $|x \backslash y|$ is the Hamming distance between x and y. The minimal distance between two different points is 1, and so $\mathfrak{c} = 1$.

In the symmetric group, the minimal number of changes required to move from one permutation to the other is 2 (corresponding to applying a transposition). Therefore, $\mathfrak{c} = 2$.

The *sensitivity* of a function $f\colon \mathcal{X} \to \{0,1\}$ is therefore defined similarly to block sensitivity with the constraint that the block size is $\mathfrak{c}$.

2.2.4 Composability

A part of the proof that all the complexity measures are related uses a reduction to the case of the Boolean cube. Given $x \in \mathcal{X}$ and $y_1, ..., y_s \in \mathcal{X}$ such that $x \backslash y_i$ are disjoint, a subset of the domain can be viewed as the cube $\{0,1\}^s$, where the vector $z_1, ..., z_s \in \{0,1\}^s$ is identified with the input

$$x \backslash \left(\bigcup_{i=1}^{s} x \backslash y_i \right) \cup \left(\bigcup_{i \in [s] : z_i = 0} x \backslash y_i \right) \cup \left(\bigcup_{i \in [s] : z_i = 1} y_i \backslash x \right) \in \mathcal{X}$$

In other words, whenever $z_i = 1$, we "replace" the block $x \backslash y_i$ with its counterpart in y_i, and otherwise we don't.

For this to work, we need to ensure that the resulting input indeed belongs to $\mathcal{X}$. This property of the domain is called *composability.*

Definition (composability). A domain $\mathcal{X}$ is *composable* if whenever $x, y_1, ..., y_s \in \mathcal{X}$ are such that $y_i \neq x$ for all i and the sets $x \backslash y_i$ are disjoint, then

$$\left(x \backslash \bigcup_{i=1}^{s} (x \backslash y_i) \right) \cup \left(\bigcup_{i=1}^{s} (y_i \backslash x) \right) \in \mathcal{X}$$

It is fairly easy to show that the Boolean cube is composable: Given an input x and a set of disjoint sets of coordinates, we can choose any subset of it and flip all the bits belonging to the corresponding coordinates, and the resulting vector is still an input in the domain.

It is less easy to show, but the symmetric group, the perfect matching scheme and the slice are composable too. [DFL+21, Lemma 2.3] gives the following simple criterion for composability:

Lemma 3. If $\mathcal{X}$, viewed as as a subset of $\{0,1\}^U$, is the intersection of $\{0,1\}^U$ and an affine subspace, then $\mathcal{X}$ is composable.

Let us demonstrate the use of the above lemma on the the symmetric group. The symmetric group is the set of solutions to the following linear system:

- For all $i,j \in [n]$: $x_{i,j} \in \{0,1\}$.
- For all $i \in [n]$: $\Sigma_{j=1}^{n} x_{i,j} = 1$.
- For all $j \in [n]$: $\Sigma_{i=1}^{n} x_{i,j} = 1$.

And thus satisfies the condition for Lemma 3.

2.2.5 Four parameters

We introduce the following four parameters of domains:

Maximum degree

The *degree* of an element $a \in \mathcal{U}$ is the number of queries mentioning it. The *maximum degree* of $\mathcal{X}$, denoted Δ, is the maximum degree of an element in $\mathcal{U}$.

In the Boolean cube, every element (i,b) is mentioned only by the query "$x_i =?$", and so $\Delta = 1$.

In the symmetric group, every element (i,j) is mentioned by two queries, "$\pi(i) =?$" and "$\pi^{-1}(j) =?$", and so $\Delta = 2$.

Conflict bound

A subset $c \subseteq \mathcal{U}$ is a *partial input* if it is a subset of some input $x \in \mathcal{X}$. Two partial inputs c_1, c_2 *conflict* if no input $x \in \mathcal{X}$ contains both. The *conflict bound*, denoted M, is the maximal value such that if c_1, c_2 are two partial inputs of size at most M and they are conflicting, then there is a query which "seperates" them, that is, a query Q such that $Q \cap c_1 \neq Q \cap c_2$.

In the Boolean cube and the symmetric group, $M = n$, where n is the size of each input in the domain (as a set). However, there are domains where $M < n$. Consider, for example, the slice $\binom{[2]}{1}$ of the vectors in $\{0,1\}^2$ of Hamming weight 1. The partial inputs $c_1 = \{(1,1)\}$ (that is, c_1 can be extended to the vector (1 0) only) and $c_2 = \{(2,1)\}$ (that is, c_1 can be extended to the vector (0 1) only) are conflicting, but no query separates them.

Sensitivity ratio and block sensitivity ratio

Given two inputs $x, y \in \mathcal{X}$, this parameter determines how many disjoint ways we can find to "bring y closer to x" in terms of the distance between x and y. The *block sensitivity ratio*, denoted β, is the largest parameter such that for every distinct $x, y \in \mathcal{X}$ there exist distinct $z_1, ..., z_s \in \mathcal{X}$, with $s \geq \beta |x \backslash y|$, such that:

- $x \backslash z_i \subsetneq x \backslash y$.
- The sets $y \backslash z_i$ are disjoint.

The *sensitivity ratio*, denoted $\beta_{\mathfrak{c}}$, is defined similarly, with the constraint that $|y \backslash z_i| = \mathfrak{c}$.

It is not hard to see that for the Boolean cube, $\beta = \beta_{\mathfrak{c}} = 1$.

For the symmetric group, $\beta = \beta_{\mathfrak{c}} = \frac{1}{3}$. The upper bound is witnessed by the following example: Take x to be the identity, and $y = (1\ 2\ 3)$. Then $|x \backslash y| = 3$ but the only way to "get closer" from y to x is to apply a transposition or y^{-1}, and any two of those changes are not disjoint. For the lower bound, assume x is the identity for simplicity, let y be any permutation, and consider some cycle in y, without loss of generality $(1\ 2\ ...\ l)$. We form z_1 by picking some i in the cycle and "shortcutting" over it, that is, replacing the cycle with $(1\ ...\ i-1\ i+1\ ...\ l)$. We can then repeat this action for any $j \notin \{i-1, i, i+1\}$ and so on, and get at least $\frac{l}{3}$ disjoint changes. Repeating this for every cycle gives $z_1, ..., z_s$ for $s \geq \frac{|x \backslash y|}{3}$.

2.2.6 Relations between the measures

In [DFL+21, Theorem 3.1] it is shown that for composable domains, the complexity measures are all polynomially related:

Theorem 4. Let $(\mathcal{X}, \mathcal{U}, n)$ be a composable domain with parameters Δ, M, β.

Every function $f \colon \mathcal{X} \to \{0, 1\}$ satisfies:

$$\sqrt{\frac{bs(f)}{6}} \leq \widetilde{\deg}(f) \leq \deg(f) \leq D(f)$$
$$bs(f) \leq C(f) \leq D(f)$$
$$bs(f) \leq 3R(f) \leq 3D(f)$$
$$D(f) \leq \beta^{-2} \max\left(\Delta, \frac{n}{M}\right) bs(f)^4$$

In particular, if $\Delta = O(1)$, $M = \Omega(n)$ and $\beta = \Omega(1)$ then all complexity measures except sensitivity are polynomially related.

In addition:

1. For the symmetric group, $s(f) \geq \sqrt{\frac{\deg(f)}{2}}$
2. For the perfect matching scheme, $s(f) \geq \sqrt{\deg(f)}$
3. For the multislice, $s(f) \geq \sqrt{\frac{\deg(f)}{2}}$

In particular, for the above domains, all the complexity measures are polynomially related.

Chapter 3

Efficient computation of functions with low sensitivity

3.1 Ball property

Gopalan et al. [GNS+16] proved the following "ball property" for functions over the Boolean cube, stating that functions with low sensitivity can be recovered from their values on a ball of small radius (with respect to the Hamming distance).

Theorem 5. Let $f\colon \{0,1\}^n \to \{0,1\}$ be a function with sensitivity $s = s(f)$, then f can be recovered from its evaluation on a ball of radius $2s+1$ around an arbitrary point.

In [DFL+21], the ball property is generalized to all composable domains. First we need to generalize the notion of *distance*: For any domain, the distance $d(x,y)$ between two inputs is $|y \backslash x|$ (note that since every input is a set of size n, this definition is symmetrical).

Recall that the chunk size $\mathfrak{c}$ is the minimal value of $|x \backslash y|$ for $x, y \in \mathcal{X}$.

Theorem 6. Let $(\mathcal{X}, \mathcal{U}, n)$ be a composable domain with chunk size $\mathfrak{c}$ and sensitivity ratio $\beta_{\mathfrak{c}}$. If $f\colon \mathcal{X} \to \{0,1\}$ has sensitivity $s = s(f)$, then f can be recovered from its evaluation on a ball of radius $r = \beta_{\mathfrak{c}}^{-1}(2s+1)$ around an arbitrary point.

Proof Suppose that we are given the values of f at all points at distance at most r from some $x \in \mathcal{X}$. Let y be an arbitrary point at distance $d \geq r$ from x. It's enough to show that $f(y)$ can be recovered from the values of f at points at distance less than d from x.

By definition of $\beta_{\mathfrak{c}}$, there exist distinct $z_1, ..., z_t \in \mathcal{X}$ with $t \geq \beta_{\mathfrak{c}} d(x,y) \geq 2s+1$ such that $d(x, z_i) \leq d$, the sets $y \backslash z_i$ are disjoint, and $|y \backslash z_i| = \mathfrak{c}$. Suppose that $f(y)$ is not the majority of $f(z_1), ..., f(z_{2s+1})$, meaning that at least $s+1$ of $f(z_1), ..., f(z_{2s+1})$ are different from $f(y)$. By definition of sensitivity, that contradicts the fact that $s(f,y) \leq s$. And so, $f(y)$ is the majority of $f(z_1), ..., f(z_{2s+1})$, thus completing the proof. ■

3.2 Small Circuits

In this section we apply the ball property to construct small circuits for low sensitivity functions on composable domains. For the rest of the section, we refer to De Morgan circuits and the size of a circuit is measured by number of gates.

Theorem 7. Let $\mathcal{X}$ be a composable domain with $|\mathcal{U}| = n^{O(1)}$. Let $f\colon \mathcal{X} \to \{0,1\}$ be a function with sensitivity s. Suppose that a membership oracle (a Boolean function taking a subset of $\mathcal{U}$ as input and specifying whether it is a member of $\mathcal{X}$ or not) can be computed by a circuit of size T. Then f can be computed by a circuit of size $n^{O\left(\beta_{\mathfrak{c}}^{-1}s\mathfrak{c}\right)}T$, where $\beta_{\mathfrak{c}}$ is the sensitivity ratio and $\mathfrak{c}$ is the chunk size.

For the rest of the section, we refer to domains with $|\mathcal{U}| = n^{O(1)}$.

One of the main ideas of the construction is computing f on a point x using majority over $2s+1$ disjoint neighbors of x (inputs whose distance from x is $\mathfrak{c}$), as done in the ball property.

As a first step, we need to be able to compute the set of neighbors of a given point:

Lemma 8. We say that $x, y \in \mathcal{X}$ are neighbors if $|x \backslash y| = \mathfrak{c}$. Given x as input, the set of all neighbors of x can be computed by a circuit of size $n^{O(\mathfrak{c})}T$.

Proof Consider the following algorithm:

- For every $A \in \binom{x}{\mathfrak{c}}$ and $B \in \binom{\mathcal{U}\backslash x}{\mathfrak{c}}$:
  - Let $z = (x_i \backslash A) \cup B$.
  - Use the membership oracle to determine whether $z \in \mathcal{X}$. If so, output z.

For every x_i, there are $n^{O(\mathfrak{c})}$ pairs of sets A, B. Therefore, the size of the circuit is $n^{O(\mathfrak{c})}T$.■

To use the idea above, we must ultimately hard-code the values of f on some set. To use this for the computation of f for an arbitrary point x, we need a way to "get from one point to the other". This is roughly done in the next lemma:

Lemma 9. Let G be the graph on $\mathcal{X}$ where (x, y) is an edge whenever x, y are neighbors. Given $x \neq y \in \mathcal{X}$ as input, a path in G of length at most n from x to y can be computed by a circuit of size $n^{O(\mathfrak{c})}T$.

Proof Consider the following algorithm:

1. $x_0 \leftarrow x$
2. For $i = 0, ..., n-1$:
   - Using Lemma 8, for every neighbor z of x_i:
     - If $y \backslash z \subsetneq y \backslash x$, then $x_{i+1} \leftarrow z$
   - If $x_{i+1} = y$, output $x_0, ..., x_{i+1}$

For $\beta_{\mathfrak{c}} > 0$, at each step there exists $z \in \mathcal{X}$ that satisfies the condition, since by the definition of $\beta_{\mathfrak{c}}$ there are at least $\beta_{\mathfrak{c}}^{-1} |y \backslash x|$ such points z.

The condition $y \backslash z \subsetneq y \backslash x$ implies that $|x_i \backslash y|$ decreases by at least 1 with each step, and therefore y is always reached after at most n steps. ■

In our construction, the neighbors the majority is taken over will all be "closer" than x to some point y. We know how to efficiently find a "fix" that gets us closer from a starting point x to an endpoint y, and furthermore, get all the different fixes that achieve that. But in order to use the sensitivity property on a point x, the neighbors $z_1, ..., z_{2s+1}$ used for the majority must be disjoint, in that the sets $x \backslash z_i$ are disjoint. The definition of $\beta_{\mathfrak{c}}$ provides that there is a set of at least $\beta_{\mathfrak{c}} r$ such fixes, where $r = |x \backslash y|$, that are disjoint. The next lemma shows that, losing a factor of $\mathfrak{c}$, such a set can be computed greedily, by repeatedly applying the search for a single fix, with the constraint that it be disjoint from the previous ones:

Lemma 10. Let $x, y \in \mathcal{X}$ and let $r = |x \backslash y|$. Then a set $z_1, ..., z_{\beta_{\mathfrak{c}} r \mathfrak{c}^{-1}} \in \mathcal{X}$ such that $y \backslash z_i \subsetneq y \backslash x$, the sets $x \backslash z_i$ are disjoint, and $|x \backslash z_i| = \mathfrak{c}$ for each i, can be computed by a circuit of size $n^{O(\mathfrak{c})} T$.

Proof Consider the following algorithm:

1. $D_0 \leftarrow \emptyset$
2. For $i = 1, ..., \beta_{\mathfrak{c}} r \mathfrak{c}^{-1}$:
   - Using Lemma 8, for every neighbor z of x:
     - If $(z \backslash x) \cap \bigcup_{j=0}^{i-1} D_j = \emptyset$ and $y \backslash z \subsetneq y \backslash x$, then $z_i \leftarrow z$ and $D_i = z \backslash x$
3. Output $z_1, ..., z_{\beta_{\mathfrak{c}} r \mathfrak{c}^{-1}}$

A z satisfying the conditions always exists: There are $z'_1, ..., z'_{\beta_{\mathfrak{c}} r} \in \mathcal{X}$ such that $y \backslash z'_k \subsetneq y \backslash x$, the sets $x \backslash z'_k$ are disjoint and $|x \backslash z'_k| = \mathfrak{c}$. For $i \leq \beta_{\mathfrak{c}} r \mathfrak{c}^{-1}$, we have $\left| \bigcup_{j=0}^{i-1} D_j \right| = (i-1)\, \mathfrak{c} < \beta_{\mathfrak{c}} r$, and therefore the union hits at most $\beta_{\mathfrak{c}} r - 1$ of the z'_k, meaning one of the z'_k satisfies the requirement. ■

The construction of the circuit in Theorem 7 is done as follows: Given a point x, we find a path from x to an arbitrary point y, where each point is of distance $\mathfrak{c}$ from the previous one. We "go from y to x" and at each step, compute the values of f on a ball around the point we're in, of a radius $r = \beta_{\mathfrak{c}}^{-1} (2s+1)\, \mathfrak{c}$, using the values on the ball of the same radius around the previous point. The way we do that is the following: We use the values of f on a ball of radius r to compute its values on a ball around the same center of radius $r + \mathfrak{c}$, and by the triangle inequality, it contains the ball around the next point.

Lemma 11. Given $x \in \mathcal{X}$ and the values of f on $\mathcal{B}(x, r)$ (the Hamming ball of radius r around the point x), the values of f on $\mathcal{B}(x, r + \mathfrak{c})$ can be computed by a circuit of size $n^{O(r)} T$.

Proof The set of all the points in $\mathcal{B}(x, r+\mathfrak{c})$ can be obtained in the following way: Start with a list of the points in $\mathcal{B}(x, r)$. Find the set of all neighbors of each point using Lemma 8, and add each one that isn't already there to the list. Repeat $\mathfrak{c}$ times. At step i, the operation requires a circuit of size $n^{O(r+i)}T \leq n^{O(r+\mathfrak{c})}T = n^{O(r)}T$. Then, for $i = 1, ..., \mathfrak{c}$, we can compute the values of f on $\mathcal{B}(x, r+i)$, using the majority of its values on $\beta_{\mathfrak{c}} r \mathfrak{c}^{-1} = 2s+1$ points in $\mathcal{B}(x, r+i-1)$ obtained by the algorithm in Lemma 10. ■

We now hold all the tools required for the proof of Theorem 7.

Proof of Theorem 7 Consider the following algorithm, taking $x \in \mathcal{X}$ as input:

1. Fix an arbitrary point x_0 and hard-code the values of f on $\mathcal{B}(x, r)$ for $r = \beta_{\mathfrak{c}}^{-1}(2s+1)\mathfrak{c}$.
2. Compute a path $x_0, x_1, ..., x_d = x$ for $d \leq n$ using Lemma 9.
3. For $0 \leq i \leq d-1$, compute the values of f on $\mathcal{B}(x_i, r+\mathfrak{c})$ using its values on $\mathcal{B}(x_i, r)$ and Lemma 11..
4. Output $f(x)$

For $1 \leq i \leq d$, it holds that $d(x_{i-1}, x_i) = \mathfrak{c}$, and therefore $\mathcal{B}(x_i, r) \subseteq \mathcal{B}(x_{i-1}, r+\mathfrak{c})$. Each iteration takes $n^{O(r)}$ time, there are at most n of them, and so we end up with the required circuit size. ■

3.3 Low depth circuits

We introduce the construction of low-depth unbounded fan-in circuits for Boolean functions with low sensitivity on domains with certain properties, including the symmetric group and the perfect matching scheme. One approach to the task of bounding circuit depth in terms of sensitivity would be to go through decision trees: a decision tree can be transformed into a circuit in the natural way, giving a circuit of depth $O(D(f))$. From Theorem 4 it follows that, for the symmetric group for example, $D(f) = O\left(bs(f)^4\right) = O(\deg(f))^8 = O\left(s(f)^{16}\right)$, and so we get a circuit of depth $O\left(s(f)^{16}\right)$. In comparison, the construction in the following theorem gives an $O(s(f)\log n)$-depth circuit.

3.3.1 Shortcuttability

The idea is to compute $f(x)$ recursively using f's values on points "closer" to a fixed set $c \in \mathcal{X}$, like in the preceding section. In the case of the symmetric group, for example, one might consider the distance of a permutation x from id. A way to get closer from x to id is to "shortcut" over some non-fixed point i of x; that is, to apply the transposition $(i\ x(i))$ to x. This motivates us to define the following:

Definition. A domain $\mathcal{X}$ is called *shortcuttable* if there exist $c \in \mathcal{X}$ and an operator $\curvearrowright\colon \mathcal{X} \times c \to \mathcal{X}$ (which we call "shortcut" and write $x \curvearrowright i$ for $x \in \mathcal{X}, i \in c$) such that:

1. For every $x \in \mathcal{X}$ and $i, j \in c$, $x \curvearrowright i \curvearrowright j = x \curvearrowright j \curvearrowright i$. Note that this allows us to define $x \curvearrowright S$ for $S \subseteq c$ in the natural way.

2. Shortcutting over a set of elements that are not in x to begin with, results in a close "enough" set to c: If $S \subseteq c \backslash x$, then $d(x \curvearrowright S, c) \leq d(x, c) - |S|$ (recall that the distance $d(x, y)$ is defined to be $|y \backslash x|$).

3. For $S \subseteq c$ and $i \in S$, $x \curvearrowright i$ is "closer" (or equally close) to $x \curvearrowright S$ than x: $(x \curvearrowright S) \setminus (x \curvearrowright i) \subseteq (x \curvearrowright S) \backslash x$, and equality occurs iff $i \in x$. Note that for $S = c$, we get that shortcutting over i brings x closer to c, since $x \curvearrowright c = c$, as follows from property 2. If $i \notin x$, we call $x \curvearrowright i$ a *proper* shortcut.

4. The change in x due to a proper shortcut is "minimal": For $x \in \mathcal{X}, i \in c \backslash x$, $d(x \curvearrowright i, x) = \mathfrak{c}$.

Next, we define a parameter similar to $\beta_{\mathfrak{c}}$, but related to the shortcut operator. When we perform a number of proper shortcuts on a set x, we want to ensure that enough of the changes are disjoint (in a sense compatible with the definition of sensitivity).

Definition. We define the *shortcut sensitivity ratio* γ to be the largest parameter such that for every $x \in \mathcal{X}$ and $S \subseteq c \backslash x$, there exist distinct $i_1, .., i_m \in S$, with $m \geq \gamma |S|$, such that the sets $x \setminus (x \curvearrowright i_j)$ are disjoint.

We are now ready to present the theorem:

Theorem 12. Let $\mathcal{X}$ be a shortcuttable domain with reference to $c \in \mathcal{X}$ and let $f \colon \mathcal{X} \to \{0, 1\}$ be a function with sensitivity s. Assume that there exists an unbounded fan-in logarithmic-depth circuit that takes as input $x \in \mathcal{X}$ and $S \subseteq c$ (in the form of flags specifying for each $i \in c$ whether $i \in S$), and computes $x \curvearrowright S$. Then f is computable by an unbounded fan-in circuit of depth $O(s \log n)$.

3.3.2 Domain Examples

We now give examples for a few shortcuttable domains.

Boolean cube and product domains

Consider the Boolean cube, and let $c = 0^n$. The shortcut operation corresponds to writing "0" in some coordinate: for $i \in [n]$, $x \curvearrowright (i, 0) = (x \setminus (i, 1)) \cup (i, 0)$. It's easy to check that all the properties hold:

1. Shortcutting over any two coordinates, at any order, corresponds to zeroing both coordinates.

2. In this case, equality holds: Shortcutting over any number s of coordinates that are not already 0 reduces the Hamming weight by exactly s.

3. Here, $(x \curvearrowright S) \setminus x = \{j \in [n] \mid (j,0) \in S \wedge (j,1) \in x\}$, and given $i = (i',0) \in S$ for some $i' \in [n]$, $(x \curvearrowright S) \setminus (x \curvearrowright i) = \{j \in [n] \mid (j,0) \in S \setminus \{i\} \wedge (j,1) \in x\}$. Obviously $(x \curvearrowright S) \setminus (x \curvearrowright i) \subseteq (x \curvearrowright S) \setminus x$, and equality occurs iff there does not exist $j \in [n]$ such that $(j,0) = i$ and $(j,1) \in x$, that is, if $x_{i'} = 0$, or in other words, $i \in x$.

4. The change in x due to a proper shortcut is flipping one coordinate. The distance of the resulting input from x is 1, which equals $\mathfrak{c}$.

Given $x \in \mathcal{X}$ and $S \subseteq [n]$, it's easy to construct a constant depth unbounded fan-in circuit that computes $x \curvearrowright \{(i,0) \mid i \in S\}$: It suffices to write "0" in all the specified coordinates.

The above can be easily modified to work for any product domain.

Perfect matching scheme

Let $\mathcal{X}$ be the perfect matching scheme on $2n$ points. Denote the elements of $[2n]$ by $a_1, ..., a_n, b_1, ..., b_n$, and let c be the matching that maps each a_i to b_i. For a matching x, denote by $x(a)$ the element matched to a by x.

For $x \in \mathcal{X}$, shortcutting over (a_i, b_i) is the action of matching a_i, b_i to each other and $x(a_i), x(b_i)$ to each other.

The shortcutting satisfies property 1: Shortcutting over a set S, at any order, can be viewed in the following way: Let $G_{x,S}$ be the graph containing x's edges and the edges (a_i, b_i) for each $i \in S$. This graph is a collection of cycles and paths (with 2-cycles whenever $x(a_i) = b_i$ for $i \in S$). Then, for every cycle, every edge in the cycle that is not of the form (a_i, b_i) is removed. For every path, every edge in the path that is not of the form (a_i, b_i) is removed, and the two ends of the path are connected. See Figure 3.1 for example. As for property 3, if $i \in x$ then $x \curvearrowright i = x$ and so equality holds. Suppose $i \notin x$. We have that $(x \curvearrowright S) \setminus x$ consists of $S \setminus x$ and all the edges connecting the ends of paths in $G_{x,S}$. If i is a part of a path or a k-cycle for $k > 4$ in $G_{x,S}$, then $(x \curvearrowright S) \setminus (x \curvearrowright i)$ consists of $S \setminus (x \cup \{i\})$ and the edges connecting the ends of paths in $G_{x,S}$. Otherwise, i is a part of a 4-cycle in $G_{x,S}$, and $(x \curvearrowright S) \setminus (x \curvearrowright i)$ contains the same as before except the edge opposite to i in the 4-cycle (as it is in $x \curvearrowright i$ as well). In both cases, $(x \curvearrowright S) \setminus (x \curvearrowright i) \subsetneq (x \curvearrowright S) \setminus x$.

As for property 4, $\mathfrak{c} = 2$ for the perfect matching scheme, and indeed, $d(x \curvearrowright i, x) = 2$ for i such that $x(a_i) \neq b_i$. As for property 2, see the following claim:

Claim 3.3.1. *Let $x \in \mathcal{X}$ and $S \subseteq [n]$ such that $x(a_i) \neq b_i$ for every $i \in [n]$. Then $d(x \curvearrowright S, c) \leq d(x, c) - |S|$.*

Proof. Note that $d(x,c) = |i \in [n] \mid x(a_i) \neq b_i|$. A k-path in $G_{x,S}$ corresponds to shortcutting over $\frac{k}{2} - 1$ indices, and decreases the distance from c either by $\frac{k}{2} - 1$, if the ends of the path are not of the form (a_i, b_i), or by $\frac{k}{2}$, if they are. This is since every inner vertex of the path, w.l.o.g. of the form a_i, is not matched to b_i by x but gets matched to b_i after the shortcut, and the same goes to the path ends in the second case, while a match (a_i, b_i) in x never gets unmatched by a shortcut. Similarly, a k-cycle in $G_{x,S}$ corresponds to shortcutting over $\frac{k}{2}$ indices, and decreases the distance from $\mathfrak{c}$ by $\frac{k}{2}$. Summing over all the

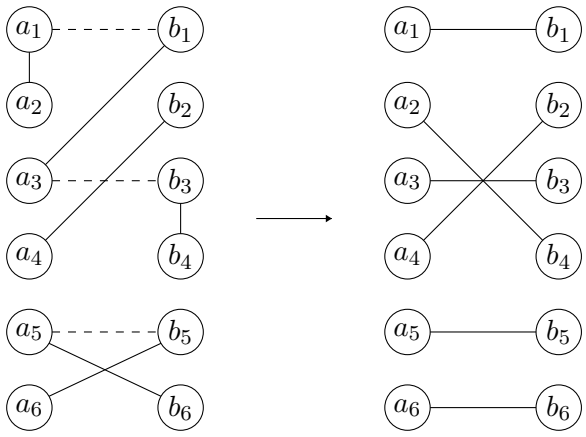

Figure 3.1: Example of a shortcut in the perfect matching scheme. The matching includes (a_1,a_2),(a_3,b_1),(a_4,b_2),(a_5,b_6) and (a_6,b_5). The shortcut is done over 1, 3 and 5.

paths and cycles, we get that overall the shortcutting over S decreases the distance from c by at least $|S|$. ■

Finally, the circuit specified in the theorem can be constructed as shown in the following claim:

Claim 3.3.2. *There exists a logarithmic-depth formula that takes as input a matching x and flags $X(i)$ specifying for each $i \in [n]$ whether (a_i, b_i) is being shortcut over, and outputs the resulting matching y.*

Proof Let $S = \{i \in [n] \mid X(i) \text{ holds}\}$. For $i, j, l \in [n]$ and $c, d \in \{a, b\}$, we construct a formula $Reach_l(c_i, d_j)$ that holds iff there is a path of length l between the vertices c_i, d_j in $G_{x,S}$, in the following way:

$Reach_0(c_i, d_j) :=$ "$c_i = d_j$".

$Reach_1(c_i, d_j) :=$ "$x(c_i) = d_j$" $\vee$ ("$i = j$" $\wedge X(i)$).

Those can obviously be computed in constant depth.

For $l > 1$: $Reach_l(c_i, d_j) = \bigvee_{k \in [n], c \in \{a,b\}} \left(Reach_{\lfloor \frac{l}{2} \rfloor}(c_i, c_k) \wedge Reach_{\lceil \frac{l}{2} \rceil}(c_k, d_j) \right)$

Now, let $Reach(c_i, d_j) = \bigvee_{l \in [n]} Reach_l(c_i, d_j)$. Computing $Reach(c_i, d_j)$ recursively results in a formula for $Reach(c_i, d_j)$ of logarithmic depth.

To complete the formula, we have $y(c_i) = d_j$ iff one of the following holds:

- $i = j$ and $x(c_i) = d_i$.
- $i = j$, $c \neq d$, $x(c_i) \neq d_j$, and $Reach(c_i, d_j)$ holds.
- $i \neq j$, $x(c_i) = d_j$ and neither of i, j is being shortcut over.
- $i \neq j$, $x(c_i) \neq d_j$, neither of i, j is being shortcut over, and $Reach(c_i, d_j)$ holds. ■

The above is described by the formula:

$$\begin{aligned}
&(\text{"}i = j\text{"} \wedge \text{"}x\,(c_i) = d_i\text{"}) \\
\vee\ &(\text{"}i = j\text{"} \wedge \text{"}c_i \neq d_j\text{"} \wedge \text{"}x\,(c_i) \neq d_j\text{"} \wedge \mathit{Reach}\,(c_i, d_j)) \\
\vee\ &(\text{"}i \neq j\text{"} \wedge \text{"}x\,(c_i) = d_j\text{"} \wedge \neg X\,(i) \wedge \neg X\,(j)) \\
\vee\ &(\text{"}i \neq j\text{"} \wedge \text{"}x\,(c_i) \neq d_j\text{"} \wedge \neg X\,(i) \wedge \neg X\,(j) \wedge \mathit{Reach}\,(c_i, d_j))
\end{aligned}$$

Symmetric group

Let $c = id$. Here, we consider the cycle decomposition of a permutation, and shortcutting means "removing" some element from its cycle, if said element is a non-fixed point, and otherwise doing nothing. Formally, given $i \in [n]$, $x \curvearrowright (i, i) = (i\ x\,(i))\,x$. Note that if $x\,(i) = i$, $x \curvearrowright i$ simply equals x. For example, consider the permutation (1 2 3 4) (5 6 7) (8 9 10 11). A shortcut over $1, 3, 4, 5, 6, 7, 10$ will result in the permutation (8 9 11). Note that even though we did not shortcut over 2, 2 was also removed from its cycle. This is an example of a case where the inequality in property 2 is proper.

The symmetric group can be viewed as a special case of the perfect matching scheme: Given $x \in S_n$, we identify x with the matching M_x over K_{2n} where $M_x\,(a_i) = b_j$ if $x\,(i) = j$. In this setting, a shortcut over an index i in S_n is simply the matching shortcut over i in the perfect matching scheme: Assume $x\,(i) = j$ and $x\,(k) = i$. In the perfect matching scheme we have $M_x\,(a_i) = b_j, M_x\,(a_k) = b_i$. The shortcutting over i in S_n results in $x\,(i) = i$ and $x\,(k) = j$, which translates to $M_x\,(a_i) = b_i$ and $M_x\,(a_k) = b_j$, which is indeed the result of the shortcutting over i in the perfect matching scheme. Thus, we conclude that S_n is shortcuttable.

Our next task is to construct a circuit that takes as input $x \in S_n$ and flags $X\,(i)$ and outputs the corresponding permutation y. We described a circuit that takes as input a matching M_x and flags $X\,(i)$ and outputs the resulting matching M_y. Note that the input x can easily transformed to an input M_x to the existing circuit, in the following way: For "$M_x\,(a_i) = b_j$" we take "$x\,(i) = j$", and "$M_x\,(c_i) = c_j$" is 0 for every i, j and $c \in \{a, b\}$. It's left to describe the formula for "$y\,(i) = j$", taking M_y as input, which is simply "$M_y\,(a_i) = b_j$".

3.3.3 Overview of the construction

We first describe a probabilistic algorithm that computes f with constant error. The idea is to compute f recursively in the following way: Given $x \in \mathcal{X}$, the output is taken to be the majority of f on sets closer to c. For sets with a low enough distance from c, the algorithm uses a hard-coded look-up table. Each of the inputs the majority is taken over is obtained from x by shortcutting over a random set of elements, whose size is (a) small enough so that the low sensitivity ensures that the resulting set has the same f-value as x with high probability; and (b) large enough so that not too many recursion steps are needed. The number of input sets chosen is taken to be a constant that ensures the conservation of the constant error.

The algorithm is translated into a circuit where every step of the recursion is of constant depth (as it's a constant fan-in majority gate), and so is the lookup table. The recursion depth is shown to be logarithmic, and so we get a logarithmic-depth, probabilistic circuit, that computes f with constant error.

The probability of success is then boosted by taking majority over a polynomial number of repetitions, and an expectation argument is used to show that there exists a suitable circuit with zero error. Since majority can be implemented with depth logarithmic in the fan-in, the boosting only adds a logarithmic-depth compound to the circuit.

3.3.4 Noise stability

The following results regard the probability that $f(x) \neq f(y)$ for a set y that is obtained from x by shortcutting over a set of elements of a size t. We first consider $t = 1$ and use the sensitivity property, then use a union bound to bound the probability for larger t in terms of s, t and the size of the set $|D|$ from which the elements are allowed to be chosen. Finally, for $t \sim \frac{d}{s}$ we get a constant error.

Lemma 13. Let $x \in \mathcal{X}$ and let $D \subseteq c$. Let i be chosen uniformly out of D. Then $Pr[f(x) \neq f(x \curvearrowright i)] \leq \frac{\gamma^{-1}s}{|D|}$, where γ is the shortcut sensitivity ratio.

Proof Let $S = \{i_1, ..., i_k\}$ be the set of elements of D such that $f(x) \neq f(x \curvearrowright i)$. Obviously, for every such i, $x \neq x \curvearrowright i$. By definition of γ, there exist $i_{j_1}, ..., i_{j_{\gamma k}}$ such that $(x \curvearrowright i_{j_1}) \setminus x, ..., \left(x \curvearrowright i_{j_{\gamma k}}\right) \setminus x$ are disjoint, and due to property 3 we have $\gamma k \leq s$, or, $k \leq \gamma^{-1}s$. There are $|D|$ elements to choose from, and so the probability of hitting one of $i_1, ..., i_k$ is at most $\frac{\gamma^{-1}s}{|D|}$. ■

Corollary 14. Let $i_1, ..., i_t$ be chosen uniformly and independently out of D. Then $Pr[f(x) \neq f(x \curvearrowright \{i_1$ $\frac{\gamma^{-1}st}{|D|-t}$.

Proof We can view the process of choosing $i_1, ..., i_t$ and applying the shortcuts in the following way: Initialize $y_0 = x$. At step $1 \leq j \leq t$:

- Choose $i_j \in D \setminus \{i_1, ..., i_{j-1}\}$ uniformly.
- Let $y_j = y_{j-1} \curvearrowright i_j$.

For each j, $|D \setminus \{i_1, ..., i_{j-1}\}| \geq |D| - t$. At each step, then, the probability that f changes is at most $\frac{\gamma^{-1}s}{|D|-t}$. The lemma follows by a union bound. ■

Corollary 15. If $t \leq \frac{|D|}{10\gamma^{-1}s+1}$ then $Pr[f(x) \neq f(y)] \leq \frac{1}{10}$.

Proof $\frac{\gamma^{-1}st}{|D|-t} \leq \frac{\frac{\gamma^{-1}s|D|}{10\gamma^{-1}s+1}}{|D|-\frac{|D|}{10\gamma^{-1}s+1}} = \frac{\gamma^{-1}s}{10\gamma^{-1}s+1-1} = \frac{1}{10}$. ■

3.3.5 Majority Tree

The above property is used to output $f(x)$ using the majority of f's values on sets obtained from x as above, for $t = \left\lfloor \frac{|D|}{10\gamma^{-1}s+1} \right\rfloor$, recursively, where the set D can roughly be thought

of as $c\backslash x$. We describe the algorithm as a tree and perform the analysis in terms of the root's properties rather than recursively, for better compatibility with the transformation to circuits. We start by considering the elements of $c\backslash x$. With every branching of the tree, we shortcut over some of them, and at every point keep track of the set of all elements that were shortcut over so far.

Let $x \in \mathcal{X}$ and let $D = c\backslash x$. Let T be a tree with the following properties:

1. Every vertex v is associated with a set $D_v \subseteq [|D|]$. For the root v_0, $D_{v_0} = \emptyset$. The mapping $v \mapsto D_v$ will be denoted D_T.

2. Every inner vertex v has α children, where α is a constant to be defined later, and for each child u of v, $D_u \supsetneq D_v$.

We associate every vertex v with a set $x_v \in \mathcal{X}$ in the following way: Write $D = \left\{i_1, ..., i_{|D|}\right\}$ and $D_v = \left\{j_1, ..., j_{|D_v|}\right\}$. Then $x_v = x \curvearrowright \left\{i_{j_1}, ..., i_{j_{|D_v|}}\right\}$. Note that $x_{v_0} = x$.

3. If u is a child of v, then $|D_u\backslash D_v| = \lfloor \frac{|D|-|D_v|}{10\gamma^{-1}s+1} \rfloor$.

4. Every leaf is of the same depth, which is the minimum depth required so that for every leaf v, $|D| - |D_v| \leq 100\gamma^{-1}s$. Note that this depth does not depend on x, but only on $|D|$.

The choice of size in property 4 is so that the set meets the requirement of Corollary 15, and due to the property 2, determines the tree's depth. The next claim shows that the resulting depth is low enough to meet our needs regarding the circuit's depth.

Claim 3.3.3. *T's depth is $O(s \log n)$.*

Proof Let $v_0, ..., v_l$ be some path from the root to a leaf v such that $|D| - \left|D_{v_{l-1}}\right| > 100\gamma^{-1}s$. Such a path exists due to the minimality condition on the depth. For $0 \leq j < l$,

$$|D| - \left|D_{v_{j+1}}\right| = |D| - \left|D_{v_j}\right| - \left|D_{v_{j+1}}\backslash D_{v_j}\right| = |D| - \left|D_{v_j}\right| - \left\lfloor \frac{|D| - \left|D_{v_j}\right|}{10\gamma^{-1}s+1} \right\rfloor < |D| - \left|D_{v_j}\right| - \frac{|D| - \left|D_{v_j}\right|}{10\gamma^{-1}s+1} + 1.$$

For $|D| - \left|D_{v_j}\right| > 100\gamma^{-1}s$, we have $\frac{1}{10\gamma^{-1}s+1}\left(|D| - \left|D_{v_j}\right|\right) > 3$ and so $1 < \frac{1}{2(10\gamma^{-1}s+1)}\left(|D| - \left|D_{v_j}\right|\right)$, meaning

$$|D| - \left|D_{v_j}\right| - \frac{|D| - \left|D_{v_j}\right|}{10\gamma^{-1}s+1} + 1 < |D| - \left|D_{v_j}\right| - \frac{|D| - \left|D_{v_j}\right|}{2\left(10\gamma^{-1}s+1\right)} < |D| - \left|D_{v_j}\right| - \frac{|D| - \left|D_{v_j}\right|}{100\gamma^{-1}s}.$$

Thus, $|D| - \left|D_{v_j}\right| < \left(1 - \frac{1}{100\gamma^{-1}s}\right)^j n$, and so $\frac{1}{n} < \left(1 - \frac{1}{100\gamma^{-1}s}\right)^{l-1}$, or, $l-1 < \frac{\log n}{-\log\left(1-\frac{1}{100\gamma^{-1}s}\right)} < 100\gamma^{-1}s \log n$, where the last inequality is due to the fact that $\log(1-\epsilon) < -\epsilon$ for every $\epsilon > 0$. ■

To complete the compatibility with Corollary 15, we introduce the following distribution on D_T: Recall that $D_{v_0} = \emptyset$, where v_0 is the root. Recursively, if u is a child of v, $D_u \backslash D_v$ is distributed uniformly (under the size constraint) over $[|D|] \backslash D_v$.

We now associate a Boolean value $A(v)$ to each vertex v, recursively, in the following way: If v is a leaf, $A(v) = f(x_v)$. For an inner vertex, $A(v)$ is the majority of $A(u)$ over all the children u of v. Next, we analyze the probability that $A(v) = f(x_v)$ for each v. Write $f(v)$ in short for $f(x_v)$. Note that $f(v_0) = f(x)$, where v_0 is the root.

The constant probability of the inequality $f(u) \neq f(v)$ for each child u of v provided by Corollary 15, combined with the constant number of children, results in an overall constant error in the computation of $f(v)$, as shown in the next claim:

Claim 3.3.4. *There exists a constant α such that for every v, $\Pr[A(v) \neq f(v)] < \frac{1}{20}$.*

Proof By induction on the distance from the furthest leaf:

If v is a leaf, $A(v) = f(v)$ always.

Let $u_1, ..., u_c$ be v's children. For each u_i, $\Pr[A(u_i) \neq f(u_i)] < \frac{1}{20}$ by the induction hypothesis. The set $D_{u_i} \backslash D_v$ is chosen as in Corollary 14 for $t = \left\lfloor \frac{|D|-|D_v|}{30s+1} \right\rfloor$, and so $\Pr[f(u_i) \neq f(v)] < \frac{1}{10}$ by Corollary 15. Overall, $\Pr[A(u_i) \neq f(v)] < \frac{1}{5}$.

Now, we can choose α such that $\Pr[A(v) \neq f(v)] = \Pr\left[\mathrm{maj}_{i \in [\alpha]}(A(u_i) \neq f(v))\right] < \frac{1}{20}$. ■

The next claim characterizes the set of sets associated with the leaves in terms of distance from c:

Claim 3.3.5. *For every leaf v, $x_v \in \mathcal{B}(id, 100\gamma^{-1}s)$.*

Proof Follows from the fact that for every v, $d(x_v, c) \leq |D| - |D_v|$ due to property 2. ■

3.3.6 Conversion to a deterministic algorithm

Let $T^{(1)}, ..., T^{(k)}$ be trees each distributed as above, and let $v_0^{(i)}$ be the root of $T^{(i)}$. Let $A_0(x) = \mathrm{maj}_{i \in [k]} A\left(v_0^{(i)}\right)$. Let $\mathcal{X} = \left|\left\{i \in [k] \mid A\left(v_0^{(i)}\right) \neq f\left(v_0^{(i)}\right)\right\}\right|$. By Chernoff bound,

$$\Pr[A_0(x) \neq f(x)] = \Pr\left[\mathcal{X} > \frac{k}{2}\right] < \left(\frac{e^9}{10^{10}}\right)^{\frac{k}{20}} < 2^{-\frac{k}{20}}.$$

Taking $k = 20n^2$, we get that majority over a polynomial number of trees gives an error of at most 2^{-n^2} for every permutation x of distance $\Delta = |D|$ from id. In other words, the expected number of inputs on which the circuit errs is less than 1. Therefore there is a choice of randomness for which the circuit never errs.

Note that the construction is not non-uniform.

3.3.7 Conversion to a circuit

The properties of the structure we described are tailored to a fixed size of $c\backslash x$. We thus construct a different circuit for each $\Delta = |c\backslash x|$, and the final circuit will compute Δ and redirect to the correct circuit.

Fix Δ. Consider a circuit that takes a permutation x with $|c\backslash x| = \Delta$, and is composed of a logarithmic-depth tree of majority gates of depth $d = O(s \log n)$, each majority gate computing the majority of α inputs (and is thus of constant depth), except for the last one having $20n^2$ inputs (which makes the majority gate of logarithmic depth). Each leaf v corresponds to a fixed set $D_v \subseteq [\Delta]$, which the circuit translates into a set of elements to be shortcut over when starting with the input x, in the following way: Fix some order on c. For $i \in c, j \in [\Delta]$, define $P(i,j)$ to hold when $i \notin x$ and $|\{k \in c\backslash x\,|, k \leq i\}| = j$. $P(i,j)$ can be computed for every i, j in logarithmic depth by counting. Now, for $i \in c$, define $X(i)$ to hold if i is being shortcut over. Note that $X(i) = \bigvee_{j \in [D]} (\text{"}j \in D_v\text{"} \wedge P(i,j))$, and so $X(i)$ can be computed for every i.

3.3.8 The Resulting Circuit

Proof of Theorem 12 The task is achieved by a circuit consisting of the following compounds:

- A logarithmic-depth circuit that computes $\Delta = |c\backslash x|$. ■
- For each $\Delta \leq n$:
  - The circuit that, for each leaf v, transforms the set $D_v \subseteq [\Delta]$ into flags $X(i)$ specifying whether i is being shortcut over.
  - A circuit that, for each leaf v, takes the flags $X(i)$ and computes the resulting set. This can be done by assumption.
  - A lookup table containing f's values on $B(id, 100\gamma^{-1}s)$: On input y, the output is $\bigvee_{\pi \in \mathcal{B}(id, 100\gamma^{-1}s): f(\pi)=1} \bigwedge_{i \in \mathcal{U}} \text{"}i \in y \iff i \in \pi\text{"}$.
  - The majority gates.

Chapter 4

Intersecting families of permutations

4.1 Background

A *t-intersecting family* over a domain $\mathcal{X}$ for $t \geq 1$ is a subset $\mathcal{F} \subseteq \mathcal{X}$ such that any two sets $S_1, S_2 \in \mathcal{F}$ satisfy $|S_1 \cap S_2| \geq t$.

For example, over the Boolean cube, $\mathcal{F}$ is t-intersecting if any two vectors $x, y \in \mathcal{F}$ agree on at least t coordinates. Over the symmetric group, $\mathcal{F}$ is t-intersecting if any two permutations $x, y \in \mathcal{F}$ agree on the images of at least t elements of $[n]$. First, we need the following definition:

Definition. A *t-star* in a domain $\mathcal{X}$ over a universe $\mathcal{U}$ is a subset of $\mathcal{X}$ of the form $\{T \in \mathcal{X} \mid S \subseteq T\}$ for some $S \in \binom{\mathcal{U}}{t}$. We define N_t to be the maximum size of a *t-star*. Note that a *t-star* is t-intersecting and so the maximum size of a t-intersecting family is at least N_t.

There has been an interest in t-intersecting families over different domains, the basic task being finding the maximum size of a t-intersecting family, and characterize the maximal families. Research began with [EKR61], who studied 1-intersecting families over the domain $\binom{[n]}{k}$. They found that the maximum size of such a family is $\binom{n-1}{k-1}$ and characterized those families as 1-stars.

Our focus will be on t-intersecting families over the symmetric group. Those were studies by Ellis, Fridgut and Pilpel [EFP11], using a spectral technique (the case $t = 1$ was studied before, by Deza and Frankl [FD77], Cameron and Ku [CK03], and Larose and Malvenuto [LM04]). For a domain $\mathcal{X}$, the idea behind the spectral approach to intersecting families is to construct an $\mathcal{X} \times \mathcal{X}$ matrix, satisfying certain properties.

Definition. A real $\mathcal{X} \times \mathcal{X}$ matrix A is *t-good* if the following properties hold:

1. A is symmetric.
2. A is supported on pairs of non-t-intersecting elements, that is, if $|x \cap y| \geq t$ then $A(x, y) = 0$.
3. $A\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ is the constant 1 vector.
4. If $\deg f \leq t$ and $\mathbb{E}[f] = 0$ then $Af = -\omega f$, where $\omega = \frac{N_t}{\mathcal{X} - N_t}$.
5. If $Af = \lambda f$ and $\deg f > t$ then $|\lambda| < \omega$.

A simple argument shows that the existence of a t-good matrix for $\mathcal{X}$ implies that a t-intersecting family is of size at most N_t, and furthermore, the characteristic function of a t-intersecting family of size N_t has degree at most t.

Ellis, Fridgut and Pilpel [EFP11] constructed a t-good matrix for $\mathcal{X} = S_n$ for large enough n (as a function of t), and thus showed that for every t, for large enough n, the maximum size of a t-intersecting family is $N_t = (n-t)!$, which is achieved by a t-star, that is, a family of the form $\{\pi \in S_n \mid \pi(i_1) = j_1, ..., \pi(i_t) = j_t\}$ for some distinct $i_1, ..., i_t \in [n]$ and some distinct $j_1, ..., j_t \in [n]$. Additionally, this result proved that the characteristic function of a maximal t-intersecting family is of degree t. In the same work it was also shown that such a family must be a t-star for large enough n, however, there was a mistake in their argument for $t > 1$, which was pointed out in [Fil17]. The result still holds, as follows from an argument of Ellis [Ell11], who actually showed something much stronger: if a t-intersecting family is not a subset of a t-star, then its size is at most $\left(1 - \frac{1}{e} + o(1)\right)(n-t)!$.

An alternative, combinatorical argument was given in [DFL+21]. The argument applies to multiple domains but we consider the case of the symmetric group.

First, let us generalize the notion of t-intersection into its bipartite version (this is not actually needed for our result, it allows us to show a stronger statement): two subsets $\mathcal{F}_1, \mathcal{F}_2$ over a domain are *cross-t-intersecting* if any $S_1 \in \mathcal{F}_1, S_2 \in \mathcal{F}_2$ have at least t elements in common, or in the case of the symmetric group, agree on the images of at least t elements.

We also need to introduce the notion of an *intersection bound*. For a domain $\mathcal{X}$ and $t \geq 1$, the intersection bound I_t is the maximal value such that for any $x \in \mathcal{X}$ and any partial input C of size at most I_t, if x t-intersects all total inputs extending C then $|x \cap C| \geq t$.

The following theorem is taken from [DFL+21, Theorem 7.1]:

Theorem 16. If $\mathcal{F}_1, \mathcal{F}_2 \subseteq S_n$ are cross-t-intersecting and the characteristic function $f_1 \colon S_n \to \{0,1\}$ of $\mathcal{F}_1$ satisfies $C(f_1) \leq I_t$, then either $\mathcal{F}_1$ is contained in a t-star, or

$$|\mathcal{F}_2| \leq \binom{C(f_1)}{t} C(f_1)(n-t-1)!$$

From the fact that for a large enough n, if $|\mathcal{F}| = (n-t)!$ then $\deg f \leq t$, where f is the characteristic function of $\mathcal{F}$, and from [Theorem 4], it follows that for such $\mathcal{F}$, $C(f)$ is is

bounded for a fixed t. Taking $\mathcal{F}_1 = \mathcal{F}_2 = \mathcal{F}$ we get that for a large enough n, either $\mathcal{F}$ is a t-star or its size is smaller than $(n-t)!$.

Theorem 16 (and similarly, its setwise version, Theorem 17) contains a stronger statement than needed for our application of it; we only need its version for t-intersecting families with $\mathcal{F}_1 = \mathcal{F}_2 = \mathcal{F}$, rather than the stated cross-t-intersection version. The theorem was originally phrased this way for historical reasons, and we being the original, more general version, as the proof is just as simple, and for better clarity of the structure of the argument.

4.2 Setwise intersection

4.2.1 Definitions and main result

Another direction is to consider *setwise-intersection* instead of intersection. A subset $\mathcal{F} \subseteq S_n$ is *t-setwise-intersecting* if for every $x, y \in \mathcal{F}$, there exists $S \in \binom{[n]}{t}$ such that $x(S) = y(S)$. We define *cross-t-setwise-intersection* similarly.

We next modify Theorem 16 for the setwise-intersection case. Here we would like to show that for every t, for a large enough n, the maximum sized t-intersecting families are the setwise equivalent of t-stars, which we call *t-setwise-stars*: families of the form $\{\pi \in S_n \mid \pi(S) = T\}$ for some $S, T \in \binom{[n]}{t}$. Note that the size of a t-setwise-star is $t!(n-t)!$. Let us define the *setwise-intersection bound*: Let I_t^s be the maximal value such that if C is a partial input of size at most I_t^s and $x \in \mathcal{X}$ t-setwise-intersects every input that contains C, then x t-setwise-intersects C.

Theorem 17. If $\mathcal{F}_1, \mathcal{F}_2$ are cross-t-setwise-intersecting and the characteristic function $f_1 : S_n \to \{0,1\}$ satisfies $C(f_1) \leq I_t^s$, then either $\mathcal{F}_1$ is contained in a t-setwise-star, or

$$|\mathcal{F}_2| \leq \max_{0 \leq m < t} \binom{t}{m} \binom{C(f_1) - t}{t-m} t!(t-m)!(n-(2t-m))!$$

Proof If $\mathcal{F}_2$ is empty, the result is trivial. Assume $\mathcal{F}_2$ is non-empty.

Suppose $\mathcal{F}_1$ is not contained in any t-setwise-star. Let C be a 1-certificate of f_1, that is, a partial input such that every input x containing C satisfies $f_1(x) = 1$. For every $S, T \in \binom{[n]}{t}$ such that $C(S) = T$ there exists $y_{S,T} \in \mathcal{F}_1$ such that $y_{S,T}(S) \neq T$.

Let $p \in \mathcal{F}_2$. Since p t-setwise-intersects every input in $\mathcal{F}_1$, and thus every input extending C, and $|C| \leq C(f_1) \leq I_t^s$, p t-setwise-intersects C. That is, there exist $S, T \in \binom{[n]}{t}$ such that $C(S) = p(S) = T$, and thus there exists $y_{S,T} \in \mathcal{F}_1$ such that $y_{S,T}(S) \neq T$. Then p also t-setwise-intersects $y_{S,T}$, that is, there exist $U, W \in \binom{[n]}{t}$ such that $p(U) = y_{S,T}(U) = W$. In conclusion, p satisfies $p(S) = T$ and $p(U \backslash S) = W \backslash T$.

Let $m = |U \cap S|$. Since $y_{S,T}(S) \neq T$, we have $U \neq S$, and so $m < t$. Note that since $p(U) = W$ and $p(S) = T$, we have $m = |W \cap T|$. There are $\binom{|C|}{t}$ choices for S and $\binom{t}{m}\binom{|C|-t}{t-m}$ choices for U. Given S, U, the number of suitable inputs p is $(n-(2t-m))!t!(t-m)!$. ■

4.2.2 Intersection bound for the symmetric group

For this section it will be convenient to again view permutations in S_n as perfect matchings over K_{2n}: we denote the elements of $[2n]$ by $a_1, ..., a_n, b_1, ..., b_n$, and given $x \in S_n$, we identify x with the matching M_x over K_{2n} where $M_x(a_i) = b_j$ if $x(i) = j$.

Lemma 18. An input x and a partial input C are t-setwise-intersecting if and only if $x \cup C$ (as a subgraph of K_{2n}) contains a set of cycles whose sizes sum to $2t$.

Proof Suppose that x, C are t-setwise-intersecting and let S, T be the witnesses. Then $(x|_{S\cup T}) \cup (C|_{S\cup T})$ (as a subgraph of $x \cup C$) is a union of cycles, since every vertex is of degree 2, and is of size $2t$, and therefore satisfies the requirement.

Suppose that $x \cup C$ contains such a set A. Let $S = A \cap \{a_1, ..., a_n\}$ and $T = A \cap \{b_1, ..., b_n\}$. For every cycle in $(x \cup C)|_A$, half of the vertices are in S and half are in T, and therefore $|S| = |T| = t$. $(x \cup C)|_A$ contains $2t$ edges, each goes from S to T, and since $x|_A, C|_A$ are perfect matchings, it must be that $x(S) = T$ and $C(S) = T$, and thus x, C are t-setwise intersecting. ■

Claim 4.2.1. *For $t < \frac{n}{2}$ we have $I^s_t = n - t - 1$.*

Proof We first show that $I^s_t < n-t$. Let $C = \{(a_i, b_i) \mid i \in [n-t]\}$. Let $x = \{(a_1, b_2), (a_2, b_3), ..., (a_{n-t}, b_1)\}$. Then x, C are not t-setwise-intersecting. However, $x(\{n-t+1, ..., n\}) = \{n-t+1, ..., n\}$, and the same holds for every input y that extends C, and so x, y are t-setwise-intersecting for every such y.

We now show that $I^s_t \geq n - t - 1$. Suppose $|C| = k \leq n - t - 1$ and x, C are not t-setwise-intersecting. By Lemma 18, it's enough to show that C can be extended to y such that no new cycles of length at most $2t$ are added to $x \cup y$. For simplicity, assume $C = \{(a_i, b_i) \mid i \in [k]\}$. Every vertex in $\{a_i \mid i \in [k]\} \cup \{b_i \mid i \in [k]\}$ is of degree 2 in $C \cup x$, and every vertex in $\{a_i \mid i \in [n] \setminus [k]\} \cup \{b_i \mid i \in [n] \setminus [k]\}$ is of degree 1, and so $C \cup x$ is a union of cycles and paths such that the ends of the paths are exactly $\{a_i \mid i \in [n] \setminus [k]\} \cup \{b_i \mid i \in [n] \setminus [k]\}$. Assume for simplicity that the pairs are $\{(a_i, b_i) \mid i \in [n] \setminus [k]\}$, and connect $(a_{k+1}, b_{k+2}), ..., (a_n, b_{k+1})$. We thus extended C to an input y by adding a cycle of length at least $2(n-k) > 2t$, as required. ■

4.2.3 Conclusion

Ellis [Ell12] shows that for every t, for a large enough n, if $\mathcal{F}$ is t-setwise-intersecting then $|\mathcal{F}| \leq t!(n-t)!$. In addition, if $\mathcal{F}$ is t-setwise-intersecting and of size $t!(n-t)!$, then f is of degree at most t, where f is the characteristic function of $\mathcal{F}$.

Fixing t and taking $\mathcal{F}_1 = \mathcal{F}_2 = \mathcal{F}$ in Theorem 17, we get that for a large enough n, either $\mathcal{F}$ is contained in a t-star or it is smaller than $\max_{0\leq m<t} \binom{t}{m}\binom{C(f_1)-t}{t-m}(n-(2t-m))!t!(t-m)!$, which From [Theorem 4] is $O_t(n-t-1)!$ and thus is smaller than $(n-t)!t!$ for a large enough n. In conclusion, for a large enough n, the maximal t-setwise intersecting families are exactly the t-setwise-stars.

Chapter 5

Open questions

Low-depth circuits for composable domains Our construction of low-depth circuits applies to functions over domains that satisfy the property we called "shortcuttability", and we showed that the symmetric group and the perfect matching scheme are shortcuttable. Can the construction be generalized to all composable domains? Conversely, can we show that for some interesting domain, the construction cannot work, for example, by showing the existence of a family of functions with low sensitivity that cannot be computed by low-depth circuits?

Setwise-intersection over composable domains Defining t-setwise intersection for families of permutations is pretty natural. Can we generalize this definition to all composable domains, and if so, obtain similar results? The answer for general composable domains is unclear. However, we believe that future attempts to generalize the theory to the perfect matching scheme would be successful. It's easy to define t-setwise intersection for the perfect matching scheme similarly to the definition over the symmetric group. The more difficult part of such work would be to repeat the spectral argument – this method was never tried in the context of the perfect matching scheme.

"Intermediate" types of intersection The notions of t-intersection and t-setwise intersection can be identified with the partitions $(1 \ldots 1)$ and (t) of t. What about other partitions? For example, families of permutations that t-intersect in the sense that every two permutations agree on the images of two (non-empty) subsets of $[n]$ whose sizes sum to t. Can we generalize the theory to all these kinds of intersections?

Improving the bound on n The results regarding the maximal size of a t-intersecting family and the characterization of the maximal families as t-stars, hold for a large enough n for any fixed t. The currently known bound on n as a function of t is exponential. Can this be improved to polynomial n?

Bibliography

[ABK+21] Scott Aaronson, Shalev Ben-David, Robin Kothari, Shravas Rao, and Avishay Tal. Degree vs. approximate degree and quantum implications of huang's sensitivity theorem. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1330–1342. ACM, 2021.

[AK97] Rudolf Ahlswede and Levon H. Khachatrian. The complete intersection theorem for systems of finite sets. *European Journal of Combinatorics*, 18(2):125–136, 1997.

[Bd02] Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288(1):21–43, 2002. Complexity and Logic.

[BGH+11] Boaz Barak, Parikshit Gopalan, Johan Håstad, Raghu Meka, Prasad Raghavendra, and David Steurer. Making the long code shorter, with applications to the unique games conjecture. *Electron. Colloquium Comput. Complex.*, page 142, 2011.

[BGJK21] Shalev Ben-David, Mika Göös, Siddhartha Jain, and Robin Kothari. Unambiguous dnfs from hex. *CoRR*, abs/2102.08348, 2021.

[CK03] Peter J. Cameron and C.Y. Ku. Intersecting families of permutations. *European Journal of Combinatorics*, 24(7):881–890, 2003.

[DFL+21] Neta Dafni, Yuval Filmus, Noam Lifshitz, Nathan Lindzey, and Marc Vinyals. Complexity measures on the symmetric group and beyond. In *12th Innovations in Theoretical Computer Science Conference*, volume 185 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 87, 5. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2021.

[EFP11] David Ellis, Ehud Friedgut, and Haran Pilpel. Intersecting families of permutations. *J. Amer. Math. Soc.*, 24(3):649–682, 2011.

[EKR61] Paul Erdös, Chao Ko, and Richard Rado. Intersection theorems for systems of finite sets. *Quarterly Journal of Mathematics*, 12:313–320, 1961.

[Ell11] David Ellis. Stability for t-intersecting families of permutations. *J. Combin. Theory Ser. A*, 118(1):208–227, 2011.

[Ell12] David Ellis. Setwise intersecting families of permutations. *J. Combin. Theory Ser. A*, 119(4):825–849, 2012.

[FD77] Peter Frankl and Mikhail Deza. On the maximum number of permutations with given maximal or minimal distance. *Journal of Combinatorial Theory, Series A*, 22(3):352–360, 1977.

[Fil17] Yuval Filmus. A comment on intersecting families of permutations. arXiv:1706.10146, 2017.

[Fri08] Ehud Friedgut. On the measure of intersecting families, uniqueness and stability. *Combinatorica*, 28:503–528, 09 2008.

[FW86] P. Frankl and R.M. Wilson. The erdös-ko-rado theorem for vector spaces. *Journal of Combinatorial Theory, Series A*, 43(2):228–236, 1986.

[GNS+16] Parikshit Gopalan, Noam Nisan, Rocco Servedio, Kunal Talwar, and Avi Wigderson. Smooth boolean functions are easy: Efficient algorithms for low-sensitivity functions. pages 59–70, 01 2016.

[Hua19] Hao Huang. Induced subgraphs of hypercubes and a proof of the sensitivity conjecture. *Annals of Mathematics*, 190(3):949–955, 2019.

[Kho02] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, page 767–775, New York, NY, USA, 2002. Association for Computing Machinery.

[KMS17] Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and grassmann graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, page 576–589, New York, NY, USA, 2017. Association for Computing Machinery.

[LM04] Benoit Larose and Claudia Malvenuto. Stable sets of maximal size in kneser-type graphs. *European Journal of Combinatorics*, 25:657–673, 07 2004.

[OW09] Ryan O'Donnell and Karl Wimmer. Kkl, kruskal-katona, and monotone nets. volume 42, pages 725 – 734, 11 2009.

[Wil84] Richard M. Wilson. The exact bound in the erdös-ko-rado theorem. *Combinatorica*, 4:247–257, 1984.

פונקציות אלו מעגלים מעומק נמוך. הרעיון בבנייה זו הוא להחיל "רעש חד-כיווני" על הקלט, כלומר, לאפס חלק מהאחדות, לחשב את ערך הפונקציה בהסתברות גבוהה מספיק, ולחזור על התהליך רקורסיבית עד שמגיעים לנקודה ממשקל המינג נמוך מספיק, שערך הפונקציה עליה כבר ידוע.

הוכחת השערת הרגישות מאפשרת גישה נוספת לבניית מעגלים מעומק נמוך עבור פונקציות בעלות רגישות נמוכה, העוברת דרך עצי החלטה: לפונקציה בעלת רגישות נמוכה יש עץ החלטה מעומק נמוך, ואת העץ אפשר להפוך למעגל. עם זאת, הבנייה הישירה נותנת חסמים טובים יותר מהבנייה המשתמשת בעצי החלטה, ולכן עודנה מעניינת.

בהמשך, תכונת הכדור הוכללה לתחומים נוספים, כולל החבורה הסימטרית. בעבודה זו אנחנו משתמשים בהכללה הזו כדי להכליל את בניית המעגלים הקטנים והמעגלים מעומק נמוך לפונקציות בעלות רגישות נמוכה מעל החבורה הסימטרית ותחומים נוספים. הפעולה המקבילה בחבורה הסימטרית למעבר מקלט אחד לשכנו, שבקוביה הבוליאנית מתקבלת ע"י הפיכת אחד מהביטים, היא הפעלת חילוף על התמורה. הפעולה המקבילה ליצירת "רעש חד-כיווני" באמצעות איפוס חלק מהאחדות, היא הסרת חלק מהאיברים מייצוג התמורה ע"י מעגלים.

בניית המעגל הקטן שלנו נותנת, עבור פונקציה f מעל החבורה הסימטרית, מעגל בגודל $n^{O(s(f))}$, כאשר $s(f)$ היא הרגישות של f, והתוצאה עבור תחומים נוספים היא דומה. בניית המעגל מעומק נמוך נותנת, עבור פונקציה f מעל החבורה הסימטרית, מעגל בעומק $O(s(f)\log n)$, ותוצאה דומה מתקיימת עבור סכמת השידוכים המושלמים.

משפחות נחתכות של תמורות

בחלק האחרון של עבודה זו, אנחנו מתמקדים במשפחות נחתכות של תמורות, ורותמים לשם כך את הקשר הפולינומי בין דרגה של פונקציה בוליאנית וסיבוכיות האישור שלה.

המחקר של משפחות נחתכות החל במקרה של משפחות נחתכות של קבוצות בתחום $\binom{[n]}{k}$. בהקשר זה, משפחה נחתכת היא משפחה $\mathcal{F}\subseteq\binom{[n]}{k}$ כך שכל שתי קבוצות $x,y\in\mathcal{F}$ הן נחתכות. בהמשך, המחקר בנושא הורחב בכיוונים שונים, כאשר השאלה היא כמה גדולה משפחה נחתכת יכולה להיות, ואיך משפחה נחתכת מקסימלית נראית. אחד הכיוונים עוסק במשפחות t-נחתכות – משפחה t-נחתכת של קבוצות היא משפחה שבה כל שתי קבוצות הן בעלות חיתוך בגודל לפחות t. כיוון אחר מרחיב את העבודות בנושא לתחומים נוספים.

שני הכיוונים הללו נפגשו בעבודות שעסקו במשפחות t-נחתכות של תמורות. משפחה t-נחתכת של תמורות היא משפחה $\mathcal{F}\subseteq S_n$ כך שכל שתי תמורות $x,y\in\mathcal{F}$ מסכימות על לפחות t איברים. מתברר כי עבור n גדול מספיק, הגודל המקסימלי של משפחה t-נחתכת של תמורות הוא $(n-t)!$, ומשפחה כזו היא t-כוכב, כלומר משפחה מהצורה $\{\pi\in S_n\mid\pi(i_1)=j_1,...,\pi(i_t)=j_t\}$ כאשר $i_1,...,i_t\in[n]$ הם איברים שונים ו-$j_1,...,j_t\in[n]$ הם איברים שונים.

אנו מכלילים תוצאה זו למקרה של משפחות t-קבוצה-נחתכות של תמורות, שבהן כל שתי תמורות מסכימות על התמונה של קבוצת איברים בגודל t. אנו נותנים הוכחה חלופית לתוצאה ידועה שלפיה, עבור n גדול מספיק, הגודל המקסימלי של משפחה t-קבוצה-נחתכת של תמורות הוא $t!(n-t)!$, ומשפחה כזו היא t-קבוצה-כוכב, כלומר משפחה מהצורה $\{\pi\in S_n\mid\pi(S)=T\}$ כאשר $S,T\in\binom{[n]}{t}$. ההוכחה המובאת על ידינו היא פשוטה בהרבה מההוכחה הקיימת (שמראה תוצאה חזקה יותר), ומתקיימת עבור ערכים קטנים יותר של n.

תקציר

הקוביה הבוליאנית ממימד n היא התחום $\{0,1\}^n$. פונקציה בוליאנית על הקוביה הבוליאנית היא פונקציה $f:\ \{0,1\}^n \to \{0,1\}$. פונקציה כזו יכולה להיות הפונקציה המחושבת ע"י מעגל בוליאני, ייצוג של בעיית הכרעה, ייצוג של אובייקט כמו גרף או תת-קבוצה, או דוגמאות רבות נוספות.

בתורת הסיבוכיות מתעניינים בשאלה עד כמה אובייקט מסוים, כמו פונקציה בוליאנית, הוא קשה לחישוב או מסובך. בספרות מופיעים מספר מדדי סיבוכיות, ביניהם סיבוכיות מעגלים, סיבוכיות עצי החלטה (המודדת את עומקם של עצי החלטה שמחשבים את הפונקציה), סיבוכיות אישור (הגרסה האי-דטרמיניסטית של סיבוכיות עצי החלטה), דרגה ורגישות. קיומם של מספר מדדי סיבוכיות שונים מעלה את השאלה: מי מהם הוא המדד ה"נכון"? טבעי, על כן, לחפש קשרים ופערים בין המדדים השונים. השאלה הנ"ל מקבלת מענה בזכות תוצאות ידועות המראות כי מדדים רבים, כולל אלו שהזכרנו (פרט לסיבוכיות מעגלים), הם שקולים פולינומית.

עם הזמן, המחקר של פונקציות בוליאניות מעל הקוביה הבוליאנית התרחב לפונקציות בוליאניות מעל תחומים נוספים, כמו הפרוסה (קבוצת כל הוקטורים מאורך קבוע ומשקל המינג קבוע), גרפים מרחיבים רב-ממדיים וסכמת גרסמן. האם ניתן להרחיב לתחומים אלה את התיאוריה של מדדי סיבוכיות מעל הקוביה הבוליאנית? לאחרונה, ההגדרות של חלק ממדדי הסיבוכיות הקלאסיים הוכללו לפונקציות בוליאניות מעל תחומים כגון החבורה הסימטרית S_n (קבוצת כל התמורות מעל n איברים), סכמת השידוכים המושלמים (קבוצת כל השידוכים המושלמים מעל הגרף המלא בעל $2n$ צמתים), הפרוסה והרב-פרוסה (גרסה מוכללת של הפרוסה שבה יש מספר צבעים, המהווה גם הכללה של החבורה הסימטרית). מתברר כי מדדי הסיבוכיות נשארים שקולים פולינומית גם עבור תחומים אלה.

חישוב יעיל של פונקציות בעלות רגישות נמוכה

רגישות היא מדד סיבוכיות של פונקציות בוליאניות המודד עד כמה הפונקציה "רגישה" לשינויים קטנים בקלט. הרגישות מוגדרת להיות הדרגה המקסימלית בגרף שצמתיו הם קודקודי הקוביה, ובין שני צמתים x, y יש קשת אם הם במרחק המינג 1 זה מזה ומתקיים $f(x) \neq f(y)$.

ידוע כבר מספר עשורים כי חלק ממדדי הסיבוכיות שהזכרנו מעל הקוביה הבוליאנית הם שקולים פולינומית, אך עד לאחרונה זה לא כלל את הרגישות. לפני שהוכחה השערת הרגישות, הגורסת כי לפונקציה בעלת רגישות נמוכה קיים עץ החלטה בעל עומק נמוך, נעשו נסיונות להוכיח קשרים בין רגישות וסיבוכיות מעגלים. רגישות נמוכה של פונקציה מאפשרת תיקון מקומי, ועובדה זו שימשה להוכחת תכונה של פונקציות מעל הקוביה הבוליאנית, הנקראת "תכונת הכדור": פונקציה בעלת רגישות נמוכה נקבעת באופן יחיד ע"י ערכיה על כדור המינג קטן. זאת כיוון שערכה של הפונקציה בכל נקודה x מחוץ לכדור נקבע באופן יחיד ע"י לקיחת רוב על פני חלק משכני x הקרובים יותר לכדור, וכך ניתן לחשב את הפונקציה רקורסיבית. תכונת הכדור שימשה לבניית מעגלים בוליאניים קטנים עבור פונקציות בעלות רגישות נמוכה. בנוסף, נבנו עבור

המחקר בוצע בהנחייתו של פרופסור יובל פילמוס, בפקולטה למדעי המחשב.

חלק מן התוצאות בחיבור זה פורסמו כמאמרים מאת המחברת ושותפיה למחקר בכנסים ובכתבי-עת במהלך תקופת המחקר של המחברת, אשר גרסאותיהם העדכניות ביותר הינן:

Neta Dafni, Yuval Filmus, Noam Lifshitz, Nathan Lindzey, and Marc Vinyals. Complexity measures on the symmetric group and beyond. In *12th Innovations in Theoretical Computer Science Conference*, volume 185 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 87, 5. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2021.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי.

מדדי סיבוכיות
על החבורה הסימטרית
ותחומים נוספים

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
מגיסטר למדעים במדעי המחשב

נטע דפני

הוגש לסנט הטכניון — מכון טכנולוגי לישראל
כסלו התשפ"ב חיפה נובמבר 2021

מדדי סיבוכיות
על החבורה הסימטרית
ותחומים נוספים

נטע דפני