Aalto University
School of Science
Bachelor's Programme in Science and Technology

Detecting multilinear monomials with algebraic fingerprints

Bachelor's Thesis

April 28, 2023

Onni Miettinen

Aalto University
School of Science
Bachelor's Programme in Science and Technology

ABSTRACT OF
BACHELOR'S THESIS

AM: prefer \cdot, $*$ usually indicates some other operation

AM: use poly from complexity package

AM: First sentence of abstract uses a bit too much jargon

AM: two "howevers" in third paragraph

Author:	Onni Miettinen
Title of thesis:	Detecting multilinear monomials with algebraic fingerprints
Date:	April 28, 2023
Pages:	27
Major:	Computer Science
Code:	SCI3027
Supervisor:	Prof. Eero Hyvönen
Instructor:	D.Sc. Augusto Modanese (Department of Computer Science)

This thesis studies how parameterized combinatorial problems can be solved by detecting a k-multilinear monomial in a multivariate polynomial by algebraic means. More specifically, the thesis focuses on the technique of algebraic fingerprinting by Koutis and Williams. Moreover, the thesis overviews the technique as a general framework for parameterized combinatorial problems, and discusses its limits. Finally, the thesis collects some ideas for improving the framework or even surpassing the lower limit of the general framework.

With algebraic fingerprints, a k-multilinear monomial can be detected in time $\mathcal{O}(2^k * \mathsf{poly}(n))$, where n is the number of variables in the multivariate polynomial. This gives a general framework for solving parameterized combinatorial problems efficiently, since they can in general be reduced to instances of k-multilinear monomial detection problems. Especially algorithms that rely on color coding can be accelerated in time by a factor of e^k with the use of algebraic fingerprints. Currently, this technique underlies fastest algorithms for many parameterized combinatorial problems.

However, $\Omega(2^k * poly(n))$ is the lower limit of the technique: multilinear monomials cannot be detected faster with the general technique of algebraic fingerprints by Koutis and Williams. However, the algebraic framework solves the very general multilinear monomial detection problem; if the underlying combinatorics of the problem are well understood and taken use of in the reduction to multilinear monomial detection, a much faster algorithm based on algebraic fingerprinting may be found for a specific problem. Using ideas from algebraic fingerprinting, Björklund managed to find an algorithm that finds a Hamiltonian path in time $\mathcal{O}(1.657^n * poly(n))$.

Keywords:	multilinear, monomial, detection, algebraic, fingerprints, fingerprinting, parameterized, combinatorial
Language:	English

Aalto-yliopisto
Perustieteiden korkeakoulu
Teknistieteellinen kandidaattiohjelma

KANDIDAATINTYÖN
TIIVISTELMÄ

Tekijä:	Onni Miettinen
Työn nimi:	Detecting multilinear monomials with algebraic fingerprints
Päiväys:	28. huhtikuuta 2023
Sivumäärä:	27
Pääaine:	Tietotekniikka
Koodi:	SCI3027
Vastuuopettaja:	Prof. Eero Hyvönen
Työn ohjaaja(t):	D.Sc. Augusto Modanese (Tietotekniikan laitos)

Tässä työssä tutkitaan, kuinka parametrisoituja kombinatorisia ongelmia voidaan ratkaista löytämällä k-asteinen multilineaarinen monomi monimuuttujaisesta polynomista algebrallisin keinoin. Kandidaatintyö keskittyy tarkemmin Williamsin ja Koutisin algebrallisia sormenjälkiä hyödyntävään tekniikkaan, jolla voidaan havaita multilineaarinen monomi tehokkaasti. Lisäksi tutkitaan tämän tekniikan suomaa yleistä kehystä parametrisoitujen ongelmien ratkaisemiseksi ja sen mahdollisia rajoja. Viimeiseksi pohditaan keinoja, joilla kehystä voitaisiin kehittää tai jopa ohittaa kehyksen antama aliraja algoritmien tehokkuudelle.

Williamsin ja Koutisin algebrallisten sormenjälkien avulla voidaan havaita k-asteen multilineaarinen monomi ajassa $\mathcal{O}(2^k * poly(n))$, jossa n on polynomin muuttujien lukumäärä. Tämä johtaa yleiseen parametrisoitujen ongelmien kehykseen, jolla niitä voidaan ratkaista tehokkaasti. Etenkin color-coding—metodiin nojaavia algoritmeja voidaan nopeuttaa tekijällä e^k hyödyntämällä algebrallisia sormenjälkiä. Tällä hetkellä algebrallisten sormenjälkien tekniikka pohjaa nopeimpia algoritmeja monille parametrisoiduille kombinatorisille ongelmille.

$\Omega(2^k * poly(n))$ on kuitenkin kehyksen alaraja: Williamsin ja Koutisin algebrallisten sormenjälkien avulla ei voida havaita multilineaarisia monomeja nopeammin. Toisaalta heidän kehys perustuu yleiseen multilineaaristen monomien havaitsemisongelmaan; jos hyödynnetään ongelmakohtaisia kombinatorisia ominaisuuksia ongelman redusoinnissa multilineaaristen monomien havaitsemiseksi, voidaan löytää algoritmi, joka suoriutuu nopeammin kyseisellä ongelmalla. Björklund kykeni kehittämään algebrallisten sormenjälkien avulla algoritmin, joka löytää Hamiltonisen polun ajassa $\mathcal{O}(1.657^n * poly(n))$.

Avainsanat:	multilineaarinen, monomi, havaitseminen, algebrallinen, sormenjälki, parametrisoitu, kombinatorinen
Kieli:	Englanti

Contents

1 Introduction

In the past fifteen years, there have been rapid advances in the algorithms for combinatorial problems. This has been greatly sparked by the development in algebraic techniques for solving the multilinear monomial detection problem, that is, deciding whether a multivariate (multiple variables) polynomial contains a multilinear monomial[1]. Multilinear monomial detection for parameterized combinatorial problems was first introduced by Koutis in an early work [Kou05], where a set packing problem is reduced to multilinear monomial detection.

AM: I do see this style of footnotes from time to time, but some people are not fond of it. Parentheses might be a better choice.

AM: (Namely,) the technique of *algebraic fingeprinting*, introduced by Koutis [Kou08] and further developed by Williams [Wil09], has found great success for many combinatorial problems. For instance with algebraic fingerprinting, the k-path problem that previously could be solved in $\mathcal{O}^*(4^k)$ [2] time by Chen et al. [Che+07], can now be solved in $\mathcal{O}^*(2^{3k/2})$ time [Kou08]. This result was quickly improved by Williams [Wil09], who gave an $\mathcal{O}^*(2^k)$ algorithm.

Most famously, Björklund [Bjö14] published an algorithm that solved the famous Hamiltonian problem (Hamiltonicity) in $\mathcal{O}^*(1.657^n)$ time with clever utilizations of this novel algebraic technique. The fastest algorithms for Hamiltonicity before this ran in $\mathcal{O}^*(2^n)$ time , and were known since 1962 AM: and date from 1962 [HK62], [Bel62]. This was a significant improvement on a problem that had seen no progress in nearly fifty years! Soon enough for k-path, an $\mathcal{O}^*(1.657^k)$ algorithm was found AM: thereafter, a $\mathcal{O}^*(1.657^k)$ algorithm was found for the related (?) k-path problem [Bjö+17].

AM: Move footnote elsewhere

1.1 Research scope and thesis structure

The goals of this bachelor's thesis are to find out how multilinear monomial detection is relevant in combinatorial problems, and how algebraic fingerprints can be utilized to design faster algorithms for these problems. For the reader with background in computer science, this thesis overviews a clever use of algebra in theoretical computer science, that is the technique of algebraic fingerprinting.

In the next subsections AM: rest of this introduction, we discuss the relation of combinatorics and polynomials, discuss the algebrization of combinatorial problems, and define the multilinear monomial detection problem as a prelude for the thesis.

AM: used twice

In Section 2, we overview the area of problems where this technique proves useful. Section 3 gives necessary preliminaries for the reader. In Section 4, we dive into the technique of algebraic fingerprints, and solve the multilinear monomial detection problem with

[1] i.e., a term of multiple variables where each variable appears at most once. See also Section 3.1

[2] $\mathcal{O}^*$ may hide factors polynomial to input size. See Section 3.3

algebraic means. Section 5 briefly overviews ideas for improvement from the general algebraic technique. Section 6 concludes the thesis.

1.2 Algebrization of combinatorial problems

A combinatorial problem asks whether a given finite set of objects satisfies some given constraints. For example, the k-path problem asks for, given a finite set of vertices and edges, a simple[3] path of k vertices. The solutions and non-solutions (solution candidate space) to combinatorial problems can be thought of as combinations of the given objects. The solution candidate space for the k-path problem consists of combinations of $k-1$ edges, where a valid solution combination contains only disjoint edges.

Valiant [Val92] observed that multivariate polynomials have natural combinatorial interpretations. Now, we see how combinatorics may be represented with multivariate polynomials. We may think of a polynomial in its sum of monomials form[4] as a set of combinations. For instance, imagine a set of elements $M = \{a, b, c\}$. A AM: The polynomial $C = a + b + c$ represents all the possible choices if we are to pick an element from M. Following the standard idea behind polynomial multiplication, we see that a polynomial

$$C^2 = (a+b+c)(a+b+c) = aa + ab + ac + ba + bb + bc + ca + cb + cc$$

represents all the possible combinations if we are to pick twice from M. Indeed, a polynomial C^k represents all combinations of k elements from M.

Utilizing this idea, Koutis [Kou05] managed to reduce a combinatorial problem into an algebraic one, that is, multilinear monomial detection. In multilinear monomial detection, we represent the combinatorial problem as a polynomial where the solution candidates are encoded as follows: the non-solutions are non-multilinear and solutions are multilinear terms.

AM: You already discuss it in Section 1.3, but you should also mention somewhere around here that complexity of problem is sensitive to how the polynomial is presented (e.g., in sum of monomials form it is trivial) and that the presentation is going to be very succinct (namely algebraic circuit). 1 sentence is enough.

For an instance of k-path problem, we may use the set of edges as M where an edge is a product of two indeterminates defined uniquely for each vertex. If we look for $k-1$ combinations of elements from M, we see that a multilinear term in the resulting polynomial C^{k-1} corresponds to a solution to the k-path problem: as the edges are disjoint, each indeterminate appears only once.

[3] i.e., a path where each vertex is visited only once

[4] an expression where there is no multiplication after addition of monomials. See also Section 3.3

Thus, the task of finding a satisfying combination to the combinatorial problem has been reduced to finding a multilinear monomial from the multivariate polynomial. It follows that a decision problem is answered by the existence of a multilinear monomial, and a counting problem by the number of multilinear monomials. This thesis, however, focuses on the decision problem [5]. AM: This thesis focuses on the former (although we give a brief overview of actually finding a solution in Section 4.3.3)

Appropriate definitions for the indeterminates for M are problem-specific. In what follows, this thesis gives another example reduction to multilinear monomial detection: we algebrize the k-3D matching problem.

The k-3D matching problem is defined as follows:

k-3D MATCHING

Input: Three disjoint sets A, B and C, and a set of triples $T \subseteq A \times B \times C$.

Question: Is there a subset $M \subseteq T$, such that $|M| = k$ and the all the triples $m \in M$ are disjoint?

We begin by defining new indeterminates corresponding to the elements in A, B and C, labeled as a_i, b_j and c_k, respectively, where $i \in [\,|A|\,]$, $j \in [\,|B|\,]$ and $k \in [\,|C|\,]$ such that $[\,n\,] = \{1, \ldots, n\}$.

For every triple $t \in T$, we define a multilinear monomial x that is a product of the elements in t. We denote the set of those monomials as X. Thus, $(a, b, c) \in T \implies abc \in X$. From an algebraic perspective, we form monomials in the commutative polynomial ring $\mathbb{Z}[X]$ (i.e., polynomials that have coefficients from $\mathbb{Z}$ and variables from X. See also Section 3.1).

Next, we define multivariate polynomials $P_1, P_k \in \mathbb{Z}[X]$ AM: $P_k \in \mathbb{Z}[X]$ for $k \in \mathbb{N}$ iteratively as follows:

OM: I'm not sure what to put for $\mathbb{Z}$ here, since we don't have any coefficients here yet

$$P_1 = \sum_{x \in X} x, \quad P_k = P_1^k$$

Following this construction, we observe that P_k, when expanded into a sum of multivariate monomials, contains a multilinear term if and only if the original k-3D matching instance can be answered in the positive. Thus, we have a successful reduction to multilinear monomial detection for the k-3D matching.

For instance with $A = \{a_1, a_2\}$, $B = \{b_1, b_2\}$, $C = \{c_1, c_2, c_3\}$ and

$$T = \{(a_1, b_1, c_1), (a_1, b_2.c_2), (a_2, b_2, c_3)\}$$

(see Figure TODO: diagram of the 3D matching problem with the matchings), we have $P_1 = a_1b_1c_1 + a_1b_2c_2 + a_2b_2c_3$. For $k = 2$, we get

AM: = after line break

[5] Section 4.3.3 briefly overviews actually finding a solution

$$P_2 = (a_1b_1c_1 + a_1b_2c_2 + a_2b_2c_3)^2$$
$$= a_1^2b_1^2c_1^2 + a_1^2b_2^2c_2^2 + a_2^2b_2^2c_3^2 + 2a_1^2b_1b_2c_1c_2 + 2a_1a_2b_1b_2c_1c_3 + 2a_1a_2b_2^2c_2c_3.$$

We may see that the multilinear monomial $2a_1a_2b_1b_2c_1c_3$ corresponds to a solution to our problem: the triples (a_1, b_1, c_1) and (a_2, b_2, c_3) are indeed disjoint, as seen in Figure TODO.

1.3 Detecting a multilinear monomial

We have now reached multilinear monomial detection from the perspective of combinatorial problems. Furthermore, we may reach this point from *parameterized* problems as well; instead of picking k triples or edges, we may pick, say, $k - 3$ combinations by only computing P_{k-3} instead of P_k. Thus, the multilinear monomial detection approach works well with parameterization. The general, parameterized multilinear monomial detection problem is defined as follows:

AM: footnote after question mark

k-MULTILINEAR MONOMIAL DETECTION

Input: A polynomial $P_1 \in \mathbb{Z}[X]$ with the solution candidates encoded as monomials.

Question: Does the polynomial $P_k = P_1^k$ extended as a sum of monomials contain a multilinear monomial of degree k [6]?

Clearly, an upper bound for solving the problem is given by a naive expansion of P_1^k into a sum of monomials and scanning all the terms. However, this is not optimal: an N-degree polynomial will have e^N [7] possible monomials, and most problems using this algebrization (see Section 2) have been solved with faster algorithms. This motivates the detection of multilinear monomials without fully expanding P_k into a sum of monomials.

Since only multilinear terms are important in P_k, any squared variable can be instantly discarded as soon as it is formed during the computation of P_1^k. This can be achieved with dynamic programming to create a polynomial P_k' that only contains multilinear monomials. Since there are 2^n multilinear monomials given n variables, this method results in a slightly faster algorithm than with naive expansion.

However, the underlying problems are usually FPT [8]. This implies that scaling exponentially with the number of variables is far from optimal. In order for the complexity of the algorithm to scale with the parameter k, we can reduce the number of variables by mapping $\rho\colon X \to Y$, where $|X| \geq |Y|$ and $|Y| = \Theta(k)$, and dynamically expand $(P_1')^k \in \mathbb{Z}[Y]$ instead of P_k, where $P_1' \in \mathbb{Z}[Y]$ represents $P_1 \in \mathbb{Z}[X]$ mapped with ρ. AM:

[6] a multilinear monomial with degree k has k indeterminates, see also Section 3.1

[7] e^N is an approximation from $\binom{n+N}{N}$, where n is the number of variables

[8] fixed parameter tractable, i.e., polynomial to n in time if k is fixed. See Section 3.3

$P'_1 = \Phi_\rho(P_1) \in \mathbb{Z}[Y]$ and Φ_ρ is the canonical extension of ρ to a ring homomorphism $\mathbb{Z}[X] \to \mathbb{Z}[Y]$ (i.e., simply replacing every $x \in X$ in P_1 with $\rho(x)$)

However, since $|X| \geq |Y|$, a multilinear monomial in P_k may not be multilinear in P'_k since we may map two different variables of a multilinear monomial in X to the same variable in Y. For an algorithm to reliably detect a multilinear monomial, it is necessary to use different mappings from X into Y until a multilinear monomial survives the mapping. However, the probability that any given multilinear monomial survives this mapping AM: a uniformly chosen mapping (?) is around $e^{-|Y|}$ [KW15]. This implies that $\Omega(e^{|Y|})$ random mappings must be tried for a multilinear monomial to survive with a reasonable constant probability. Thus, a k-multilinear monomial is detected with a randomized algorithm in $\mathcal{O}^*((2e)^{|Y|})$ time, where $|Y| = \Theta(k)$.

This is essentially the idea behind color coding introduced by Alon, Yuster, and Zwick [AYZ95]. Although here given in this algebraic form for the k-multilinear monomial detection, color coding is purely combinatorial, and does not rely on algebraic techniques. However, since k-multilinear monomial detection is a purely algebraic problem, it is reasonable to conjecture that there is an algebraic method for solving it.

Indeed, a faster algebraic method exists; the technique of algebraic fingerprinting first introduced by Koutis [Kou08] and further developed by Williams [Wil09] solves k-multilinear monomial detection in $\mathcal{O}^*(2^k)$ time. Since the technique focuses on the abstract k-multilinear monomial detection, to which many parameterized problems can be reduced to, it gives a general framework for solving parameterized problems [KW15].

2 Related problems

AM: Section looks very short, so merge it with intro

The importance of multilinear monomial detection lies in the fact that many parameterized combinatorial problems, as we see in this section, can be reduced to instances of it. Thus, solving the general k-multilinear monomial detection problem efficiently has great interest. Currently, the technique of algebraic fingerprints underlies the fastest algorithms for all the problems mentioned here. Note that only time complexity is discussed here.

Koutis was the first one to introduce and apply multilinear monomial detection for combinatorial problems, namely the *k-path* and *m-set k-packing* problems [Kou08]. In their work [KW09], Koutis and Williams reduced the *k-tree*, *k-leaf spanning tree*, and *t-dominating set* problems to k-multilinear monomial detection, and gave $\mathcal{O}^*(2^k)$ algorithms for these problems.

Björklund, Kaski, and Kowalik [BKK16] showed reductions for the *k-sized graph motif* problem and for several optimization variants such as *closest graph motif* and *maximum graph motif* problems which are relevant in e.g. bioinformatics. Cadena, Ekanayake, and

Vullikanti [CEV17] applied algebraic fingerprinting to general *graph scan statistics* which is a methodology that detects anomalies in a graph. This is relevant in general network design problems.

3 Preliminaries

It is necessary to recall basic algebraic concepts and common notation before further discussing multilinear monomial detection and algebraic fingerprinting. Section 3.1 gives definitions for some necessary algebras and algebraic concepts. Section 3.2 outlines arithmetic circuits. Section 3.3 gives a table of other notation and terminology used throughout the thesis.

3.1 Algebra

We generally refer to the following algebraic objects as *algebraic structures*, i.e., objects that contain a set of elements and operations on them.

Groups. A *group* $\mathbf{G}$ is a tuple $(G, +)$, where G is a set of elements, $+\colon G \times G \to G$ is a binary operation closed under the elements in G, $+$ is associative, every element $g \in G$ has an inverse $g^{-1} \in G$, and G contains an identity element e such that $g + e = g$, $g + g^{-1} = e$ and $e = e^{-1}$. Moreover, $\mathbf{G}$ is called *Abelian* if $+$ is also commutative. A group is called *multiplicative* if the operation is written as multiplication.

Rings. A *ring* $\mathbf{R}$ is a triple $(R, +, \cdot)$, where $(R, +)$ is an Abelian group, and $\cdot\colon R \times R \to R$ is a binary operation closed under R. We call the binary operations $+$ and $\cdot$ addition and multiplication, respectively. A *commutative ring* has commutative multiplication.

Note, from here on we may use a bold typeface to represent either the set of elements or the algebraic structure itself, i.e., we may use expressions such as $u \in \mathbf{R}$, where $\mathbf{R} = (R, +, \cdot)$ and $u \in R$. Moreover, multiplication may be denoted by juxtaposition for brevity: $a \cdot b = ab$.

$\mathbf{R}$ must contain a multiplicative identity $\mathbf{1} \in \mathbf{R}$ such that $\forall a \in \mathbf{R}\colon a \cdot \mathbf{1} = a$. We notate the additive identity e as $\mathbf{0}$ from here on. Observe that for any $R \neq \{\mathbf{0}\}$, $\mathbf{1} \neq \mathbf{0}$. Left and right distributive laws hold for rings, i.e.,

$$\forall a, b, c \in \mathbf{R}\colon a \cdot (b + c) = (ab) + (ac) \wedge (b + c) \cdot a = (ba) + (ca).$$

$u \in \mathbf{R}$ is called *unit* if it holds that $\exists v \in \mathbf{R}\colon uv = vu = \mathbf{1}$, i.e., it has a multiplicative inverse $v \in \mathbf{R}$.

Fields. A *field* $\mathbf{F} = (F, +, \cdot)$ is defined with the following conditions:

- $(F, +)$ is an Abelian group
- $(F \setminus \{\mathbf{0}\}, \cdot)$ is an Abelian group
- Left and right distributive laws hold for $\mathbf{F}$.

Equivalently, a ring is a field if every non-zero element is unit, $\mathbf{1} \neq \mathbf{0}$, and multiplication is commutative. The *characteristic* of a field $\mathbf{F}$ is defined as follows:

$$\texttt{char}(\mathbf{F}) = \begin{cases} \min\{n \in \mathbb{N} \colon n \cdot \mathbf{1} = \mathbf{0}\} \\ 0 \qquad\qquad\qquad\qquad \text{if such } n \text{ does not exist} \end{cases} \tag{1}$$

Note, that a field $\mathbf{F}$ with characteristic 2 satisfies the following:

$$\forall u \in \mathbf{F} \colon u + u = u \cdot (\mathbf{1} + \mathbf{1}) = u \cdot \mathbf{0} = \mathbf{0}$$

Throughout the thesis we may use language such as *cancelation due to characteristic* to refer to the fact that an element $f \in \mathbf{F}$ may cancel itself out in an expression if $\mathbf{F}$ has non-zero characteristic. We mainly use characteristic 2 in this thesis, and thus, we may say that f cancels out due to characteristic when it has an even coefficient: $f + f + f + f = 4f = 2f + 2f = \mathbf{0} + \mathbf{0} = \mathbf{0}$. AM: For every $k \geq 0$, $2kf = k(2f) = k \cdot \mathbf{0} = \mathbf{0}$

Characteristics are prevalent in *finite fields*, where the set of elements is finite. A simple example of a finite field is $\mathbb{Z}_2 = (\{\mathbf{0}, \mathbf{1}\}, +, \cdot)$, where the operations are standard addition modulo 2, and standard multiplication.

Finite fields can be noted as $GF(p)$, called *Galois fields*, where the field is unique up to *isomorphism* if p is a prime or a power of some prime. An isomorphism $\kappa \colon \mathbf{A} \to \mathbf{B}$ is a bijective mapping that satisfies $\forall u, v \in \mathbf{A} \colon \kappa(uv) = \kappa(u)\kappa(v) \wedge \kappa(u + v) = \kappa(u) + \kappa(v)$. Thus, if we only care for the relations between the elements, we may say that algebraic structures that isomorphic are equivalent for us. Note that if p is prime and $k \in \mathbb{N}$, the characteristic of $GF(p^k)$ is p.

Polynomial rings and multilinearity. AM: For a (finite) set X, A *polynomial ring* in X, noted as $\mathbf{K}[X]$, is a ring of polynomials with indeterminates (or variables) from X and coefficients from the commutative ring $\mathbf{K}$. In this thesis, we only handle *multivariate* polynomials, i.e., polynomials that have $|X| = n > 1$.

A *monomial* in X is of the form $X_1^{a_1} X_2^{a_2} \cdots X_n^{a_n}$, where $X_i \in X$ and $a_i \in \mathbb{N}$ marks the exponent. A monomial is *multilinear* if $\forall i \in \{1, \ldots, n\} \colon a_i = 0 \vee a_i = 1$. The *degree* of a monomial m is the sum of the exponents: $\deg(m) = \sum_{i=1}^{n} a_i$. Any element P of such polynomial ring is a finite linear combination of these monomials with coefficients $k \in \mathbf{K}$:

$$P = \sum_{p} k_p (X_1^{a_{1,p}} \cdots X_n^{a_{n,p}})$$

AM:

$$P = \sum_{p_1,\ldots,p_n \geq 0} k_{p_1,\ldots,p_n} X_1^{a_{p_1}} \cdots X_n^{a_{p_n}}$$

Alternatively, a polynomial ring $\mathbf{K}[X]$ can be thought of as a vector space AM: over $\mathbf{K}$ of infinite dimension with the AM: set of all the monomials as a basis. Polynomial rings have three operations: addition, multiplication and scalar multiplication. In terms of vectors, addition and scalar multiplication follow the standard addition and scalar multiplication for vectors. Note that the scalar value must be in $\mathbf{K}$. The multiplication is defined as follows: let $M(X)$ denote the set of all monomials in X, and P_d denote a polynomial in $\mathbf{K}[X]$ of the form $P_d = \sum_{m \in M(X)} d_m m$, where $d_m \in \mathbf{K}$. Then,

$$\forall P_a, P_b \in \mathbf{K}[X] \colon P_a \cdot P_b = (\sum_{m \in M(X)} a_m m) \cdot (\sum_{n \in M(X)} b_n n) = \sum_{m,n \in M(X)} (a_m b_n)(mn)$$

Note that the definition for the coefficient k_m essentially decides whether a monomial $g \in M(X)$ appears in a polynomial. If a polynomial P does not contain the monomial g, then it must be that $k_g = \mathbf{0} \in \mathbf{K}$.

Evaluation of a polynomial P at Z is denoted $P(Z)$, where $Z \subset \mathbf{R}$ is a set of elements from a ring $\mathbf{R}$ containing $\mathbf{K}$. Evaluating at Z, i.e., assigning indeterminates X to Z, defines an element from the ring $\mathbf{R}$, as $P(Z) \in \mathbf{R}$. We may note the unevaluated polynomial as $P(X)$ to highlight the set of indeterminates. A polynomial is identical to $\mathbf{0}$, or *identically evaluates* to $\mathbf{0}$, if $\forall W \in \mathbf{R} \colon P(W) = \mathbf{0}$.

AM: I think Z must be a map $X \to \mathbf{R}$

In this thesis, we mainly use elements of *group algebras* for evaluation, i.e., $\mathbf{R} = \mathbf{K}[\mathbf{G}]$, where $\mathbf{G}$ is a multiplicative group.

Group algebras. A group algebra is an algebraic structure very similar to a polynomial ring, and is also noted as $\mathbf{R}[\mathbf{G}]$, where $\mathbf{R}$ is a commutative ring and $\mathbf{G}$ is a multiplicative group. A group algebra may be thought of as a polynomial ring $\mathbf{R}[X]$, where the indeterminates X form a multiplicative group $\mathbf{G}$. The elements and operations in a group algebra have the same form as those in a polynomial ring, with the exception that instead of summing over monomials in X, we sum over the elements of $\mathbf{G}$.

Note that $\mathbf{0} \in \mathbf{R}[\mathbf{G}]$ corresponds to a polynomial $P \in \mathbf{R}[\mathbf{G}]$ with zero-coefficients, and $\mathbf{1} \in \mathbf{R}[\mathbf{G}]$ corresponds to the identity element of $\mathbf{G}$.

3.2 Arithmetic circuits

Computing arithmetic expressions such as polynomial evaluations can be represented by *arithmetic circuits*, that is, connected acyclic directed graphs, where terminal vertices correspond to initial and output values, and other vertices correspond to *arithmetic gates*, i.e., arithmetic operations.

In this thesis, we only consider *addition* and *multiplication* gates. These gates correspond to the binary operations of the underlying algebra which the arithmetic expression is evaluated over and given initial values from. Thus, the addition and multiplication gates may have *fan-in two* and *fan-out one*, i.e., each gate-vertex has three edges connected to it: two as an input, and one as an output. If the underlying algebra is commutative, we may call the arithmetic circuit a *commutative circuit*.

For example, an abstract evaluation of a polynomial P in $\mathbb{Z}[\{a, b, c, d\}]$ of the form $P = (ab + cd)^2$ is given by the arithmetic circuit presented in Figure 1. Note that we use the same symbols for addition and multiplication gates as seen in the figure in all arithmetic circuit diagrams in this thesis.

AM: Caption should be below image

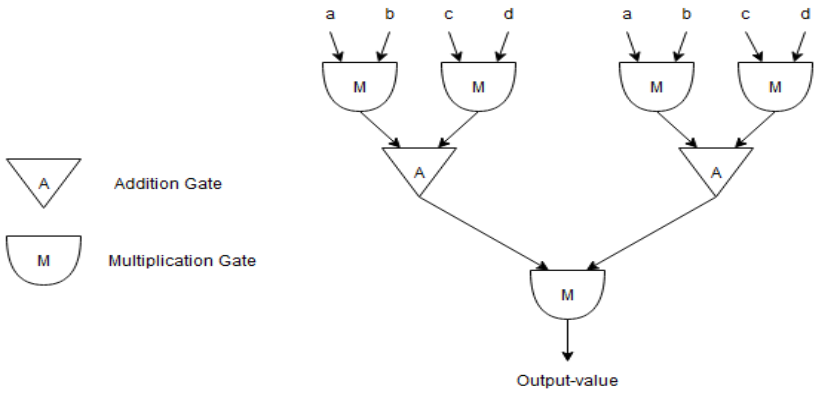

Figure 1: A circuit for evaluating $(ab + cd)^2$. Note that the input terminals have been copied for clarity.

3.3 Notation and other terminology

The following table holds useful notation and their definitions. The terms and notation are given on the left, and their definitions on the right.

AM: The `longtable` seems useful here

k-multilinear monomial	a multilinear monomial that has a degree of k
sum of monomials form	an expression for a polynomial $P(X)$ where there are no polynomial multiplications present, i.e., the expression consists of only additions between monomials
$\mathcal{O}^*$	hides factors polynomial to the input size, e.g., $\mathcal{O}(n^3k^n) = \mathcal{O}^*(k^n)$
FPT	A class of *fixed parameter tractable* problems, i.e., parameterized problems that can be solved in polynomial time if the parameter is fixed
$poly(n)$	a polynomial function in n, e.g., an^4+bn+c
Deterministic algorithm	Given a fixed input, a *deterministic* algorithm always gives the same output
Randomized algorithm	A *randomized* algorithm relies on pseudorandom functions, and may not produce the same outputs for a fixed input
Schwartz-Zippel lemma	TODO: give here or in the context?
term	def

Schwartz-Zippel ref:

J. T. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," Journal of the ACM, vol. 27, no. 4, pp. 701–717, 1980.

AM: Drop example for polynomial in n

AM: No need to define deterministic / randomized algorithm

AM: S-Z lemma in text seems better

AM: k-multilinear monomial is better defined before in context

4 General multilinear monomial detection

The detection of multilinear monomials in a multivariate polynomial is a fundamental problem, since many important problems can be reduced to it, as seen in Section 2. Therefore, any progress in general multilinear monomial detection implies faster algorithms for the problems mentioned in Section 2.

In this section, we discuss the algebraic ideas by Koutis and Williams [Kou08; Wil09; KW09] for the k-multilinear monomial detection. In Section 4.1, we dive into the technique of *algebraic fingerprinting*, and see the ideas and implementations of Koutis and Williams. In Section 4.2 we briefly discuss the time and space complexity of the general algebraic framework given by algebraic fingerprinting. Finally, Section 4.3 discusses the limits of algebraic fingerprinting.

As a prelude for the following sections, recall the relevant problem definition:

k-MULTILINEAR MONOMIAL DETECTION

Input: A commutative arithmetic circuit A representing a polynomial $P(X)$.

Question: Does the polynomial $P(X)$ extended as a sum of monomials contain a multilinear monomial of degree k?

4.1 Algebraic fingerprinting

In Section 1.3, we discussed non-algebraic methods for approaching k-multilinear monomial detection. In particular, we mention an interesting idea of discarding squared variables when dynamically computing the circuit. From this, we build towards algebraic fingerprinting.

This idea of discarding squared variables as soon as they are formed in A can be expressed mathematically: any squared variable should be identical to zero.

$$\forall x \in X : x^2 = \mathbf{0} \tag{2}$$

This implies that any non-multilinear monomial in X will evaluate to zero in $P(X)$, since any non-multilinear monomial q, assuming commutativity, can be written as x^2q' for some $x \in X$, and $x^2q' = \mathbf{0}q' = \mathbf{0}$ over any algebra. Therefore, $P(X)$ will identically evaluate to zero if there are no multilinear monomials, i.e., there exist no solutions to the original problem.

This is the basis of algebraic multilinear monomial detection introduced by Koutis [Kou08], or later referred to as *algebraic fingerprinting* [KW15]: we evaluate $P(X)$ over some algebraic structure $\mathbf{G}$, and detect a multilinear monomial from the value returned by this

evaluation. Ideally, with any evaluation of $P(X)$ at any $Z \subset \mathbf{G}, |Z| = |X|$ where $w \neq \mathbf{0}$,

$$P(Z) = \begin{cases} \mathbf{0} & \text{if no multilinear monomials exist} \\ w & \text{otherwise} \end{cases} \tag{3}$$

In the following subsections, we specify an appropriate algebraic structure such that (3) is met, as well as some requirements for efficiency. Then, we discuss the works of Koutis and Williams [Kou08; Wil09], and see how these specifications were implemented.

This thesis refers to the general framework provided by this algebraic technique for parameterized problems [KW09; KW15] as algebraic fingerprinting. However, algebraic fingerprinting can also be used to refer specifically to the idea behind solving a problem with multilinear monomials canceling out due to characteristic, which is discussed in Section 4.1.3.

4.1.1 Specifications for the algebraic structure

We have arrived at an important task for multilinear monomial detection: find an appropriate algebraic structure $\mathbf{G}$ for the evaluation of P over $\mathbf{G}$ to meet (2) and thus the first equality in (3). We specify for commutative algebraic structures such as commutative rings for multilinear monomial detection; $\mathbf{G}$ should have commutative multiplication in order for (2) to be effective. The ordering of indeterminates in monomials is generally unknown, since the specific algebrization into multilinear monomial detection is abstracted away.

For the second equality in (3), it is necessary that multilinear monomials in $P(X)$ are not identical to $\mathbf{0}$ over $\mathbf{G}$. More specifically, w should not be identical to $\mathbf{0}$. Thus, it must be that $k \leq |\mathbf{G}|$, where k is the degree of the monomials. Although this is not enough to ensure that $w \neq \mathbf{0}$ (as seen in the following section), we leave the issue for now.

These are the necessary specifications for $\mathbf{G}$ for algebraic multilinear monomial detection. However, this algebraic detection must be efficient for it to be useful. Therefore, further requirements are necessary: (a) the binary operations of $\mathbf{G}$ must be efficiently computable for a fast evaluation of $P(X)$, and (b) multilinear monomials in $P(X)$ must survive the evaluation of P under an assignment $\gamma\colon X \to \mathbf{G}$ with a constant probability. We reason the sudden appearance of *survivability* and probabilities soon.

Recall from Section 1.3 that with color coding, multilinear monomials can be detected with a randomized algorithm in $\mathcal{O}^*((2e)^k)$ time. Thus for (a), we may specify for an evaluation of $P(X)$ to take $\mathcal{O}^*(2^k)$ time. The polynomial $P(X)$ will have at most 2^k multilinear monomials. Therefore, it is necessary that the binary operations of $\mathbf{G}$ take polynomial time.

With (b), recall that multilinear monomials survive color coding with probability e^{-k}. With algebraic fingerprinting, although not identical to zero, a multilinear monomial in $P(X)$ can still evaluate to $\mathbf{0} \in \mathbf{G}$ under some assignment $\gamma\colon X \to \mathbf{G}$. However, if we specify for a constant probability of survival under random γ, we can reliably decide whether a multilinear monomial exists by evaluating $P(X)$ in $\mathbf{G}$ over a constant number of randomized assignments $X \to \mathbf{G}$. Relating to color coding, this would essentially remove the e^k factor in $\mathcal{O}^*((2e)^k)$.

To restate, we now have to find such a commutative ring $\mathbf{G}$ that meets the following specifications:

- $\forall g \in \mathbf{G}\colon g^2 = \mathbf{0}$
- Operating over $\mathbf{G}$ should be fast, i.e., evaluating $P(X)$ should take $\mathcal{O}^*(2^k)$ time.
- Multilinear monomials should evaluate to non-zero through a random assignment $\gamma\colon X \to \mathbf{G}$ with a reasonable constant probability.

In the work that introduced this algebraic fingerprinting technique [Kou08], Koutis used the group algebra $\mathbb{Z}_2[\mathbb{Z}_2^k]$. Williams developed the technique further, generalizing the algebra to $GF(2^l)[\mathbb{Z}_2^k]$ [Wil09]. Next, we look at these group algebras of $\mathbb{Z}_2^k$ for $\mathbf{G}$.

4.1.2 Using group algebras of $\mathbb{Z}_2^k$

The multiplicative group $\mathbb{Z}_2^k$ consists of k-dimensional {0,1}-vectors with the binary operation defined as component-wise addition modulo 2. For example with $k = 3$,

$$\begin{bmatrix}0\\0\\1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix} = \mathbf{0} \in \mathbb{Z}_2^3.$$

Observe that in general, every element in $\mathbb{Z}_2^k$ is its own inverse:

$$\forall z \in \mathbb{Z}_2^k\colon z^2 = \mathbf{0}. \tag{4}$$

Recall that the elements of a group algebra $\mathbf{F}[\mathbb{Z}_2^k]$ are linear combinations of the form

$$\sum_{v \in \mathbb{Z}_2^k} a_v v,$$

where $a_v \in \mathbf{F}$. From here on, we note the identity of $\mathbb{Z}_2^k$ as v_0, additive and multiplicative identities of $\mathbf{F}$ as $\mathbf{0}_F$ and $\mathbf{1}_F$, respectively, and use $\mathbf{0}$ and $\mathbf{1}$ for the identities of $\mathbf{F}[\mathbb{Z}_2^k]$. Recall that $\mathbf{1} = v_0$, and $\mathbf{0}$ corresponds to the element $\sum_{v \in \mathbb{Z}_2^k} a_v v$, where $a_v = \mathbf{0}_F$. Note

that + represents addition over $\mathbf{F}[\mathbb{Z}_2^k]$ and not addition over $\mathbb{Z}_2^k$ which is a multiplicative group.

In his introductory work for this technique [Kou08], Koutis assigned X to elements of the form $(v_0 + v_i) \in \mathbf{F}[\mathbb{Z}_2^k]$, such that for every $x_i \in X$, a random $v_i \in \mathbb{Z}_2^k$ is independently and uniformly picked for the assignment $\gamma(x_i) = v_0 + v_i$. We note the assigned values as $\overline{X}$, and the resulting element from the polynomial evaluation as $P(\overline{X}) \in \mathbf{F}[\mathbb{Z}_2^k]$. Koutis observed that due to (4), for all $v \in \mathbb{Z}_2^k$ and $(v_0 + v) \in \mathbf{F}[\mathbb{Z}_2^k]$,

$$(v_0 + v)^2 = v_0^2 + v_0 v + v v_0 + v^2 = v_0 + v + v + v_0 = 2v_0 + 2v.$$

This implies that if we pick a field with characteristic 2 for $\mathbf{F}$, $\forall v_i \in \mathbb{Z}_2^k \colon (v_0 + v_i)^2 = \mathbf{0}$. Thus, non-multilinear monomials in $P(X)$ vanish in $P(\overline{X})$, and the first equation in (3) will hold.

However, if $\mathbf{F}$ has characteristic 2, the second equation of (3) does not necessarily hold, and it may be that $w = \mathbf{0}$. Multilinear monomials may cancel each other out in $\mathbf{F}[Z_2^k]$, since they may have even leading coefficients in $P(X)$. Next, we see how Koutis approached this problem.

4.1.3 Fingerprints to prevent unwanted cancelation

Indeed, if we take the k-3D matching instance from Section 1.2, we notice that the multilinear monomials in P_2 have even coefficients, and thus would cancel out in $\mathbf{F}[Z_2^k]$ due to characteristic.

$$\begin{aligned} P_2 &= (a_1 b_1 c_1 + a_1 b_2 c_2 + a_2 b_2 c_3)^2 \\ &= a_1^2 b_1^2 c_1^2 + a_1^2 b_2^2 c_2^2 + a_2^2 b_2^2 c_3^2 + 2a_1^2 b_1 b_2 c_1 c_2 + 2a_1 a_2 b_1 b_2 c_1 c_3 + 2a_1 a_2 b_2^2 c_2 c_3. \end{aligned}$$

In general, when k is even, multilinear monomials will have even coefficients.

To tackle this issue, one idea is to add auxiliary indeterminates, called *fingerprints* [KW15], to the monomials in order to make them unique. For example, let $S = \{s_1, s_2, \ldots\}$ be the set of an appropriate number of fingerprints and augment them to P_k as follows:

AM: Actually $S = \{s_1, \ldots, s_6\}$?

AM: Use `\align`

$$\begin{aligned} P_2' &= (s_1 a_1 b_1 c_1 + s_2 a_1 b_2 c_2 + s_3 a_2 b_2 c_3)(s_4 a_1 b_1 c_1 + s_5 a_1 b_2 c_2 + s_6 a_2 b_2 c_3) \\ &= s_1 s_4 a_1^2 b_1^2 c_1^2 + s_2 s_5 a_1^2 b_2^2 c_2^2 + s_3 s_6 a_2^2 b_2^2 c_3^2 + \\ &\quad s_1 s_5 a_1^2 b_1 b_2 c_1 c_2 + s_2 s_4 a_1^2 b_1 b_2 c_1 c_2 + s_1 s_6 a_1 a_2 b_1 b_2 c_1 c_3 + \\ &\quad s_3 s_4 a_1 a_2 b_1 b_2 c_1 c_3 + s_2 s_6 a_1 a_2 b_2^2 c_2 c_3 + s_3 s_5 a_1 a_2 b_2^2 c_2 c_3. \end{aligned}$$

Then, the multilinear monomial detection algorithm could assign every $a \in X \cup S$ to $\mathbf{F}[\mathbb{Z}_2^k]$.

AM: Now the multilinear monomial $a_1 a_2 b_1 b_2 c_1 c_3$ only cancels itself out in $\rho(P_2')$ ($P_2'(Z)$?

unsure about notation) if $\rho(s_1)\rho(s_6) = \rho(s_3)\rho(s_4)$ under the assignment ρ .... We can control depending on how the $\rho(s_i)$ are chosen... With this, non-multilinear monomials will still vanish, but multilinear monomials will not identically cancel each other out when k is even.

However, introducing new indeterminates raises the degree of multilinear monomials. Therefore, it increases the probability that indeterminates in a multilinear monomial are assigned the same value from $\mathbf{F}[Z_2^k]$, which results in the multilinear monomial evaluating to zero with higher probability. Counteractively raising the dimension of $\mathbb{Z}_2^k$, however, would exponentially slow down matrix multiplications which prove to be important for efficiency in algebraic fingerprinting (see Section 4.2). AM: ??

Koutis approached this problem by assigning fingerprints to elements from a different algebraic structure: he used $S \to \mathbb{Z}_2$ and set $\mathbf{F} = \mathbb{Z}_2$ [Kou08]. In this sense, fingerprints are defined to be auxiliary coefficients instead of auxiliary indeterminates. Note that $\mathbb{Z}_2$ has characteristic 2. With this, Koutis essentially assigns multilinear monomials a coefficient randomly from $\{\mathbf{0}_F, \mathbf{1}_F\}$. The idea is that a multilinear monomial, assigned with the fingerprint $\mathbf{1}_F$, survives the cancelation due to characteristic if the canceling pair is assigned $\mathbf{0}_F$. With randomized assignments $S \to \mathbb{Z}_2$, there is a constant AM: positive probability that a multilinear monomial will have an odd coefficient, and thus survive the assignment [Kou08].

In practice, fingerprints can be implemented into the algebrization as follows (see also Figure 2): for every multiplication gate G_i feeding from the terminals in the arithmetic circuit A for $P(X)$, pick a unique $s_i \in S$. Insert a new multiplication gate $\overline{G_i}$ that takes as input s_i and the output of G_i. The output of $\overline{G_i}$ feeds to the gates that read the output of G_i. We note the new polynomial represented by this circuit as $P'(X, S)$. Note that here, Koutis picked random elements from $\mathbb{Z}_2$ for s_i. AM: inputs

4.1.4 Polynomial identity testing

From another perspective, we may look at algebraic fingerprinting as *polynomial identity testing*. That is, we compute $P'(X, S)$ over assignments $X \to \mathbf{F}[\mathbb{Z}_2^k]$ into $P'(\overline{X}, S)$, i.e., compute until the gates $\overline{G_i}$. Imagine we stop here in the circuit before assigning the fingerprints S and continuing with the multiplication. Currently, we may see $P'(\overline{X}, S) \in \mathbf{K}[S]$ as a polynomial with indeterminates from S and coefficients from $\mathbf{K} = \mathbf{F}[\mathbb{Z}_2^k]$. In a sense, we switch the roles of coefficients and indeterminates in $P'(X, S)$ to find a new polynomial $Q(S) \in \mathbf{K}[S]$.

Now, deciding whether a multilinear monomial exists in $P(X)$ is essentially given by whether the polynomial $P'(\overline{X}, S) \in \mathbf{K}[S]$ is identical to $\mathbf{0}$, i.e., with Φ representing the

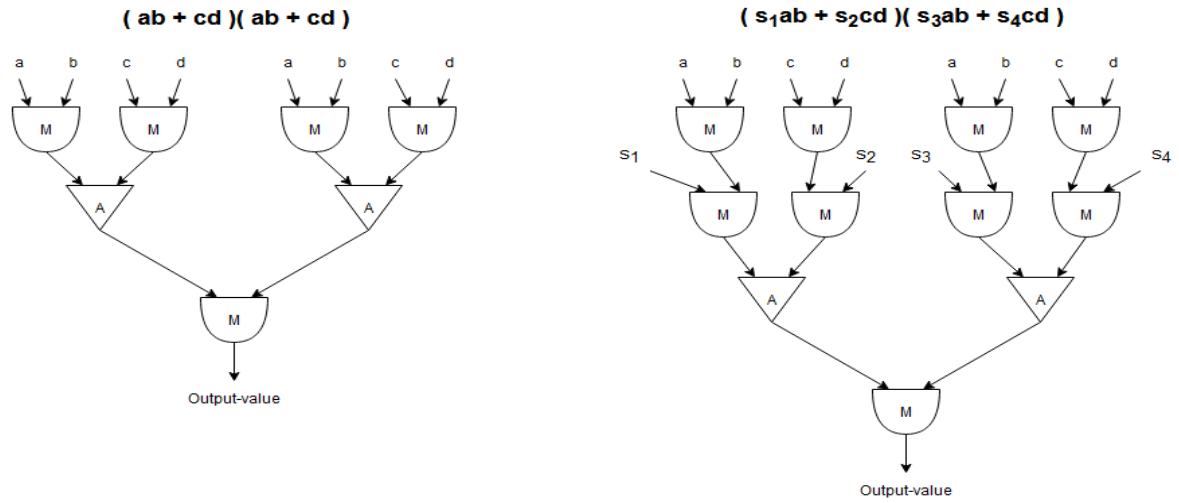

Figure 2: A fingerprinted circuit for evaluating $(ab + cd)^2$. Note the added terminals on the right circuit.

family of mappings $S \to \mathbf{F}$ and $\phi(S)$ representing the assigned values of S with ϕ,

$$\forall \phi \in \Phi \colon P'(\overline{X}, \phi(S)) = \mathbf{0}.$$

Therefore, we formulate algebraic fingerprinting as polynomial identity testing, i.e., we test wheter $P'(\overline{X}, S) \in \mathbf{K}[S]$ is identical to $\mathbf{0}$.

Polynomial identity testing

Input: An arithmetic circuit C that computes the polynomial $Q(S)$.

Question: Is $Q(S)$ identical to the zero polynomial?

We frame $P'(\overline{X}, S)$ as $Q(S) \in \mathbf{K}[S]$. Here, it is necessary to test whether $Q(S)$ is identical to *zero modulo 2*, since we used characteristic 2 to eliminate the underlying non-multilinear monomials in $P'(\overline{X}, S)$.

Thus, multilinear monomial detection is reduced via algebraic fingerprinting to polynomial identity testing. A multilinear monomial is detected if $Q(f) \neq \mathbf{0}_F \in \mathbf{F}$, where $Q(f)$ represents evaluation under an assignment $S \to \mathbf{F}$, with any field $\mathbf{F}$ of characteristic 2. We note the family of assignments $S \to \mathbf{F}$ as Φ.

Assume multilinear monomials exist in $P(X)$. Koutis achieved to detect a multilinear monomial with a probability $(1/4 + 1/4k)$ by testing whether $Q(S) = \mathbf{0}$ over the field $\mathbf{F} = \mathbb{Z}_2$ with randomized assignments $S \to \mathbb{Z}_2$ [Kou08]. Williams developed the technique of algebraic fingerprinting further by observing that due to the Schwartz-Zippel lemma, if we raise the order of the field $\mathbf{F}$ such that the number of different assignments in Φ is much larger than the number of assignments $\phi \in \Phi$ that have $Q(\phi(S)) = \mathbf{0}_F$, $Q(\delta(S)) \neq \mathbf{0}_F$ with a high probability over some $\delta \in \Phi$ [Wil09].

Of course, $\mathbf{F}$ must have characteristic 2. For this, Williams used the field $GF(2^l)$ with $l \in \mathbb{N}$. [Wil09]. By Schwartz-Zippel lemma, $Q(S)$ evaluates to $\mathbf{0}_F$ over a random assignment $S \to GF(2^l)$ with probability at most $k/2^l$ [Wil09]. For example with k-path, Williams used $l = 3 + \log_2 k$, which sets the probability of a false-negative evaluation to at most $1/2^3$. For the k-path problem, Koutis gave a randomized $\mathcal{O}^*(2^{3k/2})$ time algorithm [Kou08], and Williams developed this into a randomized $\mathcal{O}^*(2^k)$ time algorithm [Wil09] with the ideas discussed here.

In the following section, we briefly touch on the complexities of the algebras given here.

4.2 Time and space complexity

In the previous section, we discussed the specifications for an algebraic stucture to implement algebraic fingerprinting. For an appropriate algebra, we found $GF(2^l)[Z_2^k]$. However, we did not discuss the costs related to this algebra. In this section, we briefly discuss the time and space complexity of algebraic fingerprinting.

Define the evaluation of $P'(X, S)$ at any X and S to take $g(n)$ number of arithmetic operations, where $n = |X|$. In general, detecting k-degree multilinear monomials in $P(X)$ takes $\mathcal{O}^*(2^k g)$ time and $\mathcal{O}(poly(n, k))$ space [Wil09].

Actually, the complexities given by Koutis [Kou08] and Williams [Wil09] did not directly come from $\mathbb{Z}_2[Z_2^k]$ or $GF(2^l)[Z_2^k]$. To make the complexity claims in [Kou08], Koutis used a well-known [Ter99] isomorphism $\kappa\colon \mathbb{Z}[\mathbb{Z}_2^k] \to \mathcal{M}$, where $\mathcal{M} = \mathbf{M}^{2^k \times 2^k}$ is an algebra of special permutation matrices with binary entries. To translate $\mathbb{Z}[\mathbb{Z}_2^k]$ to $\mathbb{Z}_2[\mathbb{Z}_2^k]$, it is enough to take the modulo 2 of the coefficients a_v in the elements of $\mathbb{Z}[\mathbb{Z}_2^k]$:

$$\sum_{v \in \mathbb{Z}_2^k} a_v v \in \mathbb{Z}[\mathbb{Z}_2^k] \implies \sum_{v \in \mathbb{Z}_2^k} (a_v \bmod 2) v \in \mathbb{Z}_2[\mathbb{Z}_2^k]$$

AM: No need for displaystyle

AM: Use \bmod

Since Koutis used an isomorphism, all the ideas given in Section 4.1 still hold for detecting multilinear monomials. That is, the given implementation for (3) is valid through κ.

Using these matrices, Koutis [Kou08] detected k-multilinear monomials by computing the trace of the matrix resulting from the evaluation of $P_S(X)$ over $\mathcal{M}$. The trace can be computed in time $\mathcal{O}(2^k(nk + t))$ and space $\mathcal{O}(nk + s)$, where t and s are the time and space complexity of the $g(n)$ arithmetic operations, respectively.

OM: Maybe don't go further than this

AM: I would rather expand in Sect. 5 than here

OM: add to prelims?

For $GF(2^l)[Z_2^k]$, the same ideas are used, though we omit further discussion into this for the sake of brevity. Thus, we have a general $\mathcal{O}^*(2^k g)$ time and $\mathcal{O}(poly(n, k))$ space algorithm for detecting k-degree multilinear monomials in $P(X)$, where g is the number of gates in the arithmetic circuit for $P(X)$ and $n = |X|$ [KW09].

It is important to note here that this complexity is for k-degree multilinear monomial

detection. Whether a parameterized problem with a parameter k receives an $\mathcal{O}^*(2^k)$ time algorithm, is determined by the algebrization of said parameterized problem. For example for k-path, algebraic fingerprinting gives an $\mathcal{O}^*(2^k)$ time algorithm [Wil09]. For k-3D-matching however, as given in Section 1.2, algebraic fingerprinting gives an $\mathcal{O}^*(2^{3k})$ algorithm, since k-3D-matching reduces into $3k$-degree multilinear monomial detection.

4.3 Limits of algebraic fingerprinting

In this section, we discuss some limiting factors to the general algebraic framework given by algebraic fingerprinting for parameterized problems. First, we discuss the lower bound for the runtime. Then, we discuss the difficulties for derandomization. Finally, we touch on AM: the issue of finding a solution for the underlying combinatorial problem whereas algebraic fingerprinting only detects one AM: that one exists.

4.3.1 Algebraic optimization

In Section 4.2, it was established that k-multilinear monomials in $P(X)$ are detected with algebraic fingerprinting in $\mathcal{O}^*(2^k g)$ time, where g is the number of gates in the arithmetic circuit for $P(X)$ [Wil09]. This time was achieved by evaluating the circuit over matrix representations of the algebra $GF(2^l)[\mathbb{Z}_2^k]$.

Koutis and Williams showed that this particular algebra is nearly optimal for time complexity [KW09]: there is no significantly faster algebra that is appropriate for the general k-multilinear monomial detection. The authors proved that if any commutative algebra $\mathcal{G}$ is used for detecting k-multilinear monomials, the lengths of elements in $\mathcal{G}$ must be $\Omega(2^k/k)$.

Thus for the general k-multilinear monomial detection, we may say that algebraic fingerprinting is in some sense optimized. However, ideas from algebraic fingerprints can be utilized for faster algorithms for specific k-multilinear monomial detection instances (see Section 5).

4.3.2 Derandomization

Algebraic fingerprinting gives a randomized algorithm for k-multilinear monomial detection. The derandomization of the general framework seems hard, since algebraic fingerprinting uses the fact that polynomial identity testing can be solved in polynomial time with a randomized algorithm [Wil09].

If algebraic fingerprinting can be derandomized in polynomial time, it implies that polynomial identity testing can be solved with a polynomial time deterministic algorithm.

This would imply strong circuit lower bounds TODO: this is apparently significant

AM: f this gets into too hairy complexity theory territory, just say it would imply "surprising results" in complexity theory. People usually can imagine what that means...

4.3.3 Finding the solution

In many instances of multilinear monomial detection, a solution to the underlying problem can be directly found from the multilinear monomials by reverse engineering the algebrization. However in algebraic fingerprinting, we do not have this information on the monomials: when we assign the variables values and evaluate the polynomial, we may only detect the presence of a solution. As such, algebraic fingerprinting applies to decision algorithms.

AM: reverse-engineering

However, finding a solution when its presence is detected seems easy. Moreover, since most of the problems in Section 2 have search problems in FPT, a solution is found in polynomial time when its existence is detected.

AM: ??

For example, in [Kou08], [Kou08] gives an algorithm that detects a k-path, and shows that a k-path can be found by $\mathcal{O}^*(n + min(k^2, m))$ applications of the given algorithm. In general for subgraph problems, we may find a solution by calling the decision algorithm a polynomial number of times for different subgraphs of the original input graph.

AM: What are n and m here? (I can imagine, but some other reader probably won't)

5 Improving from algebraic fingerprinting

AM: dropped clearpage

As mentioned in Section 4.3.1, algebraic fingerprinting as a framework for k-multilinear monomial detection has a lower bound of $\Omega(2^k)$. However, this framework is very generic since it approaches the abstracted multilinear monomial detection without utilizing the combinatorial properties specific to the underlying problem. The idea of adding auxiliary fingerprint variables in the algebrization, though, has potential for these properties.

Indeed, faster algorithms for specific combinatorial problems have been found by designing new algebrizations with techniques similar to algebraic fingerprinting, where the fingerprints are designed to make use of the underlying combinatorial properties. In Section 5.1, we see how Björklund [Bjö14], exploited the cancelation due to characteristic to cancel non-solution monomials by clever design of fingerprints.

Section 5.2 briefly overviews other ideas or improvements in utilizing the general framework of algebraic fingerprints.

5.1 Fingerprinting for cancelation of non-solutions

TODO: go over Björklund et al. for k-path or Hamiltonicity, managed to design fingerprints such that non-solutions cancel

AM: We already have plenty of content. Either cut some content before or keep things here very brief!

In the algebraic framework by Koutis and Williams, fingerprints prevent the cancelation of multilinear monomials. Attacking the Hamiltonian path problem, however, Björklund designed fingerprints such that non-multilinear monomials, i.e. non-solution terms, cancel due to characteristic, while the multilinear terms remain with constant probability [Bjö14]. This resulted in the current fastest algorithm for undirected Hamiltonicity, running in $\mathcal{O}^*(2^n)$ time.

Before discussing the algebrization and the fingerprints, we define the Hamiltonian path problem.

Hamiltonian path (Hamiltonicity)
Input: A directed graph $G = (V, E)$.
Question: Does G contain a simple path that visits every vertex?

TODO: give the algebrization and/or show how it works

OM: Probably really brief

5.2 Other research in improving algebraic fingerprinting

- deterministic multilinear detection (partial differentials of polynomials)

As briefly discussed in Section 4.3.2, algebraic fingerprinting gives randomized algorithms, and the derandomizing the technique is hard. Since the runtime of algebraic fingerprinting is arguably optimized for the general multilinear monomial detection, much effort has been directed into matching the deterministic runtime to the randomized one, although there still is a significant gap between them.

The state of art deterministic algorithm uses the partial differentials of the polynomial, though a combinatorial set representation approach has been proven as equivalent.

- constrained multilinear detection (weighted problems?), temporal graphs

On another front, there has been research for applying algebraic fingerprints to different types of multilinear monomial detection problems. Namely, *constrained* multilinear monomial detection detects a multilinear monomial with a specific constraint, i.e., a cost or a counter. Moreover, multilinear monomial detection has been extended to temporal graphs (constraints).

- distributed multilinear detection (MIDAS-paper)

6 Conclusion

From Section 2, it can be seen that a fast algorithm for k-multilinear monomial detection gives a fast algorithm for many parameterized combinatorial problems. We found that algebraic fingerprinting solves this problem by evaluating the given multivariate polynomial over a specific algebra. Thus, we may say that algebraic fingerprinting gives a general framework for parameterized combinatorial problems.

In Section 4.1, we discussed the idea behind algebraic fingerprinting: generate a polynomial $P(X)$ from the algebrization of a combinatorial problem, and evaluate $P(X)$ over $GF(2^l)[Z_2^k]$ with randomized assignments $X \to GF(2^l)[Z_2^k]$ of form $x_i \to (v_0 + v_i)$, augmented with scalar multiplications by elements randomly chosen from $GF(2^l) \setminus \{\mathbf{0}\}$. This gives a randomized algorithm for k-multilinear monomial detection that runs in $\mathcal{O}^*(2^k)$ time.

Moreover, as seen in Section 5.1, the technique of algebraic fingerprints can be utilized for even faster algorithms. In particular, a near fifty-year stagnation in improvements for the well-studied Hamiltonian path problem saw a significant development by Björklund right after the novel algebraic technique [Kou08] saw publication.

TODO: conclude

AM: What about further work? What do people intend to work on in the near future?

References

[AYZ95] Noga Alon, Raphael Yuster, and Uri Zwick. "Color-Coding". In: *J. ACM* 42.4 (July 1995), pp. 844–856. ISSN: 0004-5411. DOI: 10.1145/210332.210337. URL: https://doi.org/10.1145/210332.210337.

[Bel62] Richard Bellman. "Dynamic programming treatment of the travelling salesman problem". In: *Journal of the ACM (JACM)* 9.1 (1962), pp. 61–63.

[Bjö14] Andreas Björklund. "Determinant Sums for Undirected Hamiltonicity". In: *SIAM Journal on Computing* 43.1 (2014), pp. 280–299. DOI: 10.1137/110839229. eprint: https://doi.org/10.1137/110839229. URL: https://doi.org/10.1137/110839229.

[BKK16] Andreas Björklund, Petteri Kaski, and Lukasz Kowalik. "Constrained Multilinear Detection and Generalized Graph Motifs". In: *Algorithmica* 74.2 (2016), pp. 947–967. DOI: 10.1007/s00453-015-9981-1. URL: https://doi.org/10.1007/s00453-015-9981-1.

[Bjö+17] Andreas Björklund et al. "Narrow sieves for parameterized paths and packings". English. In: *Journal of Computer and System Sciences* 87.C (2017), pp. 119–139. DOI: 10.1016/j.jcss.2017.03.003.

[CEV17] Jose Cadena, Saliya Ekanayake, and Anil Vullikanti. "Fast graph scan statistics optimization using algebraic fingerprints". In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE. 2017, pp. 905–910.

[Che+07] Jianer Chen et al. "Improved Algorithms for Path, Matching, and Packing Problems". In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2007, pp. 298–307. ISBN: 9780898716245.

[HK62] Michael Held and Richard M. Karp. "A Dynamic Programming Approach to Sequencing Problems". In: *Journal of the Society for Industrial and Applied Mathematics* 10.1 (1962), pp. 196–210. DOI: 10.1137/0110015. eprint: https://doi.org/10.1137/0110015. URL: https://doi.org/10.1137/0110015.

[Kou05] Ioannis Koutis. "A faster parameterized algorithm for set packing". In: *Information Processing Letters* 94.1 (2005), pp. 7–9. ISSN: 0020-0190. DOI: https://doi.org/10.1016/j.ipl.2004.12.005. URL: https://www.sciencedirect.com/science/article/pii/S0020019004003655.

AM: No need for URL if you have DOIs, also no need for ISSNs

AM: You should use math style in the title of [Wil09]

AM: use dblp.org

[Kou08] Ioannis Koutis. "Faster Algebraic Algorithms for Path and Packing Problems". In: *Automata, Languages and Programming*. Ed. by Luca Aceto et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 575–586. ISBN: 978-3-540-70575-8.

[KW09] Ioannis Koutis and Ryan Williams. "Limits and Applications of Group Algebras for Parameterized Problems". In: *Automata, Languages and Programming*. Ed. by Susanne Albers et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 653–664. ISBN: 978-3-642-02927-1.

[KW15] Ioannis Koutis and Ryan Williams. "Algebraic Fingerprints for Faster Algorithms". In: *Commun. ACM* 59.1 (Dec. 2015), pp. 98–105. ISSN: 0001-0782. DOI: 10.1145/2742544. URL: https://doi.org/10.1145/2742544.

[Ter99] Audrey Terras. *Fourier analysis on finite groups and applications*. 43. Cambridge University Press, 1999.

[Val92] Leslie G Valiant. "Why is Boolean complexity theory difficult". In: *Boolean Function Complexity* 169.84-94 (1992), p. 4.

[Wil09] Ryan Williams. "Finding paths of length k in $O(2^k)$ time". In: *Information Processing Letters* 109.6 (2009), pp. 315–318. ISSN: 0020-0190. DOI: https://doi.org/10.1016/j.ipl.2008.11.004. URL: https://www.sciencedirect.com/science/article/pii/S0020019008003396.