ABSTRACT

BREEN-MCKAY, MICHAEL J. A Combinatorial Approach to Problems in Evolutionary Biology and Sparse Networks. (Under the direction of Blair D. Sullivan and Seth Sullivant.)

Graphs allow for relationships in complex systems to be modeled effectively by relatively simple constructions. Graph theory bridges the line between theoretical computer science and discrete math. The work in this thesis focuses on two graph theoretic topics and gives combinatorial and algorithmic results to problems in both.

The r-coloring number is a graph invariant that is closely related to sparsity theory, giving a characterization of graph classes with bounded expansion. In addition, its value guarantees the existence of a linear order of the vertices with interesting properties for parameterized algorithms. We show that the decision version of the r-coloring problem is **NP**-complete via a reduction from a variant of 3-SAT. Additionally, this reduction implies that as a parameterized problem deciding the r-coloring is **para-NP**-hard and as an optimization problem it does not admit a PTAS (polynomial time approximation scheme). Using structural properties of r-coloring orders the best known approximation algorithm is presented that runs in quadratic time on the input size.

Phylogenetic trees are used in evolutionary biology to model speciation and are often inferred using genetic data. When phylogenies are formed from different genes the resulting graph structures are rarely compatible. A mixture model combines distinct probability distributions for each gene tree into a single probability measure using Markov processes. This requires that the set of input tree parameters are identifiable. Using graph isomorphism arguments we show that multisets consisting of four trees with $N \geq 6$ leaves are identifiable by their induced subtrees on six leaves. We also develop a tropical supertree method based on the relationship between weighted phylogenetic trees and tropical geometry. Based on the tropical metric we utilize techniques from linear programming and polyhedral geometry to describe the set of tropical supertrees.

A Combinatorial Approach to Problems in Evolutionary Biology and Sparse Networks

by
Michael J. Breen-McKay

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Mathematics

Raleigh, North Carolina

2022

APPROVED BY:

Ernest Stitzinger	Nathan Reading
Radmila Sazdanovic	Donald Sheehy
Blair D. Sullivan Co-chair of Advisory Committee	Seth Sullivant Co-chair of Advisory Committee

DEDICATION

For my parents, Rita Breen and Bill McKay, without their love and support I would be lost.
And for every person that showed me kindness through this process.
I am forever in your debt.

BIOGRAPHY

Michael was born down the street from Walden Pond, in Concord, Massachusetts and spent his first 23 years living just north of Boston. After stints as a banker, a sales representative, a web designer, a social worker, and an heirloom seed importer he made the decision to go back to school and become a math teacher. This led him to the mountains of western North Carolina, and UNC Asheville. On the way to graduating *summa cum laude* he realized there was a world of higher level mathematics that he had never been exposed to and made the decision to pursue a Ph.D. at North Carolina State University in Raleigh, North Carolina. Aside from math, Michael loves competitive disc golf, cooking, hiking, camping, and occasionally gets talked into rock-climbing.

ACKNOWLEDGEMENTS

This thesis is the culmination of many years of hard and sometimes discouraging research. The process has taught me a great deal about mathematics, but perhaps even more about myself. I am proud to have reached this point, and I am excited to use this knowledge in my future endeavors. With that being said, I could not have gotten here without the help of so many people.

I would like to thank my advisors, Blair D. Sullivan and Seth Sullivant. Without their guidance, hard work, and patience this would not have been possible. Through our many meetings, editing sessions and research ideas, they were always generous with their time and knowledge.

I would also like to extend my appreciation to the members of my committee: Radmila Sazdanovic, Ernest Stitzinger, Nathan Reading, and Donald Sheehy. Thank you for your support and availability throughout my time in graduate school. I was lucky enough to learn about topology from Dr. Sazdanovic, algebra from Dr. Stitzinger, and Coxeter groups from Dr. Reading and these classes were highlights of my time at NC State.

The first few years of grad school would have been much less enjoyable without the group from Laundry 108. I am particularly thankful for the conversations and intense games of pool and darts with Alex Chandler, Seth Watkins, Dustin Leininger, and Ben Hollering.

A special thanks to all the members of the research group Theory in Practice and in particular my collaborator Brian Lavallee who provided a great deal of insight and guidance.

I would also like to thank my friends and family outside of school, as they were equally important in my success, and at times the most neglected. I cannot list all of you, but to anyone that was a part of my life during grad school, thank you for putting up with me. First and foremost, my parents Rita and Bill, who have been there for me every step of the way and have always been models for the type of person I would like to be. Thanks to my brother Tim, who always told me I was doing great, even when it wasn't true. I also would like to thank my uncles, Jeff and Harry. Jeff, who passed away in 2020 inspires me every day to be as kind as possible to those around me. Harry has been one of my biggest supporters and I consider him one of my closest friends. I would also like to thank my biological mother Sandra Clark (1960-2012) and grandmother Evelyn Cronin (1934-2022). I am a better person because of them and I know they would be proud. Thank you to my GDG family. Alex, Darrus, John, and Kenny you are the best crew to throw discs through the woods with.

Finally, I would like to thank my partner Chelsea. She has given me a level of happiness and support that I did not think I could have or deserved, all while making sure that I ate vegetables and swam in rivers. I cannot wait for the rest of our adventures together.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER

1 INTRODUCTION

The field of graph theory straddles the line between discrete math and theoretical computer science. Among their many uses, graphs are studied for their combinatorial properties as discrete objects [Die05], ability to model real-world relationships [Chu10], and their efficacy as data structures in algorithm development [Eve11]. Often, when used to model real-world data, the resulting graphs are relatively sparse [Chu10].

In this thesis we present work on two topics that involve sparse graphs: the generalized coloring numbers and phylogenetic trees. The generalized coloring numbers are known to be small in sparse graph classes, notably those of bounded expansion. Additionally, they allow for the construction of linear vertex orders that have algorithmic implications. In this work we consider whether they can be computed or approximated efficiently. Phylogenetic trees use acyclic graphs with leaves labeled by species to model evolutionary relationships. They are typically studied in the field of evolutionary biology, but the underlying structure is combinatorial in nature and can also be analyzed in continuous geometric tree spaces. The work on phylogenetic trees in this thesis focuses on two problems: identifiability and supertree methods. We first look at whether fixed size sets and multisets of trees with identical labels can be identified by their induced subtrees. This is useful in proofs that the tree parameters in Markov processes can be recovered. Next we consider sets of phylogenetic trees with overlapping species and develop a method using tropical geometry to find a tree on the union of their labels that captures their structure.

The remainder of this chapter provides the basic definitions, notation, and concepts that we will need and also outlines the remainder of the thesis.

1.1 Graph Theory

We begin with a brief overview of definitions, results, and notation in graph theory. These will be used extensively in the remaining chapters as the generalized coloring numbers are graph invariants and phylogenetic trees are graphical models of speciation. Most of the graph notation will follow that of [Die05].

Definition 1.1.1. A *simple graph*, $G = (V, E)$, is a pair of sets where V is a set of *vertices* or *nodes* and E is a set of *edges*. Each edge of E is itself a pair of distinct vertices from V. If G is *undirected* an edge is a set $e = \{x, y\}$. If G is *directed* an edge is an ordered tuple $e = (x, y)$, which represents an edge pointing from x to y. Note: This definition implies there are no duplicate edges or loops (e.g. $\{x, x\}$) which is why it is considered simple.

We point out that the undirected edges $\{x, y\}$ and $\{y, x\}$ are equivalent while the directed edges (x, y) and (y, x) are not. If it is clear that a graph is undirected we will write the edge $\{x, y\}$ as *xy* or *yx*, interchangeably. If it is clear that a graph is directed we will likewise write *xy* for the edge (x, y).

Definition 1.1.2. For a graph $G = (V, E)$ and $x, y \in G$ if $xy \in E$ then we say x and y are *adjacent* or *neighbors*. Both are *incident* to the edge xy. The number of edges incident to x is the *degree of* x denoted $\deg(x)$. The set of all neighbors of x, $N(x)$, is called the *neighborhood of* x.

Definition 1.1.3. Given two graphs G, G' a *graph isomorphism* is a bijective function $f : V(G) \to V(G')$ so that $xy \in E(G)$ if and only if $f(x)f(y) \in E(G')$. If such a function exists we say G and G' are *isomorphic* and write this as $G \equiv G'$.

Definition 1.1.4. Let G be a graph and $u, v \in (G)$. An x, y*-path* P in G is a collection of distinct vertices $x = v_0, v_1, ..., v_r = y$ so that $v_i v_{i+1} \in E(G)$, for $0 \le i \le r-1$. The *length* of P, $\text{len}(P)$, is the number of edges in the path. The *distance* between x and y is the length of the shortest path between them, denoted $d_G(x, y)$. The *radius* of a graph is given by $\text{rad}(G) = \min_{x \in V(G)} \max_{y \in V(G)} d_G(x, y)$.

The notion of neighborhoods can be generalized to a radius r. Specifically, the r*-neighborhood* of a vertex v is the set of vertices that are at most distance r from v. We denote this as $N_r(v)$ and use $N_r[v] := N_r(v) \cup \{v\}$ to denote the *closed* r*-neighborhood*.

Definition 1.1.5. A graph G is called *connected* if for all $x, y \in V(G)$ there exists and x, y-path in G. Otherwise we say G is *disconnected*.

Definition 1.1.6. A *tree* $T = (V, E)$ is a connected graph that is acyclic. A *forest* is the disjoint union of trees. The degree one vertices are called *leaves* and an edge incident to a leaf is *external*. All other edges and non-leaf vertices are called *internal*. We use $\mathcal{L}(T)$ to denote the set of all leaves of the tree T.

The structure of a tree will be used throughout the paper and, in particular, they are the basis of the work in Part II. Often we will want to consider the substructures of a graph G. The following definitions provide some of the more common examples.

Definition 1.1.7. For two graphs $G = (V(G), E(G))$ and $H = (V(H), E(H))$, H is a *subgraph* of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$, which we write $H \subseteq G$. H is an *induced subgraph* of G, $H \subseteq_i G$ if it is a subgraph and for all $x, y \in V(H)$ if $xy \in E(G)$ then $xy \in E(H)$.

Definition 1.1.8. A graph G is a *subdivision* of a graph W if G can be formed from W by replacing edges in W with paths. It is an *r-subdivision* if these paths have length at most r and a *complete r-subdivision* if G results from replacing every edge in W by a path of length r.

Definition 1.1.9. A graph G is an *inflation* of a graph W if $V(G)$ admits a partition $\{V_x \subseteq V(G) | x \in V(W)\}$ into connected subsets V_x such that $xy \in E(W)$ if and only if G contains a $V_x - V_y$ edge. G is a *d-inflation* if it is an inflation and in some partition V_x has radius at most d for all $x \in V(W)$.

Definition 1.1.10. Let G and H be graphs. We say H is:

- a *topological minor* of G, denoted $H \preccurlyeq_T G$, if G contains a subdivision of H as a subgraph,
- an *a-shallow topological minor* (where a is a half-integer) of G, denoted $H \preccurlyeq_T^a$, if G contains a $2a$-subdivision of H,
- a *minor* of G, denoted $H \preccurlyeq G$, if G contains an inflation of H as a subgraph,
- a *d-shallow minor* of G, denoted $H \preccurlyeq_d G$, if G contains a d-inflation of H.

The minor and topological minor relationships can also be described in terms of graph edits. Consider the following types of edits on the graph $G = (V(G), E(G))$:

1. (Edge deletion) For $e \in E(G)$, deleting e from G gives the graph:

$$G - e = (V(G), E(G) \setminus e).$$

2. (Vertex deletion) For $v \in V(G)$, deleting v from G gives the graph:

$$G - v = (V(G) \setminus v, E(G) \setminus \{vx \in E(G)\}).$$

3. (Contraction) For H, a connected subgraph of G, contracting H gives the graph G/H:

$$\begin{aligned} V(G/H) &= (V(G) \cup \{v_H\}) \setminus V(H) \\ E(G/H) &= (E(G) \setminus E(H)) \cup \{v_H x : yx \in E(G), y \in H, x \in V(G) \setminus V(H)\}). \end{aligned}$$

4. (Vertex suppression) For $v \in V(G)$ with degree 2 and neighbors x and y, suppressing v gives the graph G':

$$V(G') = (V(G) - v)$$
$$E(G') = E(G) - \{vx, vy\} + xy.$$

A graph H is a subgraph if it can be formed from G by edge deletions and vertex deletions. If we restrict this to only vertex deletions then it is an induced subgraph. A graph H is a minor of G if H can be formed by performing edge deletion, vertex deletion and contraction on G. It is a topological minor if H can be formed through edge deletion, vertex deletion and vertex suppression. We note that if H is a topological minor of G then it is also a minor of G as vertex suppression is a special type of contraction.

Example 1.1.11. To demonstrate the definitions in this section we consider the graph $G = (V(G), E(G))$, where $V(G) = \{a, b, c, d, e\}$ and $E(G) = \{ab, ac, bc, cd, de\}$ seen on the left of Figure 1.1. Notice that if we delete the edge ac we get a tree that is a subgraph of G and is also an a-e path shown in Figure 1.2. However if we delete the vertex c we get a forest, which is the subgraph induced on a, b, d, e as in Figure 1.3.

Now we let H be the connected subgraph of G induced on a, b, and c that is highlighted in Figure 1.1. then G/H is given by contracting this set to a vertex v_H as shown in Figure 1.4. Note that since H has radius one, G/H is a 1-shallow minor of G.

The graph in Figure 1.5 is a complete 2-subdivision of G, making G a 1-shallow topological minor of it. The graph in Figure 1.6 shows the result of suppressing the vertex d in G.

1.2 Computational Complexity

In this section we give a short primer on the relevant topics in computational complexity. Unless otherwise noted these definitions follow from the graduate text by Arora and Barak [Aro09], which we also recommend for further details on the topic.

We consider the underlying question in the field of computational complexity: Can a given problem be solved efficiently? To understand what is meant by “solved efficiently” it is necessary to explain the types of problems we are considering, and how we measure the efficiency of the method. These two things are naturally linked.

Definition 1.2.1. An *algorithm* is a finite sequence of well defined instructions, that typically solves a class of problems. *Computational problems* are problems that are solvable by an algorithm.

In the literature the canonical example of an algorithm is a *Turing machine*, an abstract machine that moves along an infinite tape reading and writing symbols based on a finite number of states.

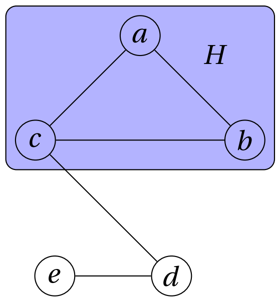


Figure 1.1 A graph G with subgraph H highlighted

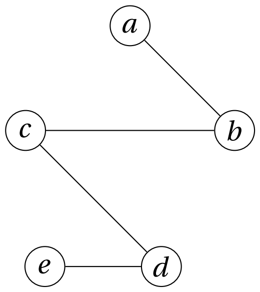


Figure 1.2 $G-(ac)$

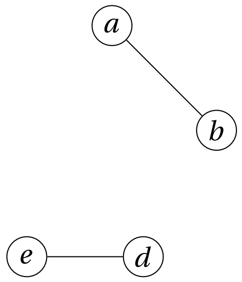


Figure 1.3 $G-c$

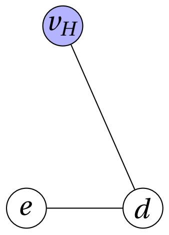


Figure 1.4 G/H

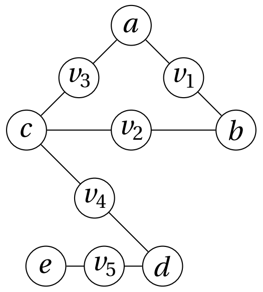


Figure 1.5 The complete subdivision of G.

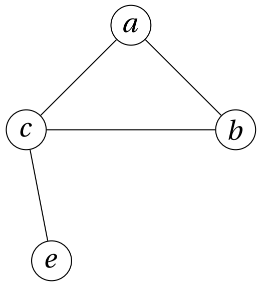


Figure 1.6 The result of suppressing the vertex d.

The combination of the machine's current state and the symbol read determines if the machine either continues to a next instruction or halts. If at each state and symbol there is only one possible instruction then the machine is said to be *deterministic* and otherwise it is *nondeterminisitic* Any algorithm can be given as a Turing machine, so its simplistic nature provides a good basis to measure complexity.

Often in computational complexity the underlying problem is given as a *decision problem*; a problem with a TRUE-FALSE solution. Such a problem encoded into binary strings can be given as a *Boolean function*, $f(x) : \{0,1\}^i \rightarrow \{0,1\}$ where i is a non-negative integer and the output value 1 represents TRUE and 0 represents FALSE. Problems that are phrased as computing a minimum or maximum value can be given as a collection of decision problems with an additional input. For example, consider the problem of finding the shortest path between two vertices in a graph. We can alternatively choose a value k and instead ask if there exists a path between two vertices of length at most k. The smallest k for, which the solution is TRUE is then the minimum such k. The *time complexity* of an algorithm is determined by the maximum number of basic steps it takes to find a solution. This quantity depends on the size of the input so time complexity is usual given as a function of the size of the input. For complicated algorithms it becomes tedious to accurately count these basic operations and the number of operations needed to perform a fixed command often depends on the implementation. Because of this, it is common to refer to the asymptotic behavior

Table 1.1 Common algorithm running times on an input of size n.

Time complexity	Running time in big-O notation
Constant time	$O(1)$
Linear time	$O(n)$
Quadratic time	$O(n^2)$
Polynomial time	$O(n^a)$, where $a \geq 1$
Logarithmic time	$O(\log(n))$
Exponential time	$O(2^{p(n)})$, where $p(n)$ is a polynomial function of n
Factorial time	$O(n!)$

of the function captured by *big-oh notation* to categorize computational problems.

Definition 1.2.2. If f and g are two functions from $\mathbb{N}$ to $\mathbb{N}$, then we say $f = O(g)$ if there exists a constant c such that $f(n) \leq c \cdot g(n)$ for every sufficiently large n.

The categories or *complexity classes* are divided based on the big-oh running times of the algorithms that solve them. For problems on graphs we often use the number of vertices, edges, or a combination to describe the size of the input. Suppose we have a hypothetical deterministic algorithm that outputs the solution to a problem by running through each vertex of a graph with n vertices, and performing 3 basic operations at each step. Then the running time of the algorithm would be $f(n) \approx 3n = O(n)$. If an algorithm has a running time equal to $O(n)$, then we say it runs in *linear time*. Table 1.1 gives some of the commonly used running times.

Definition 1.2.3. The complexity class **P** consists of all decision problems that are computable by a deterministic Turing machine in $O(n^a)$ time (polynomial time), for $a \geq 1$.

The problems in **P** are then those that in a sense can be computed efficiently. Although a problem that can be solved by an algorithm with running time $O(n^{100})$ might not be practically efficient it is polynomial in n, which is required by the definition of **P**, often a more efficient algorithm will exist.

A fundamental set of problems in the field are those that can be expressed using a select few logical operators.

Definition 1.2.4. A *Boolean formula* over the variables $x_1, \dots, x_n$ consists of the variables and the logical operators AND ($\wedge$), NOT ($\neg$) and OR ($\vee$). If φ is a Boolean formula over variables $x_1, \dots, x_n$, and $z \in \{0,1\}^n$, then $\varphi(z)$ denotes the value of φ when the variables of φ are assigned the values of z (where we identify 1 with TRUE and 0 with FALSE). If there exists z so that $\varphi(z) = 1$, or TRUE then we say the formula φ is *satisfiable*, and otherwise we say it is *unsatisfiable*. A Boolean formula over the variables $x_1, \dots, x_n$ is in *Conjunctive Normal Form* (*CNF* for short) if it can be written as

$$\bigwedge_i \left(\bigvee_j v_{i_j} \right)$$

where each v_{i_j} is either a variable x_k or its negation.

For example $\varphi = (u_1 \vee u_2 \vee \neg u_3) \wedge (\neg u_2 \vee u_1)$ is a CNF Boolean formula. The collections joined by AND are called the *clauses*. This leads to a fundamental decision problem.

3-SAT

Input:	a CNF-formula ϕ with clauses $c_1, \dots, c_m$ consisting of variables $x_1, \dots, x_n$ such that each clause contains at most 3 variables.
Problem:	is there a valid assignment of $x_1, \dots, x_n$ that satisfies ϕ?

There is no known deterministic Turing machine that can decide this for all possible inputs in polynomial time. However, for an instance of 3-SAT ϕ on variables $x_1, \dots, x_n$ and an assignment $z \in \{0,1\}^n$ a deterministic Turing machine can decide if $\phi(z)$ is TRUE in polynomial time. This distinction between efficiently deciding if the existence of a solution and efficiently *certifying* if a given solution is correct allows us to define another complexity class.

Definition 1.2.5. The class **NP** consists of all decision problems that can have TRUE solutions that can be certified in polynomial time.

We can see that $\mathbf{P} \subseteq \mathbf{NP}$, but whether or not these sets are equal is one of the most important open problems in theoretical computer science. Many results dealing with computational complexity are dependent on the assumption that $\mathbf{P} \neq \mathbf{NP}$. Assuming they are not equal, we may ask which problems are in **NP** and not in **P**? It would be ideal to have a method of showing that one problem is at least as hard as another. In Chapter 3 we will use a *reduction* to do just this.

Definition 1.2.6. Let A and B be decision problems. We say there exists a *many-one reduction* from A to B if there exists a polynomial algorithm that takes instances of A to instances of B preserving output. We denote this as $A \leq_p B$.

Many-one reductions now give us a way to order problems by their hardness. Suppose A and B are decision problems with $A \leq_p B$ and there exists a machine M_B that decides B accurately in polynomial time. Clearly, A can also be computed in polynomial time by mapping an instance of A to an instance of B and using the machine M_B. The contrapositive shows that if A cannot be computed in polynomial time, neither can B.

Definition 1.2.7. We say a problem B is **NP**-hard if for all $A \in \mathbf{NP}$ we have $A \leq_p B$. We say a problem B is **NP**-complete if it is **NP**-hard and it is in **NP**.

Of course, if **P** $\neq$ **NP**, then no problem that is **NP**-complete can be in **P** .

Theorem 1.2.8 ([Kar72])**.** 3-SAT *is* ***NP****-complete.*

In [Kar72], Karp proved the above theorem and also showed that a variety of problems from logic, graph theory, and set theory are **NP**-complete. If a problem has been shown to be **NP**-complete, we consider a few additional algorithmic techniques.

First, if the decision variant of a problem is **NP**-complete, but the underlying problem is an optimization problem, we might ask if we can compute an approximate solution that gives a theoretical guarantee on how close it is to the optimal solution.

Definition 1.2.9. Let P be an minimization (or maximization problem), $f(n) > 1$ (or $f(n) < 1$) be a computable function, and A be a polynomial-time algorithm that takes instances of P as input. We say A is a $f(n)$*-approximation algorithm* if for all $\varphi \in P$ it holds that if OPT is the minimum solution then $OPT \leq A(\varphi) \leq f(n) \cdot OPT$ (or if OPT is the maximum solution then $f(n)OPT \leq A(\varphi) \leq \rho$). If $f(n)$ is a constant function then we say P has a *constant-factor* approximation.

An important distinction in the definition is that a problem that admits a constant-factor approximation has a theoretical guarantee that is not dependent on the size of the instance. It is then natural to further refine the class **NP** into the problems that can have such an approximation.

Definition 1.2.10. The class **APX** is the set of **NP** optimization problems for which there exists a constant-factor approximation algorithm. A problem is said to have a *polynomial-time approximation scheme*, and be in the class ***PTAS***, if for every $\rho > 1$ in the minimization case (or for every $\rho < 1$ in the maximization case) there exists a ρ-approximation algorithm.

Another branch of computational complexity, *parameterized complexity*, focuses on determining if problems can be solved efficiently with respect to additional parameters.

Definition 1.2.11. A decision problem with an additional fixed parameter k is called a *parameterized problem*.

A parameterized problem considers instances of the general problem restricted by an additional parameter with the hope of finding an algorithm that computes a solution efficiently on this subset. For example, we might consider the instances of 3-SAT with at most k variables. If such an instance has n clauses, then we can check all of the 2^k assignments in $O(2^k n)$ time, which is exponential in k but linear in n.

Definition 1.2.12. Let P be a decision problem and k be a parameter and f be a computable function. If P can be solved by a deterministic algorithm with running time $f(k)n^{O(1)}$ for all k, then P is said to be *fixed parameter tractable* or ***FPT***. If P can be solved by a deterministic algorithm with running time $n^{f(k)}$ for all k then P is said to be in **XP**. If for some fixed k the problem P is **NP**-hard, then we say that P is **para-NP**-hard.

While the problems in both **FPT** and **XP** admit algorithms that are polynomial in n for a fixed k, those in **FPT**are particularly nice as the exponent of n is not dependent on k. We point out that **XP**=**para-NP**-hard if and only if **P**=**NP**.

When trying to compute or approximate values such as the r-coloring number it is useful to first determine the complexity classes that the underlying problem belongs to. This information will set some clear expectations on the quality of potential algorithms. In Chapter 3 we will look at the r-coloring number under the lens of computational complexity and give classifications for each of the variants. In Chapter 4 we look at approximating the r-coloring number and improve on the current best known approximation algorithm.

1.3 Coloring Numbers

Given a graph G on n vertices we consider a bijection $\sigma : V(G) \to [n]$. This is called a *linear order* of G, and we use $\Pi(G)$ to refer to all linear orders on G. We denote such a maps as $\sigma = (v_1, v_2, \dots, v_n)$, and the set of all linear orders on a graph G as $\Pi(G)$. As a subscript G_σ simply means that G is linearly ordered by σ. Such orders give a natural orientation to the edges of the underlying graph.

Definition 1.3.1. Let G_σ be a linearly ordered graph. For a vertex v_i we say the *forward neighbors* of v_i are its neighbors with index greater than i and denote this set as $R(v_i, G_\sigma)$. The order of this set is called the *reach of v_i with respect to σ*. The maximum over all vertices in G is the *reach of G with respect to σ*.

Definition 1.3.2. The minimum reach of G over all possible σ is the *degeneracy* of G.

$$\text{degen}(G) := \min_{\sigma \in \Pi(G)} \max_{v_i \in G_\sigma} |R(v_i, G_\sigma)|$$

The degeneracy of a graph is a well-studied graph invariant and is equivalent (although off-by-one) to the *coloring number* defined in [Erd66]. The coloring number was used to give a bound to the *chromatic number*, χ. The chromatic number is the minimum integer k so that each vertex of G can be colored with one of k colors so that no adjacent vertices are colored the same. We give this theorem in terms of degeneracy.

Theorem 1.3.3 ([Erd66])**.** *For any graph G, $\chi(G) \le \text{degen}(G) + 1$.*

Hence, for any order σ if the reach of G with respect to σ is k, then G has a proper coloring with at most $k+1$ colors. This can be seen by greedily coloring the vertices from the last vertex in σ to the first. No vertex will have more than k colored neighbors and therefore $k+1$ colors will suffice.

An alternative definition of degeneracy is that a graph G has degeneracy k if every subgraph of G has minimum degree at most k. A critical property of degeneracy is that an optimal order can be formed in linear time by greedily choosing a vertex with the minimum number of unordered

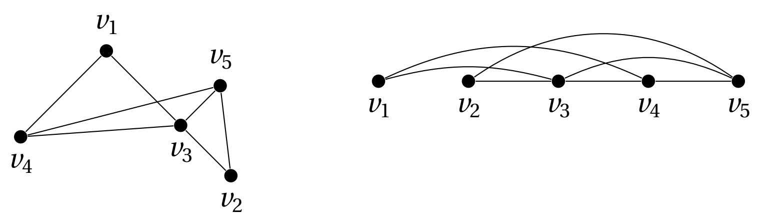


Figure 1.7 (Left) The graph G is linearly ordered using a degeneracy ordering. (Right) The graph G embedded according to the linear order. Here we see that the reach of v_1, v_2, v_3 is 2, v_4 has a reach of 1, and v_5 a reach of 0.

neighbors to be next in the order [Mat83]. Due to the relative simplicity with which degeneracy orders can be calculated they are useful parameters in a variety of algorithms [Gol08].

Simply, degeneracy guarantees an order of the vertices that bounds the intersection of each $N(v_i) \cap \{v_j\}_{j>i}$. Furthermore, for any graph G, the degeneracy of G gives an upper bound on the minimum degree vertex of every subgraph of G. In this sense it is a global property that tells us something about local relationships. It is then natural to generalize this notion to larger neighborhoods. Although one could simply try to bound $N_r(v_i) \cap \{v_j : j > i\}$, additional requirements on the paths provide a more meaningful structure. Introduced by Kierstead and Yang in 2003 [Kie03], the so-called generalized coloring numbers provide such a structure.

Definition 1.3.4. Given a linearly ordered graph G_σ and vertices $x <_\sigma y$ we say that y is *weakly r-reachable by x* if there exists an x-y path P such that $1 \leq \text{len}(P) \leq r$ and for any internal vertex of P, z, we have that $z <_\sigma y$. If there is an x-y path P such that $1 \leq \text{len}(P) \leq r$ and for any internal vertex z we have $z <_\sigma x$ we say y is *strongly r-reachable from x*, or simply r-reachable. We use $\text{wR}_r(x, G_\sigma)$ and $\text{R}_r(x, G_\sigma)$ to denote the set of vertices weakly r-reachable and strongly r-reachable from x, respectively. The size of these sets is the *weak r-reach* and the *r-reach*.

It is clear that $\text{R}_r(x, G_\sigma) \subseteq \text{wR}_r(x, G_\sigma) \subseteq N_r(x)$. Using these definitions of reach, we can give a particular linear ordering a score based on the vertex with the largest reach.

Definition 1.3.5. Given a linear ordered graph G_σ the *weak r-coloring number of G_σ* is

$$\min_{\sigma \in \Pi(G)} \max_{v \in V(G)} |\text{wR}_r(v, G_\sigma)|$$

and the *r-coloring number of G_σ* is

$$\min_{\sigma \in \Pi(G)} \max_{v \in V(G)} |\text{R}_r(v, G_\sigma)|.$$

The coloring orders provide a structurally predictable way to traverse a graph that can be useful algorithmically. They can also be used to characterize classes of graphs that have bounded expansion, which we will discuss more in Chapter 2.

1.4 Polyhedral Geometry

The work in Chapter 6 will rely heavily on definitions and results from polyhedral geometry. The main reference for this section is from the text by Ziegler [Zie12]. Generally, the spaces we consider will be convex subspaces of $\mathbb{R}^d$.

Definition 1.4.1. A point set $K \subseteq \mathbb{R}^d$ is *convex* if for any $x, y \in K$ we have that the straight line segment between x and y given by $\{\lambda x + (1-\lambda)y : 0 \leq \lambda \leq 1\}$ is also in K. The *convex hull* of a point set $K \subseteq \mathbb{R}^d$ is the smallest convex set containing K and is denoted $\text{conv}(K)$.

For a finite set of points $K = \{x_1, \dots, x_n\}$ the convex hull can be given as

$$\text{conv}(K) = \{\lambda_1 x_1 + \cdots + \lambda_n x_n : \lambda_i \geq 0, \sum_{i=1}^{n} \lambda_1 = 1\}.$$

Definition 1.4.2. A $\mathcal{V}$*-polytope* is the convex hull of a finite set of points in some $\mathbb{R}^d$. An $\mathcal{H}$*-polyhedron* is an intersection of finitely many closed half-spaces in some $\mathbb{R}^d$. This is often given as $P(A, z) = \{x \in \mathbb{R}^d : Ax \leq z\}$, where $A \in \mathbb{R}^{m \times d}$ and $z \in \mathbb{R}^m$. An $\mathcal{H}$-polyhedron that does not contain a *ray* ($\{x + ty : x, y \in \mathbb{R}^d, t \geq 0\}$ for any $y \neq 0$) is *bounded* and called an $\mathcal{H}$*-polytope*. A set of points $P \subseteq \mathbb{R}^d$ that can be presented as either an $\mathcal{H}$-polytope or a $\mathcal{V}$-polytope is called a *polytope*.

Polyhedra and polytopes will be used frequently in Chapter 6 to describe solution sets. One might notice that the definition of a polytope implies that it is an $\mathcal{H}$-polytope *or* a $\mathcal{V}$-polytope. In actuality, every polytope can be described using either presentation.

Theorem 1.4.3 (Main theorem for polytopes)**.** *A subset $P \subseteq \mathbb{R}^d$ is the convex hull of a finite point set (a $\mathcal{V}$-polytope)*

$$P = \mathit{conv}(V) \qquad \textit{for some } V \in \mathbb{R}^{d \times n}$$

if and only if it is a bounded intersection of halfspaces (an $\mathcal{H}$-polytope)

$$P = P(A, z) \qquad \textit{for some } A \in \mathbb{R}^{m \times d}, z \in \mathbb{R}^m.$$

Definition 1.4.4. A *cone* is a nonempty set of vectors $C \subseteq \mathbb{R}^d$ and all linear combinations with non-negative coefficients of any finite subset of vectors. For an arbitrary set of points $Y \subseteq \mathbb{R}^d$ the *conical hull* of Y is the intersection of all cones containing Y and can be given as
$\mathcal{C}_Y = \{\lambda_1 y_1 + \cdots + \lambda_n y_n : \{y_1, \dots, y_n\} \subseteq Y, \lambda_i \geq 0\}$.

For our purposes, we will only consider cones that are the conical hull of a finite set of points Y, in which case the points in $\mathcal{C}_Y$ are all linear combinations with non-negative coefficients of Y.

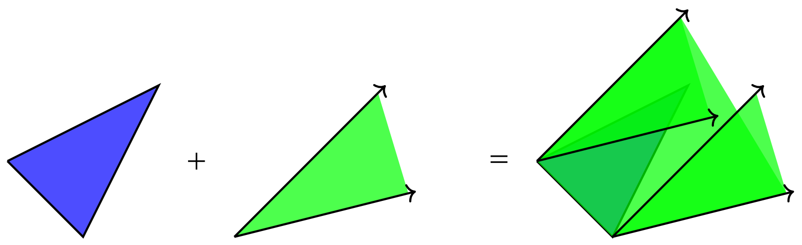


Figure 1.8 The Minkowski sum of conv(V)+cone(Y).

Definition 1.4.5. If P and Q are subsets of $\mathbb{R}^d$ then the *Minkowski Sum* of P and Q is defined to be

$$P + Q := \{x + y : x \in P, y \in Q\}.$$

A $\mathcal{V}$-polyhedron is a set $P \subseteq \mathbb{R}^d$ that is given as the Minkowski sum

$$P = \text{conv}(V) + \text{cone}(Y) \qquad \text{for some } V \in \mathbb{R}^{d\times n}, Y \in \mathbb{R}^{d\times n'}.$$

Theorem 1.4.6 (Main theorem for polyhedra)**.** *A subset $P \subseteq \mathbb{R}^d$ is a $\mathcal{V}$-polyhedron if and only if it is a $\mathcal{H}$-polyhedron.*

The Minkowski sum of a polytope and a ray will be particular useful for the analysis in Chapter 6. Example 1.4.7 gives a visual interpretation of this and the general description of a $\mathcal{V}$-polyhedron.

Example 1.4.7. We begin with the $\mathcal{V}$-polyhedron defined by conv(V) and cone(Y) in $\mathbb{R}^2$, where V is the set of points $\{(0,1),(1,0),(2,2)\}$ and Y is the set of vectors $\{\langle 1,1\rangle, \langle 1, \frac{3}{4}\rangle\}$. Figure 1.8 gives the polyhedron P that is the Minkowski sum of conv(V) and cone(Y).

Next we consider $V = \{(0,1,1),(1,0,1),(2,2,1)\}$ and $Y = \{\langle 1,1,1\rangle\}$ and the Minkowski sum of conv(V) and cone(Y). Here cone(Y) is the ray that starts at the origin and passes through the point $(1,1,1)$. The resulting polyhedron P can be seen in Figure 1.9. It is unbounded and we notice that if we intersect P with the plane $z = a$ for any $a \geq 1$ the result is a polytope that is isometric to conv(V).

1.5 Phylogenetic Trees

In this section we give the basic definitions and notation that will be used in Part II to describe phylogenetic trees.

Definition 1.5.1. Let X be a set. A *phylogenetic X-tree* (or *X-tree*) is a tree with $|X|$ leaves labeled uniquely by an element in X. If a unique interior vertex is identified as the root then the X-tree is called *rooted*, otherwise it is *unrooted*. We use $T(X)$ to refer to the set of all unrooted X-trees and $RT(X)$ the set of all rooted X-trees.

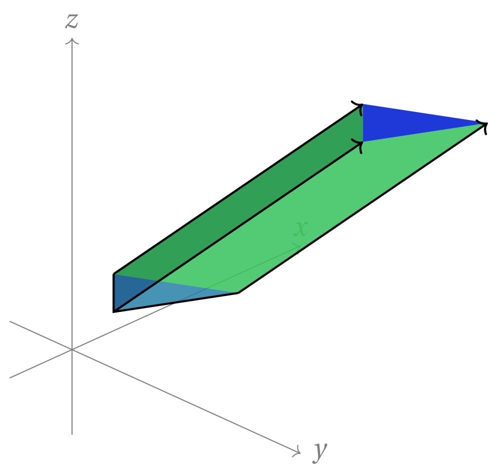


Figure 1.9 The polyhedron $P = \text{conv}(V) + \text{cone}(Y)$ where $V = \{(0,1,1),(1,0,1),(2,2,1)\}$ and $Y = \{\langle 1,1,1\rangle\}$.

Typically, the set X corresponds to some collection of species, and the branching structure represents the evolutionary relationships between them. Other than the root, vertices in an X-tree with degree-2 do not contribute to the branching structure. We can *suppress* a degree-2 vertex by deleting it and adding an edge between its two neighbors. An X-tree with no degree-2 vertices, other than perhaps the root, is said to be *refined*.

Definition 1.5.2. Let T, T' be X-trees and suppose that T results from smoothing all non-root degree-2 vertices of T'. We say that T is the *topology* of T'.

Often we will assume that $X = [n] = \{1,2,\ldots,n\}$, and refer to these as phylogenetic n-trees (or simple n-trees). When present, the root vertex represents a common ancestor. The edges in rooted trees can be thought of as being directed away from the root. The number of edges incident to a vertex and directed away from it is called the *out-degree* of the vertex. An important subset are X-trees and rooted X-trees minimize the possible branching at each interior vertex.

Definition 1.5.3. A X-tree is said to be *binary* if it is unrooted and all interior vertices have degree three, or if it is rooted and all interior vertices have out-degree two. We use $B(X)$ to refer to the set of all unrooted binary X-trees and $RB(X)$ the set of all rooted binary X-trees.

We note that this is slightly different from the typical definition of a binary tree. Vertices with degree 2 (other than a root) do not contribute to the branching structure and are typically smoothed in X-trees, resulting in full binary trees. Figure 1.10 shows both a rooted and unrooted binary 4-tree. Notice that other than the additional root vertex at the top of the tree on the right, the trees are otherwise identical in their structure.

Trees have the property that deleting any edge from the graph results in exactly two connected components. Each of these components contains at least one of the original leaves labeled by X, giving a natural partition of X.

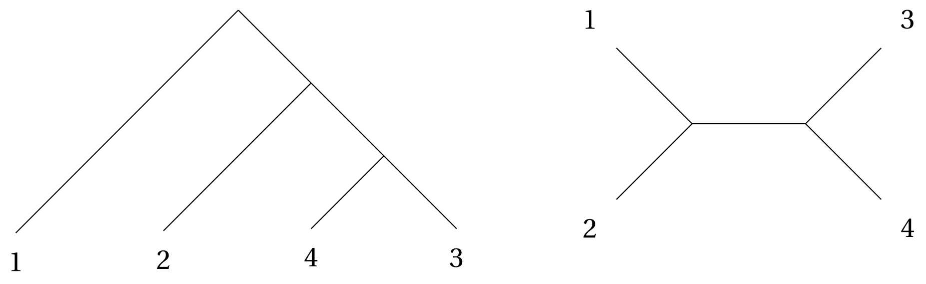


Figure 1.10 (Left) A rooted equidistant phylogenetic tree and (Right) an unrooted phylogenetic tree, both with leaves labeled from $\{1,2,3,4\}$.

Definition 1.5.4. Let T be an X-tree and e an edge of T. The graph $T-e$ that results when e is removed from T partitions X into two disjoint sets, call these A and $B = X \setminus A$. We say $A|B$ (equivalently $B|A$) is a *split* of T induced by e. The set of all splits of T is denoted $\Sigma(T)$. If e is incident to a leaf, then either A or B has order one and we call this a *trivial* or a *thin* split.

Beyond the trivial splits we distinguish two additional special structures.

Definition 1.5.5. Let T be an X-tree and $A|B$ a split. If $|A| = 2$ then A is called a *cherry* of T and if $|A| = 3$ then it is called a *cluster* of T.

One might notice that in the unrooted case every edge induces a unique split, however in the rooted case, if the root has out-degree two then both of these edges induce the same split. To distinguish between the two we imagine that a '0' leaf is attached via an edge to the root. This added leaf both transforms the rooted tree into an unrooted tree and also forces the two edges to induce distinct splits.

In Chapter 5 we will look at methods to identify sets and multisets of phylogenies. To do this we will need what it means for two trees to be equivalent.

Definition 1.5.6. Given two X-trees, T, W, we say T and W are *equivalent*, $T \cong W$, (or simply $T = W$) if there exists a graph isomorphism $f : V(T) \to V(W)$ so that $xy \in E(T)$ if and only if $f(x)f(y) \in V(W)$ and f is the identity map on the leaves.

An important operation in both identifiability and supertree computation is the reduction of a phylogeny on a set of leaves to a sub-phylogeny on a subset of the leaves.

Definition 1.5.7. Given a tree $T \in \mathcal{T}(X)$ (or $\mathcal{RT}(X)$) and $S \subseteq X$ the *subtree of T induced by S* is the unique tree in $T(S)$ (or $RT(S)$) that results when all leaves not labeled from S are deleted from T and all new degree two vertices are suppressed. We denote this tree as $T_{|K}$. (See Figure 1.14)

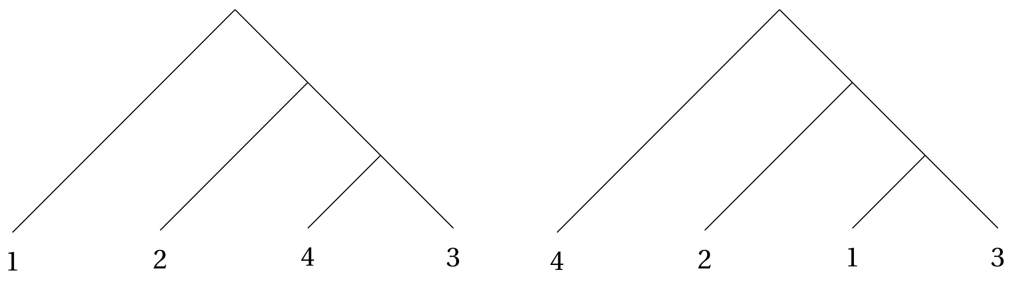


Figure 1.11 Two trees that are graph isomorphic but are not equivalent phylogenies due to the permuted leaf labels.

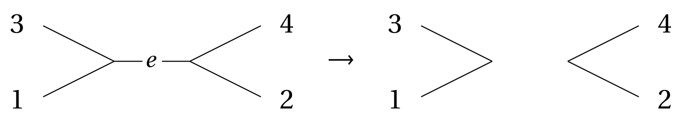


Figure 1.12 The edge e is a cut edge that partions the leaves of T into: $\{1,3\}|\{2,4\}$.

1.6 Tropical Geometry

Here we give a brief introduction to tropical geometry along with some important definitions and notation. For additional background we recommend the introductory book by Maclagan and Sturmfels [Mac15].

We use $\mathbb{R}_{\text{trop}}$ to denote $\mathbb{R} \cup \{-\infty\}$. Tropical geometry is the geometry over the *tropical semiring* $(\mathbb{R}_{\text{trop}}, \boxplus, \odot)$. The two operations are *tropical addition* ($\boxplus$) and *tropical multiplication* ($\odot$). For $a, b \in \mathbb{R}$ we define these as:

$$a \boxplus b := \max(a, b) \qquad \text{and} \qquad a \odot b := a + b$$

Under these operations we see that our additive and multiplicative identities are $-\infty$ and 0, respectively. The lack of an additive inverse prevents the tropical semiring from being a ring.

A computational benefit of tropical algebra is that the tropical operations are *linearizing*, meaning that common mathematical objects become linear. For instance if we consider a tropical multivariate polynomial such as $F(x, y, z) = 7 \odot x^2 \odot y \boxplus -5 \odot y^3 \odot z^2 \boxplus z^4$, we see that it can be given using standard arithmetic as $F(x, y, z) = \max(2x + y + 7, 3y + 2z - 5, 4z)$, where each term is a classically linear function. Tropical polynomials are then piecewise linear functions.

The space $\mathbb{R}^n_{\text{trop}}$ is a semimodule over the tropical semiring. Tropical addition and scalar multiplication are defined component wise. For $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$ we have:

$$\text{Tropical addition} : (x_1, \ldots, x_n) \boxplus (y_1, \ldots, y_n) := (x_1 \boxplus y_1, \ldots, x_n \boxplus y_n)$$

$$\text{Tropical scalar multiplication} : \alpha \odot (x_1, \ldots, x_n) := (\alpha \odot x_1, \ldots, \alpha \odot x_n)$$

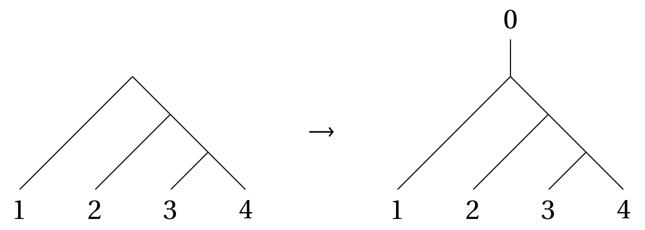


Figure 1.13 In the rooted tree on the left the two edges adjacent to the root both give the split $\{1\}|\{2,3,4\}$. Once we add the extra leaf 0, all edges give a unique partition.

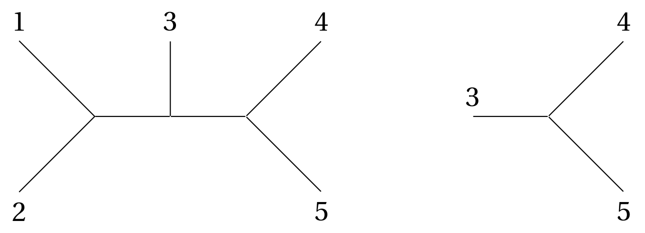


Figure 1.14 If the graph on the left is T and $K = \{3,4,5\}$, then the graph on the right is equivalent to $T_{|K}$.

In this space we can also measure the *tropical distance* between $\mathbf{x}, \mathbf{y}$, which we denote $d_{\text{tr}}(\mathbf{x}, \mathbf{y})$.

$$d_{\text{tr}}(\mathbf{x}, \mathbf{y}) := \max_i (x_i - y_i) + \max_j (y_j - x_j)$$

Unlike Euclidean geometry, geodesics are not unique in tropical geometry. The tropical line segment is used as a canonical geodesic.

Definition 1.6.1. Given $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, the *tropical line segment* is defined to be

$$\text{tline}(\mathbf{x}, \mathbf{y}) := \{\alpha \odot x \boxplus \beta \odot y : \alpha, \beta \in \mathbb{R}\}.$$

Note that formally the set of points tline$(\mathbf{x}, \mathbf{y})$ is a two dimensional set in $\mathbb{R}^n$, so the name "tropical line segment" might seem mysterious. The name makes sense when we consider that the tropical distances that we have computed are invariant for all tropical scalar multiples of the points. That is for all $\alpha, \beta \in \mathbb{R}$,

$$d_{\text{tr}}(\alpha \odot \mathbf{x}, \beta \odot \mathbf{y}) = d_{\text{tr}}(\mathbf{x}, \mathbf{y}).$$

Hence, it is natural to consider the *tropical projective torus*, $\mathbb{R}^n / \mathbb{R}\mathbf{1}$, obtained by modding out the action of tropical scalar multiplication. The torus can be mapped to $\mathbb{R}^{n-1}$ by scaling so that the first coordinate is equal to 0. For example, given $\mathbf{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ we will use $\overline{\mathbf{x}} = (0, x_2 - x_1, ..., x_n - x_1)$ as the representative of $\mathbf{x}$ in $\mathbb{R}^n / \mathbb{R}\mathbf{1}$. Considered in the tropical projective torus, for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, the tropical line segment between $\mathbf{x}$ and $\mathbf{y}$ is the concatenation of at most $n-1$ ordinary line segments. Figure 1.15 shows the tropical line segment between the points $(0,4,4)$ and $(0,-4,0)$ in the tropical

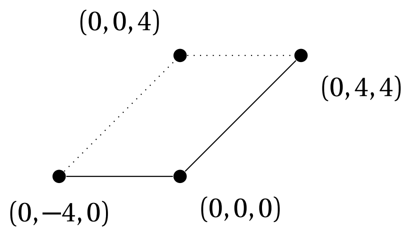


Figure 1.15 The points $(0,4,4)$ and $(0,-4,0)$ in $\mathbb{R}^3$ and two geodesics between them. The solid line gives the canonical tropical line segement. The points $(0,0,0)$ and $(0,4,0)$ are their midpoints.

projective torus.

Once we have the notion of tropical line segments, we can also talk about sets being tropically convex.

Definition 1.6.2. A set $S \subseteq \mathbb{R}^n$ is *tropically convex* if for all $x, y \in S$, $\text{tline}(x,y) \in S$.

Note that 1.6.2, like the traditional definition of convexity, includes all line segments connecting pairs of points in the set. This does not imply, however, that a tropically convex set contains all geodesics between points in the set, only the canonical geodesics. Given $S \subset \mathbb{R}^n$, the *tropical convex hull* of S, $\text{tconv}(S)$, is then the smallest convex set containing S, or, equivalently, the set of tropical linear combinations of points in S:

$$\text{tconv}(S) = \{a_1 \odot x_1 \boxplus \cdots \boxplus a_\ell \odot x_\ell : a_1, \ldots, a_\ell \in \mathbb{R}, x_1, \ldots, x_\ell \in S\}.$$

Observe that in $\mathbb{R}^n$, a point $x = (x_1, x_2, \ldots, x_n)$ does not meet the definition of a tropically convex convex set. Instead the line $\mathbb{R} \odot x$ is the convex hull of x. This shows an additional benefit of working in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$, as the set $\{\lambda \odot x : \lambda \in \mathbb{R}\}$ is projected to a single point.

1.7 Tree Metrics

In evolutionary biology, researchers can make use of a variety of methods to determine an evolutionary distance between pairs of taxa. For instance, aligning DNA or amino acids allows Hamming distance to be computed. This gives a natural distance function, $\delta : X \times X \to \mathbb{R}^{\geq 0}$. These distances can be incorporated to give a best fit weighted phylogenetic tree. Specifically, a weighted phylogenetic X-tree, (T, w) with $w > 0$, gives a distance function $d_T : X \times X \to \mathbb{R}^{\geq 0}$, where $d_T(x,y)$ is the sum of the edge weights along the path between x and y. It is natural to ask if for any arbitrary distance function δ on X, is there a weighted X-tree (T, w) so that $\delta = d_T$? It turns out that this is not always true. First we see that weighted trees give a metric on X.

Definition 1.7.1. A *metric* on a set X is a function, $d : X \times X \to [0, \infty)$ such that for all $x, y, z \in X$ we have:

- Identity of indiscernibles: $d(x,y)=0 \Longleftrightarrow x=y$
- Symmetry: $d(x,y)=d(y,x)$
- Triangle inequality: $d(x,z) \leq d(x,y)+d(y,z)$

A distance function realized by positive edge weights on a tree is a metric on the set of leaf labels. However, it is not the case that every metric can be represented on a tree. These *tree metrics* satisfy a stronger condition than the triangle inequality, called the *four point condition*.

Definition 1.7.2. A metric (M,d) satisfies the *four-point condition* if for all distinct $i,j,k,l \in M$ we have that the

$$\max\{d(i,j)+d(k,l), d(i,k)+d(j,l), d(i,l)+d(j,k)\}$$

is achieved at least twice.

The four-point condition is equivalent to the tropical quadratic *Plücker relations*, and can also be expressed using the tropical operations:

$$d(i,j) \odot d(k,l) \leq \big(d(i,k)\odot d(j,l)\big) \boxplus \big(d(i,l)\odot d(j,k), \text{ for all } i,j,k,l \in M\big)$$

Theorem 1.7.3 ([Ste16])**.** *A metric $d: X \times X \to \mathbb{R}_{\geq 0}$ satisfies the four-point condition if and only if there is an X-tree T with positive edge weights such that $d = d_T$.*

A metric that satisfies the four-point condition is a *tree metric*. For the work in Chapter 6 we consider the following restriction of tree metrics.

Definition 1.7.4. A metric (M,d) is called an *ultrametric* if it satisfies the *three-point condition*, where for all distinct $i,j,k \in M$ we have that $\max\{d(i,j), d(i,k), d(j,k)\}$ is achieved at least twice.

Under the tropical operations this relationship is equivalent to the inequality:

$$d(i,j) \leq d(i,k) \boxplus d(j,k), \text{ for all } i,j,k \in M.$$

Satisfying the three-point condition implies that the four-point condition is also satisfied. The three-point condition characterizes the tree metrics that result from equidistant trees.

Lemma 1.7.5 ([Ste16])**.** *A metric is an ultrametric if and only if it is a tree metric corresponding to an equidistant tree.*

We will use $\mathcal{T}^X$ and $\mathcal{U}^X$ to denote all tree metrics and ultrametrics on X, respectively. In the case where $X=[N]$, we use $\mathcal{T}^N$ and $\mathcal{U}^N$. As we will see later on, ultrametrics admit stronger geometric properties for analysis. Considering equidistant trees in the framework of evolutionary biology is not without reason. If we think of the leaves of a phylogeny as a set of species at some fixed time and a point on the tree to be some common ancestor. In closely related species that have roughly similar generation times, it can be reasonable to assume that the inferred phylogenetic tree could be an equidistant tree. For more distantly related species, evolutionary time and clock time might not be the same and methods that can infer more general trees are needed.

Due to the symmetric property of metrics, tree metrics are often represented as cophenetic vectors of length $n = \binom{N}{2}$ or upper triangular $N \times N$ matrices. In other words we may think of T as the vector $\mathbf{v}_T = (d_T(1,2), d_T(1,3), \dots, d_T(N-1,N))$ or as the upper triangular matrix A_T defined below.

$$A_T[i,j] = \begin{cases} d_T(i,j) & \text{if } i < j \\ 0 & \text{otherwise.} \end{cases}$$

Hence any phylogeny can be assigned a unique point in $\mathbb{R}^n$. Typically, we will use the notation $T(i,j)$ to denote the coordinate value corresponding to $d_T(i,j)$ in these equivalent representations interchangeably.

1.8 Outline of Thesis

The remainder of this thesis is structured as follows. Chapters 2, 3, and 4 describe results pertaining to the r-coloring numbers. In Chapter 2 we give a brief survey of a selection of past and current research. This covers some important relationships between the r-coloring number and bounded expansion, game coloring numbers, and r-admissibility. Chapter 3 presents results pertaining to the computational complexity of the r-coloring number. We give a reduction of the problem from a variant of 3-SAT, showing that it is **NP**-complete, **para-NP**-hard, and that it does not admit a polynomial-time approximation scheme. Finally, we present a best known approximation algorithm that produces a linear order witnessing the approximation. In Chapters 5 and 6 we present work on problem from evolutionary biology involving phylogenetic trees. The results in Chapter 5 improve on existing results for identifying sets and multisets of phylogenies based on induced subtrees. Chapter 6 outlines a method to compute supertrees from a collection of phylogenies. By utilizing a tropical variant of treespace as a metric space, we combine the information from sets of input ultrametrics and provide a geometrically useful set of minimum distance supertrees.

Part I

Coloring Numbers

CHAPTER

2 OVERVIEW: COLORING NUMBERS

2.1 Introduction

The present chapter gives a brief survey of relevant past research involving the r-coloring numbers. We hope to highlight its relationship with other important areas of research such as the game coloring number and sparsity theory. We also review a piece of work on r-admissibility, which gave a previous approximation algorithm for the r-coloring number, from which we drew inspiration.

2.2 Relationship to the Game Coloring Number

In Section 1.3 we mentioned that Kierstead and Yang [Kie03] introduced the generalized coloring numbers. Here we summarize the results from that paper.

Kierstead and Yang note that graph theoretic algorithms are often improved when the vertices are processed in a useful linear order. Recall from Theorem 1.3.3 that a greedy coloring algorithm that queues the vertices according to a degeneracy order provides a guarantee on the number of necessary colors. In fact, there exists a linear order such that a greedy coloring algorithm will use the optimal number of colors. The focus of their work was the relationship between various linear orders with an emphasis on the *r-game coloring number*.

We consider a two player turn-based coloring game played on a graph G and let r be a positive integer. The game begins with all vertices unordered. The first player to act, Alice, chooses a vertex and it is marked as v_1. The second player, Bob, chooses an unordered vertex and it is marked as v_2.

This continues with players choosing an unordered vertex to add to the order. Play continues until all vertices have been ordered, giving a linear order σ. The score of the game is

$$s := \max_{v_i \in G_\sigma} |\mathrm{R}_r(v_i, G_\sigma)|.$$

Alice's goal is to minimize the score and Bob's is to maximize the score. The *r-game coloring number* is the smallest s so that with optimal play by Alice she is guaranteed a score at most s. For a graph G we denote this value as $\mathrm{gcol}_r(G)$.

In defining this generalization of the standard game coloring number, which is equivalent to the 1-game coloring number, they also defined the r-reach and weak r-reach and the generalized coloring numbers. The main purpose of these new invariants was to provide bounds on the r-game coloring number but the authors also showed some interesting properties based on the definitions.

The first result demonstrates a relationship between the 2-coloring number and the *acyclic chromatic number.*

Definition 2.2.1. The *acyclic chromatic number* of a graph G, denoted $\chi_a(G)$ is the least k so that their exists a proper vertex coloring of G with k colors so that every pair of color classes induces a forest.

Theorem 2.2.2 ([Kie03])**.** *Every graph G satisfies $\chi_a(G) \leq \mathrm{col}_2(G) - 1$ (using our definition of $\mathrm{col}_2(G)$).*

The proof of this is similar in nature to the proof that degeneracy is an upper bound for the chromatic number. This demonstrated that their generalization of reach to higher radii could be related to other invariants. Furthermore, they showed that the weak r-coloring number and r-coloring number have a clear relationship.

Lemma 2.2.3 ([Kie03])**.** *Every graph G satisfies $\mathrm{wcol}_r(G) \leq (\mathrm{col}_r(G))^r$.*

We recreate this proof here as it provides some intuition into the relationship between the two.

Proof. Let G be a graph and $\mathrm{col}_r(G) = k$ and suppose σ is an optimal order such that $\mathrm{col}_p(G_\sigma) = k$. By induction on r we show that for every $y \in V(G)$ we have $|\mathrm{wR}_r(y, G_\sigma)| \leq (\mathrm{col}_r(G))^r$. At $r = 1$ the weak r-reach of a vertex is also its r-reach, so this is trivial. Assume it is true up to some $r = p-1$. For each $x \in \mathrm{wR}_p(y, G_\sigma)$ choose a shortest qualifying path P_{yx}. We note that if z is the first vertex on some P_{yx} such that $y <_\sigma z$ and z is distance i from y then $z \in \mathrm{R}_i(y, G_\sigma) \backslash \mathrm{R}_{i-1}(y, G_\sigma)$ and $x \in \mathrm{wR}_{p-1}(z, G_\sigma)$ is in the weak $p-1$-reach of z. Therefore,

$$|\mathrm{wR}_p(y, G_\sigma)| \leq \sum_{i=1}^{p} |\mathrm{R}_i(y, G_\sigma) \setminus \mathrm{R}_{i-1}(y, G_\sigma)| \mathrm{wcol}_{p-i}(G) \leq |\mathrm{R}_p(y, G_\sigma)| \mathrm{wcol}_{p-1}.$$

By the inductive hypothesis and the assumption that σ is optimal we get

$$|\mathrm{wR}_p(y, G_\sigma)| \leq \mathrm{col}_p(G)(\mathrm{col}_{p-1}(G))^{p-1} \leq (\mathrm{col}_p(G))^p.$$

□

Using an argument that relies on Lemma 2.2.3 they showed that for special graph classes we can bound the r-coloring number by a function dependent on r and the 1-coloring number.

Theorem 2.2.4 ([Kie03])**.** *There exists a function f such that for all positive integers k and r, if $\mathcal{C}$ is a topologically closed class of graphs such that $\mathrm{col}_1(G) \leq k$ for every $G \in \mathcal{C}$ then $\mathrm{col}_r(G) \leq f(k,r)$ for every $G \in \mathcal{C}$.*

The proof itself is constructive and the function used in the proof is defined recursively as $f(k,1) = 1$ and $f(k,r) = kf(k,r-1)^{2(r-1)^2}$. Theorem 2.2.4 implies that planar graphs have r-coloring numbers bounded by this function.

The final main result of the paper provides a relationship between the r-game coloring number and the generalized coloring numbers.

Theorem 2.2.5 ([Kie03])**.** *Every graph $G = (V,E)$ satisfies:*

$$\mathrm{gcol}_r(G) \leq 3\,\mathrm{wcol}_{2r}(G)^2 \leq 3(\mathrm{col}_{2k}(G))^{4r}.$$

The second inequality follows directly from Lemma 2.2.3. The first inequality is proved with direct strategy for Alice to follow in the game. This strategy is based on having an optimal weak $2r$-coloring number and a similar path fixing argument to the proof of of Lemma 2.2.3 given above.

Kierstead and Yang's work on linear orders that minimize certain paths had implications beyond the game coloring number. In the next section we show that they have an important relationship to sparsity theory.

2.3 Sparsity and Bounded Expansion

In 2012 Nešetřil and Ossona de Mendez [Neš12] developed a theory of sparsity intended to provide a combinatorial framework for defining and studying sparse mathematical objects. While designed to be general enough to use broadly (e.g. for sets, matrices, etc.) the main focus of their text (and also our research) is sparse undirected graphs.

We begin with an example of the nuance required in defining what it means to be sparse. Given a graph G, what feature of this graph might one consider when trying to define if it is sparse or dense? An obvious choice is an invariant such as average density, $|E|/|V|$, which reflects the average global density. However, large graphs with low average density may still contain a dense subgraph such as a large clique. Additionally, a clique with the edges subdivided many times might appear sparse but retains the structure (and "density") of the underlying clique in some sense.

With this in mind, sparsity theory uses relative definitions on infinite classes of graphs. In this setting a graph is not sparse or dense in and of itself but instead we consider the sparsity of a graph family.

Definition 2.3.1. A *class* of graphs, $\mathcal{C}$, is a set of graphs that can be either finite or infinite. A class $\mathcal{C}$ is:

hereditary if for every graph $G \in \mathcal{C}$, if $H \subseteq_i G$ then $H \in \mathcal{C}$,
monotone if for every graph $G \in \mathcal{C}$, if $H \subseteq G$ then $H \in \mathcal{C}$,
minor closed if for every graph $G \in \mathcal{C}$, if $H \preccurlyeq G$ then $H \in \mathcal{C}$,
topologically minor closed if for every graph $G \in \mathcal{C}$, if $H \preccurlyeq_T G$ then $H \in \mathcal{C}$.

We are now able to define and then refine how [Neš12] approach the notion of sparseness in infinite graph classes. We present two main classifications, *nowhere dense* and *bounded expansion*. There are many equivalent characterizations for each of these properties but we limit our attention to the most straightforward and relevant.

Definition 2.3.2. A class of graphs $\mathcal{C}$ is *nowhere dense* if for every d there exists a graph H such that $H \not\preccurlyeq_d G$ for all $G \in \mathcal{C}$.

An important subset of nowhere dense graph classes are sparser classes with so-called *bounded expansion*. We first define a graph invariant used in the classification of these families.

Definition 2.3.3. The *greatest reduced average density* (or *grad*) with rank d of a graph $G = (V, E)$ is

$$\nabla_d(G) := \max\left\{\frac{|E|}{|V|} : H \preccurlyeq_d (G)\right\}$$

We note that this mimics the idea of average density except that one must consider the average density of all d-shallow minors of G.

Definition 2.3.4. A class $\mathcal{C}$ of graphs has *bounded expansion* if for every t there exists $c(t)$ such that $\nabla_t(G) \leq c(t)$ for all $G \in \mathcal{C}$.

Intuitively, the graphs that belong to a class with bounded expansion have no t-shallow minors with average density greater than $c(t)$. The first motivation for studying graph classes that are sparse is that most real-world graphs are, in fact, sparse. The second motivation for considering graph classes with bounded expansion is that they have been shown to admit particularly efficient algorithms. In particular, graph properties that can be stated using first-order logic consist of many generally hard problems, but these problems can be efficiently solved on classes with bounded expansion. The subgraph isomorphism problem, which asks if a graph G has a subgraph isomorphic to a graph H, can be stated in first-order logic and is known to be **NP**-complete [Coo71]. But for any graph in a class with bounded expansion it was shown by Nešetřil et. al. [Neš08] that it can be solved in linear time. Furthermore, Dvořák [Dvo13a] showed that all graph properties expressible in first-order logic can be decided in linear time on graphs that are in a class with bounded expansion.

The generalized coloring numbers, in a sense, capture the property of bounded expansion. In an optimal order a vertex has high r-reach when it has a relatively dense r-neighborhood. The following alternative characterization of bounded expansion solidifies this relationship.

Lemma 2.3.5 ([Zhu09])**.** *A class $\mathcal{C}$ of graphs has bounded expansion if and only if there is a function $f : \mathbb{N} \to \mathbb{N}$ such that $\mathrm{col}_r(G) \le f(r)$ for all $r \in \mathbb{N}$ and all $G \in \mathcal{C}$.*

2.4 Approximating the *r*-Coloring Number

In Chapter 4 we will present an algorithm to produce a linear order that approximates the r-coloring number. This work follows closely from the work of Dvořák in [Dvo13b], which we present here.

We begin by introducing an additional parameter based on linear orders that is closely related to the generalized coloring numbers.

Definition 2.4.1. For a graph G, a linear order $\sigma \in \Pi(G)$, and a vertex $v \in G$, the *r-backconnectivity* of v with respect to σ, denoted $b_r(v, G_\sigma)$, is the maximum number of disjoint paths with length at most r that begin at v and end at a vertex after v in the ordering. The *r-admissibility* of G_σ $\mathrm{adm}_r(G_\sigma)$ is the $\max_{v \in G_\sigma} b_r(v, G_\sigma))$. The r-admissibility of the graph G is the $\min_{\sigma \in \Pi(G)} \mathrm{adm}_r(G_\sigma)$.

The definition of r-admissibility implies that for any graph G, $\mathrm{adm}_r(G) \le \mathrm{col}_r(G)$. We can see this by considering any order σ and vertex v. For any of the disjoint paths from v counted by the admissibility, the first vertex after v in the order is in $\mathrm{R}_r(v, G_\sigma)$.

Dvořák showed that any minimum r-admissibility ordering bounds the weak r-coloring number.

Lemma 2.4.2 ([Dvo13b])**.** *Let $r \ge 1$ and $k \ge 2$ be integers. Let G be a graph and $\sigma = v_1, v_2, \dots, v_n$ a linear order of its vertices. If the r-admissibility of G_σ is at most c, then its weak r-coloring number is at most $\frac{k^{r+1}-1}{k-1} - 1$ (under our off-by-one definition of the weak r-coloring number).*

Although it was not mentioned explicitly, the proof of Lemma 2.4.2 shows that an r-admissibility ordering also gives a bound on the r-coloring number.

Corollary 2.4.3. *Let $r \ge 1$ and $k \ge 2$ be integers. Let G be a graph and $\sigma = v_1, v_2, \dots, v_n$ a linear order of its vertices. If the r-admissibility of G_σ is at most c, then its r-coloring number is at most $k(k-1)^r$.*

We use the applicable pieces from Dvořák's proof of Lemma 2.4.2 to prove Corollary 2.4.3.

Proof. Let G be a graph $r \ge 1$ and $k \ge 2$ integers and suppose $\sigma \in \Pi(G)$ witnesses $\mathrm{adm}_r(G_\sigma) \le k$. For a vertex $v \in G$ we compute the r-reach of v. From v create a breadth-first search tree $T = T(G_\sigma)$ that is constructed so that a branch ends when a vertex after v in σ is reached. Such a tree can be "pruned" so that the leaves are the exactly the r-reach of v by continuously removing all leaves that are either before v in the order, or are greater than distance r from v in the tree. No vertex in T can have degree greater than k. If the root v has degree greater than k then it has at least $k+1$ disjoint paths G with respect to σ, contradicting the assumption that $\text{adm}_r(G_\sigma) \leq k$. The interior vertices are all before v in σ and if they have degree $k+1$ then there are k disjoint paths to the leaves below them in G and and another disjoint path up the tree to v, again a contradiction. Hence T has at most $k(k-1)^{r-1}$ leaves, which completes the proof. □

The tree used in this proof will be addressed in detail in Definition 4.2.1. An r-admissibility order would then be a $(k-1)^{r-1}$-approximation of the r-coloring number. Such an order can be found by choosing vertices with the minimum back-degree greedily, but for $r \geq 5$ computing the back-connectivity is **NP**-complete. Dvořák then shows that there exists an $O(rn^3)$ time r-approximation of the admissibility. This constructs an order with admissibility at most rk if the r-admissibility of the graph is k. Combining this approximation with Corollary 2.4.3 we get an order with max r-reach at most $(rk)(rk-1)^{r-1}$.

CHAPTER

3

COLORING NUMBER HARDNESS

The results given in Chapters 3 and 4 are the result of a collaboration with B. Lavallee and B. Sullivan at the University of Utah. Over the course of this collaboration my main focus was developing the approximation algorithm but this could not have been accomplished without the assistance of my collaborators. B. Lavallee identified an NP-hard problem and was able to work out a reduction for the weak r-coloring number. Using the same problem I was able to give an additional reduction for the r-coloring number, however it was B. Lavallee that pointed out the additional para-NP-hard and no PTAS results that follow from this reduction. I thank both of them for their insight and help throughout the process.

3.1 Introduction

As discussed in the previous chapter, the generalized coloring numbers provide a characterization of graph classes with bounded expansion; they also imply linear orders with useful algorithmic properties. Naturally, we would like to compute these values on arbitrary graphs and, when possible, construct optimal linear orderings. In this chapter, we consider whether either of these objectives can be accomplished efficiently. We recall from Section 1.2 that graph properties which can be framed as a **NP**-complete decision problem do not have efficient (polynomial time) algorithms unless $\mathbf{P} = \mathbf{NP}$. We prove in the following sections that the r-coloring number is such a graph property. Furthermore, we consider the next natural questions: can the r-coloring number be approximated efficiently? and can the r-coloring number be computed using a parameterized algorithm?

3.2 Problem Statements and Known Results

We begin by defining the decision problem associated with the r-coloring number.

r-ORDERABLE

Input: a graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Problem: is there a linear order $\sigma = (v_1, \dots, v_n)$ of V such that $\mathrm{col}_r(G_\sigma) \le k$?

For the value $r = 1$, deciding r-ORDERABLE is equivalent to computing the degeneracy of G which can be done in linear time. However, when $r = n$, deciding r-ORDERABLE is equivalent to computing the treewidth of G which is **NP**-complete [Arn87]. This means that the r-coloring number bridges between treewidth and degeneracy, but other than these extremal values of r there were previously no existing hardness results for r-ORDERABLE. The weak r-coloring number, on the other hand, was shown to be **NP**-hard when $r \ge 3$ in [Gro15]. Grohe et al. gave a reduction from BALANCED COMPLETE BIPARTITE SUBGRAPH, but it could not be extended to the weak 2-coloring number or the the r-coloring number for any r. Intuitively, one might expect that the r-coloring number would also be **NP**-hard for $r \ge 3$, but when $r = 2$ a polynomial time algorithm was plausible. The work presented in this chapter will show that r-ORDERABLE is in fact **NP**-complete, even when $r = 2$. Additionally, our reduction gives results with regard to the optimization and parameterized versions of the problem, which we discuss in Section 3.5.

3.3 A Reduction From 2-CLAUSE 3-SAT

Our approach uses a reduction from 2-CLAUSE 3-SAT, a variant of 3-SAT, which restricts the number of clauses each literal can appear in.

2-CLAUSE 3-SAT

Input: a CNF-formula ϕ with clauses $c_1, \dots, c_m$ consisting of variables $x_1, \dots, x_n$ such that each clause contains either 2 or 3 variables and each literal (x_j or $\overline{x}_j$) appears in *exactly* 2 clauses.
Problem: is there a valid assignment of $x_1, \dots, x_n$ that satisfies ϕ?

To show that 2-CLAUSE 3-SAT is **NP**-hard we require the following lemma of Tovey [Tov84].

Lemma 3.3.1 ([Tov84]). *Boolean satisfiability is **NP**-complete when restricted to instances with 2 or 3 variables per clause and at most 3 occurrences per variable.*

Proposition 3.3.2. 2-CLAUSE 3-SAT *is NP-complete.*

Proof. As a satisfiability problem, 2-CLAUSE 3-SAT is clearly in **NP**. To show it is **NP**-hard we consider an instance φ of boolean satisfiability with 2 or 3 occurrences per variable and at most 3 occurrences per variable as in Lemma 3.3.1.

We may assume that for each variable there is at least one positive and one negative occurrence. If not, this variable can be preprocessed without consequence. This implies that no *literal* appears more than twice. It remains to show that we can pad the occurrence of each literal to be exactly two without affecting satisfiability. If a literal x appears only once, then we add the new variables $y_1, \dots, y_5$ and clauses:

$$(x \vee y_1 \vee \overline{y_2}) \wedge (y_1 \vee y_3 \vee \overline{y_4}) \wedge (\overline{y_1} \vee y_3 \vee \overline{y_4}) \wedge (\overline{y_1} \vee y_4 \vee \overline{y_5}) \wedge (y_2 \vee y_4 \vee \overline{y_5}) \wedge (y_2 \vee \overline{y_3} \vee y_5) \wedge (\overline{y_2} \vee \overline{y_3} \vee y_5)$$

Note that we add a unique set of five variables for each literal that only appears once, and they only appear in the clauses given above. We can see that each new variable has 2 occurrences per literal, and these clauses can be satisfied by setting $y_1, \dots, y_5$ to true. We can repeat this process until all literals have exactly two occurrences resulting in an instance of 2-CLAUSE 3-SAT in polynomial time. □

We now give an explicit map that takes an instance of 2-CLAUSE 3-SAT to an equivalent instance of r-ORDERABLE.

Definition 3.3.3. Given an instance φ of 2-CLAUSE 3-SAT, let $G(\varphi)$ be the following graph. For each variable x_j which appears in φ, add the vertices v_j and v'_j and the edge $v_j v'_j$ to $G(\varphi)$. For each clause c_i in φ, we add a vertex u_i to $G(\varphi)$. For each literal x_j (or $\overline{x_j}$) in c_i, we add a disjoint $x_j - c_i$ path of length r to $G(\varphi)$. We call the $(r-1)$ degree-2 vertices on this path *subdivision vertices*.

We then construct a gadget $\mathcal{H}_r$, and add it to $G(\varphi)$. $\mathcal{H}_r$ consists of three parts. The first part is a 4-clique which we call the *clause clique* (H_c), the second a 6-clique called the *variable clique* (H_v), and finally three independent vertices called the *apex vertices* (H_a). Each vertex in H_a is connected to each vertex in H_v by an edge and to each vertex in H_c by a disjoint path of length r. We now describe how $\mathcal{H}_r$ is connected to the rest of $G(\varphi)$. For every clause vertex u_i, we add an edge from u_i to every vertex in H_c. We designate three vertices in H_v as positive and the other three as negative. For every variable x_i and the corresponding literal vertices v_i and v'_i, we add an edge from each v_i to each of the three positive vertices in H_v and an edge from v'_i to each of the three negative vertices in H_v. Additionally, for a clause c_i which contains only 2 variables, we add a disjoint path of length r from u_i to an arbitrary vertex in H_v. We use S to denote the set of all subdivision vertices in $G(\varphi)$. This construction is illustrated in Figure 3.1.

For an instance of 2-CLAUSE 3-SAT, let A be an assignment of the variables. We say that A induces a *canonical order* $\sigma(A)$ on the vertices of $G(\varphi)$. In fact, $\sigma(A)$ is any linear order of $G(\varphi)$ that satisfies:

$$S <_{\sigma(A)} \text{true literals in } A <_{\sigma(A)} \text{clause vertices} <_{\sigma(A)} \text{false literals in } A <_{\sigma(A)} H_c <_{\sigma(A)} H_a <_{\sigma(A)} H_v$$

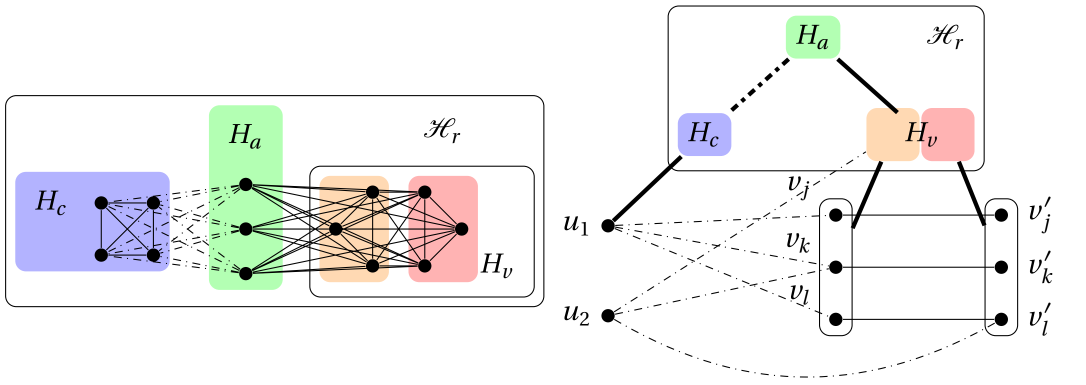


Figure 3.1 Left: The subgraph $\mathcal{H}_r$. Right: A subgraph of $G(\varphi)$ where φ contains the clauses $c_1 = (x_j \vee x_k \vee x_l)$ and $c_2 = (x_k \vee \overline{x_l})$. In both figures lines denote edges, dashed lines denote disjoint paths of length r. A thick edge denotes an edge or disjoint path of length r between all vertices in the set.

The reduction given in Definition 3.3.3 will be utilized throughout this chapter to connect instances of 2-Clause 3-SAT with r-Orderable. We first establish several key properties of $G(\varphi)$ and linear orders that are necessary in our reduction. Its purpose is to admit an order with no vertex having reach 7 if and only if the instance φ is satisfiable. The structure of $G(\varphi)$ is such that no clause vertex can be added to an order until one of the literals it contains has been added. Initially, only the literal vertices and subdivision vertices have reach less than 7. If a literal vertex v_i is added to the order then its negation $\overline{v_i}$ can now reach all the vertices in H_v. Notice that $\overline{v_i}$ can also reach some vertex on the path from it to any clause vertex corresponding to a clause that contains $\overline{x_i}$ if they are unordered. Hence, it has reach greater than 6 until all the clause vertices and subdivision vertices in its reach are ordered. In the remainder of this section, we prove some additional properties of $G(\varphi)$ that will be applicable.

Proposition 3.3.4. *For all r* $\mathrm{col}_r(\mathcal{H}_r) = 6$.

Proof. We first consider a linear order σ of $\mathcal{H}_r$ satisfying $S <_\sigma H_c <_\sigma H_a <_\sigma H_v$. Note that this is the same order these vertices would have in a canonical order for some $G(\varphi)$. We consider each set of vertices individually and count their maximum possible r-reach. Each subdivision vertex lies on a path from a vertex in H_c and a vertex in H_a and therefore has a reach of at most two in σ. A vertex in H_c can reach at most all other vertices in H_c and exactly all vertices in H_a, for a maximum reach of six. A vertex in H_a cannot reach any other vertex in H_a, and only reaches all vertices in H_v, so each has a reach of six. Finally, each vertex in H_v can only reach other vertices in H_v, as they are the only unordered vertices, and this gives a maximum of five. By the above analysis we have that $\mathrm{col}_r(\mathcal{H}_r) \leq 6$.

Now, if we consider the graph induced by a single vertex in H_a and all of H_v, we get a clique on 7 vertices which has coloring number six. This implies $\mathrm{col}_r(\mathcal{H}_r) \geq 6$, completing the proof.

□

Proposition 3.3.5. *Let σ be a linear order of $\mathcal{H}_r$ such that* $\operatorname{col}_r((\mathcal{H}_r)_\sigma) = 6$. *If $x \in H_c$ and $y \in H_v$ then $x <_\sigma y$.*

Proof. To produce a contradiction, suppose that $x \in H_c$ and $y \in H_v$, and let σ be an order such that $\operatorname{col}_r(G_\sigma) = 6$ and $y <_\sigma x$. We may assume that y is the first vertex from H_v in σ. Let z be the vertex in H_a that appears first in σ. If $z <_\sigma y$, we see that z reaches all of H_v and some vertex on the path of length r from z to x. Therefore, z has r-reach at least seven, contradicting that σ is optimal. On the other hand, if $y <_\sigma z$ then y reaches all other vertices in H_v and all vertices in H_a. This implies the r-reach of y is eight, again contradicting that σ is optimal. □

Proposition 3.3.4 also gives an important restriction to graphs such as $G(\varphi)$ that have $\mathcal{H}_r$ as a subgraph.

Corollary 3.3.6. *If G is a graph such that* $\operatorname{col}_r(G) = 6$ *and $\mathcal{H}_r \subseteq G$, then in any optimal order σ such that* $\operatorname{col}_r(G_\sigma) = 6$, *if $v \in H_c$ is the vertex from H_c that appears earliest in σ then* $\mathrm{R}_r(v, G_\sigma) \subset \mathcal{H}_r$.

The implication of Proposition 3.3.5 and Corollary 3.3.6 is that an order witnessing that $G(\varphi)$ has coloring number at most 6 must have all clause vertices appearing before all vertices in H_c and H_v.

As in the above proof, we will often assume that subdivision points are at the beginning of an order. The following lemma shows that this assumption does not increase the reach.

Lemma 3.3.7. *Let G be a graph and σ a linear order of the vertices of G so that* $\operatorname{col}_r(G_\sigma) = k > 2$. *Suppose P is a $u - v$ path in G of length r and all internal vertices of P have degree 2 in G. Define the linear order σ' to be identical to the order σ with the exception that the internal vertices of P appear first in σ'. Then* $\operatorname{col}_r(G_{\sigma'}) \le k$.

Proof. It suffices to show that all vertices in $G_{\sigma'}$ have r-reach at most k. It is clear that for a vertex $x \in G - P$ it holds that $|\mathrm{R}_r(v, G_{\sigma'})| \le |\mathrm{R}_r(v, G_\sigma)| \le k$. Any subdivision vertex can reach at most two other vertices which is less than k by assumption. Assume $u <_\sigma v$, which implies that $u <_{\sigma'} v$. Then $\mathrm{R}_r(v, G'_\sigma) \subset \mathrm{R}_r(v, G_\sigma)$. Finally, for u, we see that v is the only potentially new vertex that u could reach. If u could already reach v in σ, then we are done. If v was not in the reach of u, then another vertex on P must have been in the reach of u, specifically the first subdivision vertex on P that appears after u in σ. Therefore $|\mathrm{R}_r(u, G_{\sigma'})| \le |\mathrm{R}_r(u, G_\sigma)| \le k$. □

3.4 *r*-Orderable is NP-Complete

To complete the reduction, we must show that our map preserves output. Specifically, we will show that given an instance of 2-Clause 3-SAT φ, $G(\varphi)$ has r-coloring number six if and only if φ is satisfiable. We begin by proving the following two lemmas.

Lemma 3.4.1. *Let φ be an instance of* 2-CLAUSE 3-SAT*, and let $G(\varphi)$ be the graph defined in Definition 3.3.3. If φ is satisfiable, then* $\operatorname{col}_r(G(\varphi)) = 6$.

Proof. Let φ be an instance of 2-CLAUSE 3-SAT with n variables and $G = G(\varphi)$. Suppose A is an assignment witnessing that φ is satisfiable. For clarity, if x_1 is set to true, then we say x_1 is a true literal in A and $\overline{x_1}$ is a false literal. Otherwise, x_1 is a false literal and $\overline{x_1}$ is a true literal. We consider the canonical order $\sigma(A)$. The subdivision vertices appear first and they each have reach at most 2. Next, $\sigma(A)$ orders the vertices corresponding to true literals in A. In their reach, there are two vertices corresponding to the clauses that contain them, the vertex corresponding to their negation, and three vertices in H_v for a total reach of 6. We now proceed to the clause vertices. Since A is a satisfying assignment, they also have reach at most six: all four vertices in H_c and any vertices corresponding to literals they contain that are set to false. If a satisfied clause contains three literals, then at most two false literal vertices are in their reach of the corresponding clause vertex. If a satisfied clause contains two variables, the reach of the corresponding clause vertex may include at most one false literal vertex and one arbitrary vertex in H_v. Next in σ are the vertices corresponding to false literals in A. These can reach all the vertices in H_v, of which there are six and nothing else. Finally, no vertex in $\mathcal{H}_r$ has reach greater than six in this order by Proposition 3.3.4. □

It remains to show that any order witnessing $\operatorname{col}_r(G(\varphi)) = 6$ maps to a satisfying assignment of φ. The map is straightforward. Given a linear order σ of $G(\varphi)$, we say $A(\sigma)$ is the following assignment. For a variable x_i, if $v_i <_\sigma v_i'$, then x_i is set to true in $A(\sigma)$. If $v_i' <_\sigma v_i$, then x_i is set to false in $A(\sigma)$. We claim that if $A(\sigma)$ is not a satisfying assignment, then $G(\varphi)$ has coloring number greater than 6.

Lemma 3.4.2. *Let φ be an instance of* 2-CLAUSE 3-SAT*, and let $G(\varphi)$ be the graph given in Definition 3.3.3. If φ is not satisfiable, then $col_r(G_\varphi) > 6$.*

Proof. Let $G = G(\varphi)$. We assume that φ is not satisfiable and σ is a linear order such that $\operatorname{col}_r(G_\sigma) = 6$ to produce a contradiction. By Proposition 3.3.6 and Lemma 3.3.7, we may assume that all subdivision vertices appear first in σ and all clause vertices appear before all the vertices in H_c. Since $A(\sigma)$ does not satisfy φ by assumption, without loss of generality we suppose $c_i = (x_1 \vee x_2 \vee x_3)$ (or $c_i = (x_1 \vee x_2)$) is not satisfied by $A(\sigma)$ and that in G we have $v_1 <_\sigma v_2 <_\sigma v_3$ (or $v_1 <_\sigma v_2$). Because c_i is not satisfied, it must be the case that $v_j' <_\sigma v_j$ for all $j \in \{1,2,3\}$ (or $j \in \{1,2\}$). One of the following two cases must hold, either $u_i <_\sigma v_1$ or $v_1 <_\sigma u_i$. In the first case, by Corollary 3.3.6 u_i must be ordered before the vertices in H_c. Hence, u_i is able to reach all vertices in H_c as well as v_1, v_2, and v_3 (or an arbitrary vertex in H_v instead of v_3) for a total reach of seven, a contradiction. In the second case, v_1 can reach u_i and all six vertices of H_v, again a contradiction. Therefore $\operatorname{col}_r(G_\sigma) > 6$. □

Using these lemmas we prove the main result of this chapter.

Theorem 3.4.3. *For $r \geq 2$ and $k \geq 6$, r-*ORDERABLE *is **NP**-Complete.*

Proof. From Lemmas 3.4.1 and 3.4.2, if r-ORDERABLE were solvable in polynomial time for $k \geq 6$ then any instance of 2-CLAUSE 3-SAT would also be solvable in polynomial time. It is therefore at least as hard as 2-CLAUSE 3-SAT which is **NP**-hard by Proposition 3.3.2. Given an order σ we can check the reach of each vertex in polynomial time so r-ORDERABLE is in **NP**. By definition, it is **NP**-complete for $r \geq 2$ and $k \geq 6$. □

3.5 Additional Complexity results

In the last section we showed that r-ORDERABLE is **NP**-complete for $r \geq 2$ and sufficiently large k. As we are not able to compute the r-coloring numbers and their optimal orders in polynomial time, it is natural to ask if we can approximate the value. We first restate r-ORDERABLE as an optimization problem.

MINIMUM r-ORDERABILITY

Input:	a graph $G = (V, E)$.
Problem:	find the minimum k such that there exists a linear order $\sigma = (v_1, \ldots, v_n)$ of the vertices of G with $\text{col}_r(G_\sigma) \leq k$.

Solving MINIMUM r-ORDERABILITY would simultaneously solve r-ORDERABLE, and as we have shown, in general this cannot be done in polynomial time. However, we can look for algorithms with polynomial running time that approximate the value k. The best case scenario would be to show that MINIMUM r-ORDERABILITY is in **PTAS**. The following theorem implies that this is not possible.

Theorem 3.5.1. *For $r \geq 2$,* MINIMUM r-ORDERABILITY *does not admit a polynomial-time ρ-approximation for $\rho < \frac{7}{6}$ assuming $P \neq NP$.*

Proof. Lemmas 3.4.1 and 3.4.2 imply that 2-CLAUSE 3-SAT can be answered by determining if the auxiliary graph $G(\varphi)$ has r-coloring number at most 6. A ρ-approximation with $\rho < \frac{7}{6}$ for r-ORDERABLE would differentiate between these cases as it will return a value less than 7 if $\text{col}_r(G) \leq 6$. Since 2-CLAUSE 3-SAT is NP-hard, no such algorithm can run in polynomial time unless $P = NP$. □

Notice that this does not imply that there is no constant factor approximation of MINIMUM r-ORDERABILITY for fixed values of r. We only state that there cannot be one for every $\rho > 1$. In Chapter 4 we give a polynomial approximation of the r-coloring number that improves on the best known.

New we consider a version of the problem that is parameterized by the input k.

(k,r)-ORDERABILITY

Input: a graph $G=(V,E)$, a parameter $k\in\mathbb{N}$.

Problem: is there a linear order $\sigma=(v_1,\dots,v_n)$ of V such that $\mathrm{col}_r(G_\sigma)\le k$?.

Theorem 3.5.2. *Assuming $P\neq NP$ and $r\ge 2$, r-*ORDERABLE *is in **para-NP**-hard when parameterized by k.*

Proof. Lemmas 3.4.1 and 3.4.2 imply that for $r\ge 2$, 2-CLAUSE 3-SAT can be answered by determining if the auxiliary graph $G(\varphi)$ has r-coloring number at most 6. Our reduction shows that (k,r)-ORDERABILITY is **NP**-hard when $k=6$, and the problem is therefore in **para-NP**-hard. □

In conclusion, this chapter has shown that for $r\ge 2$ and $k\ge 6$ the target problem r-ORDERABLE cannot, in general, be solved efficiently, even when parameterized by the natural parameter. We note **para-NP**-hard problems do not admit XP-algorithms unless **P**=**NP**. Furthermore, it does not admit a polynomial time approximation scheme. In the next chapter, we will consider an approximation algorithm with a theoretical guarantee that improves on the current best known.

CHAPTER

4

APPROXIMATING THE *R*-COLORING NUMBER

4.1 Introduction

In the previous chapter, we showed that r-ORDERABLE is NP-hard to solve. This means that we cannot expect to define an algorithm that computes the r-coloring number for arbitrary graphs in polynomial time. In Section 2.4, we discussed the previous best approximation algorithm, achieving an $r(kr-1)^{r-1}$-approximation. The work depends on first approximating the r-backconnectivity of a vertex and choosing the vertex with the minimum, giving an approximate r-admissibility order that also approximates the r-coloring number. Our technique keeps track of the i-reach directly for $1 \leq i \leq r$.

The algorithm is a generalization of the exact algorithm used to find degeneracy orders extended to higher radii. It works by taking as input the graph G and a lower bound on the r-coloring number such as the degeneracy, k. It then greedily chooses any vertex to be added to an order σ provided it has i-reach below a predetermined function of k and i. These bounds are called *k-neighbor restrictive bounds* and will be defined in Section 4.3. If at any point no vertex can be chosen, we show that this implies $\mathrm{col}_r(G) > k$, and we increase k to $k+1$, recompute the bounds, and continue choosing vertices for the order. Once the algorithm completes the order σ we have an order with bounded reach. Currently, we do not have a way to describe the reach of a vertex in an graph that is not linearly ordered. In order to formally discuss our approach, we need to define partial orders.

Definition 4.1.1. Let G be a graph on n vertices. A *partial linear order* of G is an order $\sigma = (v_1, v_2, \ldots, v_i)$ for $i \leq n$. If a vertex has not yet been added to σ, we say that it is *unordered*, otherwise it is *ordered*.

By convention, if v_i is in a partial order and v is not, then we take $v_i <_\sigma v$, and unordered vertices are not related.[1] The reach of an unordered vertex is equal to its reach if were the next vertex added to σ,

4.2 *r*-Reach Tree

In this section, we describe the underlying structure of the r-reach of a vertex with respect to a linear order (or partial linear order) as a BFS tree. A tree with these properties was first mentioned in [Dvo13b] and we formalize the definition here.

Definition 4.2.1. Let G be a graph with partial linear order σ and r be a positive integer. For an ordered vertex $v \in G$ let G' be the subgraph of G with $V(G') = N_r[v]$ and $E(G') = \{xy \in E(G) : x, y \in V(G'), x \leq_\sigma v \text{ or } y \leq_\sigma v\}$. The *$r$-reach tree of v* with respect to σ, denoted $T_r(v, G_\sigma)$, is constructed as follows. Compute a breadth-first search tree in G' rooted at v, adding neighbors to the queue from low to high along σ, stopping a branch at height r or when a vertex after v in σ is reached. Remove any leaves that are less than v with respect to σ. If σ is a partial linear order and v is unordered, we compute this tree as if v has been added to σ as the next vertex.

The r-reach tree has important structural properties that we describe in the following proposition.

Proposition 4.2.2. *For a tree $T = T_r(v, G_\sigma)$ then the following hold:*

- *The leaves of T are the exactly the r-reach of v with respect to σ,*
- *If x is a leaf, and the unique v-x path in T has length d, then x is not $(d-1)$-reachable from v in G_σ,*
- *$T_i(v, G_\sigma)$ is a subtree of $T_r(v, G_\sigma)$ for $1 \leq i \leq r$.*

Proof. The properties follow from the breadth-first search (BFS) nature of the r-reach tree. From the construction, all paths from v to a leaf are paths in G. They are also qualifying paths as the interior path vertices are before v in σ. The leaves in T are then a subset of $\mathrm{R}_r(v, G_\sigma)$. If $x \in \mathrm{R}_r(v, G_\sigma)$ then there is a qualifying path consisting of edges in G', and the BFS would find x in at most r steps.

[1] In poset terminology, the vertices in σ are a *chain*, the unordered vertices are an *antichain*.

The minimum distance follows directly from the definition of a breadth-first search. The graph G' includes all edges on all qualifying paths by which x is reachable from v and the breadth-first search will always find a shortest such path.

Finally, the subtree property can be seen by showing that $T_{r-1}(v, G_\sigma)$ is a subtree of $T_r(v, G_\sigma)$. We simply remove all of the leaves from $T_r(v, G_\sigma)$ that are distance r from v and also remove any resulting leaves that appear before v in σ. The order of the breadth-first search being determined by σ guarantees that this edit gives exactly $T_{r-1}(v, G_\sigma)$. By induction, $T_i(v, G_\sigma)$ must also be a subtree of $T_r(v, G_\sigma)$. $\square$

4.3 *k*-Neighbor Restrictive Bounds

This section describes a set of bounds that can be determined from the r-reach tree. The intuition for using this interpretation of the reach is that it allows us to quantify how the earlier vertices in a linear order affect the reach of the root vertex. If we can limit the amount of branching that occurs at each vertex in a reach tree, then we also limit the number of leaves and therefore the reach of the root vertex.

We begin with a discussion of the base case, when $r = 1$. Let G be a graph with $\mathrm{col}_1(G) = k$ and $\sigma \in \Pi(G)$ witnessing this. For any $v \in V(G)$, the r-reach tree of v with respect to σ will be a star graph with at most k leaves. Also, if the 1-coloring number of G is unknown and you order the vertices by greedily choosing the next vertex to have 1-reach at most k, either you order all vertices and can state that $col_1(G) \leq k$ or at some point no vertex can be chosen, implying that $\mathrm{col}_1(G) > k$. You can then continue the order choosing vertices with 1-reach at most $k+1$. For some $k+i$, $i > 0$ the order will be completed and this implies $\mathrm{col}_1(G) = k+i$. We extend this idea to the general r.

Definition 4.3.1. Let G_σ be a (possibly partially) linearly ordered graph and consider a set $\{b_1, b_2, \ldots, b_r\} \in \mathbb{Z}^+$. We say σ *satisfies the bounds* if for all $v \in G_\sigma$, $|\mathrm{R}_i(v, G_\sigma)| \leq b_i$.

Definition 4.3.2. For a fixed integer $r \geq 1$, we call a set of functions $\{b_i(k)\}_{1 \leq i \leq r}$ *k-neighbor restrictive* if the function is increasing for $k > 0$ and for any graph G and any partial linear order σ of G, which satisfies the bounds, if v is unordered and the i-reach tree of v has more than b_i leaves, then in this tree, v has at least $k+1$ neighbors.

In the next section, we will show that if we greedily choose the vertices to satisfy a set of k-neighbor restrictive bounds and if no vertex can be chosen, then $\mathrm{col}_r(G) > k$.

4.4 Main Algorithm

In this section we present the main algorithm, which uses k-neighbor restrictive bounds to approximate the r-coloring number of a graph.

```
input  : A graph $G$, integer $r > 0$, $k$-neighbor restrictive bounds $\{b_i(k)\}_{1\le i\le r}$.
output : An order $\sigma$ and integer $k$ satisfying $\mathrm{col}_r(G_\sigma) \le b_r(k)$
1 $S \leftarrow V(G)$
2 $\sigma \leftarrow [\emptyset]$
3 $k \leftarrow 1$
4 while $S$ is not empty do
5     if there exists $v \in S$ so that $|\mathrm{R}_i(G_\sigma, v)| \le b_i(k)$ for all $i$ then
6         append $v$ to $\sigma$ and remove it from $S$
7     else
8         $k \leftarrow k+1$
9 return $\sigma$,$k$
```

Algorithm 1: BoundedColoring($G, r, \{b_i(k)\}_{1\le i\le r}$)

Theorem 4.4.1. *Given a graph G and $1 \le k \le \mathrm{col}_r(G)$, Algorithm 1 returns an order σ and value t such that $t \le \mathrm{col}_r(G) \le \mathrm{col}_r(G_\sigma) \le b_r(t)$.*

Proof. It is clear that the algorithm will return an order σ and integer k, since for sufficiently large values of k all unordered vertices will satisfy all bounds. By definition, all vertices, regardless of the value of k when they were chosen will have r-reach at most $b_r(k)$ as $b_r(k) \ge b_r(k-i)$. Therefore, $\mathrm{col}_r(G) \le \mathrm{col}_r(G_\sigma) \le b_r(k)$.

It remains to show that the output k is at most the r-coloring number of G. In the algorithm k increases by one each time when no vertex satisfies the bound $\{b_i(k)\}_{1\le i\le r}$. It suffices to show that if the algorithm is run and at some point $k = \mathrm{col}_r(G)$ then there is always a vertex that can be chosen. We suppose for some G the algorithm reaches $k = \mathrm{col}(G)$ and no unordered vertex satisfies all bounds. Therefore, every unordered vertex must not satisfy at least one $b_i(k)$ if it were chosen next. Let P be the set of vertices ordered by σ and $Q = V(G) \setminus P$. Let γ be a linear order of G so that $\mathrm{col}_r(G_\gamma) = k$. Let v be the first vertex from Q that appears in γ. Since $v \in Q$ there is some bound it did not satisfy and suppose i is the smallest such index so that $b_i(k)$ is not satisfied (i.e. $\mathrm{R}_i(v, G_\sigma) > b_i(k)$). Let $T_\sigma = T_i(v, G_\sigma)$. By definition of the reach tree T_σ has more than $b_i(k)$ leaves, all of which lie in Q. Because our bounds were k-neighbor restrictive, this implies that v had at least $k+1$ neighbors in T_σ. For each neighbor we can associate a path in T_σ from v to a leaf through this neighbor. These paths are disjoint except for v. By our choice of v, in γ the endpoints of these paths appear after v. Hence, each of these paths must have a first vertex that appears after v in γ. These vertices are i-reachable from v and there are at least $k+1$ of them. This contradicts that the r-reach of v is at most k in γ. $\square$

We note that using the right data structures to track the i-reach of every vertex Algorithm 1 can be implemented to run in time $O(r \cdot b_r \cdot n^2)$.

4.5 Deriving *k*-Neighbor Restrictive Bounds

Our focus in this section is the construction of k-neighbor restrictive bounds. Ideally, the functions will be as small as possible and it should be easy to determine the set $\{b_i(k)\}_{1\leq i\leq r}$ for an arbitrary $r > 0$. In the first part of this section, we construct minimum k-neighbor restrictive bounds and show that for $r > 4$ the process becomes particularly tedious. At the end of the section, we give an alternative set of bounds that is asymptotic to the minimum set. We are able to express this set as a family of computable polynomials.

By definition, k-neighbor restrictive bounds are recursive. Given a set of k-neighbor restrictive bounds $\{b_i(k)\}_{1\leq i\leq r-1}$, if we assume a linearly ordered graph satisfies these bounds and a vertex v has k neighbors in its r-reach tree, then $b_r(k)$ is just the maximum possible r-reach of v.

We begin with the base case.

Proposition 4.5.1. *For $r = 1$, $b_1(k) = k$ is the minimum k-neighbor restrictive bound.*

Proof. This follows trivially from the properties of degeneracy. □

Proposition 4.5.2. *For $r = 3$ and $k > 2$ the functions $\{b_1(k) = k, b_2(k) = k^2 - k, b_3(k) = 2k^3 - 3k^2\}$ are the minimum k-neighbor restrictive bounds.*

Proof. The bound $b_1(k) = k$ follows simply from Lemma 4.5.1. To compute $b_2(k)$ we suppose G is a graph and $\sigma \in \Pi(G)$ that satisfies $b_1(k) = k$. Let $v \in G$ be a vertex with exactly k neighbors in its 2-reach tree, $T = T_2(v, G_\sigma)$. We give a formula for the 2-reach of v and find a maximum. Let $N_T^+(v) = \{x \in N_T(v) : v \leq_\sigma x\}$ and $N_T^-(v) = \{x \in N_T(v) : x \leq_\sigma v\}$. These sets must be disjoint and satisfy: $|N_T^+(v)| + |N_T^-(v)| = k$. Each vertex in $N_T^+(v)$ adds exactly one to the 2-reach of v as they appear after v in σ. If $x \in N_T^-(v)$, x is earlier than v in σ and can only contribute to the 2-reach of v via a forward edge to a vertex past v. This vertex is in the 1-reach of x, which is at most k because σ satisfies $b_1(k) = k$. Clearly, v is in the 1-reach of x, so x adds at most $k-1$ to the 2-reach of v. Hence,

$$|\mathrm{R}_2(v, G_\sigma)| \leq |N_T^+(v)| + (k-1)|N_T^-(v)| = k - |N_T^-(v)| + (k-1)|N_T^-(v)| = k + (k-2)|N_T^-(v)|.$$

For $k > 2$ this is maximized by making $|N_T^-(v)|$ as large as possible, which gives $|N_T^-(v)| = k$. Therefore, the maximum 2-reach of v is at most $k^2 - k$ and by our construction we have demonstrated that this 2-reach is attainable.

We repeat this process to compute $b_3(k)$, assuming a graph G and order σ that satisfy $b_1(k) = k$ and $b_2(k) = k^2 - k$. Again, suppose $v \in G$ and v has k neighbors in its 3-reach tree $T = T_3(v, G_\sigma)$. The sets $N_T^+(v)$ and $N_T^-(v)$ are as before. As before each vertex in $N_T^+(v)$ contributes at most one to the 3-reach of v.

For each backward neighbor $w \in N_T^-(v)$, there are three types of paths that can add to the reach of v. First, if w has a neighbor that appears after v in σ, then v reaches this vertex. The next case is that w has a neighbor that is after w but before v in σ. Similar to above, the total number of forward neighbors of w is k, one of which is v because the 1-reach of w must be less than $b_1(k) = k$. If a forward neighbor of w has k neighbors after v then it adds k to the reach of v.

Additionally, w can have neighbors that appear before it in σ. We have no guarantee as to the quantity, only the guarantee that any vertices that v can reach through w along such a path must also be in the 2-reach of w and therefore there are less than $b_2(k)$. We notice that each forward neighbor of w can add at most k to the reach of v but subtracts one from the 2-reach of w. Therefore if $1 \le p_w \le k$ is the number of forward neighbors for some $w \in N_T^-(v)$ then let $r_v(w)$ be a function describing the reach added to v through a back neighbor w. To maximize this value we assume all forward neighbors of w other than v appear before v in the order and have k neighbors after v. This gives the formula:

$$r_v(w) \le b_2(k) - p_w + (p_w - 1)k \le 2k^2 - 3k.$$

For $k > 2$, for the purposes of maximizing the value, we assume that all of v's neighbors in its 3-reach tree are backward neighbors. Giving

$$\begin{aligned} |\mathrm{R}_3(v, G_\sigma)| &\le \sum_{w \in N_T^-(v)} r_v(w) \\ &\le k(2k^2 - 3k) \\ &= 2k^3 - 3k^2 \end{aligned}$$

Hence $2k^3 - 3k^2$ is the maximum attainable reach for v.

□

We see that calculating the function $b_i(k)$ is dependent on the bounds $b_1(k), \ldots, b_{i-1}(k)$. At higher radii this dependence and the interaction between interior vertices in the reach tree becomes harder to manage. Example 4.5.3 demonstrates this complexity.

Example 4.5.3. In this example we give a demonstration of the difficulty that occurs when trying to calculate $b_4(k)$.

We suppose that $r = 4$ and $k = 4$ and assume we have a graph G and order σ that satisfies $b_1(4) = 4$, $b_2(4) = 4^2 - 4 = 12$, and $b_3(4) = 2(4)^3 - 3(4)^2 = 80$. The paths of length four allow for the most branching and therefore the highest possible reach. We describe each type of maximal path by the direction of the edges with respect to σ, where '-' gives an edge that moves backwards and '+' an edge that moves forward. All paths must begin with '-' and end with a '+' to be valid. This gives the possible paths as $(-,-,-,+)$,$(-,-,+,+)$,$(-,+,-,+)$, and $(-,+,+,+)$. Figures 4.1, 4.2, 4.3, and 4.4 illustrate these paths, respectively.

In these calculations, we wish to maximize the branching to get the highest possible reach.

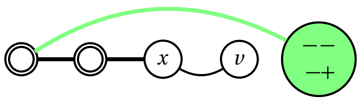


Figure 4.1 $(-,-,-,+)$: We see that x adds at most $b_3(4)-1$ to the reach of v. Any vertex v reaches through such path is also in the 3 reach of v. Since x also reaches v it is at most $b_3(4)-1$. If x reaches other vertices through other path types we must subtract this from $b_3(4)$ as well.

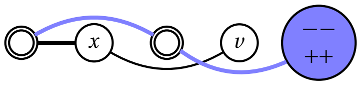


Figure 4.2 $(-,-,+,+)$:x adds at most $(b_2(4)-1)\cdot b_1$ in this case. Again any 1-reach of x must also be subtracted from the $b_2(4)$ term.

We start with $(-,+,+,+)$, which is the most straightforward. There are the maximum $k=4$ back neighbors of v. Each has at most three forward neighbors other than v, each of these have at most 4 forward neighbors, and each of these have at most 4 forward neighbors. This gives a total reach of v through this type of path as $4\cdot 3\cdot 4\cdot 4=192$. We note that this assumes the x vertices have maximum 1-reach, as do the x_i vertices.

Next, we consider paths of type $(-,+,-,+)$. Each x_1 type vertex adds $b_2(4)-b_1(4)-3$, the x_2 vertices each add $b_2(4)-b_1(4)-2$ and the x_3 vertices each add $b_2(4)-b_1(4)-1$. This gives a total of $k((b_1(4)-1)b_2(4)-(b_1(4)-1)b_1(4)-6)=72$. We point out here that 6 can be calculated as the $k-1$ triangle number. For the type $(-,-,+,+)$ we get that each x vertex adds $b_2(4)-b_1(4)$ giving $k(b_1(4)\cdot b_2(4)-b_1(4))=176$. Finally through type $(-,-,-,+)$ we have each x adds b_3-b_2 giving $k(b_3-b_2)=272$. Putting this all together $b_4(4)=712$.

The example above shows that we are able to calculate the next bound in a set of k-neighbor restrictive bounds by simply looking at all paths of length r that begin with a $-$ and end with a $+$.

In general, we can use this computation to show a k-neighbor restrictive bound $b_4(k)$ is given by:

$$b_4(k)\le k((b_1-1)(b_1)(b_1)+((b_1-1)b_2(k)-(b_1-1)b_1-(b_1)(b_1-1)/2)+(b_1b_2-b_1)+(b_3-b_2)).$$

Or, in terms of k:

$$b_4\le 5k^4-\frac{19k^3}{2}+\frac{5k^2}{2}.$$

Indeed, calculating the exact bounds becomes inefficient. The interactions demonstrated in Example 4.5.3 between vertices that can reach each other obfuscate the desired calculation. We instead construct bounds that are slightly larger, specifically taking the dominating term of the best possible. To achieve this we disregard whether the construction is feasible and just give the maximum possible for each path type in terms of the bounds.

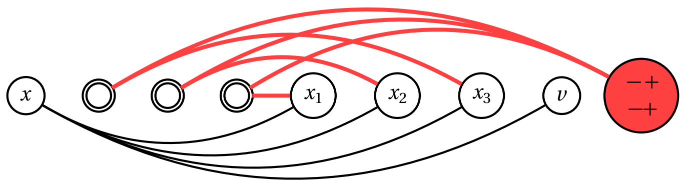


Figure 4.3 $(-,+,-,+)$: Since $b_1(4)=4$, x has at most $k=4$ forward neighbors, including v. Assuming $x_1 <_\sigma x_2 <_\sigma x_3$ we get that x_1 adds at most $b_2(4)-3$ on this path type, x_2 adds at most $b_2(4)-2$, and x_3 adds at most $b_2(4)-1$. This is because x_i can reach all x_j for $j>i$ and also v.

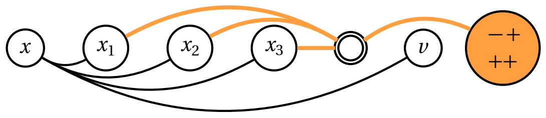


Figure 4.4 $(-,+,+,+)$: Again x can have at most $k=4$ forward neighbors, including v, and they can also have at most k forward neighbors. So through this path type each x_i adds at most $(b_1(4)-1)\cdot b_1(4)$ to the reach of v.

Specifically, for each path type we compute the maximum possible reach and ignore any interactions between vertices along these paths. Under this method, if we were to recompute the bound given by Example 4.5.3, we would not consider that the x_i can reach v or each other. We call these the *simplified* k-neighbor restrictive bounds.

Observation 4.5.4. For a given path type, the simplified reach possible through such a path can be read as a product from the -,+ description. Specifically, the first minus gives a factor of k since these are the k neighbors of the root in the r-reach tree. Starting from the left we count consecutive terms until a $+$ is reached. Since we assume all maximal paths are saturated, we subtract the next lower index from each. For instance $(-,-,-,+,-,+)$ would be broken up into $(-),(-,-,+),(-,+)$ giving $(k)(b_3-b_2)(b_2-b_1)$. Also $(-,+,-,+,+)$ would be $(-),(+),(-,+),(+)$ giving $(k)(b_1-1),(b_2-b_1),(b_1)$. If the term after the first minus is a plus we subtract one since we know this vertex can also reach the root. Since all terms have the leading k term we can remove it from our path types. So we think of the first example $(-,-,-,+,-,+)$ as k times the product $(-,-,+),(-,+)=(b_3-b_2)(b_2-b_1)$.

If p is a path type, we let $s(p)$ be the product described above.

Definition 4.5.5. We let $\mathcal{P}(r)$ denote the set consisting of the 2^{r-2} *path types* given by sequences of length $r-1$ consisting of $+$ and - such that the last term is a plus.

As in Observation 4.5.4 we omit the first negative sign, hence these are vectors of length $r-1$.

Lemma 4.5.6. *The set of bounds $b_1(k), b_2(k), \ldots, b_r(k)$, where*

$$b_1(k)=k \text{ and } b_i(k)=k\sum_{p\in\mathcal{P}(i)} s(p),$$

are k-neighbor restrictive.

Proof. This follows by induction. If $r = 1$ it is clearly true. If we assume it is true for $r \leq \ell - 1$, then $b_\ell(k)$ is constructed by definition to be greater than the max (ℓ)-reach possible by a vertex with k neighbors in its ℓ-reach tree in an order satisfying $b_1(k), \ldots, b_{\ell-1}(k)$. □

The highest degree term of the resulting polynomial is of particular interest as it is the same in the minimum k-neighbor restrictive bounds. It can also be described relatively simply.

Lemma 4.5.7. *Let C_j be the jth Catalan number. If $b_i(k)$ is a bound from Lemma 4.5.6, then the highest degree term of $b_i(k)$ is $C_{i-1}k^r$.*

We recall that the jth Catalan number can be given as an equivalence relation:

$$C_j = \sum_{i=0}^{j-1} C_i C_{j-1-i}$$

Proof. It holds for $b_1(k)$. Suppose it is true up to $b_\ell(k)$, $1 < \ell < r$. We calculate the sum in Lemma 4.5.6 in parts. Let $\mathcal{P}_i(\ell+1)$ be the set of path types in $\mathcal{P}(\ell+1)$ with the first plus sign at index i, $1 \leq i \leq \ell$. Notice that removing the first i terms from a type gives a new type in $\mathcal{P}(\ell+1-i)$. The sum of these multiplied by k is exactly $b_{\ell+1-i}(k)$. Hence we can rewrite $b_{\ell+1}$ as follows:

$$\begin{aligned} b_{\ell+1}(k) &= k \sum_{\substack{p\in\\ \mathcal{P}(\ell+1)}} s(p) \\ &= k \sum_{i=1}^{\ell} \left(\sum_{\substack{q\in\\ \mathcal{P}(\ell+1-i)}} b_i s(q) \right) \\ &= \sum_{i=1}^{\ell} \left(b_i k \sum_{\substack{q\in\\ \mathcal{P}(\ell+1-i)}} s(q) \right) \\ &= \sum_{i=1}^{\ell} b_i b_{\ell+1-i} \end{aligned}$$

Let D_j denote the dominating term of b_j. Then,

$$D_{\ell+1} = \sum_{i=1}^{\ell} D_i D_{\ell+1-i}$$

Adjusting the indices and by the inductive hypothesis we get:

$$\begin{aligned}
D_{\ell+1} &= \sum_{i=0}^{\ell-1} C_i k^{i+1} C_{\ell-1-i} k^{\ell-i} \\
&= k^{\ell+1} \sum_{i=0}^{\ell-1} C_i C_{\ell-1-i} \\
&= C_\ell k^{\ell+1}
\end{aligned}$$

□

In fact these dominating terms also give a set of k-neighbor restrictive bounds.

Lemma 4.5.8. *The set of bounds $\{b_i = C_{i-1} k^i\}$, $1 \le i \le r$ are k-neighbor restrictive.*

Proof. This set of bounds ignores all interactions when calculating a path types score. It is then calculated in the same way as the bounds in Lemma 4.5.6 giving the same dominating term and result. □

Using this set of bounds with Algorithm 1 gives an order of an arbitrary graph that witnesses a $C_{r-1} t^{r-1}$-approximation of the r-coloring number in polynomial time, where k is at most the r-coloring number of the graph.

Theorem 4.5.9. *Let G be a graph, C_j be the jth Catalan number and $r \ge 1$ be an integer. Using $\{b_i(x) = C_{i-1} x^i\}_{1 \le i \le r}$, Algorithm 1 gives a $C_{r-1} k^{r-1}$-approximation for the* MINIMUM r-ORDERABILITY *problem in time polynomial in n, where $k = \mathrm{col}_r(G)$.*

Proof of Theorem 4.5.9. By Lemma 4.5.8 the bounds are k-neighbor restrictive and our starting value $k = 1 \le \mathrm{col}_r(G)$. By Theorem 4.4.1 Algorithm 1 returns an order σ and value t satisfying:

$$k \le \mathrm{col}_r(G) \le \mathrm{col}_r(G_\sigma) \le C_{r-1} k^r$$

This gives the desired approximation. □

Part II

Phylogenetic Trees

CHAPTER

5 DISENTANGLING PHYLOGENIES

5.1 Introduction

In evolutionary biology, phylogenetic trees provide a useful model to study speciation. Given a collection of species (or *taxa*, short for taxonomic unit) the fundamental goal of phylogenetic study is to find the tree that best captures the evolutionary branching, and in some cases the evolutionary distance, between these species. Before the development of genetic analysis, most models were formed from strictly morphological data. In recent years the ability to compare species based on their genetic differences has led to more robust techniques and an influx in data points. Phylogenies may be inferred in a variety of ways, including discretely coded characters, gene frequencies, and molecular sequences. We refer the reader to [Fel04] for a more in depth survey. Each method comes with its own set of trade-offs and assumptions. Parsimony methods, for example, look for the phylogeny with the minimum number of evolutionary events necessary. Maximum likelihood and Bayesian methods use a more probabilistic approach and assumes a probability distribution. In this Chapter we focus on the identification, or *disentangling*, of sets and multisets of phylogenies and combining phylogenies with contrasting taxa. This is particularly relevant to certain statistical techniques, which we briefly touch on.

A *character* is a function from the set of species X into a set of states S. Here S could be the binary existence of a morphological feature or the nucleotide at a position in a genetic sequence. In statistical techniques, using the characters from a specific gene it is possible to estimate the evolutionary tree corresponding to that gene. Considering the set of trees for a collection of genes one goal is to recreate the phylogeny representing the evolution of the taxa. While these models tend to be easier to work with under the assumption that all genes evolve under a single Markov process, this is perhaps not the most accurate. An alternative *phylogenetic mixture model* assumes that each character is generated independently allowing for the possibility that a particular gene evolved at a different rate or along a different tree. Each character is given its own probability distribution and these are combined into a *phylogenetic mixture*. However, when mixing Markov processes it is necessary that the input tree parameters remain identifiable. This is called disentangling a set (or multiset) of phylogenies. In this chapter we study the disentangling number and improve upon existing results using a graph isomorphism argument.

5.2 Preliminaries

The work in this chapter is focused on the identification of sets and multisets of binary $[n]$-trees using the induced subtrees. Recall that the set of all binary $[n]$-trees is denote $B(n)$ The notation $\binom{B(n)}{r}$ is used to refer to all subsets of $B(n)$ consisting of r trees and likewise $\binom{RB(n)}{r}$ for subsets of $RB(n)$ of size r. A double parentheses notation, $\left(\!\binom{B(n)}{r}\!\right)$ and $\left(\!\binom{RB(n)}{r}\!\right)$, is used when we consider multisets of size r. When elements of such sets and multisets are written out explicitly we will use the notation $\{T_1,\dots,T_r\}\in\binom{B(n)}{r}$ and $[T_1,\dots,T_r]\in\left(\!\binom{B(n)}{r}\!\right)$.

Let $\mathcal{T}=\{T_1,\dots,T_r\}\in\binom{B(n)}{r}$ and $K\subseteq[n]$. We can define the *set of subtrees induced by* K as $\mathcal{T}_{|K} := \{T_{1|K},\dots,T_{r|K}\}$. Similarly, if we let $\mathcal{T}=[T_1,\dots,T_r]\in\left(\!\binom{B(n)}{r}\!\right)$ and $K\subseteq[n]$ then $\mathcal{T}_{|K} := [T_{1|K},\dots,T_{r|K}]$ is also a multiset. A key distinction here is that if $\mathcal{T}\in\left(\!\binom{B(n)}{r}\!\right)$ then $\mathcal{T}_{|K}$ is a multiset in $\left(\!\binom{B(K)}{r}\!\right)$. However, if $\mathcal{T}\in\binom{B(n)}{r}$ then $\mathcal{T}_{|K}$ is a potentially smaller set in $\binom{\mathcal{T}^K}{\le r}$.

The following example motivates this work. Suppose $\mathcal{T}\in\binom{B(n)}{r}$ and we are given $\mathcal{T}_{|K}$ for all $K\in\binom{[n]}{k}$, where k is a fixed non-negative integer. For what integer k is it always possible to identify $\mathcal{T}$ as the input? Determining this k would imply that the induced subtree map on all k element subsets gives a unique signature for all $\mathcal{T}\in\binom{B(n)}{r}$. The analog of this question but for $\mathcal{T}\in\left(\!\binom{B(n)}{r}\!\right)$ is also relevant and considered in Section 5.4. The value of k is dependent on r but is not dependent on n. Indeed, sets and multisets of trees with an arbitrary number of leaves can be identified via relatively small subsets of the leaves.

Definition 5.2.1. Let $\mathcal{S},\mathcal{T}\in\binom{B(n)}{r}$, suppose $\mathcal{S}\neq\mathcal{T}$. We say $K\subseteq X$ *disentangles* $\mathcal{S}$ and $\mathcal{T}$ if $\mathcal{S}_{|K}\neq\mathcal{T}_{|K}$. For $n\ge k$, we say that $\binom{[n]}{k}$ *disentangles* $\binom{B(n)}{r}$ if for any two sets $\mathcal{T},\mathcal{S}\in\binom{B(n)}{r}$ such that $\mathcal{S}\neq\mathcal{T}$, there exists $K\in\binom{[n]}{k}$ such that $\mathcal{T}_{|K}\ncong\mathcal{S}_{|K}$. The *set disentangling number* $D(r)$ is the smallest k such that $\binom{[n]}{k}$ disentangles $\binom{B(n)}{r}$ for all $n\ge k$. The *multiset disentangling number* $\tilde{D}(r)$ is the smallest k such that $\binom{[n]}{k}$ disentangles $\left(\!\binom{B(n)}{r}\!\right)$ for all $n\ge k$. The *rooted set disentangling number* and *rooted multiset disentangling number* given by $RD(r)$ and $R\tilde{D}(r)$, respectively, have analogous definitions.

5.3 Known Results

First introduced by Matsen et al. [Mat08], the set disentangling numbers were later generalized by Humphries [Hum08]. Much of their work was focused on reconstructing sets of trees from their combined splits.

Definition 5.3.1. A *quartet tree* is a binary phylogeny with four leaves. A *rooted triple* is a rooted binary phylogeny with three leaves. For $T \in B(n)$ we define the *quartets displayed by* T as

$$\mathcal{Q} := \{T_{|K} : K \in \binom{[n]}{4}, T_{|K} \in B(n)\}.$$

For $T \in RB(n)$ we define the *rooted triples displayed by* T as

$$\mathcal{R} := \{T_{|K} : K \in \binom{[n]}{3}, T_{|K} \in RB(n)\}.$$

Lemma 5.3.2 ([Ste16])**.** *Let* $T, T' \in B(n)$*. Then* $T \cong T'$ *if and only if* $\mathcal{Q}(T) = \mathcal{Q}(T')$*.*

Lemma 5.3.3 ([Ste16])**.** *Let* $T, T' \in B(n)$*. Then* $T \cong T'$ *if and only if* $\mathcal{R}(T) = \mathcal{R}(T')$*.*

Corollary 5.3.4. $D(1) = \tilde{D}(1) = 4$ *and* $RD(1) = R\tilde{D}(1) = 3$*.*

The above lemmas are often stated as "quartets determine binary trees" and "rooted triples determine rooted binary trees" and are commonly referenced. In [Mat08] it was shown that this value exists for $r = 2$.

Theorem 5.3.5 ([Mat08])**.** $D(2) = 6$*.*

In general, It is not immediately clear that we should expect a value of $D(r)$ to exist. However in [Hum08] it was proved that $D(r)$ is monotonic in r and well-defined. Furthermore, a first upper bound for $D(r)$ was proven.

Theorem 5.3.6 ([Hum08])**.** *The function* $D(r)$ *is monotonic, well-defined and* $3(\lfloor \log_2 r \rfloor + 1) \leq D(r) \leq 3r$ *for all* $r \geq 2$*.*

The multiset and rooted multiset disentangling numbers were introduced by Sullivant in [Sul11]. We note that when $r = 1$ and $r = 2$ there is no difference between the set and multiset variants of the problem. As early as $r = 3$, the set variant allows for multiple trees to map to the same induced subtree. For example, consider $\mathcal{T} = \{T_1, T_2, T_3\}$ and $\mathcal{S} = \{S_1, S_2, S_3\}$ in $\binom{[n]}{3}$ and let $K \subset [n]$. If $T_{1|K} = T_{2|K} = S_{1|K}$ and $T_{3|K} = S_{2|K} = S_{3|K}$ we would be unable to differentiate between $\mathcal{T}_{|K}$ and $\mathcal{S}_{|K}$ as they are sets. In the multiset case this would have disentangled $\mathcal{T}$ and $\mathcal{S}$. The multiset disentangling number is weaker and this gives us that $\tilde{D}(r) \leq D(r)$ and $R\tilde{D}(r) \leq RD(r)$. The following theorem shows that there are only two possible values for $\tilde{D}(r)$ for all $r \geq 2$.

Theorem 5.3.7 ([Sul11])**.** $3(\lfloor \log_2 r \rfloor + 1) \leq \tilde{D}(r) \leq R\tilde{D}(r) + 1 = 3(\lfloor \log_2 r \rfloor + 1) + 1$

The proofs rely in large part on arguments from polytope theory and the encoding of multisets of trees as high-dimensional contingency tables. In [Lon16] Long offered a more direct proof of the final equality that relied entirely on the tree structure. Additionally, the multiset disentangling number for $r = 3$ was proved.

Theorem 5.3.8 ([Lon16])**.** $\tilde{D}(3) = 6$.

5.4 Disentangling Multisets of Four Unrooted Binary Trees

In this section we will adapt some of the techniques used by Long to prove that $\binom{[n]}{4}$ disentangles $\left(\!\binom{B(n)}{r}\!\right)$. Most proofs will requires us to assume under an induced map two multisets are equivalent and produce a contradiction. Although these sets allow multiplicity they still require a one-to-one map between elements in two sets for them to be considered equivalent. The following definition formalizes this notion.

Definition 5.4.1. Let $\mathcal{T}, \mathcal{S} \in \left(\!\binom{B(n)}{r}\!\right)$ and K a subset of $[n]$ that does not disentangle $\mathcal{T}$ and $\mathcal{S}$. Label the trees of $\mathcal{S}$ and $\mathcal{T}$ so that $\mathcal{T} = [T_1, \ldots, T_r]$ and $\mathcal{S} = [S_1, \ldots, S_r]$. For $1 \leq m \leq r$, let

$$m_i = \min(\{m \in [r] \setminus \{m_1, \ldots, m_{i-1}\} : S_{m|K} = T_{i|K})$$

Then with respect to the chosen labeling, we say that S_{m_i} and T_i are *partners* at K.

This gives a matching between the elements of $\mathcal{T}_{|K}$ and $\mathcal{S}_{|K}$. A useful tool in our analysis will be subsets of $[n]$ that have size $n-1$. For $i \in [n]$ we use K_i to denote the set $[n] \setminus \{i\}$. Notice that if $\mathcal{T}$ and $\mathcal{S}$ in $\left(\!\binom{B(n)}{r}\!\right)$ are not disentangled by sets of size $n-1$ then each S is partnered with some T_l for each K_i. If S and T_l are partners for j of the K_i then we call them *j-partners*. We can also count the total number of matches that could happen between two trees in $\mathcal{T}$ and $\mathcal{S}$.

Definition 5.4.2. Let $S, T \in B(n)$, $n > 3$. Then $\kappa(S, T)$ is the largest subset of $[n]$ such that $S_{|K_i} = T_{|K_i}$ for all $i \in \kappa(S, T)$.

We can tell a significant amount about the structural similarities of S and T from the size of $\kappa(S, T)$.

Definition 5.4.3. Let $S, T \in \mathcal{T}_{[n]}$, $n > 3$. If $S \neq T$ but $S \simeq T$ and there exists an isormorphism $\varphi : S \to T$ that is the identity map everywhere except a cluster i, j, k. Then we call this map a *cluster permutation*.

Lemma 5.4.4. *Let $S, T \in B(n)$, $n > 3$. Then the following properties hold:*

(i) *If $|\kappa(S,T)| = 1$, with $\kappa(S,T) = \{i\}$, then there are two unique edges, e_S, e_T in $S_{|K_i} = T_{|K_i}$ that can be subdivided and have the pendant with leaf i attached to recover S and T, respectively.*

(ii) *If $|\kappa(S,T)| = 2$, with $\kappa(S,T) = \{i,j\}$, then in the graph $S_{|K_i} = T_{|K_i}$ there are three possible placements for i to recover T and S, each on an edge incident to the neighbor of j. (See Figure 5.1).*

(iii) *If $|\kappa(S,T)| = 3$, with $\kappa(S,T) = \{i,j,k\}$, then S and T are isomorphic and there is a cluster permutation from S to T (See Figure 5.2).*

(iv) *If $|\kappa(S,T)| > 3$, $\kappa(S,T) = [n]$ and $S = T$.*

Proof. This is an expansion of a proof by Long [Lon16]. Given S and T such that $\kappa(S,T) = \{i\}$, property (i) follows immediately from the observation that when the leaf labeled i is removed, S and T each have a single vertex of degree 2. The unique placement of this vertex and the edge that results from smoothing it gives e_S and e_T in $S_{|K_i} = T_{|K_i}$.

To show (ii) if $K_{i,j} = [n] \setminus \{i,j\}$ then $S_{|K_{i,j}} = T_{|K_{i,j}}$. Call this graph H. Let $v_i(S)$ be the vertex labeled i in S and $e_i(S)$ the edge incident to $v_i(S)$. Since $S|_{K_j} = T|_{K_j}$ it is clear that e_i is attached to the same edge in H for both of these. A similar argument applies to the edge that e_j is attached to in H. If these edges were distinct in H then S would be identical to T, a contradiction. This implies that in both S and T the edges that e_i and e_j attach to collapse to the same edge in H. Figure 5.1 gives all such possible placements of i with respect to j. If $\kappa(S,T) < n$ this implies that $S \neq T$ and we must have different placements of i in each.

If $\kappa(S,T) = \{i,j,k\}$ as in (iii), then Long [Lon16] showed that v_i, v_j, and v_k are at most distance three from each other in both S and T. This implies that they are the same tree apart from the labels of an i,j,k cluster. The isomorphism is clear as it takes a graph in Figure 5.2 to some other graph in Figure 5.2 by permuting the labels on the cluster.

Finally, to prove (iv) suppose $\kappa(S,T) > 3$ and $S \neq T$. Again all vertices in $\kappa(S,T)$ must be distance three from each other. This implies two different rooted trees with at least four vertices that, for all subsets of size 3 have identical induced subtrees. This contradicts $RD(1) = 3$. □

It immediately follows from the definitions that if S_i and T_j are ℓ-partners then $\kappa(S_i, T_j) \geq \ell$.

Corollary 5.4.5. *If $\varphi : T \to S$ is a cluster permutation on cluster $\{i,j,k\}$ then $\kappa(T,S) = \{i,j,k\}$.*

Lemma 5.4.6. *Let $S,T,U \in B(n)$ for $n > 5$. If $\varphi : S \to T$ is a cluster permutation and $\sigma : T \to U$ is a cluster permutation, then $|\kappa(S,U)|$ is either $0, 3$, or n.*

Proof. Since $n > 5$ all clusters in S, T, and U do not share any leaves. Hence U is isomorphic to S by a permutation of the clusters of S defined by $\sigma \circ \varphi : S \to U$. If both φ and σ permute the same cluster then the map is a cluster map or the identity map everywhere. If two clusters are permuted by the composition then for any i we have $S_{|K_i} \neq U_{|K_i}$ and $\kappa(S,U)$ is empty. □

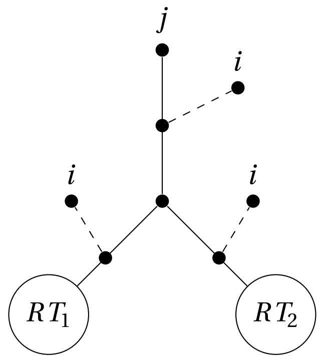


Figure 5.1 The three possible edges the leaf labeled i can be attached to if two graphs match on K_i and K_j.

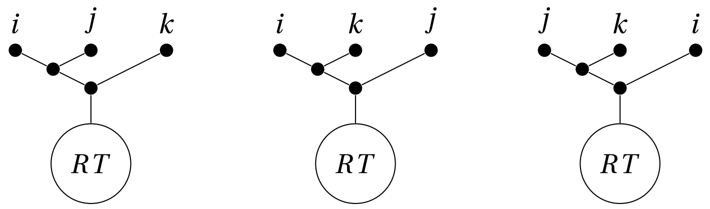


Figure 5.2 If $\kappa(S,T) = \{i, j, k\}$ then they are isomorphic graphs with a label permutation on a single cluster. Above are the 3 possible clusters [Lon16].

Corollary 5.4.7. *Let $S, T, U \in B(n)$ for $n > 5$. If $|\kappa(T,S)| = 3 = |\kappa(T,S')|$ then either $\kappa(T,S) = \kappa(T,S')$ or $\kappa(T,S) \cap \kappa(T,S') = \emptyset$.*

We see that when r is relatively low compared to the possible $D(r)$ the number of partners each tree must have to avoid being disentangled is relatively high. This technique will be less beneficial for larger values of r as r overtakes $D(r)$ relatively quickly ($r \geq 16$). For ease of notation we define two matrices to store matches and the size of κ.

Definition 5.4.8. Let $\mathcal{S}, \mathcal{T} \in \left(\!\binom{B(n)}{r}\!\right)$ and label the trees of $\mathcal{S}$ and $\mathcal{T}$ so that $\mathcal{T} = [T_1, \ldots, T_r]$ and $\mathcal{S} = [S_1, \ldots, S_r]$. Let $M_p(\mathcal{S}, \mathcal{T})$ be the matrix with $M_p[i, j] = \ell$ where T_i and S_j are ℓ-partners. Let $M_\kappa(\mathcal{S}, \mathcal{T})$ be the matrix with $M_\kappa[i, j] = |\kappa(T_i, S_j)|$.

For both matrices in Definition 5.4.8 we may assume that the values along the diagonal are at least as large as all values below them in the column, after them in their row or on a lower diagonal. If not through row and column swaps this can be achieved and $\mathcal{S}$ and $\mathcal{T}$ can be relabeled.

We can now prove the main result of this chapter.

Theorem 5.4.9. $\tilde{D}(4) = 9$.

Proof. Suppose $\tilde{D} = 10$ and let $K_i = [10] \setminus i$. Then there must exist $\mathcal{S}, \mathcal{T} \in \left(\!\binom{B(10)}{4}\!\right)$ such that $\mathcal{S}_{|K_i} = \mathcal{T}_{|K_i}$ for all $1 \leq i \leq 10$. We may assume that no tree in $\mathcal{S}$ is identical to a tree in $\mathcal{T}$. If they were we could remove them and disentangle the shorter multisets. We form the matrix $M = M_p(\mathcal{T}, \mathcal{S})$, and consider the possible values in M. T_i must match with some S_j for each K_l, hence it is a partition of 10 into four parts. Based on Lemma 5.4.4, since no $T_i = S_j$ each part must have size at most 3. Therefore every row and column will have the form $\{3,3,3,1\}$ or $\{3,3,2,2\}$ as the only possibilities up to some permutation.

First forbidden minor: We first notice that M cannot have the following matrix as a two by two minor in any orientation:

$$\begin{pmatrix} 3 & 1 \\ 3 & 3 \end{pmatrix}.$$

Suppose T_i and S_i correspond to the first row and column, respectively, and T_j, S_j to the second row and column. This implies a composition of cluster maps $T_i \to S_i \to T_j \to S_j$. By Lemma 5.4.6 T_i and S_j are isomorphic up to permutations of their clusters. Since T_i must correspond to a row vector of the form $\{3,3,3,1\}$, its structure is that of three clusters rooted from three edges adjacent to a trivalent vertex. Some leaf not in any cluster must be attached along one of these edges. Since S_j is isomorphic to T_i via cluster maps, S_j has the same structure including the placement of this leaf. If T_i and S_j match on this tenth leaf as is required then $T_i = S_j$, a contradiction to the assumption that $\mathcal{T}$ and $\mathcal{S}$ are disjoint.

If any row is of the form $\{3,3,3,1\}$ then the column containing the 1 must also be of this form. This implies no other entry in M is a 3, which is not possible base all columns and rows summing to 10. Therefore, every row and column is of the form $\{3,3,2,2\}$.

Second forbidden minor: We now assume all rows contain exactly two 3's and two 2's as entries. We consider the following possible two by four minor:

$$\begin{pmatrix} 3 & 3 & 2 & 2 \\ 2 & 2 & 3 & 3 \end{pmatrix}$$

Suppose T_1 and T_2 correspond to the first and second rows, respectively, and the columns are labeled in order by S_1, S_2, S_3 and S_4. There is then a cluster permutation $\varphi_1 : T_1 \to S_1$ and a cluster map on a distinct cluster $\varphi_2 : T_1 \to S_2$. Let c_1, c_2 be the specific clusters in T_1. Without loss of generality, suppose S_1 has a permutation of c_1 given by c_1' and an identical c_2. Therefore S_2 has the cluster c_1 and a permutation of c_2, denoted c_2'. A symmetric argument gives that T_2, S_3, and S_4 also have two distinct clusters. T_2 and S_1 are 2-partners, which implies they each share at least one identical cluster c_1' or c_2, since the tree induced by some K_i cannot affect both clusters. The same follows for T_2 and S_2, implying T_2 has either the cluster c_1 or c_2'. It follows that T_2 has one cluster that is either c_1 or c_1' and also a cluster that is c_2 or c_2'. The cluster permutations from T_2 to S_3 and S_4 imply they also have clusters permutations of c_1 and c_2. However since S_3 and S_4 are 2-partners with T_1 for $\{i, j\} \subseteq [10]$ that are not contained in either cluster they both must have exactly c_1 and c_2. This implies that T_2 cannot have c_1' and c_2' as it would be 0-partners with S_3 and S_4. It also cannot have c_1 and c_2 since

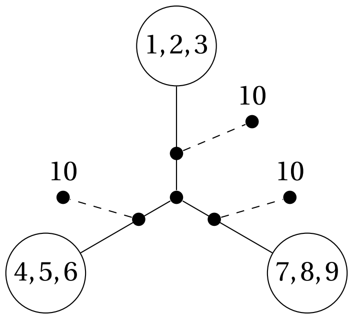


Figure 5.3 The structure of T_i corresponding to the first forbidden minor in the proof of Theorem 5.4.9 with the 3 possible placements for the vertex labeled 10.

by a counting argument it must have a cluster map to one of them along one of these clusters that permutes either c_1 or c_2. In either case T_2 either has c_1' and c_2 or c_1 and c_2' and cannot be 2-partners with one of S_1 or S_2. A contradiction.

To complete the proof we show that no four by four matrix consisting of rows and columns of the form $\{3,3,2,2\}$ does not contain the second forbidden minor. We simply take a row to be $\{3,3,2,2\}$. The first column must have two 2's. The second column then cannot have any twos on the same column as these twos which implies the second column has three 3's, a contradiction to all rows having exactly two 3's.

□

5.5 Disentangling Sets of 3 Unrooted Trees

Here we use a similar technique as above, but we will rely heavily on the matrix M_κ. In this setting, we do not have the guaranteed 1-to-1 mapping between $\mathcal{S}$ and $\mathcal{T}$, so we are unable to use the partners definition from the previous section. This alone has a significant effect on the number of possible matrices to consider. Additionally, it cannot be assumed that $\mathcal{T} \cap \mathcal{S} = \emptyset$. To demonstrate this, suppose $\mathcal{T} = \{T_1, T_2\}$ and $\mathcal{S} = \{S_1, S_2\}$ and the set K disentangles $\mathcal{T}$ and $\mathcal{S}$. One troublesome hypothetical would be the situation where $T_{1|K} = S_{1|K} = T_{2|K}$ and $S_{2|K}$ is unique to $\mathcal{S}_{|K}$. Choose any T so that $T_{|K} = S_{2|K}$, if T is added to both $\mathcal{S}$ and $\mathcal{T}$ we get sets with a non-empty intersection that are no longer disentangled by K. If the intersection of $\mathcal{T}$ and $\mathcal{S}$ is empty, then the subsets of $[n]$ of size 6 are sufficient to disentangle them.

Theorem 5.5.1. *For any $\mathcal{S}, \mathcal{T} \in \binom{B(7)}{3}$ with $\mathcal{S} \neq \mathcal{T}$, there exists K_i, $1 \le i \le 7$ so that $\mathcal{S}_{|K_i} \neq \mathcal{T}_{|K_i}$.*

Proof. We suppose not, to produce a contradiction, and let $\mathcal{S}, \mathcal{T} \in \binom{B(7)}{3}$ such that for all $i \in [7]$ we have $\mathcal{S}_{|K_i} = \mathcal{T}_{|K_i}$. Consider the matrix $M = M_\kappa(\mathcal{T}, \mathcal{S})$.

Case 1: We assume that $\mathcal{T} \cap \mathcal{S}$ is empty. All entries of M must be at most three. Furthermore, each row and column must add up to a value in $[7,8,9]$. If a row adds to less than 7 then for some K_i the corresponding tree does not match with any other tree. This implies that every entry in M is at least one. First we note that if every row and column sum to exactly seven, they are covered by Long's proof of the multiset case, as this means that for each i the matches are distinct. Suppose the sum of a row or column in M is equal to nine. Then some tree corresponds to a row or column of the form $\{3,3,3\}$. As we are assuming that every tree has a match for all K_i. This is only possible if each of the seven leaves are in a cluster, a contradiction.

Now suppose T_1 corresponds to a row of the form $\{3,3,2\}$. Again we see that strictly through a counting argument T_1 has two distinct clusters giving the specific structure of two clusters attached by an edge and the seventh leaf attached along this edge. This additional leaf and some leaf from a cluster must match with S_3. We relabel the leaves so that $\kappa(T_1,S_1)=\{1,2,3\}$, $\kappa(T_1,S_2)=\{4,5,6\}$, and $\kappa(T_1,S_3)=\{6,7\}$. As previously mentioned, the structure of T_1 must have two clusters and v_7 must lie on the path between their roots. Both S_1 and S_2 have the same structure except S_1 has a permuted $\{1,2,3\}$ cluster and S_2 a permuted $\{4,5,6\}$ cluster. From (ii) of Lemma 5.4.4 we see that v_6 and v_7 are at most distance three in both T_1 and S_3. The tree S_3 can have either a $\{4,5,7\}$ cluster or v_6 and v_7 are a cherry. It must also have an identical $\{1,2,3\}$ cluster to T_1. Note that S_3 must match with some element of $\mathcal{T}$ three times giving a cluster permutation. We assume without loss of generality that $|\kappa(T_2,S_3)|=3$. The second row and column must sum to at least 7 which gives that $|\kappa(T_2,S_1)|>0$ and $|\kappa(T_2,S_2)|>0$. As they are isomorphic by the cluster map, T_2, like S_3, does not have a $\{4,5,6\}$ cluster and is unable to match with S_1 or S_2 on K_1, K_2, or K_3. However, by the same cluster map T_2 must also have a $\{1,2,3\}$ cluster. Therefore, there exist $i,j \in \{4,5,6,7\}$ such that $T_{2|K_i}=S_{1|K_i}$ and $T_{2|K_j}=S_{2|K_j}$, implying that T_2, S_1, and S_2 all have an identical $\{1,2,3\}$ cluster, a contradiction.

Case 2: Next we consider the situation arising from the intersection of $\mathcal{T}$ and $\mathcal{S}$ containing a single tree. We construct the matrix M_κ, assuming that $T_1=S_1$.

$$\begin{pmatrix} 7 & * & * \\ * & a & b \\ * & c & d \end{pmatrix}$$

Suppose at least one of a,b,c, or d are equal to 3 and without loss of generality assume it is a. Then there is a cluster isomorphism from T_2 to S_2 and we label this cluster A in T_2 and A' in S_2 to show that they are permuted. For the second row and column to sum to at least 7, both T_2 and and S_2 must match with $T_1=S_1$ for some leaf not in A. Then for some i,j not in A we have that $T_{1|K_i}=T_{2|K_i}$ has cluster A and $T_{1|K_j}=S_{2|K_j}$ has cluster A', a contradiction.

Therefore $a = b = c = d = 2$ and we must have that M_κ is:

$$\begin{pmatrix} 7 & 3 & 3 \\ 3 & 2 & 2 \\ 3 & 2 & 2 \end{pmatrix}$$

This implies a composition of cluster isomorphisms from $T_2 \to T_1 \to S_2$. By Lemma 5.4.4 they cannot match two times.

Case 3: The final case occurs when $\mathcal{T} \cap \mathcal{S}$ has size two. This means M_κ has two sevens, which we can assume are along the diagonal.

$$\begin{pmatrix} 7 & * & * \\ * & 7 & * \\ * & * & a \end{pmatrix}$$

Again a must be greater than 0. If $a = 1$ then we have:

$$M_\kappa = \begin{pmatrix} 7 & * & 3 \\ * & 7 & 3 \\ 3 & 3 & 1 \end{pmatrix}$$

This gives a composition of cluster isomorphisms from $T_3 \to S_1 = T_1 \to S_3$ again implying that $\kappa(T_3, S_3) \in \{0, 3, 7\}$, a contradiction.

If $a = 2$ we get, up to row and column swaps, one of two possible matrices for M_κ.

$$\begin{pmatrix} 7 & * & 3 \\ * & 7 & * \\ 3 & * & 2 \end{pmatrix} \qquad \begin{pmatrix} 7 & * & 3 \\ * & 7 & * \\ * & 3 & 2 \end{pmatrix}$$

The first fails, again, by the composition of cluster isomorphisms between T_3 and S_3. In the second, to avoid such an isomorphism we can complete the matrix further.

$$\begin{pmatrix} 7 & * & 3 \\ * & 7 & 2 \\ 2 & 3 & 2 \end{pmatrix}$$

The cluster isomorphism from $T_1 = S_1 \to S_3$ implies S_1 has a cluster A and S_3 a permutation of this, A'. Notice that T_3 and S_3 do not match on any of the leaves in A. Then for some $i \in A$, T_3 must match with $T_2 = S_2$ and both must have a cluster containing i. If T_3 has the cluster A' then T_2 has a permutation of A' and cannot match with S_3 on a vertex not in A. This is a contradiction. We can assume that T_3 and T_2 have a cluster containing i but not both of the other vertices in A. Also, as $\kappa(T_3, S_2)$ and $\kappa(T_3, S_3)$ are disjoint, and we let $j \in \kappa(T_3, S_3)$. The graphs $T_{3|K_j}$ and $S_{3|K_j}$ each have a cluster containing i that are otherwise disjoint. Therefore $T_{3|K_j} \neq S_{3|K_j}$, a contradiction to $j \in \kappa(T_3, S_3)$.

If $a = 3$, again T_3 has a cluster A and S_3 a permutation of A, A'. Both match with $T_1 = T_2$ for some i and j, respectively and $i, j \notin A$. This implies $T_{1|K_i}$ has the cluster A and $T_{1|K_j}$ has the cluster A', a contradiction. □

As with the multiset case as r increases the forced number of matches in the entries of M_κ decrease to the point that some trees might never match with each other. A case based technique, such as this, could be feasible for a few additional values of $D(r)$ and $\tilde{D}(r)$, but it will be necessary to innovate new techniques to determine these values in general.

CHAPTER

6 TROPICAL MEDIAN SUPERTREES

6.1 Introduction

In this chapter, we develop a method for amalgamating sets of small trees on overlapping taxa into larger *supertrees*. By their nature, phylogenies are doomed to be incomplete. Their formation becomes exceedingly complex as the number of species grows. Due to this, they are often formed by focusing on a particular gene found in some subset of species. When coupled with the fact that different genes might evolve along different trees and at different rates it becomes quite difficult to infer a full phylogenetic picture. Instead, biologists often end up with sets of phylogenies that have overlapping taxa, and potentially different branching structures and edge lengths. *Supertree methods* are used to combine these incomplete phylogenies into a larger model of evolutionary relationships that exist [BE02]. Given a set of phylogenies $\mathcal{A} = \{T_1, \dots, T_r\}$, potentially these input trees do not conflict with each other and there are natural choices for the supertree; we say such trees are *compatible*. If the trees in $\mathcal{A}$ are compatible and only one supertree fits with the data then we say $\mathcal{A}$ is *definitive*. Of course, there is no guarantee of compatibility, and if we consider phylogenies with weighted edges it is not expected that any larger weighted tree will perfectly match with the data. Since the trees constructed by many phylogeny algorithms usually come with this additional weight information, it is natural to try to develop methods for comparing trees and computing supertrees that include this information. In Section 1.7 we showed that weighted phylogenies induce unique tree metrics. These metrics can be used to give each tree a coordinate in a continuous geometric space, called a *tree space* and compared using a distance metric from tropical geometry. The work in this chapter uses the combinatorial and geometric properties of weighted phylogenies and tree space to develop a supertree method.

6.2 Background

We recall that we use $\mathcal{T}(N)$ to denote the set of all $[N]$-trees and $\mathcal{T}^N$ to denote the set of all trees with positive edges weights (or equivalently tree metrics) on N. The additional parameter of edge weights will be used to map weighted phylogenies to a continuous geometric space called a *tree space*.

The purpose of supertree methods is the inference of large phylogenetic trees on many taxa from data points that are themselves phylogenies on subsets of these taxa. A useful supertree method should give repeatable results and also work even when the input trees have incompatible topologies. Many of the existing methods for inferring phylogenies such as maximum parsimony and maximum likelihood can be adapted to infer supertrees, although some give statistically inconsistent or misleading results [Ste08]. As an alternative to traditional methods, geometric-based supertree methods have been developed allowing for robust statistical analysis. Such methods rely on mapping weighted phylogenies to coordinates in a metric space which allow for statistical analysis.

The first fully developed tree space was described by Billera, Holmes, and Vogtman [Bil01] and is referred to as *BHV space*. Recall from Definition 1.5.4 that every edge of an X-tree defines a split which in turn partitions X into two parts. Splits defined by exterior edges are called *thin* and interior edges are called *thick*. We observe that in an X-tree, the number of splits equals the number of edges, and is therefore not exhaustive of all possible partitions of the leaves. In fact, if two trees have identical thick splits then they have the same underlying topology, and vice versa. Notice that any two splits $A|B$ and $C|D$ that appear on a single tree have the property that one of the following subsets is empty:

$$A \cap C \qquad A \cap D \qquad B \cap C \qquad B \cap D \tag{6.1}$$

Definition 6.2.1. Let X be a set and $A|B$ and $C|D$ be two partitions or splits of X. We say these splits are *compatible* if one of the intersections (6.1) is empty. For any collection of pairwise compatible splits, there exists an X-tree with exactly those splits.

To describe BHV space, we consider the maximal orthants of the space first. We fix a leaf set X and take S to be a set of compatible thick splits of maximum size. There is a unique X-tree T that achieves this set of splits and as we have assumed it to be maximum, $|S| = |X| - 3$. We order the splits lexicographically, each corresponding to a coordinate, and allow these coordinates to have values in $[0, \infty)$. These values represent the possible weights of the edges that induce the split. The value zero is included to allow for non-binary phylogenies along the boundary of the orthants to be present. This orthant is given as $\mathcal{O}_{BHV}(T)$ and is isometric to $\mathbb{R}^{|S|}_{\geq 0}$. If T_1 and T_2 are in $B(X)$ and have thick splits S_1 and S_2, respectively, then let $P = S_1 \cap S_2$. We can then glue the orthants $\mathcal{O}_{BHV}(T_1)$ and

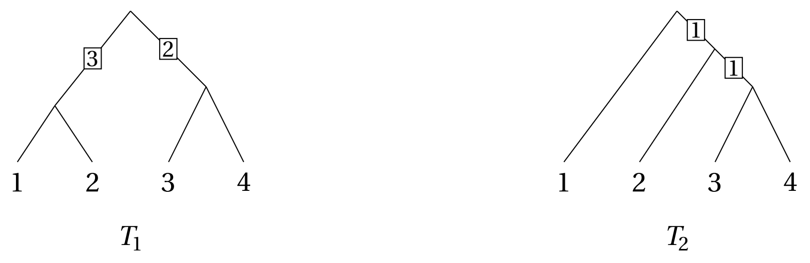


Figure 6.1 The two trees with internal branch lengths, that appear in the BHV tree space in Figure 6.2

$\mathcal{O}_{BHV}(T_2)$ along their boundary, which is isometric to $\mathbb{R}^{|P|}_{\geq 0}$. Gluing all maximal orthants results in the BHV space on the leaves X, given by BHV_X. The origin in this space is called the *cone point*, $\mathbf{0}$, and corresponds to the star shaped tree with a single internal vertex and $|X|$ leaves. The cone point is contained in all orthants of BHV_X implying there is a path between any two trees, and allowing us to give this space a distance function. Within each orthant the distance between two trees can be calculated using the standard Euclidean metric. For two trees in different orthants, T_1, T_2, we define the distance between them, denoted $d_{BHV}(T_1, T_2)$, to be the infimum of all piecewise smooth paths between them.

In this description of the original BHV space, we notice that thin splits are omitted. Their addition amounts to taking the product of BHV space with an n-dimensional orthant, and therefore had little effect on the geometry of the space. Using this expanded space, a supertree method was developed by Grindstaff and Owen [Gri18], building off of previous work on this concept By Bi et al. [Ren17]. We use BHV'_X to denote the *full BHV space*, meaning the space described above with thin splits included. In BHV'_X the cone point corresponds to the graph consisting of a single point with all leaves associated to it.

As noted previously, there is a map from rooted phylogenies on $[N]$ to unrooted phylogenies on $X \cup \{0\}$, where the leaf labeled 0 is attached to the root. If preferred, we may use this to define BHV space in terms of rooted phylogenies.

Example 6.2.2. We consider two rooted binary trees in $BHV_{[4]}$, shown in Figure 6.1. For the purposes of visualization in this example, we use traditional BHV space, rather than the full BHV space. Hence, each tree has two internal edges, and lives in a two-dimensional orthant. T_1 has the coordinates $(2,3)$ in the orthant corresponding to the splits 12|034, and 012|34, respectively. T_2 has coordinates $(1,1)$ in the orthant corresponding to 01|234 and 012|34. In the Grindstaff and Owens adaptation these would have six-dimensions.

In Figure 6.2 we see the two orthants, the ray that is their intersection, and the geodesic between them. Both trees lie in different orthants and share a ray corresponding to their common split. This ray, given by the vertical axis, corresponds to a non-binary topology. The geodesic between the two trees moves along one orthant to a point on the ray and then along the second orthant.

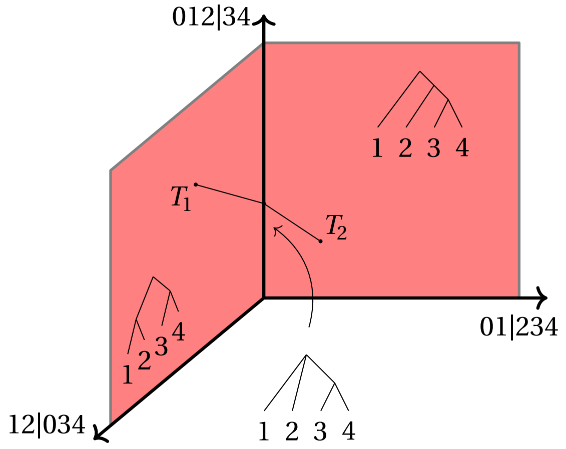


Figure 6.2 The two points $(0,2,3)$ and $(1,1,0)$ in BHV tree space and the geodesic connecting them.

In this three-dimensional space, giving T_1 the coordinates $(0,2,3)$ and T_1 the coordinates $(1,1,0)$ then the point p on the ray has coordinates $(0,5/3,0)$. The distance between T_1 and T_2 is then the Euclidean distance between T_1 and p summed with the Euclidean distance between T_2 and p.

A key aspect of the method demonstrated in [Gri18] for computing supertrees is the ability to lift a tree in $T \in \mathcal{T}^S$ to a set of trees in $\mathcal{T}^N$ that share the underlying structure of T. We begin by first describing the natural map from $\mathcal{T}^N$ to $\mathcal{T}^S$, $S \subseteq [N]$.

Definition 6.2.3. Let $T \in \mathcal{T}^X$ with weight function $w : E(T) \to \mathbb{R}^+$ and $S \subseteq X$. The *induced weighted subtree* of T induced by S is the tree $T_{|S}$ with the unique weight function $w : T_{|S} \to \mathbb{R}^+$ satisfying $d_T(i,j) = d_{T_{|S}}(i,j)$ for all $i, j \in S$.

When we compute $T_{|S}$ and a degree two vertex v is suppressed, the edge that replaces this vertex has weight equal to the sum of the weights of the two edges incident to v. In terms of the metric that T induces on $[N]$, the restriction of this metric to S is exactly the metric induced by $T_{|S}$. In [Zai16], Zairis et al. defined the *tree dimensionality map*, which maps the set of trees in $\mathcal{T}^N$ to a smaller dimension tree space on a subset of $[N]$.

Definition 6.2.4. Let $S \subseteq [N]$. The *restriction of* $\mathcal{T}^N$ *to* $\mathcal{T}^S$ is the map $\Psi_S : \mathcal{T}^N \to \mathcal{T}^S$, where $\Psi_S(T) = T_{|S}$ for all $T \in \mathcal{T}^N$.

We can then take the preimage of a tree with respect to the restriction map to find all trees in the larger space that share its structure.

Definition 6.2.5. Let $S \subseteq [N]$ and $T \in \mathcal{T}^S$. We define the *lift* of T to $\mathcal{T}^N$ given by the map $\Psi_S^{-1} : \mathcal{T}^S \to \mathcal{T}^N$, where $\Psi_S^{-1}(T)$ is the set of trees in $\mathcal{T}^N$ that have T as a subtree induced by S.

Using this lift Grindstaff and Owen [Gri18] showed that when the intersection over all lifts of a set of input trees was non-empty then it could be computed efficiently. Furthermore, they allowed for inputs that did not have compatible topologies. This method minimizes an objective function that could be either the sum of the distances to the lifts of the inputs over all trees in BHV'_N or the sum of the squares of these distances.

6.3 Tropical Tree Space

BHV space provided an innovative method for parameterizing tree space in a geometrically meaningful way, but it is not without its drawbacks. Lin et al. [Lin17] showed that given as few as three points in BHV space, the geodesic triangle formed by these points can have arbitrarily large dimension. This leads to further complications including the absence of useful projections onto BHV planes, and the possibility of *stickiness*, when the Fréchet mean remains fixed despite small changes in the data points. Also, the best known algorithm for calculating geodesics in BHV space [Owe10] runs in quartic-time on the number of leaves. With this in mind, we approach our problems in tropical tree space, also called palm tree space in [Mon18]. This space has a natural connection to the space of phylogenies and does not produce many of the hindrances of BHV space.

In Section 1.7 we mentioned that a tree metric $T \in \mathcal{T}^N$ can be given as a vector in $\mathbb{R}^n$, where $n = \binom{N}{2}$, or as an $N \times N$ matrix. This gives a natural coordinatization of each tree metric on $[N]$. The first relationship between the space of tree metrics on $[N]$ and tropical geometry was given in [Spe05]. Sturmfels and Speyer showed a homeomorphism between $\mathcal{T}^N$ and a tropical version of the Grassmanian. This follows from the four point condition, which characterizes tree metrics, being equivalent to the *Plücker relations* that define the Grassmanian. In this coordinatization, the space of rooted N-trees is tropically geometric. This is particularly evident in the *tropical projective torus*.

Definition 6.3.1. The *tropical projective torus* $\mathbb{R}^n/\mathbb{R}\mathbf{1}$, where $\mathbf{1} = (1, \dots, 1)$, is the quotient space given by the set of equivalence classes under

$$(x_1, \dots, x_n) \sim (y_1, \dots, y_n) \Longleftrightarrow x_i - y_i = x_j - y_j \text{ for all } 1 \le i, j \le n.$$

If $x = (x_1, ..., x_n) \in \mathbb{R}^n$ then we use $\overline{x}$ to denote the representative vector in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$ for the equivalence class containing x. We may think of $\overline{x}$ as the vector $(x_1 - x_1, x_2 - x_1, \dots, x_n - x_1)$. This maps the tropical projective torus onto $\mathbb{R}^{n-1}$. In terms of $\mathcal{T}^{[N]}$ we see that modding out by scalars is the same as disregarding external edge weights that differ by a constant. The main benefit of working in the projective space is that we can endow the tropical projective torus with a metric.

Definition 6.3.2. Let $x, y \in \mathbb{R}^n$ and $\overline{x}, \overline{y}$ be their representatives in the tropical projective torus. The *tropical distance* between $\overline{x}, \overline{y}$ is given by

$$d_{\text{tr}}(\overline{x}, \overline{y}) := \max_{1 \le i < j \le n} |(x_i - y_i) - (x_j - y_j)| = \max_{1 \le i \le n}(x_i - y_i) + \max_{1 \le i \le n}(y_i - x_i).$$

The function d_{tr} is called the *tropical metric*.

The tropical distance function is not a metric on $\mathbb{R}^n$ although it is a well-defined metric space on $\mathbb{R}^n/\mathbb{R}\mathbf{1}$. However, for ease of notation for $x, y \in \mathbb{R}^n$ we take $d_{\text{tr}}(x, y) = d_{\text{tr}}(\overline{x}, \overline{y})$.

Proposition 6.3.3 ([Mac15])**.** *The space $\mathcal{T}^N$ is the union of $(2N-5)!!$ polyhedral cones in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$, each with dimension $2N-3$ in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$. Each cone corresponds to a unique binary tree topology.*

A binary tree on N leaves has $2N-3$ edges and therefore the corresponding maximal cone has a dimension of $N-3$ in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$. Points along the boundary of these cones correspond to some edge weights being equal to 0, meaning these are non-binary, weighted $[N]$-trees.

In this chapter we focus on a subset of $\mathcal{T}^N$, the space of equidistant trees or ultrametrics denoted $\mathcal{U}^N$. Recall that $T \in \mathcal{T}^N$ is an ultrametric if all leaves are equidistant from a designated root vertex, or equivalently if as a metric T satisfies the three-point condition given in Definition 1.7.4. We restrict to the space of ultrametrics mainly due to some of the geometric benefits they admit, however in terms of phylogenetics one might consider the weights of edges as representing clock time. If so, we would expect nearest common ancestors to be equidistant from current species. A useful property of the space of equidistant phylogenetic trees, $\mathcal{U}^N$, is that it is tropically convex in the tropical projective torus.

Theorem 6.3.4. *The space of ultrametrics, $\mathcal{U}^N$, is tropically convex in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$.*

Proof. Let $U, W \in \mathcal{U}^N$. We show that $a \odot U \boxplus b \odot W$ is in $\mathcal{U}^N$. In the tropical projective torus $\overline{a \odot U} = \overline{U}$ so it suffices to show that $U \boxplus W$ is in $\mathcal{U}^N$. For arbitrary but distinct $i, j, k \in [N]$ we check the three point condition, that $\max(U(i,j) \boxplus W(i,j), U(i,k) \boxplus W(i,k), U(j,k) \boxplus W(j,k))$ occurs at least twice. By definition, for U (and also for W) $\max(U_{i,j}, U_{i,k}, U_{j,k})$ occurs twice. Without loss of generality assume U has a maximum in these three coordinates that is as least as large as W's. Then this value is achieved in at least two coordinates after tropical addition, and the third coordinate cannot be any larger. □

Because every ultrametric U satisfies the three-point condition for every $i, j, k \in \binom{[N]}{3}$ this implies that it must also satisfy at least one of the following sets of equations:

$$U(i,j) \le U(i,k) = U(j,k) \tag{6.2}$$

$$U(i,k) \le U(i,j) = U(j,k) \tag{6.3}$$

$$U(j,k) \le U(i,j) = U(i,k). \tag{6.4}$$

In fact, the equidistant property allows us to determine which of these equations are satisfied based entirely on the underlying topology. For an ultrametric $U \in \mathcal{U}^N$ with $i, j \in [N]$ and $i \neq j$, we use $i \vee j$ to denote the nearest common ancestor of i and j with respect to U.

Proposition 6.3.5. *Let $U \in \mathcal{U}^N$. For all $i, j, k \in \binom{[N]}{3}$ at least one of the following must hold: $i \vee j = i \vee k$,$i \vee j = j \vee k$,$i \vee k = j \vee k$. If $i \vee j = i \vee k$ then U satisfies $U(j,k) \leq U(i,j) = U(i,k)$.*

Proof. Let $U \in \mathcal{U}^N$ and $i, j, k \in \binom{[N]}{3}$. First, one of the equalities must be satisfied due to the unique path from a leaf to the root. Suppose i and j have a nearest common ancestor that is at least as far from the root as the nearest ancestor of i and k. Then the path from j to i is the same as the path from k to i after $i \vee j$. Now, without loss of generality suppose $i \vee j = i \vee k$ and U does not satisfy $U(j,k) \leq U(i,j) = U(i,k)$ to obtain a contradiction. It is clear by assumption that the $U(i,j) = U(i,k)$. Clearly, $j \vee k$ cannot be closer to the root than $i \vee j$. If it were then $i \vee j$ would be the nearest common ancestor of j and k. Hence $U(j,k) \leq U(i,j) = U(i,k)$. □

Notice that if U corresponds to a rooted binary topology then for all i, j, k one of $i \vee j = i \vee k$, $i \vee j = j \vee k$, and $i \vee k = j \vee k$. If it satisfies $i \vee j = i \vee k$ then it satisfies the strict inequality $U(j,k) < U(i,j) = U(i,k)$. On the other hand, if U is non-binary then for some i, j, k we have $U(j,k) = U(i,j) = U(i,k)$. We can use this fact to give ultrametric space as the union of polyhedral cones.

Definition 6.3.6. Let τ be a rooted binary topology with leaves labeled by $[N]$. The *cone corresponding to τ* in $\mathcal{U}^N$ is defined as:

$$\mathcal{C}_\tau := \{U \in \mathcal{U}^N : \forall \{i,j,k\} \subseteq [N] \text{ if } i \vee j =_\tau i \vee k \text{ then } U(j,k) \leq U(i,k) = U(j,k)\}.$$

Proposition 6.3.7. *The space of ultrametrics, $\mathcal{U}^N$, is the union of $(2N-3)!!$ polyhedral cones, each corresponding to a rooted binary tree topology τ and having dimension $N-2$ in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$.*

Proof. As there are $(2N-3)!!$ rooted binary tree topologies in $\mathcal{U}^N$. Clearly ever $U \in \mathcal{U}^N$ lies some $\mathcal{C}_\tau$ as we can witness the metric U on a binary tree if we allowed edge weights equal to 0. If such a binary tree had topology τ then U must be in $\mathcal{C}_\tau$. In $\mathbb{R}^n$ all ultrametrics in a fixed cone are determined by the weights of their internal edges and the weight of a single external edge. As a binary tree on N leaves has $2N-2$ edges and $N-2$ internal edges this gives that $\mathcal{C}_\tau$ ahs Hence the has dimension $N-1$ in $\mathbb{R}^n$ and $N-2$ in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$. □

Example 6.3.8. We use the three trees given in Figure 6.3 to demonstrate some of the properties discussed in this section. In $\mathbb{R}^3$ these ultrametrics have the coordinates given by:

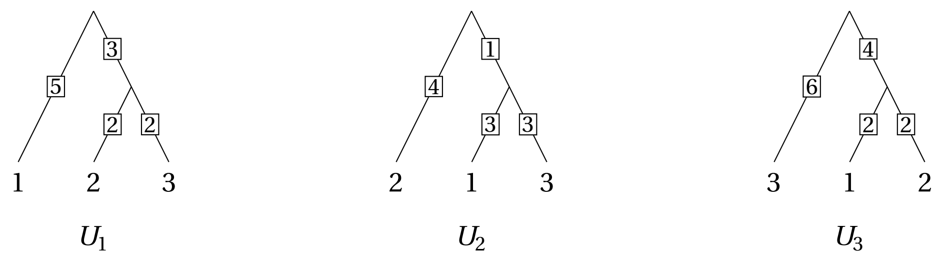
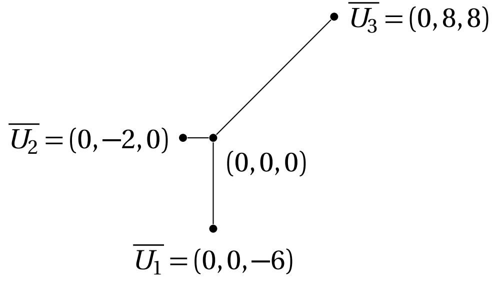


Figure 6.3 The ultrametrics $U_1, U_2, U_3 \in \mathcal{U}^3$.



$\overline{U_3} = (0,8,8)$
$\overline{U_2} = (0,-2,0)$
$(0,0,0)$
$\overline{U_1} = (0,0,-6)$

Figure 6.4

$$\begin{aligned}
U_1 &= (U_1(1,2), U_1(1,3), U_1(2,3)) = (10,10,4) \\
U_2 &= (8,6,8) \\
U_3 &= (4,12,12)
\end{aligned}$$

Alternatively, we may also think of them as the upper triangular 3×3 matrices given by:

$$U_1 = \begin{pmatrix} 0 & 10 & 10 \\ & 0 & 4 \\ & & 0 \end{pmatrix} \qquad U_2 = \begin{pmatrix} 0 & 8 & 6 \\ & 0 & 8 \\ & & 0 \end{pmatrix} \qquad U_3 = \begin{pmatrix} 0 & 4 & 12 \\ & 0 & 12 \\ & & 0 \end{pmatrix}$$

In $\mathbb{R}^3/\mathbb{R}\mathbf{1}$ we get representatives: $\overline{U_1} = (0,0,-6)$, $\overline{U_2} = (0,-2,0)$, $\overline{U_3} = (0,8,8)$. In Figure 6.4 we plot these representatives on a plane and include the tropical line segments between each pair. Each U_i lies in a distinct cone, one of each of the three possible rooted binary trees on three leaves. Three leaves is also the minimum size leaf set that has distinct cones, hence their only intersection is the cone point at $(0,0,0)$ and the tropical line segment between points in different cones must pass through this point. This is not the case for trees on more leaves. We also see that $d_{\text{tr}}(\overline{U_1}, \overline{U_2}) = \max(0,2,-6) + \max(0,-2,6) = 8$.

6.4 Tropical Median Supertrees

In this section we define a set of supertrees that uses the relationship between equidistant weighted trees and tropical geometry. Given a set of ultrametrics, $\mathcal{A} = \{U_1, ..., U_k\}$, where U_i has leaf set L_i and $\bigcup_{i=1}^k L_i = [N]$ the general goal is to find a subset of $\mathcal{U}^N$ that captures the information in $\mathcal{A}$. To achieve this we leverage the tropical metric as a quantitative measure of the closeness of a supertree to the input trees. The intuition is that we can consider the lifts of each U_i given by $\Psi^{-1}(U_i)$ and if we are able to find a subset of $\mathcal{U}^N$ that minimizes the sum of the distances to these lifts then we get a set of ultrametrics that is geometrically close to our input ultrametrics. To do this we need to define the tropical distance between sets.

Definition 6.4.1. Let $X, Y \subset \mathbb{R}^n$ and $\overline{X}, \overline{Y}$ be the collection of representatives in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$. The *tropical distance* between two sets in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$ is defined as

$$d_{\mathrm{tr}}(\overline{X}, \overline{Y}) := \min_{x \in X, y \in Y} d_{\mathrm{tr}}(x, y).$$

The tropical metric allows us to define classic statistical functions in the quotient space $\mathbb{R}^n/\mathbb{R}\mathbf{1}$. The main such function we use for analysis are *Fermat-Weber points* (see [Lin18] for an in-depth treatment).

Definition 6.4.2. For points $v_1, v_2, \ldots, v_k \in \mathbb{R}^n/\mathbb{R}\mathbf{1}$, the set of their tropical *Fermat-Weber points* is

$$\operatorname*{argmin}_{u \in \mathbb{R}^n/\mathbb{R}\mathbf{1}} \sum_{i=1}^k d_{\mathrm{tr}}(u, v_i).$$

The minimum distance is the value

$$\min_{u \in \mathbb{R}^n/\mathbb{R}\mathbf{1}} \sum_{i=1}^k d_{\mathrm{tr}}(u, v_i)$$

We generalize this definition to sets of ultrametrics that have differing leaf labels.

Definition 6.4.3. Let $\mathcal{A} = \{U_1, \ldots, U_k\}$, $U_i \in \mathcal{U}^{L_i}$, and $\bigcup_{i=1}^{k} L_i = [N]$. Then the set of *tropical median supertrees* of $\mathcal{A}$ is

$$\tilde{\mathcal{A}}_{tr} := \left\{ U \in \mathcal{U}^N : \sum_{i=1}^k d_{\mathrm{tr}}(U, \Psi^{-1}(U_i)) \text{ is minimum} \right\}.$$

The *minimum median distance* is the value

$$\mathcal{A}_{\min} := \min_{U \in \mathcal{U}^N} \sum_{i=1}^k d_{\mathrm{tr}}(U, \Psi^{-1}(U_i)).$$

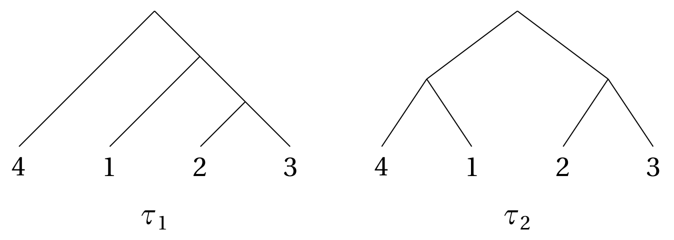


Figure 6.5 Some binary tree topologies.

Note the analogy to the ordinary median in $\mathbb{R}$ where the median of a set of numbers $\{a_1,\dots,a_k\}$ is

$$\underset{x\in\mathbb{R}}{\operatorname{argmin}}\sum_{i=1}^{k}|x-a_i|.$$

In the case where the leaf sets of all ultrametrics are the same the tropical median supertrees and the tropical Fermat-Weber points are the same. Calculating the set of tropical median supertrees will be the main focus for the rest of this chapter.

6.5 Calculating Tropical Median Supertrees

Now that we have defined what a set of tropical median supertrees is we continue with a method for computing them. As we will show, the tropical projective space admits useful geometric properties that will allow us to describe the tropical median supertree as the *Minkowski sum* of a bounded polytope and a ray. We will make use of linear programming techniques as well as the polytope solver polymake [Ass17] for visualizations and polytope descriptions.

Our method for calculating the tropical median will require us to describe the problem as a linear program. Hence, we need to express the search area ($\mathcal{U}^N$), and the lifts ($\Psi^{-1}(U_i)$) via linear equalities and also express the tropical distance function ($d_{\text{tr}}(x,y)$) as an optimization problem.

Similar to the work of Grindstaff and Owen in BHV space [Gri18] we approach the parameterization of $\mathcal{U}^N$ using a divide and conquer strategy. Despite being tropically convex in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$, $\mathcal{U}^N$ does not have a description as linear inequalities in $\mathbb{R}^n$. However, for $\tau\in RB([N])$, the cone $\mathcal{C}_\tau$ does have such a description. Definition 6.3.6 shows that from each binary τ we can derive a set of equations that all ultrametrics in $\mathcal{C}_\tau$ must satisfy.

Example 6.5.1. Consider the rooted binary tree topologies labeled by [4], τ_1 and τ_2 given in Figure 6.5. For each $i,j,k\in[4]$ we get a set of inequalities that any U in the cone corresponding to each topology must satisfy.

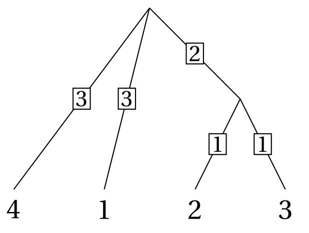


Figure 6.6 The ultrametric $(6,6,6,2,6,6)$ as a tree.

i,j,k	τ_1	τ_2
$1,2,3$	$U(2,3) \leq U(1,2) = U(1,3)$	$U(2,3) \leq U(1,2) = U(1,3)$
$1,2,4$	$U(1,2) \leq U(1,4) = U(2,4)$	$U(1,4) \leq U(1,2) = U(2,4)$
$1,3,4$	$U(1,3) \leq U(1,4) = U(3,4)$	$U(1,4) \leq U(1,3) = U(3,4)$
$2,3,4$	$U(2,3) \leq U(2,4) = U(3,4)$	$U(2,3) \leq U(2,4) = U(3,4)$

Of course, these lists are exhaustive and in practice they are not all necessary. For instance, the relationships for τ_1 can be reduced down to

$$U(1,2) = U(1,3), U(1,4) = U(2,4) = U(3,4), \text{ and } U(2,3) \leq U(1,2) \leq U(1,4).$$

Notice that the ultrametric $U = (6,6,6,2,6,6)$ maps to the tree in Figure 6.6 and its corresponding ultrametric satisfies all of the above inequalities. Hence it lies on the boundary of the cones implied by τ_1 and τ_2.

We turn our focus to the preimage of an ultrametric in the $\mathbb{R}^n$, specifically $\Psi^{-1}(U)$. Much the same way as the parameterization of $\mathcal{U}^N$ we make use of the convexity of cones. We consider the lift of an ultrametric U and restrict to a fixed rooted binary topology τ.

Definition 6.5.2. For an ultrametric U in $\mathcal{U}^S$, $S \subseteq [N]$, and a rooted binary topology τ on $[N]$ we define

$$\Psi_\tau^{-1}(U) := \Psi^{-1}(U) \cap \mathcal{C}_\tau.$$

Many topologies will not be compatible with U, so there will be certain τ where $\Psi_\tau^{-1}(U) = \emptyset$. In Proposition 6.5.3 we show that this lift to a fixed topology is classically convex in the ambient space $\mathbb{R}^n$.

Proposition 6.5.3. *Given $U \in \mathcal{U}^S$, $S \subseteq [N]$ and a rooted binary tree topology τ the set $\Psi_\tau^{-1}(U) \subseteq \mathcal{U}^N$ is convex in $\mathbb{R}^n$.*

Proof. Suppose $U \in \mathcal{U}^S$, $S \subseteq [N]$, and let $U_1, U_2 \in \Psi_\tau^{-1}(U)$. For all t so that $0 \leq t \leq 1$ we consider $U' = U_1(1-t) + U_2 t$. Because U_1 and U_2 have the same topology they satisfy the inequalities for $\mathcal{C}_\tau$.

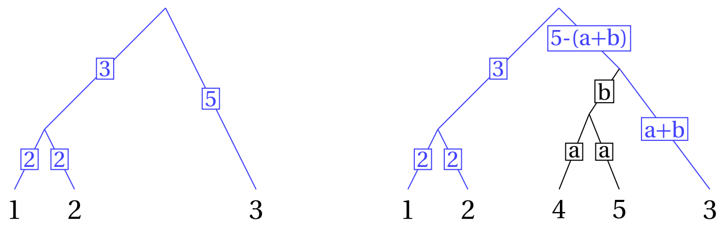


Figure 6.7 An ultrametric U and a parameterization of $\Psi_\tau^{-1}(U)$.

Convexity of $\mathcal{C}_\tau$ follows from the inequalities that define the cone, and therefore U' is also in $\mathcal{C}_\tau$. It remains to show that $U'_{|S} = U$. Given $i, j \in S$, $U_1(i,j) = U(i,j) = U_2(i,j)$, which implies that $U'(i,j)$ is also equivalent to $U(i,j)$. Hence U is an induced subtree of U'. $\square$

Example 6.5.4. We consider an ultrametric tree $U = [4,10,10]$ in $\mathcal{U}^{[3]}$ and a rooted topology τ labeled by [5] as illustrated in Figure 6.7. Every topology that has the tree on the left as an induced subtree can be formed by choosing an edge to add the leaf labeled 4, then repeating this process to add a leaf labeled 5. In this example we do not increase the height of the tree. For $U' \in \Psi^{-1}(U)$, we notice the following relationships:

- $U'(3,4) = U'(3,5) = 2(a+b)$.
- $U'(i,5) = U'(i,4) = U'(i,3)$ for $i \in [2]$.
- $0 \le a+b \le 5$, $a \ge 0$, $b \ge 0$.

Hence, $\Psi_\tau^{-1}(U)$ is exactly the ultrametrics of the form:

$$\begin{pmatrix} 0 & 4 & 10 & \mathbf{10} & \mathbf{10} \\ & 0 & 10 & \mathbf{10} & \mathbf{10} \\ & & 0 & \mathbf{2(a+b)} & \mathbf{2(a+b)} \\ & & & 0 & \mathbf{2a} \\ & & & & 0 \end{pmatrix}$$

satisfying $a, b \ge 0$ and also $a + b \le 5$.

Example 6.5.5. We again consider an ultrametric tree $U = [4,10,10]$ in $\mathcal{U}^{[3]}$ and a fixed topology τ labeled from [5], as seen in Figure 6.8. Similar to Example 6.5.4, if we think of U as having an infinite edge out of its root, denoted in the picture on the left by a dotted line, then we see that the topology τ of the tree on the right can be achieved by adding the leaf labeled 4 to this dotted edge, and then adding the leaf labeled 5 in the same way. Here we observe:

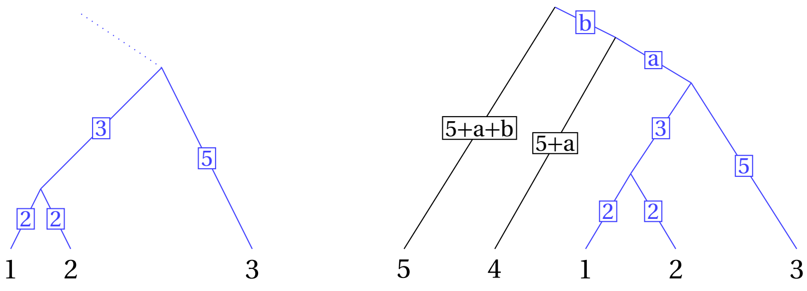


Figure 6.8 An ultrametric U and a parameterization of $\Psi_\tau^{-1}(U)$, where the height of the tree increases.

- $U'(i,4) = U'(j,4) = 2(5+a)$ for $i, j \in [3]$.
- $U'(i,5) = U'(j,5) = 2(5+a+b)$ for $i, j \in [4]$.
- $a \geq 0$, $b \geq 0$.

Here we have $\Psi_\tau^{-1}(U)$ is exactly the ultrametrics of the form:

$$\begin{pmatrix} 4 & 10 & \mathbf{2(5+a)} & \mathbf{2(5+a+b)} \\ & 10 & \mathbf{2(5+a)} & \mathbf{2(5+a+b)} \\ & & \mathbf{2(5+a)} & \mathbf{2(5+a+b)} \\ & & & \mathbf{2(5+a+b)} \\ & & & \end{pmatrix}$$

satisfying $a, b \geq 0$.

Theorem 6.5.6. *If $U \in \mathcal{U}^S$ and $S \subseteq [N]$ then $\Psi^{-1}(U)$ is tropically convex in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$.*

Proof. We suppose that $X, Y \in \Psi^{-1}(U)$ in $\mathbb{R}^n$. Convexity in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$ is equivalent to showing for all $a, b \in \mathbb{R}$ there exists $Z \in \Psi^{-1}(U)$, $c \in \mathbb{R}$, so that $c \odot Z = a \odot X \boxplus b \odot Y$. In other words any tropically linear combination is a scalar away from a supertree of U. Because X and Y both project onto U, for all $i, j \in S$ we have that $X(i,j) = Y(i,j)$. Let $c = \max\{a, b\}$ and $Z = (a-c) \odot X \boxplus (b-c) \odot Y$. $\mathcal{U}^N$ is tropically convex and hence $a \odot X \boxplus b \odot Y$ is in $\mathcal{U}^N$, as is Z. By our choice of c, for all $i, j \in S$, $Z(i,j) = X(i,j) = Y(i,j)$. Hence $Z_{|S} = U$ and $Z \in \Psi^{-1}(U)$. □

Corollary 6.5.7. *If $U \in \mathcal{U}^S$, $S \subseteq [N]$, and τ is a rooted binary topology labeled by $[N]$ then $\Psi_\tau^{-1}(U)$ is tropically convex in $\mathbb{R}^n/\mathbb{R}\mathbf{1}$.*

Proof. If $X, Y \in \Psi_\tau^{-1}(U)$ in $\mathbb{R}^n$, then for all i, j, k, they both satisfy the same set of inequalities defined by τ. Since the tropical operations preserve this property $\overline{X} \boxplus \overline{Y}$ has the same topology. Theorem 6.5.6 gives us that $\overline{X} \boxplus \overline{Y} \in \Psi^{-1}(U)$. □

The tropical convexity of $\Psi^{-1}(U)$ allows us to perform computations in the tropical projective torus, using the tropical operations, with the guarantee that our results may be translated back to a supertree of U in $\mathbb{R}^n$.

As mentioned in Example 6.5.4 some topologies in $\Psi^{-1}(U)$ may have larger heights than the original $\{U_i\}$. Example 6.5.5 demonstrates this. We make the distinction between these two types of topologies because those that do not increase the height give a bounded convex polytope, while those that do increase the height are unbounded.

For algorithm optimization, there are situations when we can restrict the unbounded polytopes to bounded versions, which we will discuss more in Section 6.6.

Up to this point we are able to define the space of ultrametrics as well as the space of lifts via linear inequalities. In order to define the set of tropical median supertrees as the solution to a linear programming problem it remains to show that the tropical distance can be defined as an optimization problem. We recall that one definition of the tropical distance is given as

$$d_{\mathrm{tr}}(\overline{U_1}, \overline{U_2}) = \max_{1\le i<j\le N}(U_1(i,j) - U_2(i,j) + \max_{1\le i<j\le N}(U_2(i,j) - U_1(i,j)),$$

and claim that this can be given as an optimization problem.

Proposition 6.5.8. *Let $x, y \in \mathbb{R}^n$ and suppose $\phi(x,y)$ is the minimum value of the following optimization problem:*

$$\begin{aligned} \textit{minimize} \quad & z_1 + z_2 \\ \textit{subject to} \quad & z_1, z_2 \in \mathbb{R} \\ & z_1 \ge x_i - y_i, \quad 1 \le i \le n \\ & z_2 \ge y_i - x_i, \quad 1 \le i \le n. \end{aligned}$$

Then $\phi(x,y) = d_{tr}(\overline{x}, \overline{y})$.

Proof. Follows directly from the definition of tropical distance. □

We now have all the individual pieces involved in calculating the set of tropical median supertrees can be given in terms of linear inequalities and optimization problems. The definition below can be used to calculate the set for a fixed collection of topologies.

Definition 6.5.9. Let $\mathcal{A} = \{U_1, U_2, \ldots, U_m\}$, with $U_i \in \mathcal{U}^{L_i}$ and $L_i \subseteq [N]$ for $1 \le i \le m$. For $B =$

$(\tau_0, \tau_1, \ldots, \tau_m)$, with $\tau_i \in RB([N])$ we consider the following optimization problem:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i=1}^{m} z_{i,1} + z_{i,2} \\
\text{subject to} \quad & U \in \mathcal{C}_{\tau_0} \\
& W_i \in \Psi_{\tau_i}^{-1}(U_i), \quad 1 \le i \le m \\
& z_{i,1} \ge U(j,k) - W_i(j,k), \quad 1 \le i \le m, \quad 1 \le j < k \le N \\
& z_{i,2} \ge W_i(j,k) - U(j,k), \quad 1 \le i \le m, \quad 1 \le j < k \le N.
\end{aligned}$$

We let $f(\tau_0, \ldots, \tau_m)$ be the minimum value and $F(\tau_0, \ldots, \tau_m)$ be the set of $U \in \mathcal{C}_{\tau_0}$ that achieve this minimum value.

Theorem 6.5.10. *Let $\mathcal{A} = \{U_1, U_2, \ldots, U_m\}$, with $U_i \in U^{L_i}$ for $L_i \subseteq [N]$, $1 \le i \le m$. Let $\mathcal{B}$ be the set of all possible lists of the form $(\tau_0, \tau_1, \ldots, \tau_m)$, where each τ_i is a rooted binary topology labeled from $[N]$. Given $f(B)$ and $F(B)$ from Definition 6.5.9 then*

$$\mathcal{A}_{\min} = \min_{B \in \mathcal{B}} f(B)$$

and

$$\tilde{\mathcal{A}}_{tr} = \{F(B) : B \in \mathcal{B} \text{ and } f(B) = \mathcal{A}_{\min}\}.$$

Proof. The optimization problem given in Definition 6.5.9 finds the minimum sum of distances for a fixed list of topologies, given by $f(B)$. Minimizing these sums over all possible lists of topologies, $\mathcal{B}$, is then the minimum possible sum, which by Definition 6.4.3 is equal to $\mathcal{A}_{\min}$. It is clear that $\{F(B) : B \in \mathcal{B} \text{ and } f(B) = \mathcal{A}_{\min}\} \subseteq \tilde{\mathcal{A}}_{tr}$. By assumption, $\mathcal{B}$ is exhaustive, hence if $U \in \tilde{\mathcal{A}}_{tr}$ then for some $B \in \mathcal{B}$, $U \in F(B)$. Therefore, $\tilde{\mathcal{A}}_{tr} = \{F(B) : B \in \mathcal{B} \text{ and } f(B) = \mathcal{A}_{\min}\}$. □

Theorem 6.5.10 gives a rigorous method for calculating the set of tropical median supertrees. However, the number of linear programs that must be run is extremely large. When running it naively, over all of $\mathcal{B}$ it requires solving and storing results from $((2N-3)!!)^{k+1}$ linear programs. To improve on this we can first consider which $B \in \mathcal{B}$ are feasible.

Proposition 6.5.11. *Let U be an ultrametric labeled from $L \subseteq [N]$. Then $\Psi^{-1}(U)$ lies in the union of at most $(2N-3)!!/(2|L|-3)!!$ maximal cones.*

Proof. We assume U is binary to maximize the calculation. Note that when we add a single leaf to a tree with $|L|$ leaves there are $(2|L|-1)$ places where we could add that leaf corresponding in order to get a new binary topology. These correspond to the $2|L|-2$ edges in the topology corresponding to U, plus the option of adding a leaf above the root. The result then follows by induction. □

Ultimately, this means that if each U_i has leaf set $L_i \subseteq [N]$ then the number of linear programs that need to be checked are at most

$$(2N-3)!!\prod_{i=1}^{r}(2N-3)!!/(2|S_i|-3)!!.$$

The term $(2N-3)!!$ comes from considering all cones that could be the location of the median, while the other terms correspond to the cones that contain the various lifts. In practice, this is only feasible when N is small and the $|L_i|$ is close to N for all i.

As we show in Example 6.5.12, a straightforward method for calculating both $\mathcal{A}_{\min}$ and $\tilde{\mathcal{A}}_{tr}$ involves iterating through the pertinent rooted binary tree topologies, solving a linear program for each and storing the value and inequalities when a minimum distance is achieved. Each set of inequalities is the H-representation of a polytope in $\mathbb{R}^{|\mathcal{A}|(n+2)+n}$. We then consider the plane, Q, given by:

$$\sum_{i=1}^{|\mathcal{A}|} z_{i,1} + z_{i,2} = \mathcal{A}_{\min}.$$

We then take each of the above polytopes, intersect them with Q, and project onto the n coordinates corresponding to U in the linear program. This gives a polytope of solutions and taking the union of all of these results in $\tilde{\mathcal{A}}_{tr}$. We can analyze these polytopes individually using tools such as polymake [Ass17] to give improved descriptions of the space. In particular, while our solutions are unbounded due to the nature of tropical scaling in $\mathbb{R}^n$, these polytopes can be described as an affine subspace of the $(n+1)$ dimensional space. Here they are the Minkowski sum of a bounded polytope and the ray from the origin that passes through $\mathbf{1}$. When it is deemed useful, we are then able to search our solution set for ultrametrics with certain heights and/or topologies.

Example 6.5.12. Consider the ultrametrics, $\mathcal{A} = \{U_1, U_2, U_3\}$ given by the matrices below.

$$U_1 = \begin{pmatrix} 0 & 18 & 34 & 100 & * \\ & 0 & 34 & 100 & * \\ & & 0 & 100 & * \\ & & & 0 & * \\ & & & & 0 \end{pmatrix} U_2 = \begin{pmatrix} 0 & 64 & 64 & 100 & * \\ & 0 & 30 & 100 & * \\ & & 0 & 100 & * \\ & & & 0 & * \\ & & & & 0 \end{pmatrix} U_3 = \begin{pmatrix} 0 & 100 & * & 86 & 56 \\ & 0 & * & 100 & 100 \\ & & 0 & * & * \\ & & & 0 & 86 \\ & & & & 0 \end{pmatrix}$$

The ultrametrics U_1 and U_2 have leaf sets $\{1,2,3,4\}$, while U_3 has leaf set $\{1,2\}$. For each of the 105 topologies in $\mathcal{U}^5$ and each of the 7 possible lifts of U_1, U_2, and U_3 we run a linear program to find the minimum distance. We find that $\mathcal{A}_{\min} = 100$ and this solution is achieved by 45 different linear programs out of the 36,015 we ran. We input these into polymake and find that 2 unbounded polytopes capture the tropical median. These polytopes can be given as the Minkowski sum of the

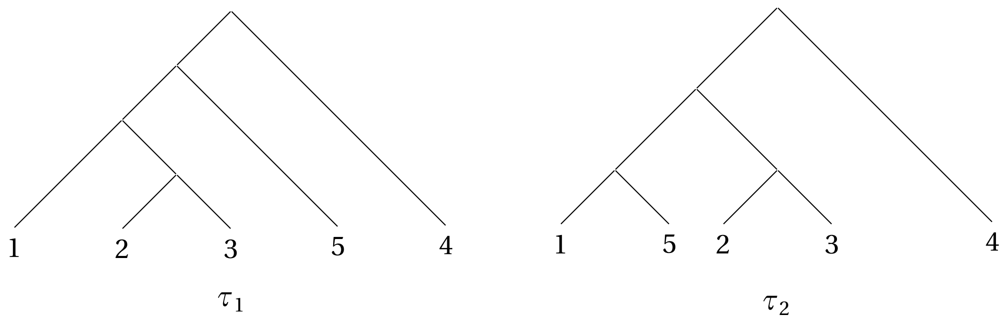


Figure 6.9 The two topologies that define the maximal cones that intersect the solution set.

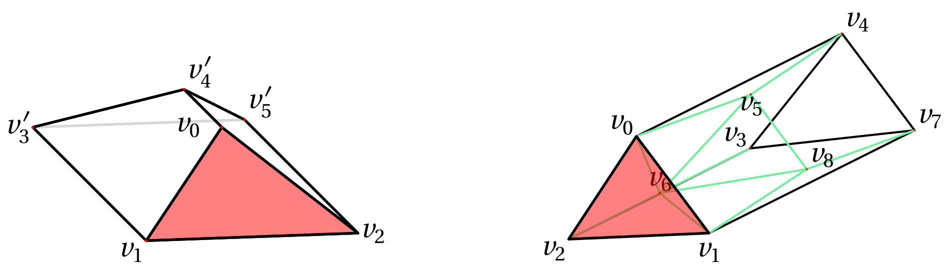


Figure 6.10 The polytope P_1 (Left), and a Schlegel diagram of P_2 (Right).

ray that passes through **1** and the bounded polytopes P_1 and P_2, with each row vector given below representing a vertex.

$$P_1 = \begin{pmatrix} 34 & 34 & 66 & 34 & 0 & 66 & 34 & 66 & 34 & 66 \\ 34 & 34 & 70 & 34 & 0 & 70 & 34 & 70 & 34 & 70 \\ 30 & 30 & 66 & 30 & 0 & 66 & 30 & 66 & 30 & 66 \\ 34 & 34 & 70 & 40 & 0 & 70 & 40 & 70 & 40 & 70 \\ 34 & 34 & 66 & 36 & 0 & 66 & 36 & 66 & 36 & 66 \\ 30 & 30 & 66 & 36 & 0 & 66 & 36 & 66 & 36 & 66 \end{pmatrix} P_2 = \begin{pmatrix} 34 & 34 & 66 & 34 & 0 & 66 & 34 & 66 & 34 & 66 \\ 34 & 34 & 70 & 34 & 0 & 70 & 34 & 70 & 34 & 70 \\ 30 & 30 & 66 & 30 & 0 & 66 & 30 & 66 & 30 & 66 \\ 30 & 30 & 66 & 0 & 0 & 66 & 30 & 66 & 30 & 66 \\ 34 & 34 & 66 & 0 & 0 & 66 & 34 & 66 & 34 & 66 \\ 44 & 44 & 76 & 0 & 10 & 76 & 44 & 76 & 44 & 76 \\ 44 & 44 & 80 & 0 & 14 & 80 & 44 & 80 & 44 & 80 \\ 34 & 34 & 70 & 0 & 0 & 70 & 34 & 70 & 34 & 70 \\ 44 & 44 & 80 & 0 & 10 & 80 & 44 & 80 & 44 & 80 \end{pmatrix}$$

From this we are able to give the tropical median explicitly as,

$$\tilde{\mathcal{A}}_{tr} = \{a \odot U : U \in P_1 \cup P_2, a \in \mathbb{R}_{\geq 0}\}.$$

We find that $\tilde{\mathcal{A}}_{tr} \subset (\mathcal{C}_{\tau_1} \cup \mathcal{C}_{\tau_2})$ for the two binary topologies shown in Figure 6.9. Note that P_1 is a subset of $\mathcal{C}_{\tau_1}$, while P_2 lies in both $\mathcal{C}_{\tau_1}$ and $\mathcal{C}_{\tau_2}$. In this case, P_1 has dimension 3, while P_2 has dimension 4. Figure 6.10 shows the structure of P_1 and a Schlegel diagram of P_2. The facet of P_1 in red defined by v_0, v_1, and v_2 denotes their intersection.

6.6 Special Case: Two Input Ultrametrics

The linear program given in the last section allows us to theoretically calculate the tropical median in $\mathcal{U}^N$ for a collection of convex lifts of ultrametrics in subspaces of $\mathcal{U}^N$. But, as we have seen, it requires running a large number of linear programs to solve. In this section we provide methods to improve the feasibility of our method. We begin by first introducing the *span* of an ultrametric U.

Definition 6.6.1. For an ultrametric U, the *span* of U denoted $\text{span}\{U\}$, is the difference between its maximum and minimum coordinate.

Clearly, scaling an ultrametric does not change the span. Also, we may use the span to form an upper bound on the minimum distance of the tropical median points.

Proposition 6.6.2. *If* $\mathcal{A} = \{U_1, U_2, ..., U_r\}$ *then* $\mathcal{A}_{\min} \leq \sum_{i=1}^{r} \mathit{span}\{U_i\}$.

Proof. For each U_i choose a lift in $\Psi^{-1}(U_i)$ that has the same span as U_i. This can always be done by simply attaching new leaves to existing vertices. Let y denote the ultrametric in $\mathcal{U}^N$ with each coordinate equal to one. The sum of the distances from the lifts to y is given by:

$$\sum_{i=1}^{r} d_{\text{tr}}(y, \Psi^{-1}(U_i)) = \sum_{i=1}^{r} \text{span}\{U_i\}.$$

□

This upper bound on the sum of the distances can potentially be used to rule out various topologies and even give height restrictions to the lifted ultrametrics and ultrametrics in the tropical median. It turns out that such restrictions are computationally advantageous, allowing us to restrict the unbounded lifts and cones to tropical polytopes.

6.6.1 Tropical Mean Supertrees

Here we consider the special case when our input is only two trees: $\mathcal{A} = \{U_1, U_2\}$. It is clear that $\tilde{\mathcal{A}}_{tr}$ is all points in $\mathcal{U}^N$ that lie on a geodesic between $\Psi(U_1)$ and $\Psi(U_2)$. These points can be calculated using the general method above. However, when we are limited to two trees there are subsets of our solution set that can also be calculated using this method. We begin by presenting bounds of $\mathcal{A}_{\min}$ that hold in this case.

Proposition 6.6.3. *Given* $\mathcal{A} = \{U_1, U_2\}$, *let* $S' = \mathcal{L}(U_1) \cap \mathcal{L}(U_2)$. *If* $|S'| \geq 3$ *and* U_1, U_2 *are binary then* $\mathcal{A}_{\min} \geq d_{tr}(U_{1|S'}, U_{2|S'})$.

Proof. This follows from $U_{1|S'}$ and $U_{2|S'}$ being induced subtrees of all their respective lifts. □

When the leaf intersection is at least 3 it may then be advantageous to check if it is feasible to lift U_1 and U_2 while maintaining this lower bound. Our distance calculation provides bounds that each added lift must remain between.

Definition 6.6.4. For U_1, U_2 with at least 2 shared leaves, let D_{max} and D_{min} denote the maximum and minimum value in $U_1 - U_2$.

If U_1, U_2 are binary and share at least 3 leaves, then $D_{max} > D_{min}$. If they share only 2 leaves, $D_{max} = D_{min}$.

If i is a leaf in both U_1 and U_2 and j is a leaf only in U_1, then to maintain the distance from 6.6.3, j must be attached to U_2, forming U_2' so that $D_{min} \leq U_1(i,j) - U_2'(i,j) \leq D_{max}$. If we consider the path in U_2 from i to the root and potentially past it onto the phantom edge, the points on this path that are $D_{min}/2$ and $D_{max}/2$ from i gives an initial boundary for the placement of i on U_2. Repeating this for all leaves only in U_1 and taking the intersection of the bounded areas gives the viable placements for i. The same can be done for leaves only in U_2. If any viable placement areas are empty, then the lower bound is not feasible. If none are empty all that is left is check if the coordinates that are new in both lifts can maintain the minimum distance.

Another benefit in the case of two trees is that we are able to bound the height of our feasible lifts, allowing us to express the lifts in a useful way.

Theorem 6.6.5. *Let $U_1 \in \mathcal{U}^X$ and $U_2 \in \mathcal{U}^Y$ and suppose $X \cup Y = [N]$ and $|X \cap Y| \geq 2$. Let $h_1 = \max(U_1)$ and $h_2 = \max(U_2)$. Any lift, $U_2' \in \Psi^{-1}(U_2)$ that lies on a minimum length geodesic between $\Psi^{-1}(U_1)$ and $\Psi^{-1}(U_2)$ must satisfy:*

$$\max(U_2')_{\leq} \max(h_2, 2(span\{U_1\} + span\{U_2\} - D_{max} + h_1)$$

Proof. Proposition 6.6.2 shows that

$$d_{\text{tr}}(\Psi^{-1}(U_1), \Psi^{-1}(U_2)) \leq 2(\text{span}\{U_1\} + \text{span}\{U_2\}).$$

Therefore, suppose $i, j \in X \cap Y$ such that $U_1(i,j) - U_2(i,j) = D_{max}$. Let k be a leaf of U_1 that is not a leaf of U_2. Then any lift U_2' must satisfy

$$U_2'(i,k) \leq 2(\text{span}\{U_1\} + \text{span}\{U_2\}) - D_{max} + U_1(i,k)$$

This allows us to give an upper bound for every coordinate in the lifts of U_1 and U_2, call these h_1 and h_2, respectively. □

In the two tree case a slightly different parameter that might be of interest is the set of midpoints of the geodesics between the two lifts. These points are more similar to a mean, as they minimize the sum of the squares of their distance from each lift. These can be calculated in the same manner.

We first run our lift over all topologies to determine $\mathcal{A}_{\min}$. Then, in our polytope we restrict each pair of slack variables to equal $\frac{\mathcal{A}_{\min}}{2}$.

Example 6.6.6. We consider the set of two ultrametrics $\mathcal{A} = \{U_1, U_2\}$, where U_1, U_2 are defined by the matrices:

$$U_1 = \begin{pmatrix} 100 & 78 & 78 & * \\ & 100 & 100 & * \\ & & 50 & * \\ & & & * \\ & & & \end{pmatrix} U_2 = \begin{pmatrix} 26 & * & 100 & 26 \\ & * & 100 & 24 \\ & & * & * \\ & & & 100 \\ & & & \end{pmatrix}$$

U_1 has leaf set $\{1,2,3,4\}$ and U_2 has leaf set $\{1,2,3,5\}$. We lift U_1 and U_2 to $\mathcal{U}^5$ and using a linear program solver determine that $d_{\text{tr}}(\Psi^{-1}(U_1), \Psi^{-1}(U_2)) = 96$. Using polymake we can then represent all ultrametrics in $\mathcal{U}^5$ that lie on a geodesic between these lifts as the Minkowski sum of the ray $\mathbf{1}$ and the union of the polytopes given below.

$$P_1 = \begin{pmatrix} 76 & 76 & 76 & 76 & 76 & 76 & 0 & 26 & 76 & 76 \\ 28 & 28 & 28 & 28 & 28 & 28 & 0 & 0 & 28 & 28 \\ 76 & 76 & 76 & 76 & 76 & 76 & 0 & 48 & 76 & 76 \\ 50 & 50 & 50 & 50 & 50 & 50 & 0 & 0 & 50 & 50 \\ 50 & 50 & 50 & 50 & 50 & 50 & 48 & 0 & 50 & 50 \\ 28 & 28 & 28 & 28 & 28 & 28 & 26 & 0 & 28 & 28 \\ 50 & 100 & 100 & 50 & 100 & 100 & 24 & 0 & 100 & 100 \\ 26 & 100 & 100 & 26 & 100 & 100 & 24 & 0 & 100 & 100 \\ 50 & 100 & 100 & 50 & 100 & 100 & 48 & 0 & 100 & 100 \\ 2 & 28 & 28 & 2 & 28 & 28 & 0 & 0 & 28 & 28 \\ 2 & 76 & 76 & 2 & 76 & 76 & 0 & 48 & 76 & 76 \\ 50 & 76 & 76 & 50 & 76 & 76 & 0 & 0 & 76 & 76 \\ 2 & 76 & 76 & 2 & 76 & 76 & 0 & 0 & 76 & 76 \end{pmatrix} P_2 = \begin{pmatrix} 76 & 76 & 76 & 76 & 76 & 76 & 0 & 26 & 76 & 76 \\ 28 & 28 & 28 & 28 & 28 & 28 & 0 & 0 & 28 & 28 \\ 76 & 76 & 76 & 76 & 76 & 76 & 0 & 48 & 76 & 76 \\ 50 & 50 & 50 & 50 & 50 & 50 & 0 & 0 & 50 & 50 \\ 50 & 50 & 50 & 50 & 50 & 50 & 48 & 0 & 50 & 50 \\ 28 & 28 & 28 & 28 & 28 & 28 & 26 & 0 & 28 & 28 \\ 50 & 28 & 28 & 50 & 50 & 50 & 48 & 0 & 50 & 50 \\ 50 & 28 & 28 & 50 & 50 & 50 & 0 & 0 & 50 & 50 \\ 98 & 76 & 76 & 98 & 98 & 98 & 0 & 48 & 98 & 98 \end{pmatrix}$$

The intersection of P_1 and P_2 is given by the vertices corresponding to the first six row vectors of each. Both P_1 and P_2 are four dimensional and their intersection is a 3-dimensional facet of both. Figure 6.13 gives the Schlegel diagram through that facet. Hence we can define the tropical median of $\mathcal{A}$ to be:

$$\tilde{\mathcal{A}}_{tr} = \{a \odot U : U \in P_1 \cup P_2, a \in \mathbb{R}^+\}$$

All ultrametrics in the tropical median are in $\mathcal{C}_{\tau_1} \cup \mathcal{C}_{\tau_2}$ for the topologies shown in Figure 6.14 with the intersection lying on the boundary between the two cones.

Additionally, in this case we can calculate the polytope of midpoints. These are the ultrametrics in $\tilde{\mathcal{A}}_{tr}$ that are equidistant to each lift. Again we give this as a Minkowski sum of a polytope and the

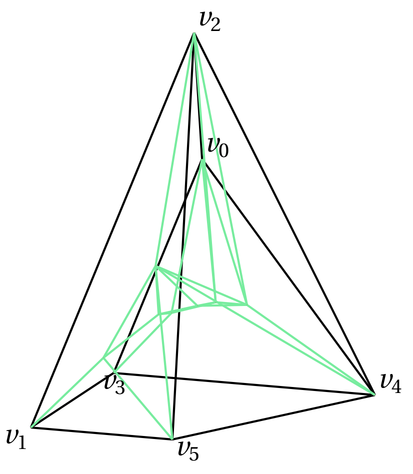


Figure 6.11 P_1

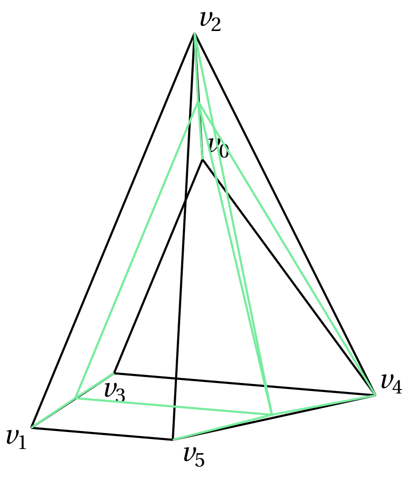
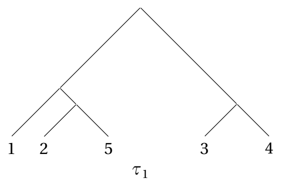


Figure 6.12 P_2

Figure 6.13 Schlegel diagram of P_1 and P_2 viewed through the facet that is their intersection and is defined by the vertices: $v_0, v_1, v_2, v_3, v_4, v_5$.



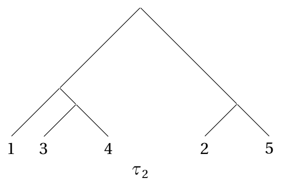


Figure 6.14 The two topologies that define the maximal cones that intersect the solution set.

ray **1**.

$$P_3 = \begin{pmatrix} 2 & 28 & 28 & 2 & 28 & 28 & 0 & 0 & 28 & 28 \\ 50 & 76 & 76 & 50 & 76 & 76 & 0 & 0 & 76 & 76 \\ 50 & 76 & 76 & 50 & 76 & 76 & 48 & 0 & 76 & 76 \\ 50 & 76 & 76 & 50 & 76 & 76 & 0 & 48 & 76 & 76 \end{pmatrix}$$

The polytope of midpoints lies in $\mathcal{C}_{\tau_1}$, for the topology τ_1 listed above and only intersects P_1.

As with the examples above, the presentation of the solution set allows us to restrict to a maximum height, and in general we can also restrict to a fixed topology if multiple topologies are present. The technique for finding the midpoint of the geodesics has the additional benefit that it can be extended to other points on the geodesic. For instance, given two input trees one may want to weight a certain tree more than the other. This only requires setting the slack variables equal to the appropriate percentage of the geodesic distance. This amounts to changing two inequalities in the H-description of the polytope.

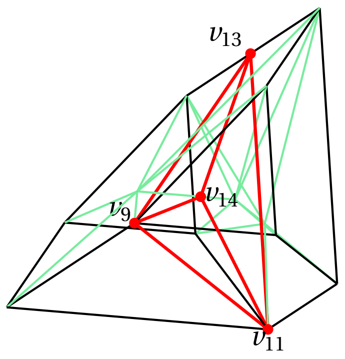


Figure 6.15 Schlegel diagram of P_1 through a different facet than the diagram above with the edges of the tropical mean in blue.

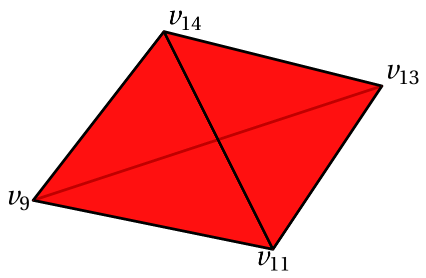


Figure 6.16 A three-dimensional rendering of the polytope outlined in the figure to the left. This simplex is the set of midpoints of geodesics between U_1 and U_2.

BIBLIOGRAPHY

[Arn87] Arnborg, S., Corneil, D. G., and Proskurowski, A. “Complexity of finding embeddings in ak-tree”. *SIAM Journal on Algebraic Discrete Methods* **8**.2 (1987), pp. 277–284.

[Aro09] Arora, S. and Barak, B. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[Ass17] Assarf, B. et al. “Computing convex hulls and counting integer points with `polymake`”. *Math. Program. Comput.* **9**.1 (2017), pp. 1–38.

[Bil01] Billera, L. J., Holmes, S. P., and Vogtmann, K. “Geometry of the space of phylogenetic trees”. *Advances in Applied Mathematics* **27**.4 (2001), pp. 733–767.

[BE02] Bininda-Emonds, O. R., Gittleman, J. L., and Steel, M. A. “The (super) tree of life: procedures, problems, and prospects”. *Annual Review of Ecology and Systematics* **33**.1 (2002), pp. 265–289.

[Chu10] Chung, F. “Graph theory in the information age”. *Notices of the AMS* **57**.6 (2010), pp. 726–732.

[Coo71] Cook, S. A. “The complexity of theorem-proving procedures”. *Proceedings of the third annual ACM symposium on Theory of computing*. 1971, pp. 151–158.

[Die05] Diestel, R. “Graph theory 3rd ed”. *Graduate texts in mathematics* **173** (2005).

[Dvo13a] Dvořák, Z., Král, D., and Thomas, R. “Testing first-order properties for subclasses of sparse graphs”. *Journal of the ACM (JACM)* **60**.5 (2013), pp. 1–24.

[Dvo13b] Dvořák, Z. “Constant-factor approximation of the domination number in sparse graphs”. *European Journal of Combinatorics* **34**.5 (2013), pp. 833–840.

[Erd66] Erdős, P. and Hajnal, A. “On chromatic number of graphs and set-systems”. *Acta Mathematica Academiae Scientiarum Hungarica* **17**.1-2 (1966), pp. 61–99.

[Eve11] Even, S. *Graph algorithms*. Cambridge University Press, 2011.

[Fel04] Felsenstein, J. and Felenstein, J. *Inferring phylogenies*. Vol. 2. Sinauer associates Sunderland, MA, 2004.

[Gol08] Golovach, P. A. and Villanger, Y. “Parameterized complexity for domination problems on degenerate graphs”. *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 2008, pp. 195–205.

[Gri18] Grindstaff, G. and Owen, M. “Geometric comparison of phylogenetic trees with different leaf sets”. *arXiv preprint arXiv:1807.04235* (2018).

[Gro15] Grohe, M. et al. “Colouring and covering nowhere dense graphs”. *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 2015, pp. 325–338.

[Hum08] Humphries, P. “Combinatorial aspects of leaf-labelled trees”. PhD thesis. University of Canterbury, 2008.

[Kar72] Karp, R. M. “Reducibility among Combinatorial Problems”. *Complexity of Computer Computations*. Boston, MA: Springer US, 1972, pp. 85–103.

[Kie03] Kierstead, H. A. and Yang, D. “Orderings on graphs and game coloring number”. *Order* **20**.3 (2003), pp. 255–264.

[Lin18] Lin, B. and Yoshida, R. “Tropical Fermat–Weber Points”. *SIAM Journal on Discrete Mathematics* **32**.2 (2018), 1229–1245.

[Lin17] Lin, B. et al. “Convexity in tree spaces”. *SIAM Journal on Discrete Mathematics* **31**.3 (2017), pp. 2015–2038.

[Lon16] Long, C. “Algebraic Geometry of Phylogenetic Models”. PhD thesis. North Carolina State, 2016.

[Mac15] Maclagan, D. and Sturmfels, B. *Introduction to tropical geometry*. Vol. 161. American Mathematical Soc., 2015.

[Mat08] Matsen, F. A., Mossel, E., and Steel, M. “Mixed-up trees: the structure of phylogenetic mixtures”. *Bulletin of Mathematical Biology* **70**.4 (2008), pp. 1115–1139.

[Mat83] Matula, D. W. and Beck, L. L. “Smallest-last ordering and clustering and graph coloring algorithms”. *Journal of the ACM (JACM)* **30**.3 (1983), pp. 417–427.

[Mon18] Monod, A. et al. *Tropical Geometry of Phylogenetic Tree Space: A Statistical Perspective*. 2018. arXiv: `1805.12400 [math.MG]`.

[Neš08] Nešetřil, J. and Mendez, P. Ossona de. “Grad and classes with bounded expansion II. Algorithmic aspects”. *European Journal of Combinatorics* **29** (2008), pp. 777–791.

[Neš12] Nešetřil, J. and Mendez, P. Ossona de. *Sparsity: graphs, structures, and algorithms*. Vol. 28. Springer Science & Business Media, 2012.

[Owe10] Owen, M. and Provan, J. S. “A fast algorithm for computing geodesic distances in tree space”. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **8**.1 (2010), pp. 2–13.

[Ren17] Ren, Y. et al. “A combinatorial method for connecting BHV spaces representing different numbers of taxa”. *arXiv preprint arXiv:1708.02626* (2017).

[Spe05] Speyer, D. and Williams, L. “The tropical totally positive Grassmannian”. *Journal of Algebraic Combinatorics* **22**.2 (2005), pp. 189–210.

[Ste16] Steel, M. *Phylogeny: discrete and random processes in evolution*. SIAM, 2016.

[Ste08] Steel, M. and Rodrigo, A. “Maximum likelihood supertrees”. *Systematic biology* **57**.2 (2008), pp. 243–250.

[Sul11] Sullivant, S. “The Disentangling Number For Phylogenetic Trees” (2011).

[Tov84] Tovey, C. A. “A simplified NP-complete satisfiability problem”. *Discrete Applied Mathematics* **8**.1 (1984), pp. 85–89.

[Zai16] Zairis, S. et al. “Genomic data analysis in tree spaces”. *arXiv preprint arXiv:1607.07503* (2016).

[Zhu09] Zhu, X. “Colouring graphs with bounded generalized colouring number”. *Discrete Mathematics* **309**.18 (2009), pp. 5562–5568.

[Zie12] Ziegler, G. M. *Lectures on polytopes*. Vol. 152. Springer Science & Business Media, 2012.