Algebraization in parameterized algorithms and complexity

and complexity

Yiannis Koutis

University of Puerto Rico,Rio Piedras

Algebraization in **parameterized algorithms** and **complexity**

A short survey of recent progress
in **algorithm design**
based on algebraic methods

Enabled by the **attitude**
of the PC paradigm towards hardness

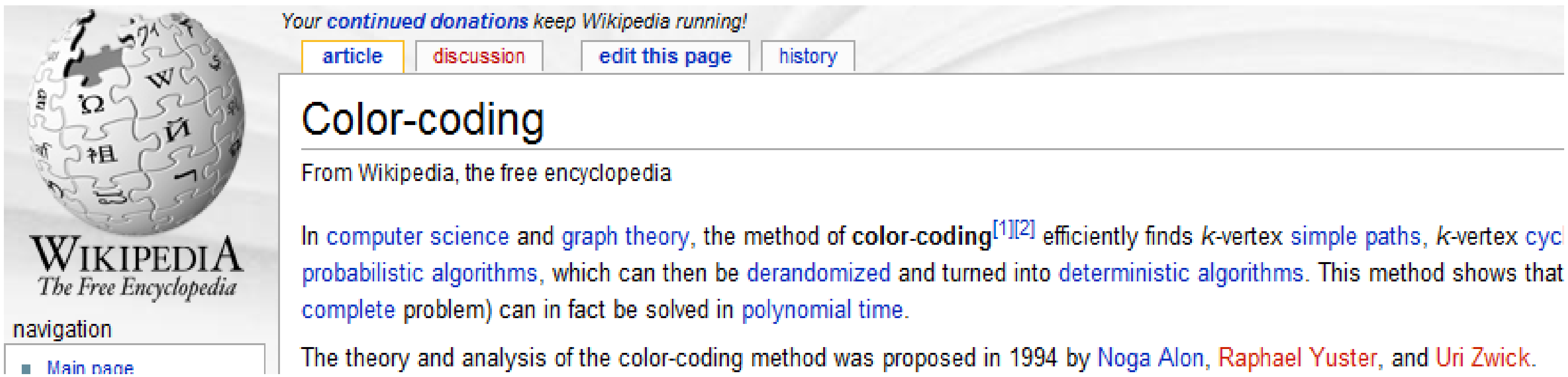
A great research **program** in the context of PC: FPT races

- Design algorithms (for your favorite problem) with better dependence on the parameter

- Obvious theoretical value
- Great **potential for impact** in practice

Impact in practice

theory, yet widely applicable and popular


WIKIPEDIA
The Free Encyclopedia
navigation
Main page

Your continued donations keep Wikipedia running!

article | discussion | edit this page | history

Color-coding

From Wikipedia, the free encyclopedia

In computer science and graph theory, the method of **color-coding**[1][2] efficiently finds *k*-vertex simple paths, *k*-vertex cyc
probabilistic algorithms, which can then be derandomized and turned into deterministic algorithms. This method shows that
complete problem) can in fact be solved in polynomial time.

The theory and analysis of the color-coding method was proposed in 1994 by Noga Alon, Raphael Yuster, and Uri Zwick.

- Recently, the color coding approach has attracted large attention in bioinformatics. One example is the detection of signaling pathways in protein-protein interaction (PPI) networks. Another example is to discover and to count the number of motifs in PPI networks. Both signaling pathways and motifs do a great help for us to better understand similarities and differences of many biological functions, processes, and structures among organisms. Due to huge amount of gene data, comparisons and searches of them are time-consuming, and this makes them a difficult job. However, by exploiting color coding method, the motifs or signaling pathways with $k = O(\log n)$ vertices in a network G with n vertices can be found very efficiently in polynomial time. Thus, this enables us to explore more complex or larger structures in PPI networks.

Impact in practice

theory, yet widely applicable and popular

[PDF] ▶ **Color-coding**

N **Alon**, R Yuster, U Zwick - Journal of the ACM, 1995 - tau.ac.il

Page 1. **Color-coding** ∗ Noga **Alon** †‡ Institute for Advanced Study and Tel Aviv University Raphael Yuster ‡ Dept. of Computer Science Tel Aviv University **...**

Cited by 240 - Related articles - View as HTML - BL Direct - All 11 versions

Color-coding

N Alon, R Yuster... - Journal of the ACM (JACM), 1995 - portal.acm.org

[nstltl[te jor A~iLatz~-edSt6{dy, Princeton, NcwJer-seyutal Te[-.4LiL UniLetxi~, Tel-A[l[, Israel **...** Abstract. We describe a novel randomized method. the method of cobm-**coding** for finding simple **...** G = (1', E). The randomized algorithms obtained using this method can be **...**

Cited by 355 - Related articles - BL Direct - All 22 versions

450

Race against the **non-parameterized** algorithm
(an alternative way of viewing "races")

- The **Hamiltonian Path** problem :
  Given a graph with n-vertices, does it contain an **n-path**, i.e. walk of length n, without loops ?

Now pretend we are back in 2007

- Best known algorithm for HP runs in $O^*(2^n)$ [Bellman, Karp, 1962]

- The **k-path** problem can be solved in time $O^*((2e)^k)$ via

Color Coding [Alon, Yuster, Zwick, 94]

Which one would you try to improve upon?

Race against the **non-parameterized** algorithm
(an alternative way of viewing "races")

- The best **k-path** algorithm of 2007 had to "switch" to the non-parameterized algorithm above a k.
- Whenever my pseudocode starts getting "bloated" by if-then-else's, I get uneasy.
- The **conjecture**: For most canonical problems we study, the fastest **parameterized** algorithm **at the limit** should also be the **fastest "exact"** algorithm for the parent NP-hard problem.

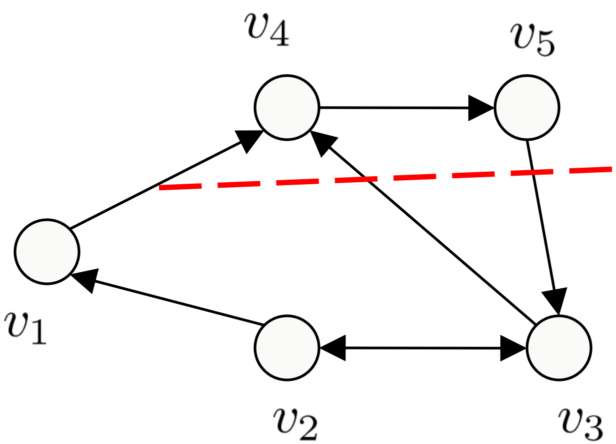
what I mean by "**algebraization**"

1. Translate the **combinatorial** problem into an **algebraic** problem about multivariate polynomials.

2. Use **algebra** (rather than **combinatorics**) to solve the latter problem

how color-coding works

algebraization step 1

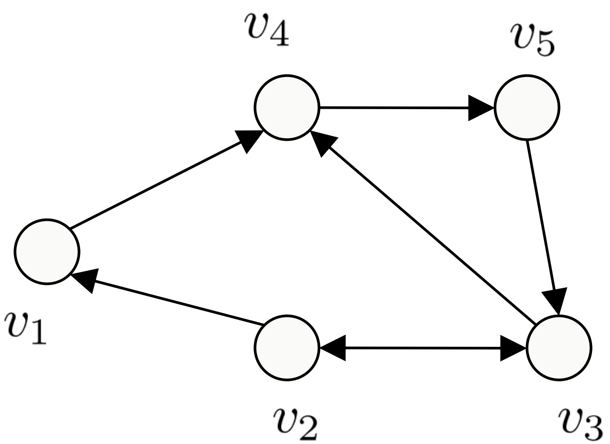


$$A = \begin{pmatrix} 0 & 0 & 0 & x_1 & 0 \\ x_2 & 0 & x_2 & 0 & 0 \\ 0 & x_3 & 0 & x_3 & x_3 \\ 0 & 0 & 0 & 0 & x_4 \\ 0 & 0 & x_5 & 0 & 0 \end{pmatrix}$$

- Adjacency Matrix with indeterminates
- $A^k(i,j)$ contains a multivariate polynomial
- There is a 1-1 correspondence between walks and terms
- A k-path is a multilinear monomial of total degree k

how color-coding works

algebraization step 1



$$A = \begin{pmatrix} 0 & 0 & 0 & x_1 & 0 \\ x_2 & 0 & x_2 & 0 & 0 \\ 0 & x_3 & 0 & x_3 & x_3 \\ 0 & 0 & 0 & 0 & x_4 \\ 0 & 0 & x_5 & 0 & 0 \end{pmatrix}$$

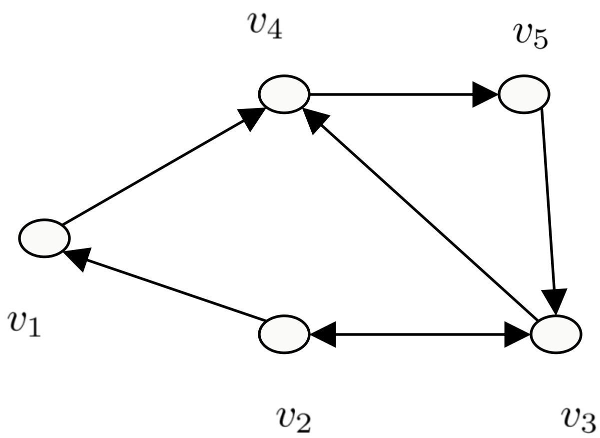
- n-path: unique monomial $x_1 * x_2 * \ldots * x_n$

multilinear detection

- How to extract it? Dynamic programming, Inclusion-Exclusion.

how color-coding works

algebraization step 2?



$$A = \begin{pmatrix} 0 & 0 & 0 & x_1 & 0 \\ x_2 & 0 & x_2 & 0 & 0 \\ 0 & x_3 & 0 & x_3 & x_3 \\ 0 & 0 & 0 & 0 & x_4 \\ 0 & 0 & x_5 & 0 & 0 \end{pmatrix}$$

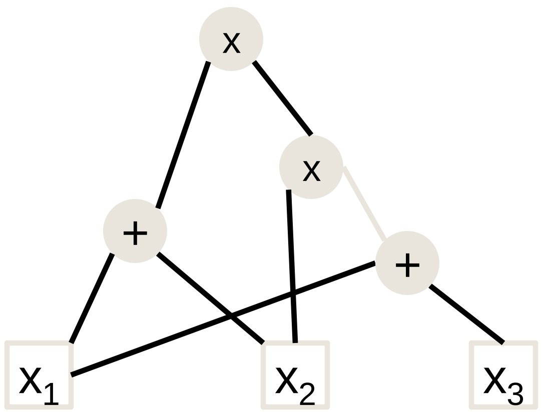
- k-path: $\binom{n}{k}$ square-free monomials

multilinear detection

- Which coefficient to extract ?
- Map $[x_1, \ldots, x_n] \rightarrow [y_1, \ldots, y_k]$
- Now there's only one monomial $y1 * y_2 * \ldots * y_k$ (with prob $1/e^k$)

multilinear k-term detection problem (k-MLD)

- **Input:** an arithmetic circuit C representing a polynomial P



$$P = x_1x_2x_3 + x_2^2x_3 + x_1^2x_2 + x_1x_2^2$$

- **Question:** does P contain a square-free term of degree k?

multilinear k-term detection problem (k-MLD)

- **Input**: an arithmetic circuit representing a polynomial P
- **Question**: does P contain a multilinear term of degree k?

- substitution and evaluation: fast, but can it be informative?
- expansion into sums: computationally expensive but helps reasoning about terms

$$(x_1x_2x_3 + x_1x_4x_5 + x_4x_5x_6)^2 = (x_1x_2x_3)^2 + (x_1x_4x_5)^2 + (x_1x_5x_6)^2 + 2(x_1^2x_2x_3x_4x_5) + 2(x_1x_4^2x_5^2x_6) + 2(x_1x_2x_3x_4x_5x_6)$$

- Computationally intensive but helps reasoning about terms
- **Idea: "exotic" substitutions, for example matrices**

algebraic detection of multilinear k-term

$$(x_1x_2x_3 + x_1x_4x_5 + x_4x_5x_6)^2 = (x_1x_2x_3)^2 + (x_1x_4x_5)^2 + (x_1x_5x_6)^2 + 2(x_1^2x_2x_3x_4x_5) + 2(x_1x_4^2x_5^2x_6) + 2(x_1x_2x_3x_4x_5x_6)$$

- squares are bad – annihilate the squares
- matrices must have zero squares: $X_i^2 = 0$ modulo 2
- square-free products are good – must survive with some probability
- square-free products different than 0
- matrices must commute: $X_iX_j = X_jX_i$
- matrices must be "small", say **O(2^k)**

algebraic detection of multilinear k-term

$$(x_1x_2x_3 + x_1x_4x_5 + x_4x_5x_6)^2 = (x_1x_2x_3)^2 + (x_1x_4x_5)^2 + (x_1x_5x_6)^2 + 2(x_1^2x_2x_3x_4x_5) + 2(x_1x_4^2x_5^2x_6) + 2(x_1x_2x_3x_4x_5x_6)$$

- Still the multilinear term can have an **even** coefficient
- **Ryan Williams' idea**: Introduce extra variables A, effectively hashing each copy of a multilinear term into a distinct multilinear term.
- Assign to the variables in A random values from a **small** field of characteristic 2.

progress in parameterized algorithms

Fast detection of multilinear terms in multivariate polynomials: an exponential speed-up over color-coding for decision problems

- Fastest known algorithms for: directed k-path, m-set packing, subgraph packing problems and other problems where color-coding has been used before.

- **A more complete algorithmic tool:** new algorithms for problems where people originally failed to see the applicability of color-coding. E.g. problems with strings (exemplar breakpoint distance).

back to the k-path problem: can we do better ?

- The k-path polynomial has special properties.
  The k-MLD problem is much more general.

- So, bring back the combinatorics:
  - Algebraic graph theory : e.g. Tutte matrix and its determinant

- Use special properties of these "smarter" polynomials

- Andreas Bjorklund 2010 (later extended to k-path):
  n-path can be solved in time $O(1.7^n)$

The "attitude" of parameterized complexity and how it enabled **progress in algorithm design**

- A race against the **"non-parameterized"** algorithm

→

- New ideas and techniques for the parameterized problem

→

- Progress after nearly 50 years
  for the Hamiltonian problem.

The **main ideas** so far

- Algebraization
- Computations over fields of characteristic 2
- Use of the Schwartz-Zippel Lemma -- R. Williams
- Self-cancellation modulo 2 of "undesired" terms of polynomials (even number of copies) – A. Bjorklund

- Algorithms are randomized

Cut & Count

Cygan, Nederlof, Pilipczuk, Pilipczuk, van Rooij, Wojtaszczyk 2011

- Algorithms for problems parameterized by treewidth **tw**
- Hamiltonian Path, Longest Path, Connected Dominating Set, Feedback Vertex Set etc..

- Previous algorithms worked in time $O(tw^{tw})$
- Now algorithms work in time $O(c^{tw})$ (for c=4, or c=6)

- Conditional lower bounds for some problems where $O(tw^{tw})$ didn't get breached by the technique.

K-Cycle problem

Bjorklund, Husfeldt, Taslaman 2011

- Find if graph contains a cycle that goes through **K** pre-specified vertices
- Previously O(2^2^K^10)
- Now $O(2^K)$

Kernelization vs Compressibility ?

Wahlstrom 2012

- Recall that k-path, k-cycle probably doesn't have a kernel
- Status of **kernelization** for K-cycle is unknown

- K-cycle is **compressible** down to space **$O(K^3)$.**
- The compressed object is a matrix with variables and the K-cycle instance is answered via computing an **exponential** number of determinants

Algebraization in **parameterized algorithms** and **complexity**

A short survey of recent progress
in **algorithm design**
based on algebraic methods

Enabled by the **attitude**
of the PC paradigm towards hardness

Some informal thoughts on algebraization

A study of k-MLD that is sensitive to the underlying polynomials?

k-path problem

k-packing of 3-sets

$$(Y_1 + \ldots + Y_m)^n$$

$$Adj^k(x_1, \ldots, x_n)$$

$$Y_j = x_{j1}x_{j2}x_{j3}$$

Hard to believe that these two have the same upper bound

Branching algorithms

Branching algorithms reduce an instance to a small number of slightly smaller instances, based on **branching rules**

- Whenever my pseudocode starts getting "bloated" by if-then-else's, I get uneasy.

- With **branching algorithms**…. **I despair!**

I hate reviewing branch &bound papers

Branching algorithms

- Daniel Marx calls for a systematic study of branching algorithms similar to the one for kernelization.
- The main question would be:

  Which problems admit a branching algorithm?

Branching algorithms

- Algebraization **works** for almost all problems where branching algorithms have been successful. It just gives slower algorithms.
- On the other hand, there are **no known branching algorithms** for several problems where algebraization works.

Branching algorithms

- So I propose, a

Race against the branching algorithm

- In particular I would be interested in seeing progress via algebra on:
  - VERTEX COVER
  - K-LEAF DIRECTED TREE

Generalizations of k-MLD

- The k-MLD problem can be formulated as follows:

Given a polynomial P, presented as an arithmetic circuit, and the ideal $I=\langle x1^2, \ldots, xn^2 \rangle$, is $P/I = 0$?

Given a polynomial P, presented as an arithmetic circuit, and an ideal I, is P/I =0?

A generalization

- The polynomial

$$P = (x_1 + \ldots + x_n)^k$$

- The ideal

$$I = < x_1^2, \ldots, x_n^2 >$$

Independent Set

- Augmented with

$$I = < x_i x_j >$$

for your favorite (i,j)

Algebraization provides opportunities for connections with other kinds of mathematics

- Assume you have a set of 0-1 vectors, addition is entry-wise modulo 2

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

- How many k-subsets of d-dimensional vectors sum to 0?

$$2^{(k-d)}-1$$

Summary

- Algebraization is a powerful technique in parameterized algorithms and complexity
- Algorithm Design in general can benefit by the study of algebraic approaches for selected PC problems.
- Pursuing algebraization approaches also has the potential of an impact in other areas of mathematics.

Algebraization in PC is fun

Thanks!