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

Detecting multilinear monomials with algebraic fingerprints

Bachelor's Thesis

March 25, 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:	deadline date
Pages:	13
Major:	Computer Science
Code:	SCI3027
Supervisor:	Prof. Eero Hyvönen
Instructor:	D.Sc. Augusto Modanese (Department of Computer Science)
abstract to be	
Keywords:	key, words, the same as in FIN/SWE
Language:	English

Contents

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.

AM: No need for this since you already have ToC

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 Algebraization of combinatorial problems

AM: Algebrization

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.

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

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.

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 into 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 $\sum_X$?

2 Related works

TODO: go through publications that utilize this algebraic fingerprinting technique

3 Preliminaries

It is necessary to recall basic algebraic concepts before further discussing multilinear monomial detections and algebraic fingerprinting. This section gives definitions for a group, ring and field, and some useful concepts regarding them. The second subsection goes through other notation and terminology used throughout the thesis.

3.1 Groups, rings and fields

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.

AM: There is also `\mathbf`
AM: See TeX source
AM: I think $e = e^{-1}$ follows from the other requirements (but that is not so important)

A ring $\mathbf{R}$ is a tuple $(R, \cdot)$ AM: $(R, +, \cdot)$, where $R = (G, +)$ is an Abelian group, $\cdot :$ $G \times G \longrightarrow G$ is a binary operation closed under G. We call the binary operations $+$ and $\cdot$ addition and multiplication, respectively. Note, that from here on we use R as the set of elements defined for $\mathbf{R}$. In general, a bold typeface $\mathbf{X}$ represents a group, ring or field and X its set of elements. R must contain a multiplicative identity $\mathbf{1} \in R$ such that $\forall a \in R$: $a \cdot \mathbf{1} = a$. 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.,

$$\forall a, b, c \in R\colon\ a \cdot (b + c) = (a \cdot b) + (a \cdot c) \wedge (b + c) \cdot a = (b \cdot a) + (c \cdot a).$$

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

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:

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

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

AM: characteristic

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

TODO: group algebra, polynomial ring, linear dependency

3.2 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 framework for detecting multilinear monomials

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.

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

TODO: give structure of this section

4.1 Problem definition

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

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 (TODO: quick example?). This motivates the detection of multilinear monomials without fully expanding the polynomial into a sum of monomials.

Since only multilinear terms are important in $P(X)$, any squared variables can be instantly discarded as soons as they are 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.

TODO: Rewrite the algebra here (first quickly in English, then in maths) 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 algorithm to scale with the parameter, 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 $P(X)$.

However, since $|X| \geq |Y|$, a multilinear monomial in $P(X)$ may not be multilinear in $P(Y)$. This implies a non-deterministic decision algorithm with no false positives.

4.2 Algebraic multilinear monomial detection

TODO: go through idea of algebraic assignment and specifications for a suitable algebra, then lead to fingerprinting (fingerprints solve an issue of unwanted cancellation due to characteristic)

4.2.1 Algebraic fingerprinting to prevent unwanted cancellation

Algebraic fingerprints are introduced to solve the problem of unwanted cancellation. To prevent the multilinear monomials from cancelling each other, we augment the polynomial with new unique indeterminates, i.e. fingerprints, such that multilinear monomials in the expanded polynomial are unique. The squared variables, and thus non-solution monomials, will still vanish, but the multilinear terms will prevail.

However, introducing new indeterminates raises the degree of the multilinear monomials. Therefore it may be, that there are not enough matrices to assign such that there would be no duplicates in a multilinear monomial. As a result, higher dimension matrices are needed, which implies exponentially slower matrix multiplication (TODO: check the time complexity of matrix multiplication). Thus, introducing new indeterminates slows the algorithm. (TODO: slower than $\mathcal{O}^*(2^k)$?)

In [Kou08] instead of uniqueness, Koutis uses random assignment from $\{\mathbf{0}, \mathbf{1}\}$. With this assignment, there is a possibility that a multilinear monomial will have an odd coefficient, thus surviving the cancellation due to characteristic.

Koutis uses random assignment from $\{0, 1\}$ in order to find an odd k-mld problem, which can be solved by previous methods, since there is no cancellation (multilinear terms have odd coefficients).

AM: What is cancellation?

TODO: explain how fingerprints prevent cancellation, give the general algebraic framework that appears in [Wil09], (also polynomial identity testing)

[Kou08] uses commutative group algebras of $\mathbb{Z}_2^k$ (to have squared variables equal zero) for ODD k-multilinear detection (multilinear monomials have odd coefficients). randomized assignment, no false positives

AM: ODD = ordered decision diagram?

Koutis reduces an algebraization into ODD k-multilinear detection by randomly assigning fingerprints from 0,1 in the hopes of getting odd number of solutions (k-multilinear terms), i.e., hoping that for a pair that cancels each other, one of them gets removed by 0-assignment so that cancelation is prevented

[Wil09] reduces multilinear detection (after assinging variables values from some algebra) to polynomial identity testing for a polynomial P(A) that has fingerprints as variables. Fingerprints are then assigned values from a field that has more elements than P(A) has roots, which means that P(A) evaluates to non-zero with high probability (if there are solutions)

4.2.2 Limits of general multilinear monomial detection

TODO: give some cons wrt. color coding (difficult derandomization, are we able to add weights for optimization problems?)

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]

TODO: lead to problem specific implementations (we can use fingerprinting cleverly, when we understand the underlying problem well)

4.3 Finding the solution

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

[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.

[Bjö+17] solves the decision problem for k-path starting at some vertex s. A k-path can be found by just applying the algorithm for every vertex.

5 Problem-specific applications of algebraic fingerprints

As mentioned in section 4.2.2, the algebraic framework proposed by Koutis and Williams [Kou08; Wil09] for k-multilinear monomial has a lower bound of $\mathcal{O}^*(2^k)$. However, this framework approaches the abstract multilinear monomial detection without utilizing any information about the underlying problem.

This section showcases how the auxiliary fingerprints can be abused, when the underlying problem is well understood.

TODO: add subsections for problem specific implementations with different utilizations of fingerprints

5.1 Fingerprinting for cancellation 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 cancellation 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].

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?

6 Conclusion

TODO: conclude

References

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

[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.

[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.

[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.

[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*(2k) 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`.