COMPUTATION OF CYCLE BASES IN SURFACE EMBEDDED GRAPHS

by

Thomas Stanley

APPROVED BY SUPERVISORY COMMITTEE:

Kyle Fox, Chair

Sergey Bereg

Benjamin Raichel

COMPUTATION OF CYCLE BASES IN SURFACE EMBEDDED GRAPHS

by

THOMAS STANLEY, BS

THESIS

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN

COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS

December 2021

ACKNOWLEDGMENTS

I would like to extend my sincere thanks to Kyle Fox for his help with this thesis, chairing the committee, and guiding my research interests. I would also like to extend my deepest gratitude to John Cole for providing support and advice throughout my undergraduate and graduate educations. Many thanks to Hal Sudborough, Linda Morales, and Sergey Bereg for allowing me to join in their research and beginning my interest in academic computer science. I very much appreciate Xiaohu Guo and Darin Okuda for giving me many opportunities to further my research experience. Special thanks to Raquel Bromberg for her support allowing me to complete this project. And of course, many thanks to my family for always helping and supporting me.

November 2021

COMPUTATION OF CYCLE BASES IN

SURFACE EMBEDDED GRAPHS

Thomas Stanley, MSCS
The University of Texas at Dallas, 2021

Supervising Professor: Kyle Fox, Chair

Abstract. We study the problem of finding a cycle basis, a minimum weight set of independent cycles that form a basis of the cycle space for a given graph. We focus on finding the minimum cycle basis of directed graphs. This is a more complicated problem compared to the undirected case as the underlying field is $\mathbb{Q}$ for directed graphs instead of $\mathbb{Z}_2$ for undirected, which causes problems in the speed of calculations. Previously the fastest known deterministic algorithm to find the minimum cycle basis of a directed graph runs in $O(m^3n + m^2n^2 \log n)$ time [11]. We concentrate on graphs embedded on a surface of genus g. We modify algorithms for undirected graphs to work on directed graphs. We present an $O(mn^2g^2 \log g + m^{\omega+1})$ time algorithm to find the minimum cycle basis of a directed graph embedded on a surface of genus g. We also give an improvement on the minimum cycle basis in the undirected case.

TABLE OF CONTENTS

ACKNOWLEDGMENTS iv
ABSTRACT v
LIST OF FIGURES vii
LIST OF ALGORITHMS viii
CHAPTER 1 INTRODUCTION 1
1.1 Preliminaries 2
CHAPTER 2 ALGORITHM 6
2.1 Computing Support Vectors 6
2.2 Finding the Minimum Cycle 10
2.2.1 Results 14
REFERENCES 15
BIOGRAPHICAL SKETCH 17
CURRICULUM VITAE

LIST OF FIGURES

1.1 A simple directed graph 2
1.2 A directed graph with its underlying undirected graph [11] 3
2.1 A region tree for a set of edges with non-trivial homology signature [2] 12
2.2 A region tree for a set of edges with null-homologous homology signature [2] . . 13

LIST OF ALGORITHMS

1 An algorithm to compute N_i's and C_i's 7

2 A faster algorithm to compute N_i's and C_i's 8

3 An algorithm select a suitable R 9

4 An algorithm to create a region tree for a homology class 12

5 An algorithm find the shortest cycle non-orthogonal to a given support vector S modulo p for a given region tree T_C 13

CHAPTER 1

INTRODUCTION

For a given connected graph $G = (V, E)$ with weights in $\mathbb{R}$, let $m = |E|$ and $n = |V|$ be the number of edges and vertices. We define a cycle to be a subset of the edges such that each vertex is incident to an even number of edges in the subset.

For undirected graphs we can form a vector space over cycles in G with addition defined as symmetric difference of the edges. It is well known that this vector space is isomorphic to $\mathbb{Z}_2^{m-n+1}$. For directed graphs we can then form a vector space over cycles in G with a more complicated definition of addition. Given a cycle we say an edge is traveling the correct direction if the direction of the edge agrees with the cycle. We say the the edge is traveling the incorrect direction otherwise. For example in Figure 1.1 for the cycle (e_1, e_2, e_3) all edges would be traveling in the correct direction, but for the cycle (e_3, e_2, e_1) all edges would be traveling in the incorrect direction. Edges traversed by a cycle in the correct direction are defined to be "positive" where cycles traversed in the incorrect direction are said to be "negative". Addition of cycles is then defined to be an element wise sum of the edges.

For either case using the vector space we can find a basis of the set, known as a cycle basis in the context of graphs, which is defined as a maximal set of $d = m - n + 1$ independent cycles. It is well known that the cycle basis of G can be generated by the fundamental cycles of any spanning tree of G. Given a weighted graph one can define the minimum cycle basis as a cycle basis of minimum weight. We note that for the minimum cycle basis of a directed graph, cycles are defined using only the underlying undirected graph, the direction of the edges is used only for determining if cycles are independent from each other. The minimum cycle basis has applications in many fields for both the directed [5, 9] and undirected [17, 4, 15] cases.

From the definition of independence in vector spaces, sets of independent cycles form a matroid. Therefore one can use the standard greedy algorithm of sorting and eliminating

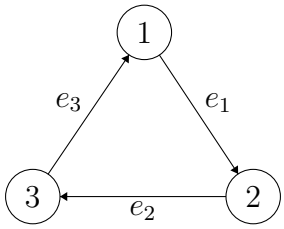

Figure 1.1. A simple directed graph

to find the minimum cycle basis. However the number of cycles in G can be potentially exponential. Horton reduced the search space for the greedy algorithm to $O(nm)$ cycles by showing that every cycle in the minimum cycle basis must be a fundamental cycle of a shortest path tree, giving the first polynomial time algorithm [13]. Several other deterministic polynomial time algorithms have been given [1, 12, 6, 16], the fastest for the undirected case being an $O(nm^2/\log n + n^2 m)$ time alrogithm by Mehlhorn and Michail [16] and an $O(m^3 n + m^2 n^2 \log n)$ time algorithm for the directed case by Hariharan, Kavitha, and Mehlhorn [11]. For graphs embedded on a surface of genus g, Borradaile, Chambers, Fox, and Nayyeri presented a $O(n^\omega + 2^{2g} n^2 + m)$ time algorithm for computing the minimum cycle basis, where $O(n^\omega)$ is the time is takes to multiply two $n \times n$ matrices using fast matrix multiplication [2].

We give an $O(mn^2 g^2 \log g + m^{\omega+1})$ time algorithm to find the minimum cycle basis for a surface embedded directed graph.

1.1 Preliminaries

The problem of finding a minimum cycle basis in a directed graph is very similar to the case of an undirected graph. The underlying undirected graph of a directed graph is the directed graph with the edge directions removed leaving just undirected edges. When finding cycles in the basis, we search for cycles in the underlying undirected graph, the direction of the edges are only used for determining if the new cycle is independent with existing cycles in the basis. However it is not possible to run the undirected algorithm to produce a minimum

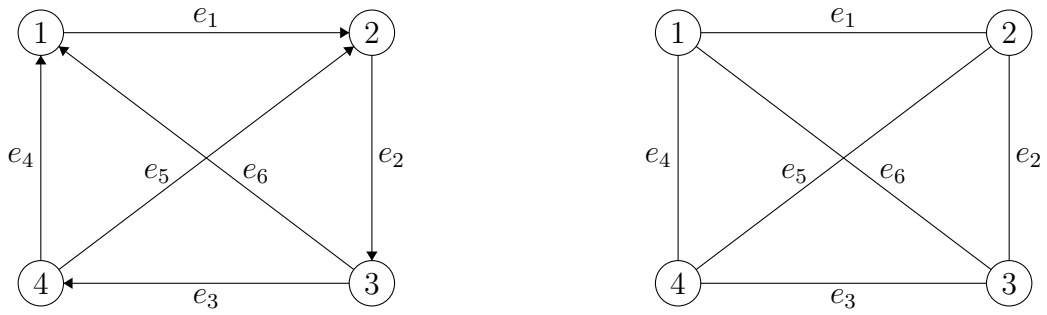

Figure 1.2. A directed graph with its underlying undirected graph [11]

cycle basis for the directed graph. A counter example is given in Figure 1.2 by Hariharan, Kavitha, and Mehlhorn [11]. In this figure we look at three cycles (e_1, e_2, e_3, e_4), (e_1, e_5, e_3, e_6) and (e_2, e_6, e_4, e_5). In the directed graph these cycles are linearly independent. However in the underlying undirected graph the sum of two of these cycles is the third implying they are linearly dependent. We do note that for the case of planar graphs the cycle bases of the directed graph are exactly that of the undirected graph, hence searching for a minimum cycle basis in the undirected graph is sufficient to find the minimum cycle basis of the directed graph.

A surface or a 2-manifold with boundary Σ is defined as a compact Hausdorff space such that every point lies in an open neighborhood homeomorphic to the Euclidean plane or the closed half plane. The boundary of the surface is the set of all points whose open neighborhood is homeomorphic to the closed half plane, and every boundary component is homeomorphic to the circle. A cycle in the surface is a continuous mapping from the unit circle to the surface, and the cycle is called simple if the mapping is injective. A path in the surface is a continuous mapping from $[0, 1]$ to the surface and is also called simple if the mapping is injective. A loop is a path with the same starting and ending positions or equivalently a cycle with a designated starting/ending point. A surface is said to be orientable if it does not contain a subset homeomorphic to the Möbius band, and non-orientable otherwise. The genus of a surface, g, is the maximum number of disjoint cycles

in a surface such that their removal leaves a surface that is still connected. Surfaces are homeomorphic if their genus, number of boundary components, and whether or not they are orientable all agree.

For a given graph $G = (V, E)$, we say G is embedded on Σ if there exists a mapping which maps vertices to distinct points on Σ and edges to internally disjoint paths on Σ with endpoints that lie on their incident vertices' points. A face of the embedding is defined to be a maximally connected subset of Σ such that the subset does not intersect the embedded graph. If every face on an embedding is homeomorphic to an open disk we say the embedding is cellular. Only orientable cellular embeddings of graphs will be considered from now on.

Every embedded graph G has a dual graph G^* which is constructed by creating a vertex for every boundary component in Σ as well as every face of G. Edges are then created between two vertices in G^* if the corresponding faces and boundary components are separated by an edge. Finally faces in G^* now correspond to vertices in G. The original graph is then known as the primal graph, where primal vertices are dual to dual faces, and dual vertices are dual to primal faces. Usually no distinction is made between primal and dual edges. We assume without loss of generality that the surfaces our graphs are embedded on contain only a single boundary component.

A spanning tree of G is a subset of edges of G that form a tree containing every vertex of G. A spanning coforest is a subset of dual edges that form b trees in the dual that contain every dual vertex, such that every tree contains a dual-boundary vertex, where b is the number of boundary components. A tree-coforest decomposition is a partition of the edges of G into three sets, T a spanning tree of G, C a spanning coforest of G, and L the leftover edges [7, 8].

Let (T, L, C) be an arbitrary tree-coforest decomposition of G. Let $\beta = |L|$. Define c_i for $i \in \{1, \ldots, \beta\}$ to be the unique simple co-cycle or unique co-path between distinct dual boundary verticies created by adding the ith edge in L to C, we also assign an arbitrary correct direction to this cycle for the purpose of the directed cycle signature. Let

$f_{\beta+1}, \ldots f_{m-n+1}$ denote the faces of G. Define c_i for $i \in \{\beta+1, \ldots, m-n+1\}$ to be the simple co-path from f_i to the dual boundary vertex in f_i's component of C. We define the cycle signature $[e] \in \{-1, 0, 1\}^{m-n+1}$ of an edge e as a vector with the ith component defined as follows.

$$[e]_i = \begin{cases} 1 & \text{if } e \text{ is traveling in the correct direction for } i \in \{1, \ldots, \beta\} \\ 1 & \text{if } e \text{ is traveling towards the root for } i \in \{\beta+1, \ldots, m-n+1\} \\ -1 & \text{if } e \text{ is traveling in the incorrect direction for } i \in \{1, \ldots, \beta\} \\ -1 & \text{if } e \text{ is traveling away from the root for } i \in \{\beta+1, \ldots, m-n+1\} \end{cases}$$

The cycle signature of any cycle C is the sum of the cycle signatures of the edges of C. Borradaile, Chambers, Fox, and Nayyeri show that a similar cycle signature definition for the undirected case produces an isomorphism to the cycle space [2]. To show the directed cycle signatures are isomorphic to the directed cycle space one follows the same argument.

The homology of G is an algebraic description of the topology of the surface as well as G's embedding. We are only concerned with the one-dimensional cellular homology over the finite field $\mathbb{Z}_2$ and use the underlying undirected graph when referencing the homology of G. We say a cycle is null-homologous if it is the boundary of a subset of faces. Two cycles are homologous if their symmetric difference is null-homologous. These definition allows us to split an orientable G into $2^{2g+\max\{b-1,0\}}$ homology classes. We define the operation $G \nmid C$ as cutting both G and Σ along some cycle C, creating two copies of C. Cutting G using any cycle from the null-homology class cuts Σ into two separate surfaces, and cutting G along any two non-crossing cycles from a single homology class cuts Σ into two separate surfaces.

CHAPTER 2

ALGORITHM

Our algorithm computes cycles of the minimum basis one by one. We do this by maintaining a set of support vectors that form the basis for the subspace orthogonal to the set of cycles already computed for the basis. To compute a new cycle in the basis we choose a support vector that we have not used so far and find the cycle of minimum weight that is not orthogonal the chosen support vector. The unchosen support vectors are then updated so they remain orthogonal to the current incomplete basis we have computed. This is the method that many algorithm have used to compute minimum cycle bases [2, 6, 11].

Specifically our algorithm uses the prime field modifications for directed graphs made by Hariharan, Kavitha, and Mehlhorn for dealing with the potentially large numbers generated by the coefficients from $\mathbb{Q}$ [11]. For choosing the non-orthogonal cycle our algorithm follows the basic idea of Borradaile, Chambers, Fox, and Nayyeri of constructing region trees based on homology signatures to improve the cycle selection time [2]. However, modifications must be made to the cycle selection procedure to account for the large coefficients from $\mathbb{Q}$.

2.1 Computing Support Vectors

The method of computing support vectors that are used to calculate the minimum cycle basis follows from the following theorem given and proved by Hariharan, Kavitha, and Mehlhorn [11].

Theorem 1. *Cycles $C_1, \ldots C_d$ form a minimum cycle basis if there are vectors $N_1 \ldots N_d$ in $\mathbb{Q}^m$ such that for all i, $1 \leq j \leq i$:*

1. *Prefix orthogonality: $\langle N_i, C_i \rangle = 0$ for all j, $1 \leq j < i$.*

2. *Nonorthogonality: $\langle N_i, C_i \rangle \neq 0$.*

3. Shortness: C_i is a shortest cycle with $\langle N_i, C_i \rangle \neq 0$

Algorithm 1 is a simple deterministic algorithm that was given by Kavitha and Mehlhorn to compute N_i's and C_i' [14].

Algorithm 1 An algorithm to compute N_i's and C_i's

$N_1, \ldots, N_d \leftarrow \hat{u_1}, \ldots \hat{u_d}$ ▷ ($\hat{u_d}$ has a 1 in the ith position and 0's everywhere else)
for $i \leftarrow 1$ to d **do**
  $C_i \leftarrow$ shortest cycle with non-zero dot product with N_i
  for $j \leftarrow i+1$ to d **do**
    $N_j \leftarrow N_j - N_i \frac{\langle C_i, N_j \rangle}{\langle C_i, N_i \rangle}$
    $N_j \leftarrow N_j \frac{\langle C_i, N_i \rangle}{\langle C_{i-1}, N_{i-1} \rangle}$
  end for
end for

The correctness of this algorithm is based on a lemma given and proved by Kavitha and Mehlhorn [14].

Lemma 2. *For any i, at the end of iteration $i-1$, the vectors $N_i, \ldots, N_d$ are orthogonal to $C_1, \ldots, C_{i-1}$ and moreover for any j with $i \leq j \leq d$,*

$$N_j = \langle C_{i-1}, N_{i-1} =\rangle (x_{j,1}, \ldots, x_{j,i-1}, 0, \ldots, 0, 1, 0, \ldots, 0)$$

where 1 occurs in the jth coordinate and the vector $\boldsymbol{x} = (x_{j,1}, \ldots, x_{j,i-1})$ is the unique solution to the set of equations:

$$\begin{pmatrix} \tilde{C}_1^T \\ \vdots \\ \tilde{C}_{i-1}^T \end{pmatrix} \boldsymbol{x} = \begin{pmatrix} -c_{1,j} \\ \vdots \\ -c_{i-1,j} \end{pmatrix}$$

Where $\tilde{C}_k$, $1 \leq k < i$, is the restriction of C_k to its first $i-1$ coordinates and $c_{k,j}$ is the jth coordinate of C_k.

Furthermore the running time of this algorithm was shown by Kavitha and Mehlhorn to be $\tilde{O}(m^4) + mO(\text{cycle})$, where $O(\text{cycle})$ is the time taken to find the shortest non-orthogonal cycle to N_i [14].

This simple algorithm was then improved upon by Hariharan, Kavitha, and Mehlhorn. By using a divide and conquer approach the calculations spent updating the N_i vectors can be done in bulk [11]. Algorithm 2 gives the recursive step from index l to index h is as follows.

Algorithm 2 A faster algorithm to compute N_i's and C_i's

$mid \leftarrow \lceil (l+h)/2 \rceil$
Find cycles $C_l, \ldots, C_{mid}$ using $N_l, \ldots, N_{mid}$ recursively
Update the vectors $N_{mid+1}, \ldots, N_h$
Find cycles $C_{mid+1}, \ldots, C_h$ using $N_{mid+1}, \ldots, N_h$ recursively

To compute the minimum cycle basis, we call this algorithm with $N_1, \ldots, N_d$ initialized to the first d unit vectors, $l = 1$, and $h = d$. Our goal when updating the vectors is to make the vectors $N_{mid+1}, \ldots, N_h$ orthogonal to the newly computed cycles $C_l, \ldots, C_{mid}$. To update the vectors, we make the following definitions.

- $A \in \mathbb{Q}^{k \times m}$, A's ith row is C_{l+i-1}
- $D \in \mathbb{Q}^{(h-k)\times(h-k)}$, D has the value $\langle N_{mid}, C_{mid} \rangle / \langle N_{l-1}, C_{l-1} \rangle$ in every diagonal
- $X \in \mathbb{Q}^{k \times (h-k)}$
- $N_d \in \mathbb{Q}^{m \times (h-k)}$, N_d's jth column is N_{mid+j}
- $N_u \in \mathbb{Q}^{m \times k}$, N_u's jth column is N_{l+j-1}

As shown by Hariharan, Kavitha, and Mehlhorn, we can update the vectors by solving the following system for X [11].

$$AN_dD = -AN_uX$$

By construction (NN_u) is lower triangular with non-zero diagonal entries, and therefore is invertible. Hence we can write.

$$X = -(AN_u)^{-1}AN_dD$$

Finally our updated vectors $N_{mid+1}, \ldots, N_h$ can be found by computing $N_uX + N_dD$. This process of updating the vectors takes $O(mk^{\omega-1})$ arithmetic operations in total, where ω is the time it takes to multiply two matrices using fast matrix multiplication. However, because we are in a directed graph, the elements of $(AN_u)^{-1}$ can be as large as $d^{\Theta(d^2)}$. This causes arithmetic operation to take up to $\tilde{\Theta}(d^2)$ time giving a runtime of $\tilde{\Theta}(m^{\omega+2})$ for the outermost step which is slower than the simpler algorithm for directed graphs [11].

In order to solve the problem of large intermediate elements we run the above algorithm over a ring $\mathbb{Z}_R$ where R is a specially chosen prime. Working over this ring allows us to do arithmetic operations in $O(1)$ time. In order to be able to recover our N_j vectors we must choose a R such that R is relativity prime to $\langle N_l, C_l\rangle, \langle N_{l+1}, C_{l+1}\rangle, \ldots, \langle N_{mid}, C_{mid}\rangle$ (to ensure AN_u is invertible in $\mathbb{Z}_R$), and relativity prime to $\langle N_{l-1}, C_{l-1}\rangle$ (to ensure that $\langle N_{mid}, C_{mid}\rangle / \langle N_{l-1}, C_{l-1}\rangle$ is well defined in $\mathbb{Z}_R$) [11]. We can select R for each iteration of the recursive step using Algorithm 3.

Algorithm 3 An algorithm select a suitable R

Require: $p_1, \ldots, p_{d^2}$, primes each of which is at least d, the products $P_1 = p_1 \ldots p_d, P_2 = p_1 \ldots p_{2d}, \ldots, P_d = p_1 \ldots p_{d^2}$ precomputed before running algorithm.
$L \leftarrow \langle N_{l-1}, C_{l-1}\rangle \langle N_l, C_l\rangle \ldots \langle N_{mid}, C_{mid}\rangle$
Binary search $P_1 \ldots P_d$ to find the smallest $s \geq 0$ such that $P_{s+1} \nmid L$
Determine a $p \in \{p_{sd+1}, \ldots, p_{sd+d}\}$ such that $p \nmid L$
return p^d

Pre-computing d^2 primes takes $\tilde{O}(d^2)$ and pre-computing the products $P_1, \ldots, P_d$ takes $\tilde{O}(d^3)$. The algorithm to select R runs in $\tilde{O}(d^2)$ [11]. Finally all together the total time complexity for the update step with modulo arithmetic is $\tilde{O}(m^2k^{\omega-1} + d^2k)$ or $\tilde{O}(m^2k^{\omega-1})$.

2.2 Finding the Minimum Cycle

What remains is to find the cycle of minimum weight that is non-orthogonal to the current support vector. To do this quickly we modify an algorithm given by Borradaile, Chambers, Fox, and Nayyeri [2]. This algorithm first reduces the candidate cycles to $O(2^{2g}n)$ cycles for a graph of genus g. The cycles are then partitioned into sets and a tree is built using these sets in $O(2^{2g}n^2)$ time. This tree can then be searched in $O(2^{2g}n)$ time to find the minimum non-orthogonal cycle [2].

A Horton cycle is defined to be a simple cycle given by a shortest x, u path, a shortest x, v path, and the edge uv. The set of all Horton cycles on a graph is given by the set of $m - n + 1$ elementary cycles of the n shortest path trees [13]. A simple cycle C is said to be isometric if for all $x, y \in C$, C contains the shortest x, y path.

Theorem 3. *All cycles in the minimum cycle basis of a directed graph G are isometric Horton cycles.*

Proof. For a minimum cycle basis C of G, let $c \in C$ and $C' = C \setminus \{c\}$. We use the definition of three path condition as given by Cabello, Colin de Verdière, and Lazarus [3]. For any three path α, β, γ in G we have that if $\alpha - \beta$ is dependent on cycles in C' and $\beta - \gamma$ is dependent on cycles in C', $\alpha - \gamma$ is dependent on cycles in C' by simple algebra on the cycle signatures. Therefore the set of cycles dependent on C' forms a family that follows the three path condition. Cabello, Colin de Verdière, and Lazarus then show that the shortest cycle that is not in a family that follows the three path condition can be given by a fundamental cycle of a shortest path tree, and that this shortest path tree can be rooted anywhere along the cycle [3]. Hence the shortest cycle that is independent from those in C' is isometric. □

Building off the work of Hartvigsen and Mardon, Borradaile, Chambers, Fox, and Nayyeri show that there are $O(2^{2g})$ diffrent homology classes each with $O(n)$ distinct isometric cycles,

giving $O(2^{2g}n)$ distinct isometric cycles in a graph of orientable genus g with unique shortest paths [2]. We now assume that our graphs are orientable have unique shortest paths. Because we have isometric cycles they cross at most once. Greene shows that cycles that cross at most once live in at most $O(g^2 \log g)$ homotopy classes [10]. The number of homotopy classes also bounds the number of homology classes, which would reduce the number of isometric cycles to $O(ng^2 \log g)$. This improvement on the bound on the number of homology classes also improves the runtime given by Borradaile, Chambers, Fox, and Nayyeri.

In order to apply the algorithm given by Borradaile, Chambers, Fox, and Nayyeri, we must first convert it to use cycle signatures in $\mathbb{Q}^d$ instead of $\mathbb{Z}^d$. However the elements of the support vector we are given can have elements of up to size $d^{d/2}$ so naively modifying the algorithm will lead to problems with the speed of arithmetic. Instead we use a similar approach to that of Hariharan, Kavitha, and Mehlhorn and first find the shortest cycle that is non-orthogonal to our chosen support vector modulo some prime p.

We first describe how to construct the trees. We note that this step is done once before running the main algorithm and does not depend on the prime p. We begin by computing all Horton cycles of the graph and extracting the isometric cycles as shown by Amaldi et al [1]. Next we compute the homology signatures for these cycles and split them into $O(g^2 \log g)$ homology classes. Then for each homology class we compute the region tree. Each vertex of the region tree will correspond to a set of faces, where each edge will correspond to an edge. The region trees are then computed in Algorithm 4.

In Algorithm 4 the operation $G \nmid \gamma$ takes $O(n)$ time, so the entire algorithm takes $O(n^2)$ for each region tree. Therefore we can preprocess G in $O(n^2 g^2 \log g)$ time. Examples of constructed region trees can be seen in Figures 2.1 and 2.2.

We can then use the region trees to find the shortest cycle that is non-orthogonal modulo p with a given support vector N_i as shown in Algorithm 5. This algorithm is based on that of Borradaile, Chambers, Fox, and Nayyeri with modifications to account for the prime p

Algorithm 4 An algorithm to create a region tree for a homology class

Require: Set of cycles C all belonging to the same homology class and the graph G

if C has non-trivial homology **then**
  T_C starts out with one vertex containing all faces
  We choose arbitrary $\gamma_0 \in C$
  T_C has a single edge looping on the single vertex referring to γ_0
  $G' \leftarrow G \nmid \gamma_0$
  for $\gamma \in \{C \setminus \gamma_0\}$ **do**
    $G' \leftarrow G' \nmid \gamma$
    Cutting G' splits some component of G' into two new components
    Split vertex corresponding to cut component into two new vertices with corresponding faces from the newly created components
    Assign γ to the new edge
  end for
  Remove edge corresponding to γ_0
  Root T_C at the vertex containing the boundary component
else
  T_C starts out with one vertex containing all faces
  $G' \leftarrow G$
  for $\gamma \in C$ **do**
    Cutting G' splits some component of G' into two new components
    Split vertex corresponding to cut component into two new vertices with corresponding faces from the newly created components
    Assign γ to the new edge
  end for
  Root T_C at the vertex containing the boundary component
end if

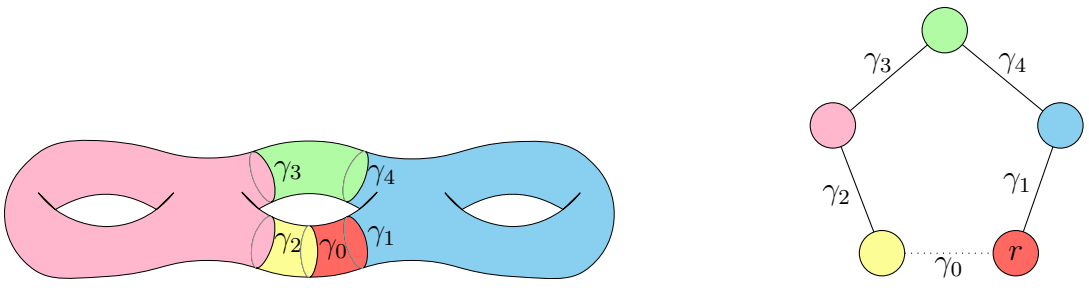

Figure 2.1. A region tree for a set of edges with non-trivial homology signature [2]

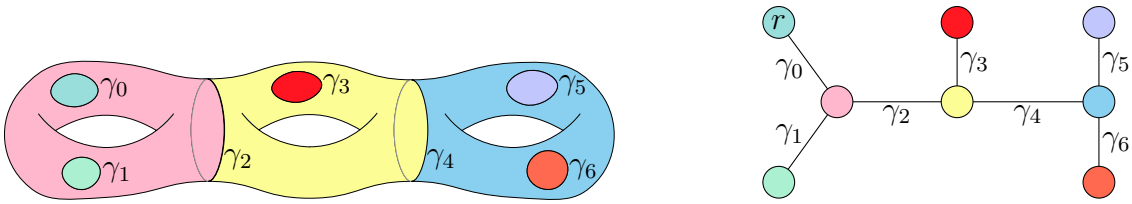

Figure 2.2. A region tree for a set of edges with null-homologous homology signature [2]

needed for directed cycles [2]. The *sign* function as used in Algorithm 5 takes the first edge above the given face in the spanning coforest and returns 1 if the edge is directed towards the root and -1 otherwise, this can be precomputed and stored while constructing the region trees. This algorithm considers each face of the graph at most once, so the total runtime to walk up the tree is $O(n)$. We must run this algorithm once for every homology class, giving a total runtime to find a non-orthogonal cycle modulo p of $O(ng^2 \log g)$.

Algorithm 5 An algorithm find the shortest cycle non-orthogonal to a given support vector S modulo p for a given region tree T_C

Require: Set of cycles C all belonging to the same homology class and the graph G
  $m \leftarrow \infty$ ▷ The current minimum weight
  $c \leftarrow$ NULL ▷ The current cycle referring to the minimum weight
  for edge $e \in T_C$ going to a leaf **do**
    $z \leftarrow 0$
    for $f_i \in F(\text{bottom}(e))$ **do** ▷ Every face referenced by vertex at the lower end of e
      $z \leftarrow z +_p S_i * \text{sign}(f_i)$
    end for
    if $z \neq 0$ and $\text{weight}(e) < m$ **then** update m and c with e
  end for
  loop Walk up the tree from the leaves, with current edge e
    Combine computed z's from children
    for $f_i \in F(\text{bottom}(e))$ **do**
      $z \leftarrow z +_p S_i * \text{sign}(f_i)$
    end for
    if $z \neq 0$ and $\text{weight}(e) < m$ **then** update m and c with e
  end loop
  return c

In order to obtain the shortest non-orthogonal cycle from a collection of shortest non-orthogonal cycles modulo p, we first pre-compute primes $p_1, \ldots, p_{d/2}$ each of which is at least d. The ring $\mathbb{Z}_{\prod p_i}$ is isomorphic to $\mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_{d/2}}$, which implies that any non-zero element whose magnitude is less than $\prod_{i=1}^{d/2} p_i$ is mapped to a tuple of values that is not the zero vector. Therefore if run our algorithm for cycle searching $d/2$ times, once for each prime we pre-computed, each cycle that is non-orthogonal will also be non-orthogonal from some $p \in \{p_1, \ldots, p_{d/2}\}$. This gives us a total runtime of $O(mng^2 \log g)$ to find the shortest non-orthogonal cycle for some given support vector.

2.2.1 Results

Using the recurrence from calculating the support vectors we have our final runtime to be.

$$T(k) = \begin{cases} 2T(k/2) + \tilde{O}(m^2 k^{\omega-1}) & \text{if } k > 1 \\ mng^2 \log g & \text{if } k = 1 \end{cases}$$

This recurrence solves to $T(O(n)) = O(mn^2g^2 \log g) + \tilde{O}(m^{\omega+1})$. Giving a final runtime of computing the minimum cycle basis of an embedded directed graph of $O(mn^2g^2 \log g + m^{\omega+1})$. We note that any method to improve the speed of selecting support vectors would improve the time required to find the minimum cycle basis. We would also like to highlight that the improvements made by Greene [10] improve the results by Borradaile, Chambers, Fox, and Nayyeri for finding the minimum cycle basis of an undirected embedded graph from $O(n^\omega + 2^{2g}n^2 + m)$ [2] to $O(n^\omega + n^2g^2 \log g + m)$.

REFERENCES

[1] Amaldi, E., C. Iuliano, T. Jurkiewicz, K. Mehlhorn, and R. Rizzi (2009). Breaking the o(m2n) barrier for minimum cycle bases. In A. Fiat and P. Sanders (Eds.), *Algorithms - ESA 2009*, Berlin, Heidelberg, pp. 301–312. Springer Berlin Heidelberg.

[2] Borradaile, G., E. W. Chambers, K. Fox, and A. Nayyeri (2017). Minimum cycle and homology bases of surface-embedded graphs. *J. Comput. Geom. 8*(2), 58–79.

[3] Cabello, S., E. Colin de Verdière, and F. Lazarus (2010). Finding shortest non-trivial cycles in directed graphs on surfaces. In *Proceedings of the Twenty-Sixth Annual Symposium on Computational Geometry*, SoCG '10, New York, NY, USA, pp. 156–165. Association for Computing Machinery.

[4] Cassell, A. C., J. C. D. C. Henderson, K. Ramachandran, and A. W. Skempton (1976). Cycle bases of minimal measure for the structural analysis of skeletal structures by the flexibility method. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences 350*(1660), 61–70.

[5] Chua, L. and L.-K. Chen (1973). On optimally sparse cycle and coboundary basis for a linear graph. *IEEE Transactions on Circuit Theory 20*(5), 495–503.

[6] de Pina, J. C. (1995). *Applications of shortest path methods*. Ph. D. thesis, University of Amsterdam.

[7] Eppstein, D. (2003). Dynamic generators of topologically embedded graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '03, USA, pp. 599–608. Society for Industrial and Applied Mathematics.

[8] Erickson, J. and A. Nayyeri (2011). Minimum cuts and shortest non-separating cycles via homology covers. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, USA, pp. 1166–1176. Society for Industrial and Applied Mathematics.

[9] Gleiss, P., J. Leydold, and P. Stadler (2001, 01). Circuit bases of strongly connected digraphs. *Santa Fe Institute, Working Papers 23*.

[10] Greene, J. E. (2019, Dec). On loops intersecting at most once. *Geometric and Functional Analysis 29*(6), 1828–1843.

[11] Hariharan, R., T. Kavitha, and K. Mehlhorn (2008). Faster algorithms for minimum cycle basis in directed graphs. *SIAM Journal on Computing 38*(4), 1430–1447.

[12] Hartvigsen, D. and R. Mardon (1994, August). The all-pairs min cut problem and the minimum cycle basis problem on planar graphs. *SIAM J. Discret. Math. 7*(3), 403–418.

[13] Horton, J. (1987, 04). A polynomial-time algorithm to find the shortest cycle basis of a graph. *SIAM J. Comput. 16*, 358–366.

[14] Kavitha, T. and K. Mehlhorn (2007, 06). Algorithms to compute minimum cycle basis in directed graphs. *Theory of Computing Systems 40*, 485–505.

[15] Knuth, D. E. (1997). *The Art of Computer Programming, Volume 1 (3rd Ed.): Fundamental Algorithms*. USA: Addison Wesley Longman Publishing Co., Inc.

[16] Mehlhorn, K. and D. Michail (2010, December). Minimum cycle bases: Faster and simpler. *ACM Trans. Algorithms 6*(1).

[17] Tewari, G., C. Gotsman, and S. Gortler (2006, 12). Meshing genus-1 point clouds using discrete one-forms. *Computers & Graphics 30*, 917–926.

BIOGRAPHICAL SKETCH

Thomas Stanley was born in Covington, Louisiana. He attended San Marcos High School in San Marcos Texas. He received a Bachelor of Science with majors in both Computer Science and Mathematics from The University of Texas at Dallas in Fall of 2019. He entered the Computer Science graduate program at The University of Texas at Dallas in the Spring of 2020. He is currently employed at Ligo Analytics as a Software Engineer developing structural biology analysis programs. He also does contracting work for The University of Texas Southwestern Medial Center developing algorithms to assist quantifying brain cancer progression.

CURRICULUM VITAE

Thomas Stanley

December 7, 2021

Contact Information:

tas150430@utdallas.edu

Education:

BS, Computer Science and Mathematics, University of Texas at Dallas, 2019

Employment History:

Research Assistant, The University of Texas at Dallas, January 2018 – December 2020
Software Engineer Contractor, The University of Texas Southwestern, January 2021 – Present
Software Engineer, Ligo Analytics, May 2021 – Present

Publications:

[1] Hansen, M. R., E. Pan, A. Wilson, M. McCreary, Y. Wang, T. Stanley, M. C. Pinho, X. Guo, and D. T. Okuda (2018, Sep). Post-gadolinium 3-dimensional spatial, surface, and structural characteristics of glioblastomas differentiate pseudoprogression from true tumor progression. *J Neurooncol 139*(3), 731–738.

[2] Bereg, S., B. Malouf, L. Morales, T. Stanley, I. H. Sudborough, and A. Wong (2019). Equivalence relations for computing permutation polynomials. *CoRR abs/1911.12823*.

[3] Sivakolundu, D. K., M. R. Hansen, K. L. West, Y. Wang, T. Stanley, A. Wilson, M. McCreary, M. P. Turner, M. C. Pinho, B. D. Newton, X. Guo, B. Rypma, and D. T. Okuda (2019, 09). Three-Dimensional Lesion Phenotyping and Physiologic Characterization Inform Remyelination Ability in Multiple Sclerosis. *J Neuroimaging 29*(5), 605–614.

[4] Moog, T. M., M. McCreary, T. Stanley, A. Wilson, J. Santoyo, K. Wright, M. D. Winkler, Y. Wang, F. Yu, B. D. Newton, B. Zeydan, O. Kantarci, X. Guo, and D. T. Okuda (2020, Oct). African americans experience disproportionate neurodegenerative changes in the medulla and upper cervical spinal cord in early multiple sclerosis. *Multiple Sclerosis and Related Disorders 45*.

[5] Okuda, D. T., T. M. Moog, M. McCreary, J. N. Bachand, A. Wilson, K. Wright, M. D. Winkler, O. G. Ramos, A. P. Blinn, Y. Wang, T. Stanley, M. C. Pinho, B. D. Newton, and X. Guo (2020, Nov). Utility of shape evolution and displacement in the classification of chronic multiple sclerosis lesions. *Scientific reports 10*(1), 19560–19560.

[6] Sivakolundu, D. K., K. L. West, M. D. Zuppichini, A. Wilson, T. M. Moog, A. P. Blinn, B. D. Newton, Y. Wang, T. Stanley, X. Guo, B. Rypma, and D. T. Okuda (2020, Oct). Bold signal within and around white matter lesions distinguishes multiple sclerosis and non-specific white matter disease: a three-dimensional approach. *Journal of Neurology 267*(10), 2888–2896.

[7] Bereg, S., B. Malouf, L. Morales, T. Stanley, and I. H. Sudborough (2021). Improved lower bounds for permutation arrays using permutation rational functions. In J. C. Bajard and A. Topuzoğlu (Eds.), *Arithmetic of Finite Fields*, Cham, pp. 234–252. Springer International Publishing.