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

Detecting multilinear monomials with algebraic fingerprints

Bachelor's Thesis

April 22, 2023

Onni Miettinen

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

ABSTRACT OF
BACHELOR'S THESIS

Author:	Onni Miettinen
Title of thesis:	Detecting multilinear monomials with algebraic fingerprints
Date:	April 22, 2023
Pages:	**??**
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 $\mathcal{O}(2^k * poly(n))$, 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:	22. huhtikuuta 2023
Sivumäärä:	**??**
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 **6**

1.1 Research goals and thesis structure 6

1.2 Algebrization of combinatorial problems 7

1.3 Reducing k-3D matching to multilinear monomial detection 8

2 Related works **9**

3 Preliminaries **10**

3.1 Algebra 10

3.2 Arithmetic circuits 11

3.3 Notation and other terminology 11

4 General multilinear monomial detection **13**

4.1 Problem definition 13

4.2 Algebraic fingerprinting 14

4.2.1 Specifications for the algebra 15

4.2.2 Using group algebras of $\mathbb{Z}_2^k$ 16

4.2.3 Fingerprints to prevent unwanted cancelation 17

4.2.4 Polynomial identity testing 18

4.3 Time and space complexity 19

4.4 Algebraic fingerprinting as a framework for parameterized problems . . . 20

4.5 Limits of algebraic fingerprinting 20

4.5.1 There are no faster algebras 20

4.5.2 Derandomization seems difficult 21

4.6 Finding the solution 21

5 Improving algebraic fingerprinting **22**

5.1 Fingerprinting for cancelation of non-solutions 22

5.2 Parallelizing multilinear monomial detection 23

6 Conclusion **24**

References **25**

1 Introduction

In recent 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, i.e., finding whether a multivariate polynomial contains a multilinear monomial, first introduced by Koutis in [Kou05], where the set packing problem is reduced to multilinear monomial detection.

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

AM: What is the meaning of $\mathcal{O}^*()$?

Of course, this technique was further developed, and soon after in [Bjö14] Björklund et al. showed an algorithm AM: This technique was further developed by Björklund [Bjö14], who showed ... that solved the Hamiltonian problem (Hamiltonicity), i.e., finding whether a given graph contains a simple path that visits every vertex, in $\mathcal{O}^*(1.657^n)$ time. Soon enough, for k-path, an $\mathcal{O}^*(1.66^k)$ algorithm was found [Bjö+17]. The fastest algorithms for Hamiltonicity before this ran in $\mathcal{O}^*(2^n)$ and were known since 1962 [HK62], [Bel62]. This was a significant improvement on a problem that had seen no progress in nearly fifty years.

AM: Note use of `\citeauthor`
AM: Did Björklund really develop the technique further? or did he only show how to apply it to another problem?
AM: n or k?

1.1 Research goals and thesis structure

AM: This section can be shortened to one paragraph

The goals of this 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 problems that use multilinear monomial detection. Also, interesting ideas regarding algebraic fingerprints are explored for.

Multilinear monomial detection is a fundamental problem, since many important combinatorial problems can be reduced into it via a problem specific algebraization. Thus, faster algorithms and new ideas for multilinear monomial detection are important.

Multilinear monomial detection is essentially searching for solutions among non-solutions, both of which are encoded as monomials in a polynomial. The technique of algebraic fingerprinting is present in multilinear monomial detection. With algebraic fingerprints, unwanted cancelation of solution monomials due to the characteristic of a field can be prevented. Moreover, algebraic fingerprints can be used to cancel non-solution monomials by abusing the characteristic.

In the next subsections, the thesis discusses algebraization and reduction into multilinear monomial detection. The section 2 covers preliminaries. In section 3, the thesis discusses general multilinear monomial detection. In section 4, some problem specific instances of multilinear monomial detection are given, and clever utilizations of algebraic fingerprints are shown. Section 5 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 path of k vertices. The solutions and non-solutions (solution space) to combinatorial problems can be thought of as combinations of the given objects. The solution space for the k-path problem consists of combinations of k vertices and $k-1$ edges. A non-solution combination would contain duplicate vertices, or edges that contain vertices outside the combination.

AM: Probably one can drop the vertices and think only about edges?

AM: This sounds like a very simple problem... But I guess the catch is that we want an algorithm with complexity depending only on k?

Algebraization is reducing a given problem into an algebraic form, i.e. AM: that is, a question regarding some algebraic property of some algebraic entity. In an algebraization of a combinatorial problem, the algebraic entity can be constructed from algebraic elements defined from the set of objects given as an input. The motivation behind the construction is some algebraic property that, when satisfied, gives a solution to the problem.

In [Val92], it was observed that multivariate polynomials in certain algebras have natural combinatorial interpretations. Utilizing this idea, [Kou05] managed to reduce a combinatorial problem into an algebraic form, that is, multilinear monomial detection. First, we introduce multiple variables that correspond to elements from the set of objects given as input. Then, we construct an arithmetic circuit representing a multivariate polynomial, such that it encodes all solutions and non-solutions as multivariate monomials, with multilinear monomials corresponding to solutions. 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.

AM: Somewhere we need to reference the reader to Sect. 2 for non-familiar terms like multivariate, algebras, etc.

Appropriate definitions for the variables are problem-specific. In the following section, this thesis gives a reduction into multilinear monomial detection, shown in [KW15], for the k-3D matching problem. Another simple example can be found, for the set packing problem, in [Kou05].

AM: I would merge the two sections and use the k-3D matching as an example to the explanation above

1.3 Reducing k-3D matching to multilinear monomial detection

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 $T \subset A \times B \times C$.

Question: Is there a subset $M \subseteq T$, such that $|M| = k$ and $\forall m \in M$: None of the elements in m appear in $M \backslash \{m\}$ AM: $M \setminus \{m\}$

We begin by defining new variables 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 $k \in [|C|]$.

For every triple $t \in T$, we define a multilinear monomial x that is a product of the elements in t. We introduce a set X that satisfies the following:

$$\forall x \in X : x = abc : (a, b, c) \in T.$$

AM: Probably meant $\forall x \in X, x = abc...$?

Next, we define multivariate polynomials P_1 and P_k as follows:

$$P_1 = \sum_X , \; P_k = P_1^k.$$

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. Furthermore, every multilinear monomial in the expanded P_k corresponds to a solution to the problem, and the solutions can be directly found from the variables in the multilinear monomial. Thus, a successful reduction into multilinear monomial detection has been given for the k-3D mapping.

An example instance of k-3D matching with this exact algebraization can be found in [KW15]. TODO: show the example here

AM: Use `\setminus` instead of `\backslash`
AM: $T = A \times B \times C$ not allowed?
AM: Either convert $\forall$ into text or what follows into maths
AM: Def. of $|\cdot|$?
AM: Over what object are we doing multiplication?
AM: Use \[... \] for display math
AM: Meaning of $\sum_X$?
AM: Remove clearpage

2 Related works

The importance of multilinear monomial detection lies in the fact that many 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 been of great interest. Moreover, the technique of algebraic fingerprints underlie the fastest algorithms for all the problems mentioned here. Note that only time complexity is discussed here.

Ideas behind representing combinatorial problems with multivariate polynomials have been considered before, e.g., by Valiant [Val92] and Koutis [Kou05]. However, 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]. For k-path, this improved the runtime from $\mathcal{O}^*(4^k)$ [Che+07] to $\mathcal{O}^*(2^{3k/2})$, and for m-set k-packing, the runtime was improved from $\mathcal{O}^*(5.44^{mk})$ [Kou05] to $\mathcal{O}^*(2^{mk})$.

Soon after, Williams developed this algebraic technique for an $\mathcal{O}^*(2^k)$ k-path [Wil09], from which Koutis and Williams [KW09; KW15] gathered and proposed a general $\mathcal{O}^*(2^k)$ algebraic framework, *algebraic fingerprinting*, for k-multilinear monomial detection as a general technique for parameterized combinatorial problems. In their work [KW09], Koutis and Williams gave faster $\mathcal{O}^*(2^k)$ algorithms using this technique for the k-tree, k-leaf spanning tree, and t-dominating set problems. The runtimes of the previously fastest algorithms for these problems, respectively, were $\mathcal{O}^*((2e)^k)$ [AYZ95], $\mathcal{O}^*(4^k)$ [KLR11], $\mathcal{O}^*((4+\varepsilon)^t)$ [KMR07].

Due to the development in algebraic fingerprinting, Björklund [Bjö14] found an $\mathcal{O}^*(1.657^n)$ algorithm for Hamiltonicity, improving the runtime from $\mathcal{O}^*(2^n)$ [Bel62; HK62], by clever algebrization and utilizations of algebraic fingerprints. Using similar ideas, Björklund et al. [Bjö+17] gave faster algorithms for k-path, m-set k-packing, and its special case of 3-set k-packing. The runtimes for these were $\mathcal{O}^*(1.657^k)$, $\mathcal{O}^*(2^{(m-2)k})$ and $\mathcal{O}^*(1.493^{3k})$, respectively.

k-multilinear monomial detection itself was widely studied by Chen et al. [Che+07; CF11], Chen et al. [Che+13; Che+11] and Chen [Che13]. TODO: what did they research (deterministic stuff, primality of the field) (In this thesis though, we focus on the randomized technique of algebraic fingerprinting.)

TODO: deterministic multilinear detection

TODO: constrained multilinear detection (weights? functional motifs for biological networks)

TODO: parallelization (MIDAS?)

Stuff: Fast graph scan statistics, temporal graphs,

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 goes through other notation and terminology used throughout the thesis.

3.1 Algebra

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 written with 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 required for the group 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
- $(F \backslash \{\mathbf{0}\}, \cdot)$ 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:

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

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: $f + f + f + f = 4f = 2f + 2f = \mathbf{0} + \mathbf{0} = \mathbf{0}$.

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 with 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 isomorphic algebras 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. 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 $|X| = n > 1$.

The monomials in these polynomials are of the form $kX_1^{a_1}X_2^{a_2}\cdots X_n^{a_n}$, where $a_i \in \mathbb{N}$ marks the exponent and $k \in \mathbf{K}$ the coefficient. A monomial is *multilinear* if $\forall i \in \{1 \ldots n\}\colon a_i = 0 \vee a_i = 1$. The *degree* of a monomial $m \in \mathbf{K}[X]$ is the sum of the exponents: $deg(m) = \sum_{i=1}^{n} a_i$. Any element P of such polynomial ring is a linear combination of these monomials.

TODO: group algebra, polynomial ring, linear dependency,

3.2 Arithmetic circuits

3.3 Notation and other terminology

TODO: create a table or like, list terms: multilinearity, multivariety, sum of monomials form & generating form (arithmetic circuit) of polynomial, degree of multivariate monomial, $\mathcal{O}$, Θ, FPT, $\mathcal{O}^*$, determinism & non-determinism [or use 'Monte Carlo' :)], Schwartz-Zippel lemma

AM: Standard $\mathcal{O}$ and Θ notation should be known

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 [TODO: quick examples (just refs?)]. Therefore, any progress in general multilinear monomial detection directly implies faster algorithms for all problems, that are reduced to and solved with general multilinear monomial detection. TODO: relate to section 2: related works

AM: These problems should "live" in their own subsection, then you can just call them "problems from Sect. X.Y"

In this section, we first define the parameterized multilinear monomial detection problem, and give some non-algebraic background on solving it. Then in Section 4.2, we discuss the algebraic ideas by Koutis and Williams [Kou08; Wil09; KW15]. Section 4.2 is long, but the overall idea as a general framework for parameterized problems is summarized and discussed in Section 4.4. Finally, we briefly discuss finding the solution when its existence is detected in Section 4.6.

4.1 Problem definition

The general, parameterized multilinear monomial detection problem is defined as follows:

AM: Arithmetic circuit has bounded or unbounded fan-in / fan-out?

k-MULTILINEAR MONOMIAL DETECTION

Input: A commutative arithmetic circuit A over a set of variables X representing a polynomial $P(X)$.

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

Clearly, an upper bound for solving the problem is given by a naive expansion of A into $P(X)$ and evaluation of $P(X)$. However, this is not optimal: an N-degree polynomial will have $2^{\Theta(N)}$ possible monomials, and most problems that can use this algebraization, have been solved with faster algorithms. This motivates the detection of multilinear monomials without fully expanding A into a sum of monomials.

Since only multilinear terms are important in $P(X)$, any squared variable can be instantly discarded as soons as it is formed in A. This can be achieved with dynamic programming to create a polynomial $P'(X)$ that only contains multilinear monomials. Since there are 2^N multilinear monomials in $P'(X)$ with N variables, this method results in a faster algorithm than with naive expansion.

AM: but still exponential

However, the underlying problems are usually FPT. 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 X into Y, where $|X| \geq |Y|$ and $|Y| \propto k$, and dynamically evaluate $P(Y)$ instead of $P(X)$.

AM: Def?

AM: P is unchanged, so it still expects $|X|$ arguments...?

However, since $|X| \geq |Y|$, a multilinear monomial in $P(X)$ may not be multilinear in $P(Y)$. For an algorithm to 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 is around $e^{-|Y|}$ [KW15]. This implies that $\mathcal{O}(e^{|Y|})$ 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 non-deterministic algorithm in $\mathcal{O}^*((2e)^{|Y|})$ time, where $|Y| \propto k$.

AM: Ω

AM: randomized

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 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 most parameterized problems can be reduced to, it gives a general framework for solving parameterized problems [KW15].

OM: I think this should maybe be in the introduction section rather than here?
AM: I agree

4.2 Algebraic fingerprinting

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

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

AM: Quotient ring

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

AM: $\mathbf{0}q'$?

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 algebra $\mathbf{G}$, and detect a multilinear monomial from the value returned by this evaluation. Ideally, with any $\gamma\colon X \to G$, $P(X')$ representing $P(X)$ over $\mathbf{G}$ by γ and $w \neq \mathbf{0}$,

AM: What is X'?

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

AM: Evaluation under γ needs its own notation, e.g., $\varphi_\gamma(P)$

In the following subsections, we specify an appropriate algebra 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 [KW09; KW15] by these authors as algebraic fingerprinting. However, algebraic fingerprinting can also be used to refer to the idea behind solving a problem with multilinear monomials canceling out due to characteristic, which is discussed here in Section 4.2.3.

4.2.1 Specifications for the algebra

We have arrived at an important task for multilinear monomial detection: find a field $\mathbf{G}$ for the assignment $\gamma\colon X \to \mathbf{G}$ to meet (2) and thus the first equality in (3). We specify for fields since rings are not enough 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.

AM: field or algebra?

AM: How about a commutative ring then?

For the second equality in (3), it is necessary that multilinear monomials in $P(X)$ can map to multilinear monomials in $P(X')$, i.e., it must be that $k \leq |G|$, where k is the degree of the monomials. Moreover, multilinear monomials should not evaluate to $\mathbf{0}$ over $\mathbf{G}$, and more specifically, w should not be identical to $\mathbf{0}$.

AM: Isn't $P(X')$ the evaluation of $P(X)$ under γ? If so, then $P(X')$ is just an element of $\mathbf{G}$, not a polynomial

These are the necessary specifications for the field $\mathbf{G}$ for algebraic multilinear monomial detection. However, this algebraic detection must be faster than color coding for it to be useful. Therefore, further requirements are necessary: (a) the binary operations of $\mathbf{G}$ must be fast AM: efficiently computable for a fast evaluation of $P(X')$, and (b) multilinear monomials must survive the assignment γ with a *reasonable* constant probability. We may specify a reasonable probability as something around at least $1/4$, since that is the probability of survival reached in the original work for algebraic fingerprinting [Kou08].

AM: Does it really make a difference if the probability is 1/4 and not (say) 1/100?

Recall 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 the 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 can still evaluate to zero over $\mathbf{G}$. However, if we specify for a constant probability of survival, 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 field $\mathbf{G}$ 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, utilizing the algebra $GF(2^{3+log_2(k)})[\mathbb{Z}_2^k]$ [Wil09]. Next, we look at these group algebras of $\mathbb{Z}_2^k$ for $\mathbf{G}$.

AM: $2^{\log_2(k)} = k$

4.2.2 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

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

AM: Actually an algebra over a field?

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 AM: the identities of $\mathbf{F}[\mathbb{Z}_2^k]$. Note that $\mathbf{1} = v_0$, and $\mathbf{0}$ corresponds to the element $\sum_{v \in \mathbb{Z}_2^k} a_v v$, where $a_v = \mathbf{0}_F$.

In [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 $x_i \to (v_0 + v_i)$. AM: $\gamma(x_i) = v_0 + v_i$ We note the assigned values as $\overline{X}$, and the resulting polynomial 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.

AM: So if we represent the elements of $\mathbf{F}[\mathbb{Z}_2^k]$ as vectors, we get vectors with 2^k entries

AM: Add note that "+" here represents addition over $\mathbf{F}[\mathbb{Z}_2^k]$ and *not* addition over $\mathbb{Z}_2^k$ (which is represented by a dot)

AM: No longer a polynomial

4.2.3 Fingerprints to prevent unwanted cancelation

Indeed, if we take the example in Section 1.3, we notice that the multilinear monomials have even coefficients, and thus would cancel out in $\mathbf{F}[Z_2^k]$ due to characteristic. TODO: write the example in the section and then add here some example polynomial from there that has even coefficients

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 (EXAMPLE POLYNOMIAL) as follows: (TODO: add fingerprinted polynomial here)

Then, the algorithm could assign every $a \in X \cup S$ to $\mathbf{F}[\mathbb{Z}_2^k]$. 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 variables 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. Raising the dimension of $\mathbb{Z}_2^k$, however, would exponentially slow down matrix multiplications which are important for efficiency in the full algebraic framework (see Section 4.4).

Koutis approached this problem by assigning fingerprints to a different algebra: he used $S \to \mathbb{Z}_2$ and set $\mathbf{F} = \mathbb{Z}_2$ [Kou08]. 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 probability (TODO: what is the probability?) that a multilinear monomial will have an odd coefficient, and thus survive the assignment [Kou08].

AM: 1/2? (for every fixed monomial)

In practise AM: practice, fingerprints can be implemented into the algebrization as follows [Wil09]: for every multiplication gate G_i 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 $\mathbb{Z}_2$ instead of picking unique fingerprints s_i.

In AM: 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

AM: Don't these last two paragraphs belong in the next section?

fingerprints S to some algebra and continuing with the multiplication. Now, deciding whether a multilinear monomial exists in $P(X)$ is essentially given by whether the polynomial $P(\overline{X}, S) \in \mathbf{F}[\mathbb{Z}_2^k]$ is identical to $\mathbf{0}$, i.e., with Φ representing the family of mappings $S \to \mathbf{F}$,

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

AM: You're going to have to explain this to me :)

4.2.4 Polynomial identity testing

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 $Q(S) \in \mathbf{F}[\mathbb{Z}_2^k]$. 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, where a multilinear monomial is detected if $Q(S) \neq \mathbf{0}$ over $\mathbf{F}[\mathbb{Z}_2^k]$ 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 $Q(\phi(S)) = \mathbf{0}$, $Q(\delta(S)) \neq \mathbf{0}$ with a high probability over some $\delta \in \Phi$ [Wil09].

AM: References for Schwartz-Zippel:

- J. T. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," Journal of the ACM, vol. 27, no. 4, pp. 701–717, 1980.
- S. Yekhanin, Locally Decodable Codes. Now Publishers, 2012. (To appear). R. Zippel, "Probabilistic algorithms for sparse polynomials," in EUROSAM, vol. 72 of Lecture Notes in Computer Science, (E. W. Ng, ed.), pp. 216–226, Springer, 1979.
- also maybe R. A. DeMillo and R. J. Lipton, "A probabilistic remark on algebraic program testing," Information Processing Letters, vol. 7, no. 4, pp. 193–195, 1978.

Of course, $\mathbf{F}$ must have characteristic 2. For this, Williams used the field $GF(2^{3+log_2 k})$ [Wil09]. By Schwartz-Zippel lemma, $Q(S)$ evaluates to $\mathbf{0}$ over a random assignment

$S \to GF(2^{3+log_2 k})$ with probability at most $1/2^3$ [Wil09]. 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.

TODO: talk briefly about the other requirement (fast operations), results in a $\mathcal{O}^*(2^k)$ time algorithm.

In the following section, we see this as a whole and look further into implementing this stuff. TODO: rephrase

4.3 Time and space complexity

In the previous section, we discussed the specifications for an algebra 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 $P_S(X)$ at any X 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 $\mathbb{Z}[\mathbb{Z}_2^k]$:

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

Since Koutis used an isomorphism, all the ideas given in Section 4.2 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
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 **??**, algebraic fingerprinting gives an $\mathcal{O}^*(2^{3k})$ algorithm [KW15], since k-3D-matching reduces into $3k$-degree multilinear monomial detection.

4.4 Algebraic fingerprinting as a framework for parameterized problems

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. Thus, we may say that algebraic fingerprinting gives a general framework for parameterized combinatorial problems.

In Section 4.2, we discussed the idea behind algebraic fingerprinting: generate a polynomial $P(X)$ from the algebrization of a combinatorial problem, and evaluate $P(X)$ over $GF(2^{3+log_2 k})[Z_2^k]$ with randomized assignments $X \to GF(2^{3+log_2 k})[Z_2^k]$ of form $x_i \to (v_0 + v_i)$, augmented with scalar multiplications by elements randomly chosen from $GF(2^{3+log_2 k})$. This gives a randomized algorithm for k-multilinear monomial detection that runs in $\mathcal{O}^*(2^k)$ time.

This section gives a general overlook on algebraic fingerprinting as a framework for parameterized problems. First, we discuss the time and space complexities of algebraic fingerprinting. Then, we go over an example implementation of algebraic fingerprinting for the k-path problem. Finally, we discuss the limits of this technique: the chosen algebra $GF(2^{3+log_2 k})[Z_2^k]$ is optimal for the general multilinear monomial detection.

OM: give all the numbers here?

4.5 Limits of algebraic fingerprinting

SHORT SECTION

TODO: explain the limit in general multilinear detection with this algebraic framework (impossible to find better algebra than what is used for the current fastest k-mld algorithm), [KW09]

4.5.1 There are no faster algebras

In Section 4.3, 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 the algebraic fingerprinting is in some sense optimal. However, ideas from algebraic fingerprints can be utilized for faster algorithms for specific k-multilinear monomial detection instances (see Section **??**).

4.5.2 Derandomization seems difficult

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**], which is hard??

4.6 Finding the solution

SHORT SECTION

Multilinear monomial detection has only been given as a detector for a solution, i.e., a decision algorithm. Talk about actually finding the solution.

AM: Yes, but probably restrict it to $\sim$ 1 page (there is enough content as it is)

[Kou08] gives an algorithm that solves the decision problem for k-path. k-path is found with $\mathcal{O}^*(n + min(k^2, m))$ applications of the algorithm.

[Wil09] solves the decision problem for k-path. [Wil09] also gives an algorithm that finds a path when it is known that a k-path exists.

5 Improving algebraic fingerprinting

As mentioned in Section 4.5, the algebraic fingerprinting framework proposed by Koutis and Williams [Wil09; KW15] for k-multilinear monomial detection has a lower bound of $\mathcal{O}^*(2^k)$. However, this framework AM: is very generic since it approaches the abstract 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. TODO: rephrase more clearly

AM: O and "lower bound" don't go well together

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

AM: for...?

AM: make use of

Furthermore, the evaluation of the polynomial in the algebraic fingerprinting framework is done sequentially. However, the matrix representations of the group algebras used in [Wil09] offer possibilities for parallelization. In section 5.2, AM: we discuss the ideas of Ekanayake et al. behind the distributed multilinear monomial detection [Eka+19] (are discussed) AM: .

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

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

5.2 Parallelizing multilinear monomial detection

AM: Probably very ambitious to describe the full thing here... (You already need to focus on getting the details right in Sects. 4 and 5.1.) Maybe instead just give a broad picture of what other things people have considered? (unless this is really the only other thing out there)

TODO

6 Conclusion

TODO: conclude

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

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.

[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.

[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.

[Che13] Shenshi Chen. "Monomial testing and applications". In: *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management: Third Joint International Conference, FAW-AAIM 2013, Dalian, China, June 26-28, 2013. Proceedings*. Springer. 2013, pp. 106–117.

[CF11] Zhixiang Chen and Bin Fu. "The complexity of testing monomials in multivariate polynomials". In: *Combinatorial Optimization and Applications: 5th International Conference, COCOA 2011, Zhangjiajie, China, August 4-6, 2011. Proceedings 5*. Springer. 2011, pp. 1–15.

[Che+11] Zhixiang Chen et al. "Algorithms for testing monomials in multivariate polynomials". In: *Combinatorial Optimization and Applications: 5th International Conference, COCOA 2011, Zhangjiajie, China, August 4-6, 2011. Proceedings 5*. Springer. 2011, pp. 16–30.

[Che+13] Zhixiang Chen et al. "On testing monomials in multivariate polynomials". In: *Theoretical Computer Science* 497 (2013), pp. 39–54.

[Eka+19] Saliya Ekanayake et al. "MIDAS: Multilinear detection at scale". In: *J. Parallel Distributed Comput.* 132 (2019), pp. 363–382. DOI: 10.1016/j.jpdc.2019.04.006. URL: https://doi.org/10.1016/j.jpdc.2019.04.006.

[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.

[KLR11] Joachim Kneis, Alexander Langer, and Peter Rossmanith. "A new algorithm for finding trees with many leaves". In: *Algorithmica* 61 (2011), pp. 882–897.

[KMR07] Joachim Kneis, Daniel Mölle, and Peter Rossmanith. "Partial vs. complete domination: t-dominating set". In: *SOFSEM 2007: Theory and Practice of Computer Science: 33rd Conference on Current Trends in Theory and Practice of Computer Science, Harrachov, Czech Republic, January 20-26, 2007. Proceedings 33*. Springer. 2007, pp. 367–376.

[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.

[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.