GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Master Thesis

Title

Author:
Julian Lorenz
Mtr.: 3383863

Supervisor:
Prof. Dr. Holger Dell

2022 – 01 – 19

Table of Contents

1 Acknowledgements

2 Statutory Declaration

I confirm that the submitted thesis is original work and was written by me without further assistance. Appropriate credit has been given where reference has been made to the work of others.

The thesis was not examined before, nor has it been published.
The submitted electronic version of the thesis matches the printed version.

Datum? Unterschrift!

3 Abstract

ToDo

4 Introduction

ToDo

4.1 Preliminaries

5 Main 1

5.1 Main 1.1

5.2 Main 1.2

5.3 Main 1.3

6 Main 2.1

6.1 Main 2.1

6.2 Main 2.2

6.3 Main 2.3

7 Implementation

7.1 Idea

7.2 Code

7.3 Analysis

8 Conclusion

ToDo

9 List of Figures

10 List of Tables

11 Work in Progress

11.1 Definitions

11.1.1 Homomorphismus

- $f(x \cdot y) = f(x) \cdot f(y)$ structure preserving.
- Hom: relation holds -> relation holds
- Strong Hom: relation holds/fails -> relation holds/fail
- 2 vertices no edge -> 2 vertices with edge ==> hom, but not strong hom
- P_3 -> P_3, P_2
- C_4 -> C_4, P_3, P_2
- any bipartit -> K_2
- non-edge -> edge or vertex or non-edge
- proper coloring: adjacent vertices are not assigned the same color
- proper coloring with r colors <=> hom from G to K_r (complete)

11.1.2 Automorphism

- a, b autom -> a verknüpft b is automorphism
- a autom -> a^{-1} autom
- ID always autom, reflection (mirror axis), rotation
- form group
- complete graph: $n!$
- Fact: For every G, $Aut(G) \cong Aut(\overline{G})$
- Theorem: G graph with n vertices (order n): $|Aut(G)|$ divides $n!$ and $|Aut(G)| = n! \iff$ G is K_n or $\overline{K_n}$

```
# Sage
K3 = graphs.CompleteGraph(3)
C4 = graphs.CycleGraph(4)
K3.show()
K3.automorphism_group().list()
```

11.2 Homomorphism Vector

- homomorphism vector $\mathrm{HOM}(G) := (\mathrm{Hom}(F,G))_F$, infinite vector
- computing entries NP-complete -> restrict to certain classes (poly time)
- bounded by tree-width <-> poly time
- $\mathrm{HOM}_C(G)$ describes the spectrum

-> adj matrices same eigenvalues multiplicities

11.3 Color-Refinement / Weisfeiler-Leman

Properties:

- different coloring ⇒ not isomorph, same color ⇒ nothing
- distinguishes almost all non-isomorphic graphs (Erdös)
- after first refinement: colors ≈ degrees
- 1-WL & 2-WL distinguish the same graphs
- k-WL distinguishes less Graphs than (k+1)-WL for all $k \geq 2$
- one-vs-many test trivially possible

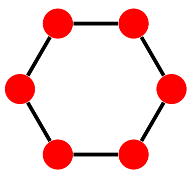
Paper:

- Color Refinement and its Applications (2017)
- Pseudocode
- The k-dimensional Weisfeiler-Leman Algorithm (2019)
- Pseudcode

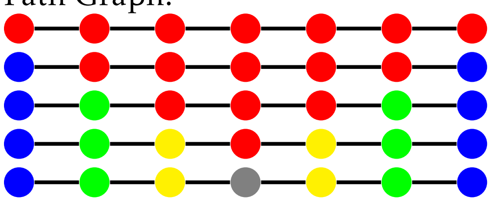
Pseudocode:

11.3.1 Examples

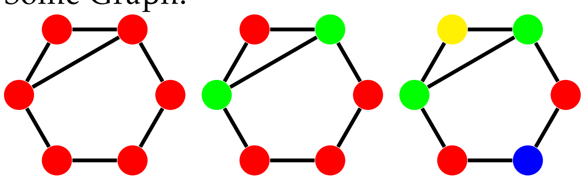
Regular Graph:



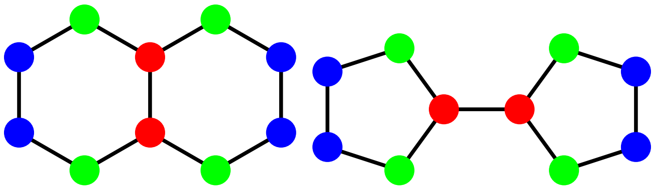
Path Graph:



Some Graph:



Fail Graphs:



11.4 k-WL

- CR: $C_{i+1}(v) = (C_i, \{\{C_i(w) | vw \in E\}\})$
- k-WL: $C^k_{i+1}(\overline{v}) = \left(C^k_i(\overline{v}), \{\{(atp(v_1, \ldots, v_k, w), C^k_i(v_1, \ldots, v_{k-1}, w), \ldots, C^k_i(w, v_2, \ldots, v_k)) | w \in V\}\}\right)$

11.4.1 Atomic Type

11.5 Tree-Decomposition

- all in 1 bag = gültige TD

11.6 Dell 1

Paper [1]

11.6.1 Abstract / Introduction

- quasi-lin time algo to decide whether there is a tree T with Hom(T,G) != Hom(T,H)
- isomorphic iff number Hom(F,G) of hom from F to G equals number Hom(F,H) of hom from F to H
- hom vector: HOM(G):=$(\mathrm{Hom}(F,G))_{F \text{ graph}}$, computing entries is NP-complete
- $\mathrm{HOM}_{\mathcal{F}}(G) := (\mathrm{Hom}(F,G))_{F\in\mathcal{F}}$ in poly for bounded tree width, not poly if unbounded
- $\mathrm{HOM}_{\mathcal{C}}(G)$ characterizes the spectrum of a graph, i.e. $\mathrm{HOM}_{\mathcal{C}}(G) = \mathrm{HOM}_{\mathcal{C}}(H)$ iff adja mat same EV with same multi
- edit distance: how many edges and vertices deleted or removed to obtain other
- $\mathrm{F}_{\mathrm{iso}}(G,H)$ describes (fractional) isomorphismus
- CR dist iff fractionally isomorphic (non-negative real solutions)
- k-WL colors k-tuples of vertices
- level-k-Sherali-Adams relaxation?? (iso to frac iso)

11.6.2 Our results

- (ii) <-> (iii) Tinhofer (23,24)
- $\mathrm{HOM}_{\mathcal{T}}(G) = \mathrm{HOM}_{\mathcal{T}}(H)$
- dif number of vertices/edges => hom counts differ on 1-vertex or 2-vertex tree

check yourself

- every known algo for computing the entry $\mathrm{Hom}(T,G)$ of the vector $\mathrm{HOM}_{\mathcal{T}}(G)$
- proof 1 => characterize an n-vertex graph G up to frac iso, suffices to restrict the hom vector $\mathrm{HOM}_{\mathcal{T}}(G)$ to trees of height at most $n-1$.
- FISO int -> iso, FISO real -> path, FISO non-neg real -> Tree

Theorem 11.1. $\mathrm{HOM}_{\mathcal{T}}(G) = \mathrm{HOM}_{\mathcal{T}}(H)$ <-> Color doesnt disting <-> frac iso

Theorem 11.2. $\mathrm{HOM}_{\mathcal{T}}(G) = \mathrm{HOM}_{\mathcal{T}}(H)$ <-> $\mathrm{F}_{\mathrm{iso}}(G,H)$ lin equ has real solution

Theorem 11.3. For every k, there are graphs G and H with $\mathrm{HOM}_{\mathcal{P}_k}(G) = \mathrm{HOM}_{\mathcal{P}_k}(H)$ and $\mathrm{HOM}_{\mathcal{T}_2}(G) \neq \mathrm{HOM}_{\mathcal{T}_2}(H)$

Corollary 5

11.7 Dell 2

Paper [?]

11.7.1 Abstract

- Homomorphism polynomials enumerate all homomorphisms from a pattern graph H to n-vertex graphs
- looking for a pattern graph H in another graph G, called the host graph -> homomorphism
- looking for subgraphs of host graph G which are isomorphic to H
- host + pattern as input -> NP-complete
- pattern graph fixed size -> better possible
- suffices to consider the hom polys from H to K_n as the homomorphism polynomial of H
- arithmetic circuit constructions of hom polys can be used to obtain almost all known better algorithms for detecting induced subgraph isomorphisms as well
- hom poly yield poly families that are complete for classes VP (bounded treewidth) and VNP (?)
- Graph parameter: Treewidth, parthwidth, treedepth (from being a star)
- H has bounded treewidth, then there are small-sized ari circ for hom polys
- ari circ constructions of hom polys based on treewidth do not use negative constants -> monotone
- Schnorr: expo lower bound for clique poly (special case hom poly: H is clique)

11.7.2 contributions

- treew, pathw and treed exactly characterize the complexity of hom poly for arithmetic circuits, ABPs and arithmetic formulas
- colored subgraph iso polys instead of hom polys
- Q: in how many monomial computations can a single gate participate? A: dictated by treewidth
- comp polynomial and use it to construct tree decomposition
- gate can participate in the computation of at most $n^{k-tw(H)-1}$ monomials (degree k)
- monotone circ can not produce invalid submonomial (no cancellations)
- treewidth of a clique on n vertices is $n-1$

Theorem 11.4. The monotone arithmetic circuit complexity of homomorphism polynomial for a pattern graph H is $\Theta(n^{tw(H)+1})$, where $tw(H)$ is the treewidth of H. plural?

Theorem 11.5. The monotone ABP complexity of homomorphism polynomial for a pattern graph H is $\Theta(n^{pw(H)+1})$, where $pw(H)$ is the pathwidth of H.

Theorem 11.6. The monotone formula complexity of homomorphism polynomial for a pattern graph H is $\Theta(n^{pw(H)+1})$, where $pw(H)$ is the treedepth of H.

11.7.3 Preliminaries

- for any poly: size of smallest ABP and size of smallest skew circuit are within constant factors of each other.
- colored iso poly: enumerates all col iso from patttern to host where there are n vertics of each color. This polynomial can be used to count col iso in n-vertex host graphs by setting the variables corresponding to edges not in the host graph to 0.
- size of a tree decomposition is the size of the largest bag minus one.
- Treeewidth is the size of a smallest tree decomposition of H.
- if path: path decomposition (pathwidth)
- For all graphs $tw \leq pw \leq td - 1$
- for $p \geq 2$ there is a tree X_k on $k = 2^{p+1} - 1$ vertices that have pathwidth p.
- all paths have pathwidth 1 and the k-vertex path has treedepth $\lceil \log_2(k+1) \rceil$.

Definition 11.1. A polynomial over $\mathbb{Q}$ is called monotone if all its coefficients are non-negative.

Definition 11.2. An arithmetic circuit over the variables $x_1, \ldots, x_n$ is a rooted DAG where each source node (also called an input gate) is labeled by one of the variables x_i or a constant $a \in \mathbb{Q}$. All other nodes (called gates) are labeled with either + (addition) or $\times$ (multiplication). The circuit computes a polynomial over $\mathbb{Q}[x_1, \ldots, x_n]$ in the usual fashion. directed, acyclic Graph auss-chreiben?
The circuit is called monotone if all constnats are non-negative.
The circuit is a skew circuit if for all $\times$ gates, at least one of the inputs is a variable or a constant. what else??
The circuit is a formula if all gates have out-degree at most one.
The size of a circuit or skew circuit or formula is the number of edges in the circuit.
The depth of a circuit is the number of gates in the longest path from the root to an input gate.

Definition 11.3. An Algebraic Branching Program (ABP) is a DAG with a unique source ndoe s and a unique sink node t. Each edge is labeled with a variable from $x_1, \ldots, x_n$ or a constant $a \in \mathbb{Q}$. Each path in the DAG from s to t corresponds to a term obtained by multiplying all edge labels on that path. The polynomial computed by the ABP is the sum of all terms over all paths from s to t. The ABP is called monotone if all constants are non-negative. The size of the ABP is the number of edges. ABP notwendig?

Definition 11.4. For graphs H and G, a homomorphism from H to G is a function $\phi : V(H) \mapsto V(G)$ such that $\{i,j\} \in E(H)$ implies $\{\phi(i), \phi(j)\} \in E(G)$. For an edge $e = \{i,j\}$ in H, we use $\phi(e)$ to denote $\{\phi(i), \phi(j)\}$.

Definition 11.5. Let H be a k-Vertex graph where its vertices are labeled $[k]$ and let G be a graph where each vertex has a color in $[k]$. Then, a colored isomorphism of H in G is a subgraph of G isomorphic to H such that all vertices in the subgraph have different colors and for each edge $\{i,j\}$ in H, there is an edge in the subgraph between vertices colored i and j.

Definition 11.6. For a pattern graph H on k vertices, the n^{th} homomorphism polynomial for H is a polynomial on $\binom{n}{2}$ variables x_e where $e = \{u,v\}$ for $u,v \in [n]$.

$$\mathrm{Hom}_{H,n} = \sum_{\phi} \prod_{e} x_{\phi(e)}$$

where ϕ ranges over all homomorphisms from H to K_n and e ranges over all edges in H.

Definition 11.7. For a pattern graph H on k vertices, the n^{th} *colored isomorphism polynomial* for H is a polynomial on $|E(H)|n^2$ variables x_e where $e = \{(i,u),(j,v)\}$ for $u,v \in [n]$ and $\{i,j\} \in E(H)$.

$$\mathrm{ColIso}_{H,n} = \sum_{u_1,\ldots,u_k} \prod_{i,j} x_{\{(i,u_i),(j,u_j)\}}$$

where $u_1,\ldots,u_k \in [n]$ and $\{i,j\} \in E(H)$.

New labeling can be obtained by the substitution $x_{\{(i,u),(j,v)\}} \mapsto x_{\{(\xi(i),u),(\xi(j),v)\}}$.

Definition 11.8. Let g be a gate in a circuit C. A *parse tree* rooted at g is any rotted Notwendig?
tree which can be obtained by the following procedure, duplicating gates in C as neccessary to preserve the tree structure.

1. The gate g is the root of the tree.

2. If there is a multiplication gate g in the tree, include all its children in the circuit as its children in the tree.

3. If there is an addition gate g in the tree, pick an arbitrary child of g in the circuit and include it in the tree.

Definition 11.9. A *tree decomposition* of H is a tree where each vertex (called a *bag*) in the tree is a subset of vertices of H. This tree must satisy two properties.

1. For every edge $\{i,j\}$ in H, there must be at least one bag in the tree that contains both i and j.

2. for any vertex i in H, the subgraph of the tree decomposition induced by all bags containing i must be a subtree. This subtree is called the *subtree induced by* i.

Definition 11.10. For a connected graph H, an *elimination tree* of H is a rooted, directed tree that can be constructed by arbitrarily picking a vertex u in H and adding edges from the roots of elimination trees of connected components of $H - u$ to the root vertex labeled u. In particular, if H is a single vertex, then the elimination tree of H is the same single vertex graph. Notwendig?
The *depth* of an elimination tree is the number of vertices in the longest path from a leaf to the root. The *treedepth* of H is the depth of the smallest depth elimination tree of H.

11.7.4 Algebraic complexity of homomorphism polynomials

- Hom image of P_3 will either be another P_3 or an edge P_2.

Definition 11.11. A *separating set* for a polynomial p is a set of monomials in p such that for any two monomials s and t in the seperating set, there does not exist a monomial m of p such that m divides the product st.

The size of a separating set lower bounds the size of the monotone arithmetic circuit computing the polynomial. Hom_{K_k} requires monotone arithmetic circuits of size n^k.

Theorem 11.7. Any separating set for $\mathrm{Hom}_{P_3,n}$ has size at most n for sufficiently large n. Proposition

Proof...
Hom image of P_3 will either be another P_3 or an edge P_2.
monomials of $\mathrm{Hom}_{P_3,n}$ correspond to P_3 or edge.

Theorem 11.8. Any separating set for $\mathrm{Hom}_{C_4,n}$ has size at most n^2 for sufficiently large n. Proposition

Proof...

Theorem 11.9. The *monotone circuit complexity* of Hom_H is $\Theta(n^{\mathrm{tw}+1})$.

Proof... Über tree-decomposition verstehen

11.8 Dell 3

- Counting subgraph isomorphisms corresponds to counting linear combinations of homomorphisms
  => Counting homom suffices

References

[1] Dell, H., Grohe, M., & Rattan, G. (2018):
Lovász Meets Weisfeiler and Leman
arXiv preprint: arXiv:1802.08876

[2] Curticapean, Radu and Dell, Holger and Marx, Daniel (2017):
Homomorphisms are a good basis for counting small subgraphs
49th Annual ACM SIGACT Symposium on Theory of Computing

[3] Komarath, Balagopal and Pandey, Anurag and Rahul, CS (2020):
Graph Homomorphism Polynomials: Algorithms and Complexity
arXiv preprint: arXiv:2011.04778