ON THE COMPUTATIONAL COMPLEXITY OF INFERRING EVOLUTIONARY TREES

HAROLD TODD WAREHAM

ON THE COMPUTATIONAL COMPLEXITY OF INFERRING EVOLUTIONARY TREES

BY

© Harold Todd Wareham

A thesis submitted to the School of Graduate
Studies in partial fulfillment of the
requirements for the degree of
Master of Science

Department of Computer Science
Memorial University of Newfoundland
December 1992

St. John's Newfoundland

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

ISBN 0-315-82642-8

Abstract

The process of reconstructing evolutionary trees can be viewed formally as an optimization problem. Recently, decision problems associated with the most commonly used approaches to reconstructing such trees have been shown to be NP-complete [Day87, DJS86, DS86, DS87, GF82, Kri88, KM86]. In this thesis, a framework is established that incorporates all such problems studied to date. Within this framework, the NP-completeness results for decision problems are extended by applying theorems from [CT91, Gas86, GKR92, JVV86, KST89, Kre88, Sel91] to derive bounds on the computational complexity of several functions associated with each of these problems, namely

- *evaluation functions*, which return the cost of the optimal tree(s),
- *solution functions*, which return an optimal tree,
- *spanning functions*, which return the number of optimal trees,
- *enumeration functions*, which systematically enumerate all optimal trees, and
- *random-selection functions*, which return a randomly-selected member of the set of optimal trees.

Where applicable, bounds are also presented for the versions of these functions that are restricted to trees of a given cost or of cost less than or greater than a given limit. Based in part on these results and theorems from [BH90, GJ79, KMB81, Kre88], bounds are derived on how closely polynomial-time algorithms can approximate optimal trees. In particular, it is shown using the recent results of [ALMSS92] that no phylogenetic inference optimal-cost solution problem examined in this thesis has a polynomial-time approximation scheme unless P = NP.

Acknowledgements

The research reported in this thesis and the studies leading to it have taken place over several years. Over that time, a person will accumulate many debts to others; this is particularly true of graduate students. My apologies to those whose help I have forgotten to or did not mention. Their various kindnesses must be acknowledged by the fact that I am, at last, writing these acknowledgements.

I would like to thank the School of Graduate Studies, the Alumni Association of Memorial University of Newfoundland, William Day, and Wlodek Zuberek for the financial assistance under which I carried out the first two years of my studies. For the past two years, I have been very fortunate to work part-time for Richard Greatbatch and Brad de Young of the Physical Oceanography group at the MUN Department of Physics. I would like to thank both them and the staff of the Physical Oceanography group for providing the money, the environment, and the patience that have allowed me to complete and write up my research.

I would like to thank the Dean of Science, the Dean of Student Affairs and Services, the Canadian Institute for Advanced Research, William Day, and Paul Gillard for the generous travel funding I have received over the last four years. The conferences they have enabled me to attend will benefit me always.

I would like to thank the staff of the Queen Elizabeth II Library for much friendly assistance over the years. I owe a similar debt to the staff and faculty of the Department of Computer Science, particularly Elaine Boone, Jennifer Button, Sharon Hynes, Patricia Murphy, Mike Rayment, and Nolan White. Paul Fardy, David Fifield, and Allan Goulding helped with typesetting this thesis in LaTeX. I know that I am only partially aware of all that has been done for me by Jane Foltz and Paul Gillard, who have served as Heads of my Department over the course of my studies; this small thanks must for now suffice.

I would like to thank all of the people who sent me reprints and manuscripts; in particular, I would like to thank Mark Krentel, Alan Selman, Madhu Sudan, and Seinosuke Toda, whose manuscripts are the basis for many results in this thesis. I would also like to thank the people who have clarified my thinking and my results via e-mail and personal conversations, notably Joe Felsenstein, Bill Gasarch, Arun Jagota, Jim Kadin, Johannes Köbler, Mark Krentel, Manrique Mata-Montero, Gary Olsen, Alan Selman, and Jacobo Torán. In particular, I would like to thank Johannes Köbler, for suggesting simplifications and completions for several results in Section 4.1.2, Mark Krentel, for the discussions that led (among other things) to the NP-hardness reduction for the FUGT[≥] problem, and Bill Gasarch, who is indirectly responsible for almost all results reported in this thesis.

I would like to thank my many friends, for putting up with my erratic grad student behavior these last few years and simultaneously making such continued behavior possible.

I would like to thank my parents for their individual and invaluable support to me over my years of study.

Finally, I would like to thank my supervisor, William Day, for his many years of patience, sound advice, and assistance.

Contents

List of Tables

List of Figures

By and large, as the mass of knowledge grows, men devote little attention to the dead. Yet it is the dead who are frequently our pathfinders, and we walk all unconsciously along the roads they have chosen for us.

Loren Eiseley, *The Man Who Saw Through Time*

Dedicated in Memory of

Loren Eiseley

(1907 - 1977)

1 Introduction

Phylogenetic systematics is the subdiscipline of evolutionary biology that deals with reconstructing the tree that represents the evolutionary relationships of a set of species. Such trees are used to create taxonomic classifications for species and to evaluate alternative hypotheses of adaptation, evolutionary mechanism, and ancient geographical relationships [EC80, FB90, Nei87, NP81]. As the data are seldom available to reconstruct the actual historical tree, one takes as an estimate of this tree a subset of the set of all possible trees that are best relative to some biologically-relevant criterion.

Many approaches to reconstructing evolutionary trees have been developed over the last thirty years [Fel88, PHS92, SO90]. These approaches are of two types [SO90, p. 412]: method-based approaches, which integrate the criterion for tree selection directly into the method for searching the set of all possible trees, and criterion-based approaches in which the criterion and search method are distinct. Method-based approaches obtain optimal trees quickly, but do not rank the suboptimal trees and the alternative hypotheses encoded by these trees. Criterion-based methods do give such rankings but are much slower because known algorithms have to evaluate all possible trees; hence, practical implementations of criterion-based approaches settle for deriving approximations to the optimal trees rather than the optimal trees themselves.

Consider the formal computational problems associated with these types of approaches. The problems associated with method-based approaches typically have polynomial-time algorithms, and are thus of little interest here. In this thesis, I will be concerned with the problems associated with criterion-based approaches. Since 1982, decision problems associated with the most commonly-used approaches in phylogenetic systematics have been shown to be NP-complete [Day87, DJS86, DS86, DS87, GF82, Kri88, KM86]. While this implies that related problems such as producing optimal trees are harder than NP, it is not known exactly how much harder these problems are, or how closely fast algorithms may approximate optimal trees. This latter problem is especially important because trees of slightly different or even the same cost can imply very different evolutionary hypotheses [Mad91, p. 315]. There are many examples in the biological literature of hypotheses that have been modified or retracted in light of different estimates of the optimal tree e.g. the "Out of Africa" hypothesis for the origin of the human mitrochondrial DNA gene pool [MRS92, SSV92].

In this thesis, I will derive bounds on the the computational complexities of several functions based on phylogenetic inference problems:

- *evaluation functions*, which return the cost of the optimal tree(s);
- *solution functions*, which return an optimal tree;
- *spanning functions*, which return the number of optimal trees;
- *enumeration functions*, which systematically enumerate all optimal trees;

and

- *random-selection functions*, which return a randomly-selected member of the set of optimal trees.

Bounds are also derived for those functions that return trees of a given cost or of cost less than or greater than a given limit. In addition, I will derive bounds on the approximability of the solution functions associated with the most commonly-used approaches to phylogenetic inference. Results are given not only for evolutionary trees based exclusively on dichotomous speciation events but also for trees incorporating such events as hybridization and recombination.

The results in this thesis have been obtained by applying existing techniques to a set of closely-related problems. These results are of significance to computational complexity theory to the extent that, by isolating aspects of problems that cause unexpected increases or decreases in complexity, they suggest further avenues for research. The biological relevance of these results is more problematic. Some biologists have argued that these results are not applicable because (1) the defined problems are too general, and problems of practical interest may be solvable in polynomial time, and (2) the framework of asymptotic worst-case analyses in which these results were derived is unrealistic. (J. S. Farris and M. F. Mickevich, personal communication). In formulating the problems examined in this thesis, there have been undeniable tradeoffs of fidelity to biological reality for the sake of tractability of analysis. However, such tradeoffs underlie many applications of mathematics to real problems, and are not only unavoidable but necessary in the initial stages of an investigation. The purpose of this thesis is not to present results of direct relevance to biologists, but to lay a theoretical framework in which such results may one day be derived.

1.1 Organization of This Thesis

This thesis is laid out in four sections.

In Section 2, I give various definitions used in this thesis, including graph-theoretic definitions of non-reticulate and reticulate evolutionary trees and an introduction to computational complexity theory.

In Section 3, I review basic concepts in phylogenetic analysis as well as all previously-defined phylogenetic inference decision problems and the reductions by which they have been shown to be NP-complete. A framework is given that incorporates all such problems studied to date. This section also includes definitions and reductions for new problems involving reticulate trees, as well as several new reductions for previously-defined problems. The tree of reductions among all problems examined in this thesis is shown in Figures 7 and 8, and the correspondence between phylogenetic inference problems examined in this thesis and those in the literature is given in Table 19.

In Section 4, I use the OptP hierarchy [GKR92, Kre88] and paddability [CT91, Gas86] to classify the phylogenetic inference evaluation problems into two groups within FP^{NP}. The complexities of these problems, along with several other properties of these problems and theorems from [JVV86, KST89, Sel91], are used to derive bounds on the complexities of the associated solution, spanning, enumeration, and random-generation functions. All bounds and hardness results derived in this section are summarized in Table 20.

In Section 5, I use results from [BH90, GJ79, KMB81, Kre88] to derive lower bounds on the approximability of phylogenetic inference problems by polynomial-time algorithms. In particular, it is shown using the recent results of Arora et al. [ALMSS92] that no phylogenetic inference optimal-cost solution problem examined in this thesis has a polynomial-time approximation scheme unless P = NP. All bounds on approximability derived in this section are summarized in Table 24.

Each of Sections 3, 4, and 5 begins with a subsection on notation particular to that section and concludes with a summary of the results derived in that section. Brief discussions of the biological relevance of these results are given at the end of each such summary.

2 Notation

This section consists of graph-theoretic definitions of evolutionary trees and an introduction to computational complexity theory. First there are some general definitions.

Define alphabet $\Sigma = \{0, 1\}$, all strings x as being members of Σ^*, and all languages L as being members of 2^{Σ^*}. Let $|x|$ be the length of string x, $|L|$ be the cardinality of L, and L^l be the set of all strings in L with length l. For a language L, co-$L = \Sigma^* - L$. Define $\langle x, y \rangle$ as an invertible function that encodes pairs of strings into a single string, and χ_L as the characteristic function of language L i.e. $\chi_L(x) = 1$ if $x \in L$ and 0 otherwise.

Let $\mathcal{N} = \{0, 1, 2, \ldots\}$ be the nonnegative integers, $\mathcal{Q}^+$ be the nonnegative rational numbers, and $\mathcal{R}^+$ be the nonnegative real numbers. Given functions $f : X \rightarrow Y$ and $g : Y \rightarrow Z$, let $dom(f)$ and $rng(f)$ be the domain and range of f, respectively, and $g \circ f : X \rightarrow Z$ be the composition of f and g i.e. $(g \circ f)(x) = g(f(x))$. If function f is not defined on input x, then $f(x) = \perp$. If $\forall x \in X\{|rng(f(x))| = 1\}$, f is *single-valued*; else, f is *multivalued*. A function $f : \mathcal{N} \rightarrow \mathcal{N}$ is *smooth* if the function $g : 1^n \rightarrow 1^{f(n)}$ is polynomial-time computable and $f(x) \leq f(y)$ for all $x \leq y$ [Kre88, p. 493]. For an arbitrary total order R on binary strings, define the *ordering* P_R as the pair of functions (E, L) such that $E(i, l)$ returns the ith member of $(\Sigma^*)^l$ under R and $L(x, y, l)$ indicates if $x \leq y$ under R for $x, y \in (\Sigma^*)^l$; this thesis will focus on those orderings for which E and

L are computable in polynomial time (see Section 4.5).

There are several types of bounds for a numerical function. These bounds can be represented by classes of functions [BDG88, p. 35]:

- $O(f)$ is the set of functions g such that for some $r > 0$ and for all but finitely many n, $g(n) < r \cdot f(n)$.
- $o(f)$ is the set of functions g such that for every $r > 0$ and for all but finitely many n, $g(n) < r \cdot f(n)$.
- $\Omega(f)$ is the set of functions g such that for some $r > 0$ and for infinitely many n, $g(n) > r \cdot f(n)$.

Classes $O(f)$, $o(f)$, and $\Omega(f)$ correspond to loose upper, strict upper, and loose lower bounds, respectively. These classes can also be defined over whole classes of functions rather than a single function e.g. $O(poly)$, $o(polylog)$, where $poly = \bigcup_k n^k = n^{O(1)}$ and $polylog = \bigcup_k \log^k n = \log^{O(1)} n$. All logarithms in this thesis will be to base 2.

2.1 Graphs, Hypergraphs, and Trees

A *graph* $G = (V, E)$ is a set V of vertices and a set E of edges such that each edge links a pair of vertices. Edges in which one vertex is designated the source and the other the target are called *arcs*; a graph composed of arcs is a *directed graph*. A *path* between vertices u and v in a graph G is a sequence of alternating vertices and edges $v_1 e_1 v_2 \ldots v_n e_n v_{n+1}$ such that e_i is an edge in G between v_i and v_{i+1}, e_i and e_{i+1} are distinct edges in G, $v_1 = u$, and $v_{n+1} = v$; directed paths are defined similarly. A graph is *connected* if there is a path between each pair of vertices in the graph; if there is an edge between each pair of vertices in the graph, the graph is *complete*. If all edges in a graph lie on a single path between a pair of vertices, the graph is *linear*. A *hypergraph* $H = (V, E)$ is a set V of vertices and a set E of hyperedges such that each hyperedge links a group instead of a pair of vertices. A hyperedge whose vertex-set has been partitioned into disjoint target and source vertex-sets is called a *hyperarc*; a hypergraph composed of hyperarcs is a *directed hypergraph*. A *hyperpath* between vertices u and v in a hypergraph H is a sequence of alternating vertices and hyperedges $v_1 e_1 v_2 \ldots v_n e_n v_{n+1}$ such that e_i is a hyperedge in H linking v_i and v_{i+1}, e_i and e_{i+1} are distinct hyperedges in H, $v_1 = u$, and $v_{n+1} = v$; directed hyperpaths are defined similarly. If a graph has a function associating numbers (i.e. *weights*) with its vertices (edges), the graph is called a *vertex- (edge-) weighted graph*; otherwise, it is an *unweighted graph*. Weighted and unweighted hypergraphs are defined similarly. Hypergraphs are useful in simplifying and generalizing results from graph theory, especially those results dealing with combinatorial problems [Ber73, p. viii]. For other standard graph and hypergraph definitions, see [Ber73, Ber85]; the definition of hyperarc is from [ADS86].

A path from any vertex to itself is called a *cycle*. A graph that does not

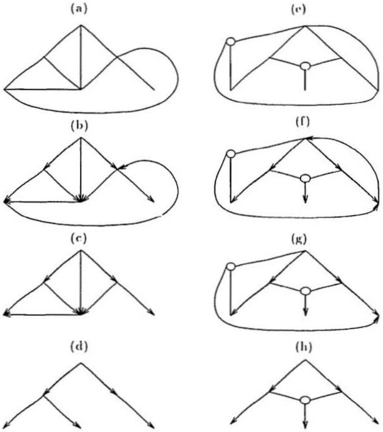


Figure 1: Graphs and hypergraphs: (a) graph; (b) directed graph; (c) directed acyclic graph; (d) directed tree; (e) hypergraph; (f) directed hypergraph; (g) directed Berge acyclic hypergraph; (h) directed hypertree.

contain any cycles is *acyclic*. An acyclic connected graph is called a *tree*. Directed cycles and directed acyclic graphs are defined similarly. Define a *directed tree* as a directed acyclic graph that satisfies three additional restrictions:

1. there is a distinguished vertex called the *root*,
2. there is at least one directed path from the root to every vertex in the tree, and
3. the root cannot be the target of any arc and every other vertex is the target of exactly one arc.

A hyperpath from any vertex to itself is called a *Berge cycle*. A hypergraph that does not contain any Berge cycles is *Berge acyclic*. Directed Berge cycles and directed Berge acyclic hypergraphs are defined similarly. Unlike graphs, there are many types of acyclicity for hypergraphs [Duk85] which are based on Berge cycles that satisfy additional restrictions; however, Berge acyclicity implies each of these other types of acyclicity [Fag83, Theorem 6.1]. Define a *directed hypertree* as a directed Berge-acyclic hypergraph that satisfies four additional restrictions:

1. there is a distinguished vertex called the *root*,
2. there is at least one directed hyperpath from the root to every vertex in the hypertree,
3. the root cannot be in the target-set of any hyperarc and every other vertex is in the target-set of exactly one hyperarc, and

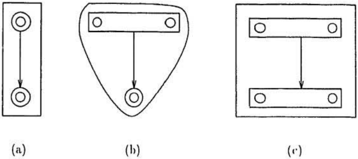


Figure 2: Types of hyperarcs: (a) 2-hyperarcs; (b) 3-hyperarcs; (c) 4-hyperarcs.

4. there are only three types of hyperarcs in the hypertree (see Figure 2):

   (i) one source vertex and one target vertex (*2-hyperarc*),

   (ii) two source vertices and one target vertex (*3-hyperarc*), or

   (iii) two source vertices and two target vertices (*4-hyperarc*).

Note that the correspondence of arcs to 2-hyperarcs makes directed trees special cases of directed hypertrees.

Trees are used in evolutionary biology to represent evolutionary relationships between species. In the biological literature, directed trees are called *rooted trees* and undirected trees are called *unrooted trees*. In evolutionary trees, edges are interpreted as species undergoing evolutionary change (*lineages*), vertices are interpreted as speciation events in which new species are generated, and the root vertex is interpreted as the most recent common ancestor of the species being studied. All types of trees give an estimate of the pattern of speciation events; however, only directed trees hypothesize the direction in which evolutionary change has proceeded. Many of the trees in this thesis will be edge-weighted trees, in which each edge's weight is interpreted as the amount of evolutionary change undergone by the species corresponding to that edge.

The restrictions above on directed trees and hypertrees guarantee the following biologically necessary properties: (1) no species can give rise to one of its ancestors, and (2) each species arises from exactly one speciation event. All types of trees can represent dichotomous speciation events; directed hypertrees can also represent two more complex evolutionary events using their 3- and 4-hyperarcs

hybridization (the creation of a third entity from two parent entities) and recombination (an altering of one or both of two entities). Such events involving the creation of two or more paths between pairs of vertices in a tree are called *reticulations*, and trees incorporating these events are said to be *reticulate*. Reticulation as defined here is applicable not only to problems involving hybridization and introgression as defined in evolutionary biology [Fun85, StaC75], but also to problems involving horizontal gene transfer [Sne75], multi-allele recombination events [Hei90], and transmission of copying errors in medieval manuscripts [Lee88]. See Appendix A for further discussion of reticulation.

Note that in the biological literature, graph-theoretic trees are often given different names depending on what they represent and the methods by which they were derived i.e. phylogram, dendrogram, cladogram, and that a single tree may on occasion imply a whole class of trees [HP84].

2.2 Computational Complexity Theory

For a more in-depth treatment of computational complexity theory, see [BDG88, BDG90, GJ79, HS78, Joh90, WW86]. Biologists will find [Day92] a good introduction to certain topics in this section.

There are many types of formal computational problems e.g. decision, evaluation, counting. These problems can be unified using the framework developed in [WW86, pp. 100–101] cf. [JVV86]. Define a relation $R : \Sigma^* \times \Sigma^*$ on pairs of objects e.g. (boolean formulas) × (truth assignments to boolean variables); (graphs) × (cliques). Formal computational problems can be viewed as functions defined on the projection of R onto a given element x.

- *Decision Problem (PROB)*:

  For some boolean-valued predicate G defined on Σ^*,

  PROB(x) = $\exists y\,[(x,y) \in R \wedge G(y)]$.

- *Solution Problem (SOL-PROB)*:

  For some boolean-valued predicate G defined on Σ^*,

  SOL-PROB(x) = $\{\, y \mid (x,y) \in R \wedge G(y)\}$.

If relation R has an associated valuation function $b : R \rightarrow \mathcal{N}$, R corresponds intuitively to an optimization problem. Define the following problems on such R.

- *Given-cost Solution Problem (SOL-VAL.EQ-PROB)*:

  $\text{SOL-VAL.EQ-PROB}(\langle x, k\rangle) = \{\, y \mid (x,y) \in R \wedge b(x,y) = k\}$

- *Given-limit Solution Problem (SOL-VAL.LE-PROB, SOL-VAL.GE-PROB)*:

  $\text{SOL-VAL.LE-PROB}(\langle x, k\rangle) = \{\, y \mid (x,y) \in R \wedge b(x,y) \leq k\}$

  $\text{SOL-VAL.GE-PROB}(\langle x, k\rangle) = \{\, y \mid (x,y) \in R \wedge b(x,y) \geq k\}$

- *Optimal-cost Evaluation Problem (MIN-PROB, MAX-PROB)*:

  $\text{MIN-PROB}(x) = \min_{(x,y)\in R} b(x,y)$

  $\text{MAX-PROB}(x) = \max_{(x,y)\in R} b(x,y)$

- *Optimal-cost Solution Problem (SOL-$\mathbf{X}$-PROB, $\mathbf{X} \in \{MIN, MAX\}$)*:

  $\text{SOL-}\mathbf{X}\text{-PROB}(x) = \{\, y \mid (x,y) \in R \wedge b(x,y) = \mathbf{X}\text{-PROB}(x)\}$

Three other types of problems may be defined on the ranges of $\mathbf{Y}$, $\mathbf{Y} \in$ {SOL-PROB,SOL-$\mathbf{X}$-PROB} (X $\in$ {MIN, MAX, VAL.EQ, VAL.LE, VAL.GE}).

- *Spanning Problem (SPAN-$\mathbf{Y}$)*:

  $\text{SPAN-}\mathbf{Y}(x) = |\{\mathbf{Y}(x)\}|$.

- *Random-Selection Problem (RAND-$\mathbf{Y}$)*:

  RAND-$\mathbf{Y}(x) = y$, where y is a randomly-selected member of $\{\mathbf{Y}(x)\}$.

- *Enumeration Problem (ENUM-$\mathbf{Y}$)*:

  ENUM-$\mathbf{Y}(x,\ i) = y$, where y is the i-th member of $\{\mathbf{Y}(x)\}$ under some polynomial-time ordering P.

Each of these problems corresponds to a function. A decision problem also corresponds to the language composed of the subset of its instances whose solution is "yes". A problem X is said to be solved by an algorithm if for any input x, that algorithm computes a single value from $\{f(x)\}$ for the function f embodied in that problem. Let X_f denote the set of single-valued functions corresponding to algorithms that solve problem X. Inputs to a problem will be called *instances* and outputs will be called *solutions*.

Define deterministic Turing machines (DTM), nondeterministic TM (NTM), and deterministic and nondeterministic oracle TM (DOTM, NOTM) that recognize languages (*acceptors*) and compute functions (*transducers*) in the standard manner [BDG88, GJ79]. A DTM transducer N computes y on input x ($N(x) \rightarrow y$) if y is the final contents of N's output tape for the computation of N on x. A NTM transducer N computes y on input x ($N(x) \mapsto y$) if there is an accepting computation of N on x such that y is the final contents of N's output tape [Sel91, p. 3]. DTM transducers compute single-valued functions, and NTM transducers compute partial multivalued functions. Any input to a TM transducer that does not have an accepting computation computes symbol $\perp$. An OTM that forces all queries to be made simultaneously is *non-adaptive*, while an OTM that allows queries to be made on the basis to answers to previous queries is *adaptive*.

The computational resources used by an algorithm to solve an instance of a problem can be visualized as the computational resources used by the TM which corresponds to that algorithm. Let $r_A(x)$ be the amount of resource R used by algorithm A on input x. For a function $f : \mathcal{N} \rightarrow \mathcal{N}$ and a computational resource R, an algorithm A is f-R (f-R *computable*) e.g. polynomial-time, polynomial-time computable, if $r_A(|x|) \in O(f)$. A problem is f-R if there is an f-R algorithm that solves that problem.

Problems can be grouped into complexity classes based on bounds on computational resources required to solve those problems e.g. DTIME($poly$), which is the set of all problems solvable by polynomial-time DTM. Some standard complexity classes for decision problems are:

P All decision problems solvable by polynomial-time DTM.

NP All decision problems solvable by polynomial-time NTM.

PSPACE All decision problems solvable by polynomial-space DTM.

EXPTIME All decision problems solvable by exponential-time DTM.

It is known that $P \subseteq NP \subseteq PSPACE \subseteq EXPTIME$, and that P $\subset$ EXPTIME [BDG88, Proposition 3.1]. Several classes of complexity between P and NP are:

UP All decision problems solvable by polynomial-time NTM such that for each input, there is at most one accepting computation.

FewP All decision problems solvable by polynomial-time NTM such that for each input I and a fixed polynomial p, there are at most $p(|I|)$ accepting computations.

R All decision problems solvable by polynomial-time NTM such that for each input I, either there are no accepting computations or else at least half of all computations are accepting.

Many problems are of complexity between NP and PSPACE, and are within the levels of the Polynomial Hierarchy [MS72]:

$$\Theta_0^p = \Delta_0^p = \Sigma_0^p = P$$

$$\Theta_{k+1}^p = P^{\Sigma_k^p[O(\log n)]}$$

$$\Delta_{k+1}^p = P^{\Sigma_k^p}$$

$$\Sigma_{k+1}^p = NP^{\Sigma_k^p}$$

$$\Pi_{k+1}^p = \text{co-}NP^{\Sigma_k^p}$$

$$\text{PH} = \bigcup_{k=1} \Delta_k^p$$

where P^Y (NP^Y) is the class of problems solvable by polynomial-time DOTM (NOTM) that can use any oracle in class Y, and $P^{Y[f(n)]}$ ($P_{||}^{Y[f(n)]}$) is the class of problems solvable by polynomial-time adaptive (non-adaptive) DOTM that can ask up to $f(n)$ queries to an oracle in class Y. Levels Δ_k^p, Σ_k^p, and Π_k^p were defined in [MS72], and level Θ_k^p was defined in [WagK88]. It is known that $\Theta_k^p \subseteq \Delta_k^p \subseteq \Sigma_k^p \cup \Pi_k^p \subseteq P^{\Sigma_k^p[1]} \subseteq \Theta_{k+1}^p$, that $PH \subseteq PSPACE$, and that if for some k, $\Pi_k^p = \Sigma_k^p$ then PH $= \Sigma_k^p$ [Sto77, WagK90, Wra77]. Two working hypotheses in complexity theory are that P $\neq$ NP and that PH does not collapse to any finite level.

Many of the language complexity classes above can be restated as classes of single-valued functions. Let **FX** denote the class of functions computed by TM used to define language class **X** e.g. FNP, $F\Delta_k^p$, FPSPACE. Define FPSPACE(poly) as those functions in FPSPACE whose outputs are polynomially bounded in the length of the input, and FPH $= \bigcup_{k=1} F\Delta_k^p$. It is known that $\Delta_k^p \subset F\Delta_k^p$, $FP^{\Sigma_k^p[f(n)]} \subseteq FP^{\Sigma_k^p[f(n)+1]}$, $FP^{\Sigma_k^p[f(n)]} \not\subset FP^{\Sigma_{k+1}^p[f(n)-1]}$ unless P $=$ NP, and $FP^{\Sigma_{k+1}^p[1]} \not\subset FP^{\Sigma_k^p[f(n)]}$ unless $FPH = FP^{\Sigma_k^p}$ [Gas92, Kre92b]. The relationships within and between classes in FPH have been established only at the lowest levels (see Section 4.1.1); those relationships known to date suggest that classes in FPH behave very differently from their analogues in PH [Gas92]. Classes of multivalued functions are also possible. There are many restrictions of polynomial-time NTM transducers that generate such classes [Sel91]; one such restriction is F_g, the subset of functions f in class F such that graph(f) $= \{\langle x, y\rangle | f(x) \mapsto y\} \in P$ i.e. outputs can be checked in polynomial time [Sel91, Val76]. Define FMPH $= \bigcup_{k=1} F\Sigma_k^p$ and $FM_gPH = \bigcup_{k=1}(F\Sigma_k^p)_g$. Classes $F\Sigma_1^p = FNP$, $(F\Sigma_1^p)_g = FNP_g$, and $F\Sigma_k^p$ are called NPMV, $NPMV_g$, and $NPMV^{\Sigma_{k-1}^p}$ in [FHOS92, Sel91], and NTM in FNP_g which compute total functions are called NP metric TM in [Kre88]. Functions in FNP_g compute the solutions associated with decision problems in NP. For further discussion about these and other function classes, see Section 4.1.

A reduction $\Pi \propto \Pi'$ is an algorithm that solves problem Π by using an algorithm for problem Π'. Reductions order problems by computational hardness. The two main types of reductions between decision problems are:

- **many-one** ($\leq_m^p$): $A \leq_m^p B$ if there is a polynomial-time function f such that $x \in A$ if and only if $f(x) \in B$.

- **Turing** ($\leq_T^p$): $A \leq_T^p B$ if there is a polynomial-time function using B as an oracle that determines if $x \in A$.

A generalization of many-one reducibility called *metric reducibility* holds between single-valued functions.

Definition 1 (adapted from [Kre88], p. 493) *Let $f, g : \Sigma^* \rightarrow \Sigma^*$. A* metric reduction from f to g *is a pair of polynomial-time functions (T_1, T_2), where $T_1 : \Sigma^* \rightarrow \Sigma^*$ and $T_2 : \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$, such that $f(x) = T_2(x, g(T_1(x)))$ for all $x \in \Sigma^*$.*

The following variant holds between problems.

Definition 2 *Let Π and Π' be problems and SOL-$\mathbf{X}$(I) be the set of solutions associated with instance I of problem* $\mathbf{X}$. *A* metric reduction from Π to Π' *is a pair of polynomial-time functions (T_1, T_2). where $T_1 : I \rightarrow I'$ and $T_2 : I \times S' \rightarrow S$, such that for any single-valued function f that solves Π', $T_2(I, f(T_1(I))) \in$ SOL-Π(I) for any instance I of* Π.

This reducibility is a restricted version of the Turing reducibility between partial multivalued functions defined in [FHOS92]. Note that the definitions of these metric reducibilities are equivalent for problems that are single-valued. Another relation called refinement can also hold between multivalued functions. Given multivalued functions f and g, g is a *refinement* of f if $dom(f) = dom(g)$ and for all $x \in dom(g)$ and all y, if $g(x) \mapsto y$ then $f(x) \mapsto y$. These relations can also hold between whole classes of functions. For instance, if F and G are two classes of partial multivalued functions, then $F \subseteq_c G$ if every $f \in F$ has a refinement in G [Sel91, p. 4]. Both inclusion and refinement relations can hold between multivalued function classes, and single-valued classes can be included in multivalued classes (indeed, this is equivalent to refinement); however, only the single-valued refinement relation can hold between multivalued function classes and single-valued classes.

Given two problems x and y and a reducibility r, x and y are *computationally equivalent* if x r-reduces to y and y r-reduces to x. Given a class of problems X and a reducibility r, a problem y is said to be X-*hard* if each problem in X

r-reduces to y. If y is X-hard and is also in X, y is *X-complete*; if y is X-hard and is not in X, y is *properly X-hard*. Two limited types of reductions are:

- *arithmetic-equivalence reductions*: Problems Π and Π' differ only in their cost-functions b_Π and $b_{\Pi'}$, and there exists a pair of polynomial-time functions (T_1, T_2) such that for all instances x of Π and Π', $b_\Pi(x) = T_1(b_{\Pi'}(x))$ and $b_{\Pi'}(x) = T_2(b_\Pi(x))$.

- *restriction reductions*: Problems Π and Π' differ only in that $dom(\Pi) \subset dom(\Pi')$ i.e. Π is a subproblem of Π'.

By definition, restriction and arithmetic-equivalence reductions are many-one and metric reductions.

Though some of the problems discussed in this thesis are most naturally defined on $\mathcal{R}^+$, all problems will be restricted to $\mathcal{Q}^+$. Real numbers in general cannot be used because irrational numbers e.g. $\sqrt{2}$, cannot be represented within a computer whose running time is bounded by a function of the length of its input. All irrational numbers that arise in calculations must also be eliminated or approximated e.g. $\sqrt{x} \rightarrow \lceil \sqrt{x} \rceil$. A case study in how a real-number problem is modified to be computable is given for the Euclidean Minimal Steiner Tree problem in [GGJ77]. The lower bounds given by such modified problems on the complexity of the actual problems is the best that can be done within computational complexity theory as it currently exists. However, there may be other options [BSS89, Ko91].

Though problems will be defined on $\mathcal{Q}^+$ for the convenience of readers, all problems will actually operate on $\mathcal{N}$. This is easily done by multiplying out the rational denominators i.e. $\frac{1}{3}, \frac{3}{4} \rightarrow \{4, 9\} \div 12$. Thus, the bit-representation length of numbers will be proportional to their value. This ensures that the length of certain small rational numbers will not exceed that of larger numbers (e.g. though $\frac{13}{14} < 13$, $|13| + |14| > |13|$; however, $|13| < |13 \cdot 14|$). This property is necessary in several proofs in Sections 3.2.1 and 3.2.3.

3 Computational Problems in Phylogenetic Systematics

This section begins with an overview of various concepts in phylogenetic systematics. This is foilowed by a review of certain decision problems associated with phylogenetic analysis using the phylogenetic parsimony, character compatibility, and various of the distance matrix fitting criteria, and a review of the r• luctions by which these problems have been shown to be NP-complete [Day83, Day87, DJS86, DS86, DS87, Kri88, KM86]. This section also includes definitions and reductions for several new phylogenetic parsimony problems that allow limited amounts of reticulation, as well as a new reduction for the Additive Evolutionary Tree problem [Day83].

3.1 Phylogenetic Systematics

Systematics is the subdiscipline of biology that deals with ordering species into sets of groups (*systems*) according to various kinds of relationships between species (e.g. ecological roles, geographical proximity, overall similarity) [Ax87, Hen66]. Phylogenetic systematics is in turn the subdiscipline of biological systematics concerned with ordering species based on their evolutionary relationships; specifically, species are grouped together by descent from a common ancestor, and these groups are nested hierarchically to make an evolutionary tree. The process of reconstructing evolutionary trees is called *phylogenetic analysis* (*phylogenetic inference*), and the evolutionary trees so reconstructed are called *phylogenies.*

The units that are ordered in phylogenetic analysis are called *taxa.* Two types of data are typically available to reconstruct evolutionary relationships among taxa:

- **Discrete Character Matrix:** The data are an m-by-d matrix giving the values possessed by each of a set of m taxa for each of a set of d characteristics. These characteristics are called *characters* and their values are *character states* For example, a character **flower colour** might have character states **blue, yellow,** and **red.** Character-states are grouped into characters by the relation of homology [Ax87, EC80, Wil81].[1] The vector of character-states over all taxa for a particular character is a *character pattern*, and the vector of character-states over all characters for a particular taxon is a *character distribution.* If a character has only two states, it is *binary*; else, it is *unconstrained* (*multistate*). If a character has a graph

[1] *Homology* is the relation among different structures in different species that evolved from a common ancestral species (e.g. the character **mammalian fore-limb** that has states *arm* (human beings, apes), *foreleg* (dogs, horses, tigers), *wings* (bats), and *flippers* (dolphins, whales)). There are other kinds of relations among observed character-states, such as *analogy*, the relation of similar structures in different species that have arisen independently in several ancestral species (e.g. the character **wings** that groups together the wings of insects, birds, and bats). All character-state relations give evolutionary information of some sort; however, only homology delimits groups of species sharing a common ancestor, and thus only characters formed by homology are useful in reconstructing evolutionary trees.

In the case of molecular sequences, homology can hold among different sequences from different species, as well as between different positions in different sequences; indeed, the problem of establishing sequence-position homology is that of deriving a sequence alignment [SO90, pp. 416–417]. Note that in molecular biology, the term "homology" is also a synonym for sequence similarity [MH90, pp. 7–9].

imposed on its states whose edges specify the allowable changes from one state to another, the character is *ordered*; else, the character is *unordered*. If the edges of an ordered character's graph are directed, the character is *polarized*; else, it is *unpolarized*. If the edges of an ordered character's graph have weights, the character is *weighted*; else, it is *unweighted*. Ordered characters are typically based on linear, complete, or tree graphs. In polarized characters, if state X is the source of a directed path to state Y, X is *ancestral* to Y and Y is *derived* relative to X. The state that is ancestral to all character-states in a polarized character is the ancestral state for that character. By convention, the ancestral and derived states in polarized binary characters are written as 0 and 1.

- **Distance Matrix:** The data are an m-by-m matrix giving a measure of dissimilarity or similarity between each pair of taxa in a set S of m taxa. The terms "similarity" and "dissimilarity" denote quantities that are precisely defined and inversely related; when rigor is not required or specified, both will be denoted by the term "distance" [SO90, p. 423]. Let M_n be the set of non-negative rational real-valued matrices on n taxa, and $B_n \subset M_n$ be the set of all matrices whose off-diagonal values are in $\{1, 2\}$. Call members of B_n and M_n *binary* and *unconstrained* matrices respectively, by analogy with discrete characters. Let $X_{S'}$ be matrix X on S restricted to $S' \subset S$. Every matrix represents a distance function $d : S^2 \rightarrow \mathcal{R}$, which may satisfy some subset of the following properties.

1. $\forall x \in S$, $d(x,x) = 0$.

2. $\forall x, y \in S$, $d(x,y) = 0$ implies $x = y$.

3. $\forall x, y \in S$, $d(x,y) = d(y,x)$.

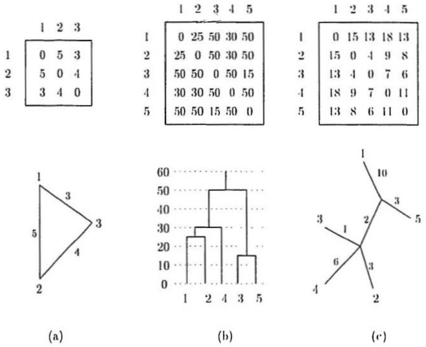
4. $\forall x, y, z \in S$, $d(x,y) \leq d(x,z) + d(z,y)$.

5. $\forall x, y, z \in S$, $d(x,y) \leq \max[d(x,z), d(z,y)]$.

6. $\forall x, y, z, w \in S$,

   $d(x,y) + d(z,w) \leq \max[d(x,z) + d(y,w), d(x,w) + d(y,z)]$.

Conditions (4), (5), and (6) are known as the *triangle*, *ultrametric*, and *additive* inequalities, respectively. A function that satisfies conditions (1), (2), and (3) is a *semimetric*; if condition (4) is also satisfied, the function is a *metric*. Metrics that satisfy conditions (5) and (6) are known as *ultrametrics* and *tree metrics*, respectively (see Figure 3). The number of distinct off-diagonal values in an ultrametric is the *height* of that ultrametric. Tree metrics and ultrametrics can be represented as trees; let U_n (A_n) be the set of all ultrametric (additive) trees on n taxa, $U_{n,q} \subset U_n$ be the set of all ultrametric trees on n taxa of height at most q, $1 \leq q \leq n(n-1)/2$, and $\pi_U : U_n \rightarrow M_n$ ($\pi_A : A_n \rightarrow M_n$) be the function that maps an ultrametric (additive) tree onto its ultrametric (tree metric). In this thesis, A_n will be restricted to A_n^d (*discretized additive trees*) whose edges have length $k/2, k >$



$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 5 & 3 \\ 2 & 5 & 0 & 4 \\ 3 & 3 & 4 & 0 \end{array}$$

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 25 & 50 & 30 & 50 \\ 2 & 25 & 0 & 50 & 30 & 50 \\ 3 & 50 & 50 & 0 & 50 & 15 \\ 4 & 30 & 30 & 50 & 0 & 50 \\ 5 & 50 & 50 & 15 & 50 & 0 \end{array}$$

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 15 & 13 & 18 & 13 \\ 2 & 15 & 0 & 4 & 9 & 8 \\ 3 & 13 & 4 & 0 & 7 & 6 \\ 4 & 18 & 9 & 7 & 0 & 11 \\ 5 & 13 & 8 & 6 & 11 & 0 \end{array}$$

Figure 3: Types of metrics (taken from [Day88]): (a) a metric and a possible representation in the 2-D Euclidean plane; (b) an ultrametric and its associated ultrametric tree; (c) a tree metric and its associated additive tree.

0; note that ultrametrics drawn from A_n^d have integer off-diagonal entries. Ultrametric trees are by definition rooted while additive trees can be either rooted or unrooted. Ultrametric trees correspond to rooted additive trees in which each leaf is the same distance from the root.

Discrete character matrices are generated by examining the taxa of interest. Distance matrices are generated directly via certain techniques (i.e. immunological assay, DNA – DNA hybridization) or derived from discrete character matrices by applying a distance function defined on pairs of character distributions. Raw distance matrices must often be transformed into matrices that reflect "true" evolutionary distances [SO90, pp. 422–436]. These types of data are not ideal for the task of reconstructing evolutionary history, but they are sufficient: as taxa originate by inheritance with modification, each ancestral lineage in the evolutionary tree has left its signature in its descendents, either as character states that have propagated to that lineage's descendents, or as a certain evolutionary distance by which each such descendent is separated from every other taxon in the tree. Hence, many of the ancestral lineages, as well the details of the process by which ancestral lineages gave rise to the observed taxa, can be reconstructed using the types of data above [EC80].

There are several other useful representations for evolutionary trees besides tree graphs. In trees constructed using discrete character data, each vertex in the tree has its own set of character-state values. These trees can be summarized by the character-state sets of their vertices (see Figure 4). Alternatively, for each character, one can map the set of vertices possessing each character-state onto that character's character-state graph to create a *cladistic character*, and summarize a tree by its set of cladistic characters. Cladistic characters are often easier to visualize as trees of subsets (see Figure 5). Discrete character matrices and individual characters may also be summarized by cladistic characters. Non-reticulate edge-weighted trees can be summarized by their *patristic matrices*,

$P = [p_{ij}]$, where p_{ij} is the sum of the weights of all edges on the path between taxa i and j in the given tree. A tree whose patristic matrix is an ultrametric of height q can be represented [JS71, pp. 48–50] [KM86, p. 312] as a $(q+1)$-length sequence of pairs (P_i, l_i) such that

1. $P_1, P_2, \ldots, P_{(q+1)}$ are partitions of S,

2. l_i is an integer such that $0 = l_1 < l_2 < \ldots < l_{(q+1)}$,

3. P_i is a *proper refinement* of P_{i+1} $(1 \leq i \leq q)$, and

4. $P_1 = \{\{s_1\}, \{s_2\}, \ldots, \{s_{|S|}\}\}$ and $P_{(q+1)} = \{S\}$.

For example, the partition representation of the ultrametric tree of height 4 in Part (b) of Figure 3 is

$$\begin{aligned}
(P_1, l_1) &= (\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\}, 0), \\
(P_2, l_2) &= (\{\{1\}, \{2\}, \{4\}, \{3, 5\}\}, 15), \\
(P_3, l_3) &= (\{\{1, 2\}, \{4\}, \{3, 5\}\}, 25), \\
(P_4, l_4) &= (\{\{1, 2, 4\}, \{3, 5\}\}, 30), \text{ and} \\
(P_5, l_5) &= (\{\{1, 2, 3, 4, 5\}\}, 50).
\end{aligned}$$

By convention, the weight of an edge in a tree reconstructed using discrete-character data is the sum of the weights of all character-state changes (*character-state transitions*) between the vertices defining that edge (see Figure 4).

Each approach to phylogenetic analysis considered in this thesis embodies a criterion that assigns a cost to each possible tree relative to a particular data

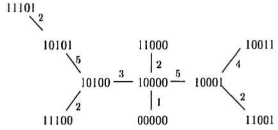
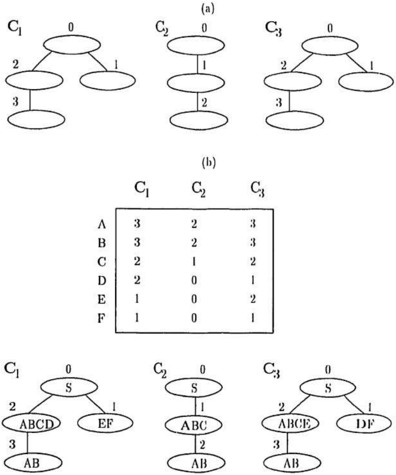


Figure 4: A discrete character tree. This unrooted tree is based on 5 unweighted binary characters, and has a length of 9. The number on an edge denotes the character whose state has changed on that edge. Note that there are multiple character-state transitions in characters 2 and 5.

set. The trees selected by each approach as the best estimates of the actual evolutionary tree for a data set are the trees whose cost is optimal for that data set under that approach's criterion. Hence, each approach to phylogenetic analysis is an optimization problem.

Several of the most popular approaches to phylogenetic analysis that use discrete character data are:

- **Phylogenetic Parsimony** [Hen66, KF69]: Selects the evolutionary tree of shortest length that reproduces the character distributions for the given taxa, where the *length* of a tree is the sum of the weights of all edges in the tree. The hypothesis encoded in this tree is preferred because it explains as much of the observed character distributions as possible by character-state transitions in a common ancestor, and invokes the fewest *ad hoc* hypotheses of subsequent character-state change [Far83].



	C_1	C_2	C_3
A	3	2	3
B	3	2	3
C	2	1	2
D	2	0	1
E	1	0	2
F	1	0	1

Figure 5: Character-state trees (adapted from [Day88]). Part (a) shows three character-state trees C_1, C_2, and C_3. Part (b) shows the trees of subsets corresponding to each of these characters as determined by discrete character matrix X on the set of taxa $S = \{A, B, C, D, E, F\}$.

Criterion		Character-State Transition Restrictions	Character Order Type
Wagner (Linear)	**WL**	No restrictions.	Linear
Wagner (General)	**WG**	No restrictions.	Ordered
Fitch	**Fi**	No restrictions.	Complete
Camin-Sokal	**CS**	No transitions from derived to ancestral states.	Ordered
Dollo	**Do**	One transition from ancestral to derived state per character.	Linear
Chromosome Inversion (Polymorphism)	**CI**	One transition from ancestral to heterozygous state per character; no transitions from ancestral to derived or from derived to ancestral or heterozygous states.	Linear
Generalized	**Ge**	Specified for each character.	Ordered

Table 1: Phylogenetic parsimony criteria.

There are several phylogenetic parsimony criteria, each of which encodes a different model of evolution by placing different restrictions on the types and numbers of character-state transitions allowable in a tree (see Table 1). The Wagner Linear [KF69], Wagner General, and Fitch [Fit71] criteria assume the simplest model of evolution, in which character-state change is reversible. The Camin-Sokal criterion [CS65] assumes that character change is irreversible, while the Dollo criterion [Far77] assumes that character-state change is reversible but character-state origin is unique. The Chromosome Inversion criterion [Far78] is a restricted Dollo criterion whose characters have three states: ancestral (**A**), derived **D**), and heterozygous (**H** = {**A**, **D**}). The Generalized parsimony criterion [SC83] represents characters as matrices of distances between character states (*stepmatrices*), which allows this criterion to simulate all possible parsimony criteria by placing appropriate restrictions on the state-transition weights [SO90, Figure 11, p. 464].

Note that in the biological literature, phylogenetic parsimony methods are also called cladistic parsimony or cladistic methods, and that the term "phylogenetic systematics" is on occasion restricted to the inference of evolutionary trees by phylogenetic parsimony methods.

- **Character Compatibility** [ME85]: Reconstructs the evolutionary tree from the largest subset of the given characters that are pairwise compatible, where two cladistic characters K and L are compatible if there exists a tree of subsets $\mathbf{M}$ such that the trees of subsets $\mathbf{K}$ and $\mathbf{L}$ of these characters are subsets of $\mathbf{M}$. For example, in Figure 5, characters C_1 and C_2 are compatible and C_2 and C_3 are compatible, but C_1 and C_3 are not compatible.

- **Maximum Likelihood** [Fel81]: Selects the evolutionary tree that has the greatest probability of producing the frequencies of each type of character-pattern in the data, i.e. the maximum likelihood *P(Characters | Tree)*, relative to some probabilistic model of character-state change.

- **Invariants (Evolutionary Parsimony)** [CF87, Lak87]: Select the evolutionary tree that best satisfies its associated *invariants*, which are algebraic constraints on the observed frequencies of each type of character-pattern that hold for that tree over all possible discrete character matrices. The set of invariants for each evolutionary tree is derived relative to some probabilistic model of character-state change.

There are many approaches to phylogenetic analysis using distance matrix data, all of which assume that the given distances represent or closely approximate actual evolutionary distances between taxa. Most of these approaches compute the ultrametric or tree metric corresponding to the tree that has the minimal distance from the given semimetric according to some statistic. Many of these statistics are based on the Minkowski metrics L_q, $q \geq 1$, defined on pairs of matrices D and P on taxa S.

$$L_q(D,P) = \{ \sum_{x,y \in S} |D_{xy} - P_{xy}|^q \}^{1/q} \tag{1}$$

$$L_\infty(d,p) = \max_{x,y \in S} |D_{xy} - P_{xy}| \tag{2}$$

Several such statistics for semimetric D and ultrametric or tree metric P are

$$F_\alpha(D,P) = \sum_{x,y \in S} |D_{xy} - P_{xy}|^\alpha \qquad [\alpha \in \{1,2\}] \tag{3}$$

and

$$F(D,P) = 100 \times \frac{\sum_{x,y \in S} |D_{xy} - P_{xy}|}{\sum_{x,y \in S} D_{xy}} \tag{4}$$

where F_1 is the f-statistic [Far72], F_2 is the least-squares fit criterion [CE67], and F was defined in [PW76]. Note that each statistic in both of the groups

$\{L_1, F_1, F\}$ and $\{L_2, F_2\}$ is arithmetically equivalent to other members of its group. One can also view the given distances not as targets to be approximated but as lower bounds on what should be approximated. This is embodied in the concept of *dominance*, i.e. for metrics D and D' on a set of objects S, D *dominates* $D'(D \geq D')$ if $\forall x, y \in S, D_{xy} \geq D'_{xy}$ [JS71, p. 52]. Though dominance was originally proposed for fitting ultrametric trees, it has also been used in some methods for fitting additive trees [SO90, p. 451].

Each approach to phylogenetic analysis embodies some model of the evolutionary process of character change; some are more explicit than others in the statement of the model that they use. The trees produced by each approach are useful to the extent that one believes in the model embodied by that approach. See [Fel88, PHS92, SO90] for a complete review of approaches to phylogenetic analysis and computer programs implementing these approaches.

3.2 NP-Complete Problems in Phylogenetic Systematics

Since 1982, decision problems for the major phylogenetic parsimony criteria [Day83, DJS86, DS87, GF82], the character compatibility criterion [DS86], and various distance matrix fitting criteria for ultrametric and additive trees [Day83, Day87, Kri86, Kri88, KM86] have been shown NP-complete using reductions from the NP-complete problems given in Table 2. As later sections of this thesis will make extensive use of both these definitions and these reductions, they will be

VERTEX COVER (VC) **[GT1]**

Instance: A graph $G = (V, E)$ and a positive integer $K \leq |V|$.

Question: Is there a *vertex cover* of size K or less for G, that is, a subset $V' \subseteq V$ such that $|V'| \leq K$ and, for each edge $\{u, v\} \in E$, at least one of u or v belongs to V'?

EXACT COVER BY 3-SETS (X3C) **[SP2]**

Instance: A set X with $|X| = 3q$ and a collection C of 3-element subsets of X.

Question: Does C contain an *exact cover* for X, that is, a subcollection $C' \subseteq C$ such that every element of X occurs in exactly one member of C'?

CLIQUE **[GT19]**

Instance: A graph $G = (V, E)$ and a positive integer $J \leq |V|$.

Question: Does G contain a *clique* of size J or more, that is, a subset $V' \subseteq V$ such that $|V'| \geq J$ and every two vertices in V' are joined by an edge in E?

Table 2: Basic NP-complete decision problems (taken from [GJ79]). The reference numbers assigned to these problems in the list of NP-complete decision problems in [GJ79] are given in square brackets.

reviewed in this section. Each reduction will be given a formal definition in the style of [Kar72], followed by a sketch of its proof of correctness.

3.2.1 Phylogenetic Parsimony

Each of these problems is given as input a discrete character matrix for m taxa and d characters, and operates on an implicit graph G whose vertices are the set of all d-dimensional points defined by the states of the given characters and whose edges are specified by the allowable transitions between the states in these characters. Each phylogenetic parsimony problem seeks the evolutionary tree in G of minimum length that includes the given taxa, subject to the restrictions on character-state transitions that are particular to that problem's criterion (see Table 1). The given characters can be restricted in various ways to generate a family of phylogenetic parsimony problem "schemata" (see Tables 3, 4, and 8); each phylogenetic parsimony criterion can then be applied to these schemata to generate problems. The hierarchy of subproblems generated by these schemata will be useful in later sections of this thesis.

Consider the following restrictions on the given characters:

- **Cladistic vs. Ordered vs. Qualitative:** A *cladistic* (**C**) problem is given polarized characters, an *ordered* (**O**) problem is given ordered characters, and a *qualitative* (**Q**) is given unordered characters. Each problem finds solutions that are consistent with its characters; however, ordered and qualitative problems must also find character polarizations and orderings for which solutions exist. The cladistic / qualitative distinction was made in [DJS86, EM77, EM80] for binary characters; as qualitative and ordered problems are equivalent for binary characters, the distinction of ordering is only applicable to unconstrained characters.

  Cladistic problems correspond to phylogenetic analysis procedures that produce explicitly rooted trees, ordered problems correspond to procedures that produce either rooted or unrooted trees, and qualitative problems correspond to procedures such as Transformation Series Analysis [Mic82] that simultaneously produce trees and derive character ordering and polarization from the given data (cf. [Lip92]).

- **Binary vs. Unconstrained:** A problem is *binary* (**B**) if it is restricted to binary characters; otherwise, the problem is *unconstrained* (**U**).

- **Weighted vs. Unweighted:** A problem is *unweighted* (**U**) if it is restricted to unweighted characters; otherwise, the problem is *weighted* (**W**).

The five schemata generated by the first two of these restrictions are given in Tables 3 and 4; the remaining restriction yields a total of ten schemata. The validity of these restrictions for each of the phylogenetic parsimony criteria is shown in Table 5. Restrictions do not apply to a particular criterion if they conflict with the restrictions imposed by that criterion e.g. Dollo criterion characters can only have three states; Fitch criterion characters are by definition unweighted and ordered. The application of all phylogenetic parsimony criteria to valid schemata yields 39 phylogenetic parsimony problems (see Tables 6 and 7).

Additional phylogenetic parsimony problems may be generated by allowing evolutionary trees to include limited amounts of reticulation. Consider the problem schemata in Table 8 defined for each non-reticulate phylogenetic parsimony problem **X**. These schemata restrict the amounts of available (the SH**X** and SR**X** schemata) or allowable (the k-H**X** and k-R**X** schemata) reticulation that can oc-

Binary Cladistic **X** (BCX)

Instance: Positive integer d; a subset S of $\{0, 1\}^d$; and a positive integer B.

Question: Is there a phylogeny satisfying criterion **X** that includes S, is rooted at the **root-type** vertex, and has length at most B?

Binary Qualitative **X** (BQX)

Instance: Same as BCX, except that no character is required to be directed.

Question: Is there a phylogeny satisfying criterion **X** that includes S and has length at most B?

Table 3: Phylogenetic parsimony decision problem schemata (non-reticulate trees) (adapted from [DJS86]). These schemata are stated relative to a phylogenetic parsimony criterion **X**. If $\mathbf{X} \in \{\mathrm{CI}, \mathrm{CS}\}$, **root-type** is "all-ancestral"; if $\mathbf{X} = \mathrm{Do}$, **root-type** is "all-derived".

Note that the statements of problems given above differ from [DJS86, DS87] in that the bound B is on the number of edges rather than the number of vertices in the tree. The two formulations are equivalent; however, the former allows a more natural interpretation of weighted problems.

UNCONSTRAINED CLADISTIC **X** (UCX)

Instance: Positive integer d; sets $A_1, \ldots, A_d$ of character-states, and directed character-state graphs $G_1, \ldots, G_d$ specifying allowable transitions among these states; a subset S of $A_1 \times \ldots \times A_d$; and a positive integer B.

Question: Is there a phylogeny satisfying criterion **X** and the given directed character-state graphs that includes S, is rooted at the **root-type** vertex, and has length at most B?

UNCONSTRAINED ORDERED **X** (UOX)

Instance: Same as UC**X**, except that none of the character-state graphs are directed.

Question: Is there some polarization of the given character-state graphs that allows a phylogeny satisfying criterion **X** that includes S and has length at most B?

UNCONSTRAINED QUALITATIVE **X** (UQX)

Instance: Same as UC**X**, except that none of the character-state graphs are ordered.

Question: Is there some ordering and polarization of the given character-state graphs that allows a phylogeny satisfying criterion **X** that includes S and has length at most B?

Table 4: Phylogenetic parsimony decision problem schemata (non-reticulate trees) (cont'd from Table 3).

Criterion		Unweighted / Weighted	Binary / Unconstrained	Cladistic / Ordered / Qualitative	# Prob.
Wagner Linear	**WL**	✓	✓	**O,Q**	8
Wagner General	**WG**	✓	✓	**O,Q**	8
Fitch	**Fi**		✓	**O**	2
Camin-Sokal	**CS**	✓	✓	**C,O,Q**	12
Dollo	**Do**	✓	✓	**C,O,Q**	12
Chromosome Inversion	**CI**			**C,O,Q**	3
Generalized	**Ge**	✓	✓	**O**	4
				Total	39

Table 5: Applicability of input character restrictions to phylogenetic parsimony criteria. The given total number of problems is smaller than expected because some of these problems are equivalent; see Tables 6 and 7 for details.

cur in a tree. See Appendix A for further discussion of these schemata. Each can be applied to all phylogenetic parsimony problems created so far, giving a total of 156 phylogenetic parsimony problems. One such problem is k-RUUOWL, the k-Recombination under Unweighted Unconstrained Ordered Wagner Linear parsimony problem. Note that as reticulation is always directed, the trees produced by these problems are rooted.

It is not obvious at first glance that the problems above are in NP. Conventional tree-traversal algorithms can be modified to check all parsimony criteria for both non-reticulate and reticulate trees in time polynomial in the size of the candidate solution [StaT80, Section 3], but such solutions are not guaranteed to be of size polynomial in the size of the instance because they might be as large as the implicit graph on $O(2^d)$ vertices from which they are taken. However, under

Acronym		Problem
UBW		Unweighted Binary Wagner
	UBOWL	Unweighted Binary Ordered Wagner Linear
	UBQWL	Unweighted Binary Qualitative Wagner Linear
	UBOWG	Unweighted Binary Ordered Wagner General
	UBQWG	Unweighted Binary Qualitative Wagner General
	BFi	Binary Fitch
UUW		Unweighted Unconstrained Wagner
	UUQWG	Unweighted Unconstrained Qualitative Wagner General
	UFi	Unconstrained Fitch
WBW		Weighted Binary Wagner
	WBOWL	Weighted Binary Ordered Wagner Linear
	WBQWL	Weighted Binary Qualitative Wagner Linear
	WBOWG	Weighted Binary Ordered Wagner General
	WBQWG	Weighted Binary Qualitative Wagner General
UUOWL		Unweighted Unconstrained Ordered Wagner Linear
UUQWL		Unweighted Unconstrained Qualitative Wagner Linear
WUOWL		Weighted Unconstrained Ordered Wagner Linear
WUQWL		Weighted Unconstrained Qualitative Wagner Linear
UUOWG		Unweighted Unconstrained Ordered Wagner General
WUOWG		Weighted Unconstrained Ordered Wagner General
WUQWG		Weighted Unconstrained Qualitative Wagner General

Table 6: Phylogenetic parsimony decision problems (non-reticulate trees). Each group of equivalent problems is indented, and appears after the acronym for that group.

Acronym	Problem
UBCCS	Unweighted Binary Cladistic Camin-Sokal
UBQCS	Unweighted Binary Qualitative Camin-Sokal
UBOCS	Unweighted Binary Ordered Camin-Sokal
UBQCS	Unweighted Binary Qualitative Camin-Sokal
UUCCS	Unweighted Unconstrained Cladistic Camin-Sokal
UUOCS	Unweighted Unconstrained Ordered Camin-Sokal
UUQCS	Unweighted Unconstrained Qualitative Camin-Sokal
WBCCS	Weighted Binary Cladistic Camin-Sokal
WBOCS	Weighted Binary Ordered Camin-Sokal
WBQCS	Weighted Binary Qualitative Camin-Sokal
WUCCS	Weighted Unconstrained Cladistic Camin-Sokal
WUOCS	Weighted Unconstrained Ordered Camin-Sokal
WUQCS	Weighted Unconstrained Qualitative Camin-Sokal
UBCDo	Unweighted Binary Cladistic Dollo
UBQDo	Unweighted Binary Qualitative Dollo
UBODo	Unweighted Binary Ordered Dollo
UBQDo	Unweighted Binary Qualitative Dollo
UUCDo	Unweighted Unconstrained Cladistic Dollo
UUODo	Unweighted Unconstrained Ordered Dollo
UUQDo	Unweighted Unconstrained Qualitative Dollo
WBCDo	Weighted Binary Cladistic Dollo
WBODo	Weighted Binary Ordered Dollo
WBQDo	Weighted Binary Qualitative Dollo
WUCDo	Weighted Unconstrained Cladistic Dollo
WUODo	Weighted Unconstrained Ordered Dollo
WUQDo	Weighted Unconstrained Qualitative Dollo
CCI	Cladistic Chromosome Inversion
OCI	Ordered Chromosome Inversion
QCI	Qualitative Chromosome Inversion
UBGe	Unweighted Binary Generalized
UUGe	Unweighted Unconstrained Generalized
WBGe	Weighted Binary Generalized
WUGe	Weighted Unconstrained Generalized

Table 7: Phylogenetic parsimony decision problems (non-reticulate trees) (cont'd from Table 6).

Select Hybridization Under **X** (SHX)

Instance: Same as for problem **X**, with an additional parameter R, a given polynomial-sized (in the parameters of **X**) set of 3-hyperarcs.

Question: Same as for **X**, with the additional condition that the phylogeny can include any subset of the 3-hyperarcs in R.

k-Hybridization Under **X** (k-HX)

Instance: Same as for problem **X**, except that the implicit graph incorporates a fixed type of 3-hyperarc, and there is an additional parameter k, a positive integer.

Question: Same as for **X**, with the additional condition that the phylogeny can include $\leq k$ 3-hyperarcs of the fixed type.

Select Recombination Under **X** (SRX)

Instance: Same as for problem **X**, with an additional parameter R, a given polynomial-sized (in the parameters of **X**) set of 4-hyperarcs.

Question: Same as for **X**, with the additional condition that the phylogeny can include any subset of the 4-hyperarcs in R.

k-Recombination Under **X** (k-RX)

Instance: Same as for problem **X**, except that the implicit graph incorporates a fixed type of 4-hyperarc, and there is an additional parameter k, a positive integer.

Question: Same as for **X**, with the additional condition that the phylogeny can include $\leq k$ 4-hyperarcs of the fixed type.

Table 8: Phylogenetic parsimony decision problem schemata (reticulate trees). These schemata are stated relative to a non-reticulate phylogenetic parsimony problem **X**.

certain additional restrictions, the problems defined above can be shown to be in NP. Consider the relationship between solution cost and size.

Lemma 3 *A polynomial-time nondeterministic computation is guaranteed to find all solutions Y to an instance I of an unweighted (weighted) parsimony problem X such that $b_X(Y) \leq p(|I|)$ $(b_X(Y) \leq p(|I|)W_{\min}(I))$ for some polynomial p.*

Proof: Observe that the largest solution of cost k to an instance I unweighted parsimony problem is a tree on $k+1$ vertices, and that the largest solution of cost k to an instance I of a weighted parsimony problem is a tree on $k/W_{\min}(I)$ vertices with edges of weight $W_{\min}(I)$, where $W_{\min}(I)$ $(W_{\max}(I))$ is the smallest (largest) character-transition weight in the given instance. ▮

Solutions satisfying these bounds exist for each non-reticulate phylogenetic parsimony problem defined above. For Wagner Linear, Wagner General, Fitch, Camin-Sokal, and Generalized problems, this solution is a tree rooted at the all-ancestral vertex which has paths to each taxon that use the appropriate character-state transitions to generate the states for that taxon; the solution for the Dollo (Chromosome Inversion) problem is a tree rooted at the all-ancestral (all-A) vertex that has a path to the all-derived (all-H) vertex, which then has paths to each taxon that use the appropriate "reversal" transitions to generate the states for that taxon. Each of these solutions is of size $O(mdl\,(\log W_{\max}))$ and cost $O(mdl\,(W_{\max}))$, where l is the length of the longest path between two states in any character-state graph in the instance and $W_{\max} = 1$ if the problem is unweighted. As the cost of a solution is proportional to its size in unweighted problems, any solutions (including optimal solutions) better than those given above have costs that satisfy the bound in Lemma 3.

Corollary 4 *A polynomial-time nondeterministic computation is guaranteed to find all optimal solutions of any instance of a non-reticulate unweighted phylogenetic parsimony problem.*

This relationship does not hold for weighted problems; solutions of lower cost may exist that are larger than solutions of higher cost. However, if the problem is restricted to those instances I such that $W_{\max}(I) < p(|I|)W_{\min}(I)$ for some polynomial p, any solutions (including optimal solutions) better than those given above must have cost $k \leq O(mdLW_{\max}) \leq p'(|I|)W_{\max}(I) \leq p''(|I|)W_{\min}(I)$ for some polynomials p', p'', and thus have costs that satisfy the bound in Lemma 3.

Corollary 5 *A polynomial-time nondeterministic computation is guaranteed to find all optimal solutions of any instance I of a non-reticulate weighted phylogenetic parsimony problem such that $W_{\max}(I) \leq p(|I|)W_{\min}(I)$ for some polynomial p.*

Thus, all non-reticulate unweighted and weighted phylogenetic parsimony problems defined above whose weights are so restricted are in NP. As solutions to cladistic problems are also solutions to ordered and qualitative problems, and as each reticulate problem can incorporate a number of reticulations at most polynomial in the parameters of its non-reticulate counterpart, all parsimony problems defined above are in NP.

The reductions given in [DJS86, DS87] that establish the NP-hardness of the non-reticulate unweighted binary Camin-Sokal, Dollo, and Chromosome Inversion phylogenetic parsimony problems are given in Tables 9 and 10. These reductions use the basic idea of Karp's reduction from EXACT COVER to STEINER TREE IN GRAPHS [Kar72] – namely, reduce some problem involving the selection of a subcollection of a collection of subsets on a set of items onto a three-level tree in which the leaves correspond to the items, the root to the selected subcollection, and the remaining internal vertices to the subsets in this subcollection.[2] In the reductions in Tables 9 and 10, the items are the edges of G and the subsets in the collection are the sets of edges adjacent to each of the vertices in G. The trees that are solutions in each of the reduced instances contain subtrees that have three levels ([DJS86, Lemma 1]; [DS87, Lemma 2]), where the internal vertices selected on the second level of each tree correspond to satisfying vertex covers for the original instances. In the case of the Dollo and Chromosome Inversion problems, each solution tree has a "tail" composed of the vertices in Y which ensures that the tree has a root that is consistent with its problem's criterion. Moreover, one can construct trees from satisfying vertex covers that correspond to satisfying

[2] The reduction as given in [Kar72] is flawed, as the reader can verify for items $T = \{a, b, c, d, e, f, g\}$ and collection of subsets $S = \{\{a, b, c\}, \{c, d, e\}, \{e, f, g\}\}$; the edges of weight 0 allow a solution tree of length 3 to the reduced instance, even though the original instance has no exact cover. Krentel has fixed this problem by a variant on [Kar72] that yields a reduction from SET COVER to STEINER TREE IN GRAPHS [GKR92, Theorem 3.4].

VC $\leq^p_m$ *UBCCS / UBQCS [DJS86]*

$d = |V|$,

where each character corresponds to a particular vertex $v \in V$.

$S = 0 \bigcup X$,

where 0 is the all-ancestral vertex, and X is the set of vertices corresponding to the edges in E (for $e = \{u, v\}$ in E, there are 1's in the characters corresponding to u and v and 0's elsewhere).

$B = K + |E|$

VC $\leq^p_m$ *UBCDo / UBQDo (adapted from [DJS86])*

$d = 3|V|$,

where characters $2|V| + 1$ to d correspond to the vertices in V.

$S = 0 \bigcup X \bigcup Y$,

where 0 is the all-ancestral vertex, X is the set of vertices corresponding to the edges in E, and Y is the set of vertices y_i, $1 \leq i \leq d$, such that y_i has 1's in characters 1 to i and 0's elsewhere.

$B = K + 3|V| + |E|$

Table 9: Reductions for phylogenetic parsimony decision problems.

Note that the reductions given for the Dollo and Chromosome Inversion problems differ from [DJS86, DS87] in that the $d = 3|V|$ instead of $2K + |V|$. The proofs given in [DJS86, DS87] still work for these modified reductions; moreover, these modifications simplify the transformation of these many-one reductions to metric reductions in Section 4.2.

VC $\leq_m^p$ *UCCI / UQCI (adapted from [DS87])*

$d = 3|V|$,

where characters $2|V| + 1$ to d correspond to the vertices in V.

$S = H \cup X \cup Y$,

where H is the all-H vertex, X is the set of vertices corresponding to the edges in E (for $e = \{u, v\} \in E$, there are D's in the characters corresponding to u and v and H's elsewhere), and Y is the set of vertices y_i, $1 \leq i \leq d$, such that y_i has A's in characters 1 to i and H's elsewhere.

$B = K + 3|V| + |E|$

Table 10: Reductions for phylogenetic parsimony decision problems (cont'd from Table 9).

trees for the reduced instances ([DJS86, Theorems 2 and 3]; [DS87, Theorem 3]). The trees will have the three-level structure as long as the all-ancestral (or all-H) vertex is included in S; hence, these reductions simultaneously show that the cladistic and qualitative versions of each problem are NP-hard. For the same reason, the reduction for the Camin-Sokal problems also shows that the unweighted binary Wagner problem is NP-hard [DJS86, p. 41].

The non-reticulate binary unweighted problems are restrictions of all other non-reticulate and reticulate problems (set $k = 0$ (k-**HX**,k-**RX**) and $R = \emptyset$ (S**HX**,S**RX**)); thus, all Wagner, Fitch, Camin-Sokal, Dollo, and Chromosome Inversion problems are NP-hard. As any ordered problem can be solved by an appropriately structured instance of the Generalized parsimony problem, all Generalized parsimony problems are also NP-hard. Hence, all phylogenetic parsimony problems considered above are NP-complete.

A proof that UBW and UUW are NP-complete was given previous to that in [DJS86] by Graham and Foulds [GF82], using a reduction from X3C. The elegant reduction from UUW to WUOWL given in [Day83] does not work as stated there, because Day uses a version of UFi that includes the implicit graph in the instance and this version has not been shown to be NP-complete (see Appendix B). However, with slight modifications, this reduction does work for UUW as defined above.

The phylogenetic parsimony problems described above are closely related to the STEINER TREE IN GRAPHS (STG) and RECTILINEAR STEINER TREE (RST) problems (see Table 11). The phylogenetic parsimony problems are like STG in that the solution is drawn from a graph, and like RST in that this solution domain is implicit. The relationship is not exact in either case because none of the phylogenetic parsimony problems defined above include their implicit graphs in their instances (cf. Appendix B), and only the simplest phylogenetic parsimony problems are defined on d-dimensional rectilinear spaces. Despite these differences, certain of the STG and RST solution and approximation algorithms [BR91, Ric89, Sny92, Win87] can be modified to solve particular phylogenetic parsimony problems; see Section 5.4 and Appendix B.

Steiner Tree in Graphs (STG) **[ND12]**

Instance: Graph $G = (V, E)$, subset $S \subseteq V$, positive integer $K \leq |V| - 1$.

Question: Is there a *Steiner tree* T for S in G with length $\leq K$, that is, a subtree T of G that includes all vertices in S and contains no more than K edges?

Rectilinear Steiner Tree (RST) **[ND13]**

Instance: Set $P = \{(x_1, y_1), ..., (x_n, y_n)\}$ of integer co-ordinates in the plane; positive integer L.

Question: Is there a *rectilinear Steiner tree* T with length $\leq L$, that is, a tree T composed of horizontal and vertical line segments linking the points in P such that the sum of the lengths of all line segments in that tree is no more than L?

Table 11: Steiner Tree decision problems (taken from [GJ79]).

3.2.2 Character Compatibility

Each of these problems is given as input a set of d characters defined on a set of m objects, and seeks the largest pairwise compatible subset of the given characters. The cladistic / ordered / qualitative and binary / unconstrained character restrictions made in the last section are also applicable to character compatibility problems. A collection of ordered (qualitative) characters is compatible if its character-state sets can be polarized (ordered and polarized) to make the collection a compatible set of cladistic characters [DS86, p. 225]. The five character compatibility problems so defined are given in Tables 12 and 13.

Each of these character compatibility problems is obviously in NP, as solution

Binary Cladistic Compatibility (BCC)

Instance: Set of m objects; a collection C of d binary cladistic characters, as described by a d-by-m character-by-object matrix X; and a positive integer $B \leq d$.

Question: Does the collection of characters C have a compatible collection $C' \subseteq C$ such that $|C'| \geq B$?

Binary Qualitative Compatibility (BQC)

Instance: Set of m objects; a collection C of d binary qualitative characters, as described by a d-by-m character-by-object matrix X; and a positive integer $B \leq d$.

Question: Does the collection of characters C have a polarization such that there is a compatible collection $C' \subseteq C$ such that $|C'| \geq B$?

Table 12: Character compatibility decision problems (adapted from [DS86]).

sets of characters are subsets of the given set of characters. The reductions given in [DS86] which establish that BCC and BQC are NP-hard are given in Table 14. The problems CLIQUE and BCC are very similar: both problems are looking for the subset of largest size such that a particular relation holds between every pair of elements in that subset. Let K be the character-pattern for a particular character, and $K(x)$ be the character-state in K of taxon x; for two binary characters K_i and K_j on the set of taxa S, K_i and K_j are incompatible if and only if all three of the elements (1,0), (0,1), and (1,1) are in $(K_i \times K_j)(S)$ [EJM76, Theorem 2.3]. By this result, pairs of characters in the reduced instance that correspond to vertices not joined by an edge in G are incompatible. Hence, any

Unconstrained Cladistic Compatibility (UCC)

Instance: Collection C of d cladistic characters defined on a set of m objects; a positive integer $B \leq d$.

Question: Does the collection of characters C have a compatible collection $C' \subseteq C$ such that $|C'| \geq B$?

Unconstrained Ordered Compatibility (UOC)

Instance: Collection C of d ordered characters defined on a set of m objects; a positive integer $B \leq d$.

Question: Does the collection of characters C have a polarization such that there is a compatible collection $C' \subseteq C$ such that $|C'| \geq B$?

Unconstrained Qualitative Compatibility (UQC)

Instance: Collection C of d qualitative characters defined on a set of m objects; a positive integer $B \leq d$.

Question: Does the collection of characters C have a polarization and an ordering such that there is a compatible collection $C' \subseteq C$ such that $|C'| \geq B$?

Table 13: Character compatibility decision problems (cont'd from Table 12).

CLIQUE $\leq^p_m$ *BCC [DS86]*

$d = |V|$
$m = 3d(d-1)/2$
$X = [x_{i,j}], 1 \leq i \leq d, 1 \leq j \leq m$

X has a character-column for each vertex in V, and three taxon-rows for each unordered pair of vertices in V. For each edge $\{u, v\}$ not in E, set the row-entries in column u for that edge to 011, and the row-entries in column v to 110. All other entries in X are 0.

$B = J$

BCC $\leq^p_m$ *BQC [DS86]*

$d' = d$
$m' = 2m$
$X' = [x'_{i,j}], 1 \leq i \leq d', 1 \leq j \leq m'$

where the taxa corresponding to rows $(m+1) \leq i \leq m'$ exhibit the ancestral character-states of the characters in X.

$B' = B$

BQC $\leq^p_m$ *BCC [DS86]*

$d' = d$
$m' = m$
$X' = [x'_{i,j}], 1 \leq i \leq d', 1 \leq j \leq m'$

where a character's most frequently occurring state becomes that character's ancestral state in X'.

$B' = B$

Table 14: Reductions for character compatibility decision problems.

collection of pairwise compatible characters must correspond to a set of vertices in G that form a clique [DS86, Proposition 4], completing the proof. The key to the reduction from BCC to BQC is that binary qualitative characters behave like binary cladistic characters in which the state occurring most frequently has been set to ancestral [McM77, Lemma and Theorem 1]. This can be forced by adding taxa [DS86, Proposition 2]. The reduction from BQC to BCC holds by similar reasoning [DS86, Proposition 2]. As binary characters are restrictions of unconstrained characters, problems UCC, UOC, and UQC are also NP-complete.

3.2.3 Distance Matrix Fitting

Each of these problems is given as input a semimetric on m taxa. Some problems seek either the ultrametric or additive tree that has the closest fit to this semimetric according to that problem's statistic; others seek the ultrametric or additive tree of shortest length that is dominant to this semimetric. The distance matrix fitting problems defined in [Day83, Day87, Kri88, KM86] are given in Table 15. Many of these problems were shown to be NP-complete via reductions from certain of their subproblems given in Table 16.

As with the phylogenetic parsimony problems, the distance matrix fitting problems are in NP subject to certain restrictions on instance weights. For an instance I and associated solution Y, let $W_{maxdiff}(I, bc)$ is the maximum difference between any weight in I and any weight in Y.

Fitting Unconstrained Matrices to Ultrametric Trees via Statistic $\mathbf{X}$ (FUUT[X]) [$X \in \{F_1, F_2\}$]

Instance: Set S of n taxa; semimetric $D \in M_n$; and a positive integer B.

Question: Does there exist an ultrametric tree $U \in U_n$ such that $\mathbf{X}(D, \pi_U(U)) \leq B$?

Fitting Unconstrained Matrices to Dominant Ultrametric Trees via Statistic $\mathbf{X}$ (FUUT[X,≥]) [$X \in \{F_1, F_2\}$]

Instance: Set S of n taxa; semimetric $D \in M_n$; and a positive integer B.

Question: Does there exist an ultrametric tree $U \in U_n$ such that $\mathbf{X}(D, \pi_U(U)) \leq B$ and $\pi_U(U)) \geq D$?

Fitting Unconstrained Matrices to Discretized Additive Trees via Statistic $\mathbf{X}$ (FUDT[X]) [$X \in \{F_1, F_2, F\}$]

Instance: Set S of n taxa; semimetric $D \in M_n$; and a positive integer B.

Question: Does there exist an additive tree $T \in A_n^d$ such that $\mathbf{X}(D, \pi_A(T)) \leq B$?

Fitting Unconstrained Matrices to Graph-Based Dominant Additive Trees (FUGT[≥])

Instance: Complete graph $G = (V, E)$, $|V| = n$; semimetric $D \in M_n$ defined on all pairs of vertices in G; set of taxa $S \subseteq V$; and a positive integer B.

Question: Is there a subtree T of G that includes S such that $\sum_{(x,y) \in T} D(x,y) \leq B$ and $[\pi_A(T)]_S \geq D_S$?

Table 15: Distance matrix fitting decision problems (adapted from [Day83, KM86, Day87, Kri88]).

Lemma 6 *A polynomial-time nondeterministic computation is guaranteed to find all solutions Y to an instance I of a distance matrix fitting problem X such that* $\log W_{maxdiff}(I,Y) \leq p(|I|)$, *for some polynomial p.*

Proof: Observe that all of these problems have solutions in which the number of elements in the solution is polynomial in the size of the instance, i.e. ultrametrics of size $|S|^2$ (FUUT[**X**], FUUT[**X**,≥]), trees with at most $2|S| - 1$ vertices (FUDT[**X**]), trees with at most $|G|$ vertices (FUGT[≥]). To complete the proof, observe that the costs of solutions Y to instances I of distance matrix fitting problems whose statistics are based on L_1 and L_2 are $O(|S|(W_{maxdiff}(I,Y)))$ and $O(|S|^2(W_{maxdiff}(I,Y)))$, respectively. ∎

Corollary 7 *A polynomial-time nondeterministic computation is guaranteed to find all solutions Y to an instance I of a distance matrix fitting problem X such that $b_X(Y) < O(2^{p(|I|)})$, for some polynomial p.*

Each distance matrix fitting problem defined above has solutions satisfying this bound i.e. ultrametrics with off-diagonal entries = $D_{\max}$ (FUUT[**X**], FUUT[**X**,≥]), any tree containing S such that every edge has weight $D_{\max}$ (FUDT[**X**]), any valid solution tree (FUGT[≥]). As solution size is proportional to solution cost, all optimal solutions satisfy the bound in Corollary 7.

Corollary 8 *A polynomial-time nondeterministic computation is guaranteed to find all optimal solutions of any instance of a distance matrix fitting problem.*

Fitting Binary Matrices to Ultrametric Trees of Height 2 VIA STATISTIC X (FBUT2[**X**]) [$X \in \{F_1, F_2\}$]

Instance: Set S of n taxa; semimetric $D \in B_n$; and a positive integer B.

Question: Does there exist an ultrametric tree $U \in U_{n,2}$ such that $\mathbf{X}(D, \pi_U(U)) \leq B$?

Fitting Binary Matrices to Dominant Ultrametric Trees of Height 2 VIA STATISTIC X (FBUT2[**X**,≥]) [$X \in \{F_1, F_2\}$]

Instance: Set S of n taxa; semimetric $D \in B_n$; and a positive integer B.

Question: Does there exist an ultrametric tree $U \in U_{n,2}$ such that $\mathbf{X}(D, \pi_U(U)) \leq B$ and $\pi_U(U) \geq D$?

Table 16: Auxiliary decision problems for NP-hardness proofs of distance matrix fitting decision problems (adapted from [KM86, Day87, Kri88]).

Hence, all distance matrix fitting problems defined above are in NP.

Problem FUUT[F_1] was shown to be NP-hard via a reduction from FBUT2[F_1]. A reduction which establishes that FBUT2[F_1] is NP-hard is given in Table 17. This reduction is adapted from a Turing reduction from X3C to SOL-MIN-FBUT2[F_1] given in [KM86]. An instance of X3C has a solution if and only if the graph G created by this reduction has a vertex-partition into 3-vertex triangles [KM86, Lemma 6]. It can be shown that the ultrametric of minimal cost for the reduced instance of FBUT[F_2] will always have such a partition if and only if there is an exact cover for the original instance of X3C; moreover, the maximum nonoverlapping (not necessarily exact) cover for the original instance of X3C can be easily derived from this ultrametric. An optimal tree for any instance of

FBUT2[F_1] will always have off-diagonal entries $\in \{1,2\}$ [KM86, Lemma 3]. For such an ultrametric tree $U = \{\{\{\{s_1\},\ldots,\{s_{|S|}\}\},0\},\{\{I_1,\ldots,I_r\},1\},\{\{S\},2\}\}$, let $i_\rho = |I_\rho|$ and $j_\rho = |\{\{i,j\} \in I_\rho | d_{i,j} = 1\}|$, $1 \leq \rho \leq r$. By [KM86, Lemma 4],

$$\begin{aligned} F_1(D,U) &= \sum_{\rho=1}^{r} \sum_{\{i,j\}\subset I_\rho} |d_{i,j} - 1| + \sum_{1\leq\rho'<\rho''\leq r} \sum_{i\in\rho'} \sum_{j\in\rho''} |d_{i,j} - 2| \\ &= \sum_{\rho=1}^{r} \left(\binom{i_\rho}{2} - j_\rho \right) + |\{\{i,j\} \mid d_{i,j} = 1\}| - \sum_{\rho=1}^{r} j_\rho \end{aligned} \tag{5}$$

No subpartition in the second partition of an ultrametric U that is minimal under F_1 can group together vertices from different subgraphs G_i and G_j, as the tree in which these vertices are grouped separately by subgraph would have lower cost by equation 5 above. Hence, each subpartition in the second partition must be based on vertices from the same subgraph G_α. Note that $F_1(D,U)$ is minimal when the subpartitions in the second partition specify a partition of G (and thus individual G_α) into the largest possible complete subgraphs. The reader can verify that the optimal partition of any subgraph G_α into complete subgraphs under F_1 is either into the four triangles

$$\begin{gathered} \{x_{\alpha,1}, y_{\alpha,1,1}, y_{\alpha,1,2}\}, \quad \{x_{\alpha,2}, y_{\alpha,2,1}, y_{\alpha,2,2}\}, \\ \{x_{\alpha,3}, y_{\alpha,3,1}, y_{\alpha,3,2}\}, \quad \{x_{\alpha,1,3}, y_{\alpha,2,3}, y_{\alpha,3,3}\} \end{gathered} \tag{6}$$

or the three triangles,

$$\{y_{\alpha,1,1}, y_{\alpha,3,2}, y_{\alpha,3,3}\}, \{y_{\alpha,2,3}, y_{\alpha,3,1}, y_{\alpha,2,2}\}, \{y_{\alpha,1,2}, y_{\alpha,1,3}, y_{\alpha,2,1}\} \tag{7}$$

plus single vertices drawn from the set $\{x_{\alpha,1}, x_{\alpha,2}, x_{\alpha,3}\}$, depending on whether the single vertices in this set have or have not been partitioned into G_α. Let these two sets of G_α be denoted by G_M and G_U; note that $|G_M| + |G_U| = |C|$. As single-vertex groupings do not affect the cost of U under F_1, the cost of a minimal ultrametric U is

$$\begin{aligned} F_1(D,U) &= 0 + |\{\{i,j\} \mid d_{i,j} = 1\}| - \sum_{p=1}^{r} j_p \\ &= |E| - 3(4|G_M| + 3|G_U|) \\ &= |E| - 3(|G_M| + 3|C|) \end{aligned} \tag{8}$$

A subgraph G_α is partitioned into four triangles if and only if the corresponding 3-set is in the maximal nonoverlapping cover of the original instance of X3C, i.e. the elements $\{x_{\alpha,1}, x_{\alpha,2}, x_{\alpha,3}\}$ are partitioned into that G_α. Hence, the original instance of X3C has an exact cover if and only if $|G_M| = q$, i.e. $F_1(D,U) = |E| - 3(q + 3|C|)$, completing the proof. As FBUT2[F_1] is computationally equivalent to FBUT[F_1], the corresponding problem with no restrictions on the height of the ultrametric [KM86, Lemma 2], and as FBUT[F_1] is a restriction of FUUT[F_1], these problems are also NP-hard, and thus NP-complete.

Problem FUUT[F_2] can be shown NP-complete in a similar fashion. By a variant of [KM86, Lemma 3], the optimal trees for any instance of FBUT2[F_2] will always have off-diagonal entries $\in \{1,2\}$. As $|d-p| = |d-p|^2$ when $d,p \in \{1,2\}$, $F_1(D, \pi_U(U)) = F_2(D, \pi_U(U))$ for any $D \in B_n$ and $U \in U_{n,2}$. Thus FBUT2[F_2] and FBUT2[F_2] are arithmetically equivalent. As FBUT2[F_2] and FBUT[F_2]

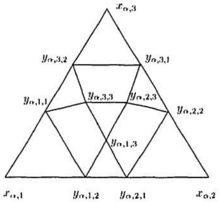


Figure 6: Structure of subgraph G_α used in reduction from X3C to FBUT2[F_1] (Figure 2 from [KM86]).

are also computationally equivalent by a variant on [KM86, Lemma 2], and as FBUT[F_2] is a restriction of FUUT[F_2], these problems are NP-complete as well.

Problem FUUT[F_1,$\geq$] was originally shown to be NP-hard via a reduction from a restricted version of VERTEX PARTITION INTO TRIANGLES [Kri88]. The NP-hardness of FBUT2[F_1, $\geq$] can also be established by a reduction from X3C analogous to that given above for FBUT[F_1], whose proof is more intuitive because dominance forces the partition of individual G_α into complete subgraphs. The latter reduction will be used in later sections of this thesis. As [KM86, Lemmas 2 and 3] can be modified to work for the corresponding dominant ultrametric problems, the reasoning above by which FUUT[**X**], **X** $\in \{F_1, F_2\}$ was shown NP-complete also shows that FUUT[**X**,$\geq$], **X** $\in \{F_1, F_2\}$, are NP-complete.

X3C $\leq^p_m$ *FBUT2[*F_1*] (adapted from [KM86])*

$n = 3(q + 3|C|)$

$S = X \bigcup \{ y_{\alpha,\beta,\gamma} \mid \alpha \in \{1, 2, \ldots, |C|\}, \beta, \gamma \in \{1, 2, 3\} \}$

$D = [d_{i,j}]$

where D is defined relative to a graph $G = (S, E)$ composed of the union of the graphs $G_\alpha = (V_\alpha, E_\alpha)$, $1 \leq \alpha \leq |C|$. Each subgraph G_α corresponds to an element of $c_\alpha \subset C$, $c_\alpha = \{x_{\alpha,1}, x_{\alpha,2}, x_{\alpha,3}\}$, $x_{\alpha,\{1,2,3\}} \in X$, and has the structure shown in Figure 6. Given G, define D as

$$\begin{array}{ll} d_{i,j} = 0 & \text{if } i = j, \\ d_{i,j} = 1 & \text{if } \{i, j\} \in E, \\ d_{i,j} = 2 & \text{otherwise.} \end{array}$$

$B = |E| - 3(q + 3|C|)$

FBUT2[$\mathbf{X}$*]* $\leq^p_m$ *FUDT[* $\mathbf{X}$*] (*$\mathbf{X} \in \{F_1, F_2\}$*) [Day87]*

$n' = n + \varphi$,

where $\varphi = \psi^2 n^4$ and $\psi = 1.5n - 1$.

$S' = S + y_i, 1 \leq i \leq \varphi$

$$D' = [d'_{i,j}] = \begin{bmatrix} D & M \\ M' & \mathbf{1} \end{bmatrix},$$

where $M = [m_{i,j}]$. $m_{i,j} = \psi$ for all $1 \leq i \leq n$ and $1 \leq j \leq \varphi$, M' is the transpose of M, and $\mathbf{1}$ is a square matrix with zeros on, but ones off, the main diagonal.

$B' = B$

Table 17: Reductions for distance matrix fitting decision problems.

The reduction given in [Day87] which establishes that FUDT[**X**], $\mathbf{X} \in \{F_1, F_2\}$ is NP-hard is given in Table 17. Day requires that $|S|$ be an even integer ≥ 4 in his version of FBUT2[**X**], which can be ensured by replicating some $c_i \in C$ in the given instance of X3C. An optimal discretized additive tree T for the reduced instance of FUDT[**X**] can be transformed in polynomial time into a tree consisting of two subtrees, an ultrametric tree U of height 2 on S attached by an edge of length 0.5 to a subtree rooted at vertex v that is attached to all vertices y_i, $1 \leq i \leq \varphi$ by edges of length 0.5, such that $\mathbf{X}(D, \pi_U(U)) = \mathbf{X}(D, \pi_A(T))$ [Day87, Proposition 3]; moreover, an optimal ultrametric for the original instance of FBUT2[**X**] can be similarly transformed into an optimal solution for the reduced instance of FUDT[**X**] [Day87, Proposition 5]. Hence, FUDT[**X**] is NP-complete, and by the arithmetic equivalence of the F_1 and F statistics, FUDT[F] is NP-complete as well.

A reduction which establishes that FUGT[$\geq$] is NP-complete is given in Table 18. This reduction is based on the reduction from VERTEX COVER to UBQCS and UBCCS given in Section 3.2.1. For graph G in an instance of FUGT[$\geq$] created by this reduction, define a *canonical tree* as a subtree T of G that contains S and is composed of edges of the types $\{*, v_i\}$ and $\{v_i, e_j\}, e_j = \{v_i, \mathbf{x}\} \in E_{VC}$.

Lemma 9 *Every instance of FUGT[$\geq$] created by the reduction in Table 18 has a minimal-length tree that is canonical.*

Proof: Let T be a minimal-length tree of length B for a reduced instance I of

$VC \leq^p_m FUGT[\geq]$

$$
\begin{aligned}
&V = \{*\} \cup \{\, v_i \mid 1 \leq i \leq |V_{VC}|\} \cup \{\, c_j \mid 1 \leq j \leq |E_{VC}|\}\\
&D = [d_{i,j}],\\
&\qquad \text{where} \begin{array}{ll}
d(*, v_i) & = 1\\
d(*, c_j) & = 2\\
d(v_i, v_j) & = 4\\
d(v_i, c_j) & = 1 \text{ if } c_j = \{v_i, x\} \in E_{VC},\\
d(v_i, c_j) & = 3 \text{ otherwise}\\
d(c_i, c_j) & = 2
\end{array}\\
&S = \{*\} \cup \{\, c_j \mid 1 \leq j \leq |E_{VC}|\}\\
&B = K + |E_{VC}|
\end{aligned}
$$

Table 18: Reductions for distance matrix fitting decision problems (cont'd from Table 17).

FUGT[≥]. If T is not a canonical tree, it contains one or more edges of the types $\{*, c_i\}$, $\{v_i, v_j\}$, $\{c_i, c_j\}$, or $\{v_i, c_j\}$ such that v_i is not a vertex of c_j. Create tree T' from T by replacing each non-canonical edge X as follows:

1. $X = \{*, c_i\}$: If there is a vertex $v_j \in T$ such that $c_i = \{v_j, z\} \in E_{VC}$, replace X by the edge $\{v_j, c_i\}$; else, replace X by the edges $\{*, v_k\}$ and $\{v_k, c_i\}$ such that $c_i = \{v_k, z\} \in E_{VC}$. The former case cannot occur because it would create a tree with a length $B' < B$; the latter case produces a tree T' of equal length.

2. $X = \{v_i, v_j\}$: Assume without loss of generality that there is already an edge $\{*, v_i\}$ or $\{*, v_j\}$ in T, and replace X by the edge of this pair that is not in T. This cannot occur because it would create a tree with a length

$B' < B$.

3. $X = \{e_i, e_j\}$: Assume without loss of generality that there is an edge $\{v_k, e_i\}$ in T. If there is a vertex $v_l \in T$ such that $e_j = \{v_l, \mathbf{z}\} \in E_{VC}$, replace X by the edge $\{v_l, e_j\}$; else, replace X by the edges $\{*, v_l\}$ and $\{v_l, e_j\}$ such that $e_j = \{v_l, \mathbf{z}\} \in E_{VC}$. The former case cannot occur because it would create a tree with a length $B' < B$; the latter case produces a tree T' of equal length.

4. $X = \{v_i, e_j\}$ *such that* v_i *is not a vertex of* e_j: If there exists a vertex v_k in T such that $e_j = \{v_k, \mathbf{z}\} \in E_{VC}$, replace X by edge $\{v_k, e_j\}$ to T; else, replace X by the pair of edges $\{*, v_k\}$ and $\{v_k, e_j\}$. Neither case can occur because each would create a tree with length $B' < B$.

The created tree T' has the same length as T and still connects all vertices in S; moreover, as T' contains no non-canonical edges, it is a canonical tree. ▌

Canonical trees have several useful properties. The path lengths of a canonical tree T are such that $[\pi_A(T)]_S \geq D_S$. Moreover, the vertices in the second level of a canonical tree T correspond to a satisfying vertex cover for the original instance.

Theorem 10 *FUGT[≥] is NP-complete.*

Proof: By Corollary 8, FUGT[≥] is in NP. Consider the reduction from VERTEX COVER to FUGT[≥] given in Table 18. This reduction is polynomial time. Moreover, optimal solutions for an original instance of VERTEX COVER and its reduced instance of FUGT[≥] can be created from each other. If the original instance of VERTEX COVER has a satisfying vertex cover $V^* \subseteq V_{VC}$ of size $K' \leq K$, construct the canonical tree linking the vertices of V^* with the vertices $\{e_j\}$ and $*$; this tree has length $K' + |E_{VC}| \leq B$, and is thus a solution to the reduced instance. If the reduced instance of FUGT[≥] has a solution tree T of length $B \leq K + |E_{VC}|$, construct the canonical tree T'' corresponding to T of length $B' \leq B$. The vertex cover V^* defined by the vertices in the second level of T' has size $|V^*| = B' - |E_{VC}| \leq B - |E_{VC}| = K$,; thus, V^* is a satisfying vertex cover for the original instance. ▮

A Turing reduction from SOL-MIN-FBUT[F_1] to SOL-MIN-FBUT2[F_2] is given in [Kri86, Theorem 4]; however, unlike the reduction from X3C to SOL-MIN-FBUT2[F_1] given in [KM86], it is not obvious how to convert this Turing reduction into a many-one reduction. Several problems that involve fitting semimetrics to dominant and subdominant ultrametrics using statistics L_1, L_2, and L_∞ are shown to be solvable in polynomial time in [Kri86, Kri88]; related problems involving other statistics are examined in [Day92]. The reduction from UUW to FUGT[≥] given in [Day83] does not work for the same reasons as Day's reduction from UUW to WUOWL (see Section 3.2.1); however, the former reduction cannot be fixed by using the implicit-graph version of UUW defined above because the reduction uses an intermediate problem (CONSENSUS PROBLEM IN CLASSIFICATION) which requires that the implicit graph be included in the problem instance.

3.2.4 Summary

Figures 7 and 8 show the various reductions described in this section, and Table 19 gives the correspondence between problems in these figures and problems described in the literature. Note that all but ten of these reductions are either by restriction or by arithmetic equivalence. All of these problems are inter-reducible by virtue of being NP-complete; however, the pattern of reductions in this diagram will be significant in later sections of this thesis. Note that each of these reductions require only that a solution exist that has a given cost, not that a solution have a cost above or below a given limit; moreover, the proofs of each of these reductions $\Pi \leq_m^c \Pi'$ give algorithms for converting solutions of cost c to instances of Π into solutions of cost c' for reduced instances of Π' and vice versa, such that these c and c' are related arithmetically. The former property, along with the reductions for CLIQUE and VERTEX COVER given in [GJ79, Section 3.1], establishes that all corresponding given-cost phylogenetic inference decision problems are NP-complete. Both of these properties will also be useful in later sections of this thesis.

The decision problems given in this section do not answer questions typically asked by systematic biologists, and are thus not relevant in themselves. However, the NP-completeness results for these problems do suggest that fast i.e. polynomial time algorithms do not exist for these problems, and that efforts should be

$$\begin{tikzcd}
\text{CLIQUE} \arrow[r] & \text{BCC} \arrow[r, "\text{r}"] \arrow[d, leftrightarrow] & \text{UCC} \\
& \text{BQC} \arrow[r, "\text{r}"] \arrow[rd, "\text{r}"] & \text{UOC} \\
& & \text{UQC}
\end{tikzcd}$$

$$\begin{tikzcd}
& & \text{FUDT}[F_1] \arrow[r, leftrightarrow, "\text{a}"] & \text{FUDT}[F] \\
\text{X3C} \arrow[r] \arrow[dddr] & \text{FBUT2}[F_1] \arrow[ur] \arrow[r, "\text{r}"] \arrow[d, leftrightarrow, "\text{a}"] & \text{FBUT}[F_1] \arrow[r, "\text{r}"] & \text{FUUT}[F_1] \\
& \text{FBUT2}[F_2] \arrow[r, "\text{r}"] \arrow[dr] & \text{FBUT}[F_2] \arrow[r, "\text{r}"] & \text{FUUT}[F_2] \\
& & \text{FUDT}[F_2] & \\
& \text{FBUT2}[F_1, \geq] \arrow[r, "\text{r}"] \arrow[d, leftrightarrow, "\text{a}"] & \text{FBUT}[F_1, \geq] \arrow[r, "\text{r}"] & \text{FUUT}[F_1, \geq] \\
& \text{FBUT2}[F_2, \geq] \arrow[r, "\text{r}"] & \text{FBUT}[F_2, \geq] \arrow[r, "\text{r}"] & \text{FUUT}[F_2, \geq]
\end{tikzcd}$$

$$\begin{tikzcd}
& \text{FGUT}[\geq] & \\
& \text{UB\{C,Q\}CS, UBW} \arrow[dr, "\text{r}"] & \\
\text{VERTEX COVER} \arrow[uur] \arrow[ur] \arrow[r] \arrow[dr] & \text{UB\{C,Q\}DI} \arrow[r, "\text{r}"] & \text{UBGe} \\
& \text{UB\{C,Q\}CI} \arrow[ur, "\text{r}"'] &
\end{tikzcd}$$

Figure 7: Reductions among phylogenetic inference decision problems. Reductions $\Pi \leq_m^p \Pi'$ are denoted by arrows from Π to Π'. Arrows marked by a and r correspond to reductions by arithmetic equivalence and restriction, respectively.

	Problem	
	Thesis	Literature
Phylogenetic Parsimony	UB{C,Q}CS	{C,Q}CS [DJS86]
	UB{C,Q}Do	{C,Q}DO [DJS86]
	UB{C,Q}CI	{C,Q}CI [DJS86]
	UBW	SPQ [GF82, Day83]
	UUW	SPP [GF82, Day83]
	WUOWL	WTP [Day83]
Character Compatibility	B{Q,C}C	B{Q,C}C [DS86]
	U{Q,C}C	U{Q,C}C [DS86]
Distance Matrix Fitting	FBUT[F_1]	^{b}HIC† [KM86]
	FBUT2[F_1]	^{b}HIC$_3$† [KM86], Δ_1^2† [Kri86], FUT[1] [Day87]
	FBUT2[F_2]	Δ_2^2† [Kri86], FUT[2] [Day87]
	FUUT[F_1]	Δ_1† [Kri86], HIC† [KM86]
	FUUT[F_2]	Δ_2† [Kri86]
	FUUT[$F_1, \geq$]	P4 [Kri88]
	FUDT[α], $\alpha \in \{F_1, F_2\}$	FAT[α], $\alpha \in \{1, 2\}$ [Day87]
	FUGT[$\geq$]	AET [Day83]

Table 19: Correspondence between phylogenetic inference problems in this thesis and problems in the literature. All solution problems are marked with daggers (†); all other problems are decision problems.

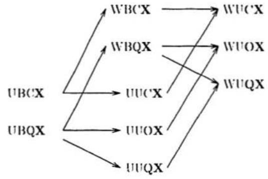


Figure 8: Restriction reductions among phylogenetic parsimony decision problems. These problems are stated relative to a phylogenetic parsimony criterion **X**. Note that each problem above is also linked by restriction reductions to each of its four corresponding reticulate problems (see Table 8).

focused on the design of polynomial-time approximation algorithms which guarantee solutions that are close to optimal [Day83, GF82]. These reductions can also be used to determine the computational hardness of more complex problems (Section 4) and to place limits on the kinds of polynomial-time approximations that can exist for phylogenetic inference problems (Section 5).

4 The Computational Complexity of Phylogenetic Inference Functions

In the last section, certain decision problems associated with various phylogenetic inference criteria were shown to be NP-complete. By a folklore result in theoretical computer science, each of the corresponding solution problems is solved by a function in FP^{NP} [GJ79, Chapter 5]. However, this says little about the hardness of more complex problems based on these criteria. In this section, I will derive various bounds on the complexities of the evaluation, solution, spanning, enumeration, and random-generation functions associated with the optimal-cost, given-cost, and given-limit versions of the phylogenetic inference problems.

The reader should remember that results given below for phylogenetic parsimony and distance matrix fitting given-cost and given-limit problems apply only to those problems in which the cost-parameter k satisfies the restrictions given in Lemma 3 and Corollary 7.

It will often be convenient below to have a single binary encoded representation (***canonical representation***) of each solution to a problem, so that individual solutions are not output more than once. Such representations exist for all problems examined in this thesis:

- *Character Compatibility*: Represent characters by character-state adjacency matrices whose character-states are in instance input order, and represent sets of characters by such matrices in instance input order.

- *Phylogenetic Parsimony*: Represent characters as above, and represent trees by vertex-adjacency matrices whose vertices are in lexicographic order relative to character-state instance input order. Reticulations are stored in a separate list by source-vertex set lexicographic order relative to character-state instance order.
- *Distance Matrix Fitting*: Represent ultrametrics by their corresponding ultrametric matrices whose vertices are in input instance order. Represent additive trees by their inorder traversal sequences [StaT80, Section 3], where the tree is rooted at the least vertex in input instance order and the left-right ordering of subtrees is replaced by an ordering on the basis of the least vertex in a subtree.

These canonical representations can be encoded and decoded in polynomial time using slightly modified standard algorithms [StaT80, Section 3]. All TM solving problems examined in this section will be assumed to operate on and to output canonical representations.

4.1 Function Complexity Classes

This section will give a brief overview of some function classes that will be used below. These classes fall mainly into two regions – within FP^{NP} and within

FPSPACE(poly). The relations between all classes defined in this section are shown in Figures 9 and 10.

4.1.1 Classes Within FP^{NP}

There are two hierarchies of interest within FP^{NP}:

1. The function bounded NP query hierarchy, $FP^{NP[f(n)]}$.

2. The OptP$[f(n)]$ hierarchy, where f is smooth and $f \in O(poly)$ [GKR92, Kre88]: For polynomial-time NTM $N \in FNP_g$, let $opt^N(x)$ be the optimal value (largest for a maximization problem, smallest for a minimization problem) computed by N for input x.

Definition 11 (adapted from [Kre88], p. 493) *A function $f : \Sigma^* \rightarrow \mathcal{Z}$ is in OptP (optimization polynomial time) if there is a polynomial-time NTM $N \in FNP_g$ such that $f(x) = opt^N(x)$ for all $x \in \Sigma^*$. We say that f is in OptP$[z(n)]$ if $f \in$ OptP and the length of $f(x)$ in binary is bounded by $z(|x|)$ for all $x \in \Sigma^*$.*

Though all OptP$[f(n)]$ functions are contained in $FP^{NP[f(n)]}$, each function in $FP^{NP[f(n)]}$ metrically reduces to some OptP$[f(n)]$ function [Kre88, Theorem 3.2(i)]. Thus, all OptP$[f(n)]$-complete functions are also $FP^{NP[f(n)]}$-complete.

Note that metric reductions can stretch input by a polynomial amount. One consequence of this stretching is that a function that is complete for OptP[$f(n)$] is also complete for OptP[$f(n^{O(1)})$] [GKR92, p. 7]. It is most noticeable in the names for certain classes defined using big-O notation e.g. OptP[$O(\log\log n)$] is more properly written as OptP[$c\log\log n + O(1)$].

The following class relations are known:

- For every smooth function f,
  - For $f(n) \leq \frac{1}{2}\log n$, $FP^{NP[f(n)-1]} \subset FP^{NP[f(n)]}$ unless P = NP [Kre88, Theorem 4.2].
  - For $f(n) \leq (1-\epsilon)\log n$, $\epsilon \in (0,1]$, $FP^{NP[f(n)-1]} \subset FP^{NP[f(n)]}$ unless P = NP [Bei88, Theorem 21].
  - For $f(n) \in O(\log n)$, $FP^{NP[f(n)-1]} \subset FP^{NP[f(n)]}$ unless $\Sigma_3^p = \Pi_3^p$ [ABG91, Theorem 42].
- $FP^{NP[O(\log n)]} \subset FP^{NP}$ unless P = NP [Kre88, Theorem 4.1].
- $FP^{NP[O(\log n)]} \subset FP_{||}^{NP}$ unless R = NP [Sel91, Theorem 12] and FewP = P [Sel91, Corollary 4(i)].
- $FP_{||}^{NP} \subset FP^{NP}$ if and only if $P^{NP[O(\log n)]} \subset P^{NP}$ [Sel91, Theorem 1].

Other separations hold under more exotic assumptions [Bei88, Bei91]. Research to date has focused on all classes below $FP^{NP[O(\log n)]}$ and the class FP^{NP}.

Though many results have been imported directly from language-based to function-based classes, there have been some notable surprises, in particular the non-equivalence of $FP_{\|}^{NP}$ and $FP^{NP[O(\log n)]}$ and the separation of $FP^{NP[O(\log n)]}$, $FP_{\|}^{NP}$, and FP^{NP}.

Metric reducibility suffices to show hardness for most single-valued function classes. Functions are shown $FP_{\|}^{NP}$-hard via the property of *paddability* [CT91, Gas86]. Recall that all problems are based on relations $R : I \times S$ on instances I and solutions S. Let SOL-$\Pi(x)$ be the set of solutions for an instance x of a problem Π.

Definition 12 ([CT91], Definition 4.2) *A problem* Π *is* paddable *if there is a pair of polynomial-time functions* $h_1 : 2^I \to I$ *and* $h_2 : 2^I \times S \to 2^S$ *such that for all finite sets* $\{x_1, x_2, \ldots, x_m\} \in 2^I$ *and all single-valued functions* f *that solve* Π, *if* $x = h_1(\langle x_1, x_2, \ldots, x_m\rangle)$ *then* $h_2(\langle x_1, x_2, \ldots, x_m\rangle, f(x)) = \langle y_1, y_2, \ldots, y_m\rangle$, *where* $y_i \in SOL\text{-}\Pi(x_i)$ *for* $1 \leq i \leq m$.

Paddability was defined implicitly in [Gas86]. Gasarch realized that if instances of a paddable problem Π can encode an X-hard problem then Π is $FP_{\|}^{X}$- hard. If X = NP, paddable problems are $FP^{NP[O(\log n)]}$-hard [Gas86, Theorem 8] (the essential idea is that a binary $O(\log n)$-depth NP query tree contains at most a polynomial number of NP queries). Chen and Toda defined and named paddability independent of Gasarch's work, and stated their results in terms of $FP_{\|}^{NP}$-hardness.

Theorem 13 ([CT91], Lemma 4.1) *Let* Π *be a paddable problem whose associated decision problem* L_Π *is NP-hard. Then* Π_f *is* $FP_{||}^{NP}$*-hard.*

Proof: (sketch): Define function $Q_\Pi(x_1, x_2, \ldots, x_m) = \chi_{L_\Pi}(x_1)\chi_{L_\Pi}(x_2)\ldots\chi_{L_\Pi}(x_m)$, where $x_1, x_2, \ldots, x_m$ are instances of L_Π. As L_Π is NP-hard, any function in $FP_{||}^{NP}$ can be solved using a single call to Q_Π; however, as Π is paddable, any instance of Q_Π can be solved using a single call to any solution function for Π. ▮

Their interpretation is the more powerful in light of Selman's results showing that $FP_{||}^{NP}$ is intermediate in hardness between $FP^{NP[O(\log n)]}$ and FP^{NP} (see above). The following variant of paddability will be useful below:

Definition 14 *A problem* Π *is* paddable with respect to a problem Π' *if there is a pair of polynomial-time functions* $h_1 : 2^{I'} \rightarrow I$ *and* $h_2 : 2^{I'} \times S \rightarrow 2^{S'}$ *such that for all finite sets* $\{x'_1, x'_2, \ldots, x'_m\} \in 2^{I'}$ *and all single-valued functions* f *that solve* Π, *if* $x = h_1(\langle x'_1, x'_2, \ldots, x'_m\rangle)$ *then* $h_2(\langle x'_1, x'_2, \ldots, x'_m\rangle, f(x)) = \langle y'_1, y'_2, \ldots, y'_m\rangle$, *where* $y'_i \in SOL\text{-}\Pi'(x'_i)$ *for* $1 \leq i \leq m$.

Theorem 15 *If problem* Π *is paddable with respect to an NP-hard problem* Π' *then* Π_f *is* $FP_{||}^{NP}$*-hard.*

Proof: The proofs by which the analogous result holds for paddable problems ([Gas86, Theorem 8]; Theorem 13 above) require only that Q_Π be based on some NP-hard problem, which need not be L_Π. ▮

Theorem 16 *If problem Π metrically reduces to problem Π' and problem Π is paddable with respect to a problem Π'', then problem Π' is paddable with respect to Π''.*

Proof: By definition, there exist polynomial-time functions h_1 and h_2 such that for $x = \{x_1, \ldots, x_m\} \in 2^I$, $y = \{y_1, \ldots, y_m\} \in 2^S$, and any single-valued function f that solves Π, $\langle y \rangle = h_2(\langle x \rangle, f(h_1(\langle x \rangle)))$, and functions T_1, T_2, such that for every single-valued function g that solves Π', there exists a function f that solves Π such that $f(x) = T_2(x, g(T_1(x)))$. Define functions $h'_1 : 2^I \rightarrow I_g$ as $h'_1(\langle x \rangle) = T_1(h_1(\langle x \rangle))$ and $h'_2 : 2^I \times S_g \rightarrow 2^S$ as $h'_2(\langle x \rangle, y) = h_2(\langle x \rangle, T_2(T_1(h_1(\langle x \rangle)), y))$ such that $\langle y \rangle = h'_2(\langle x \rangle, g(h'_1(\langle x \rangle)))$. Functions h'_1 and h'_2 are polynomial time and show that Π' is paddable with respect to Π''. ▮

Note that paddability as defined here is distinct from paddability as traditionally defined in computational complexity theory ([BDG88, pp. 74–75]; [BDG90, pp. 122–123]).

Four classes of multivalued functions will also be used below:

- $NPMV = FNP$ and $NPMV_g = FNP_g$ [Sel91].
- $NPMV \circ FP^{NP}$ is the set of all partial, multivalued functions that are computed by polynomial-time NTM transducers that are allowed to ask up to a polynomial number of adaptive NP queries before nondeterminism is invoked.

- $NPMV_g \circ FP^{NP}$ is the set of all functions $f \in NPMV \circ FP^{NP}$ such that the nondeterministic phase of the computation is restricted to $NPMV_g$.

- $(NPMV \circ FP^{NP})_g$ is the set of all functions $f \in NPMV \circ FP^{NP}$ such that graph$(f) \in P$.

The NPMV-composition class notation is adapted from that in [FHOS92]. Valiant noticed that all solution problems associated with NP decision problems are in $NPMV_g$ ([Sel91, p. 4]; [Val76]). Class $NPMV_g \circ FP^{NP}$ is useful because it corresponds to those solution problems whose associated decision and evaluation problems are in NP and OptP, respectively. The following class relations are known:

Lemma 17 ([Sel91], Proposition 7) *If $f \in FP^{NP[O(\log n)]}$ and graph$(f) \in P$ then $f \in P$.*

Proof: Implicit in the proof of [Kre88, Theorem 4.1]. ∎

Corollary 18 *The following hold:*

1. *$(NPMV \circ FP^{NP})_g = NPMV_g$.*

2. *$NPMV_g \subseteq NPMV$, $NPMV_g \circ FP^{NP} \subseteq NPMV \circ FP^{NP}$, $NPMV_g \subseteq NPMV_g \circ FP^{NP}$, and $NPMV \subseteq NPMV \circ FP^{NP}$.*

3. *$NPMV \subseteq_c FP^{NP}$ [Sel91, p. 10].*

4. $FP^{NP[O(\log n)]} \subset NPMV$ *if and only if NP = co-NP [Sel91, Theorem 4, Part 12].*

5. $FP^{NP[O(\log n)]} \subset NPMV_g$ *if and only if* $P = NP$ *[Sel91, Theorem 3, Part 27].*

6. $NPMV_g \subseteq_c FP^{NP[O(\log n)]}$ *if and only if* $P = NP$ *[Sel91, Theorem 3, Part 16].*

7. $FP^{NP} =_c NPMV_g \circ FP^{NP}$.

8. $NPMV_g \circ FP^{NP} \subset NPMV$ *if and only if NP = co-NP.*

Proof:

Proof of (1): The leftwards inclusion is trivial. The rightwards inclusion follows by this simulation: for any machine M corresponding to a function f in $(NPMV \circ FP^{NP})_g$, nondeterministically guess all possible sequences of NP query answers, compute nondeterministically relative to these queries, and accept a computed output if it is is valid (which can be checked in polynomial time, as graph$(f) \in P$).

Proof of (3): Follows from the prefix-search technique (see Section 4.3).

Proof of (4): The leftwards implication follows from the collapse of the Polynomial Hierarchy. The rightwards implication follows because the characteristic function for any language in co-NP is in $FP^{NP[1]}$.

Proof of (5): The leftwards implication follows from the collapse of the Polynomial Hierarchy. The rightwards implication follows from Lemma 17.

Proof of (6): Similar to proof for (5).

Proof of (7): Follows from definitions and proof for (3).

Proof of (8): Follows from parts (4) and (7). ∎

The major relations that are still open are $NPMV_g \subseteq_c FP^{NP}_{||}$ [Sel91, p. 23], $NPMV \subseteq NPMV_g \circ FP^{NP}$, and $NPMV \circ FP^{NP} \subset NPMV_g \circ FP^{NP}$.

Note that any optimal-cost solution problem can be simulated by asking the number of NP queries required to determine the optimal cost (see Section 4.2) and then using this cost as the input to the corresponding given-cost problem. Hence, if enough computational power is available, any function in $NPMV_g \circ FP^{NP}$ can be reduced to a function in $NPMV_g$ i.e. the given-cost solution problem. This simulation will be used in many of the proofs given below.

4.1.2 Classes Within FPSPACE(poly): Counting Classes

The classes of interest within FPSPACE(poly) belong to three hierarchies of counting classes, which are based on two different modes of counting solutions. For a polynomial-time NTM transducer $N \in FMPH$, let $\#^N(x)$ be the number of accepting paths i.e. the total number of solutions, and $\mathrm{Span}^N(x)$ be the number of different solutions computed by N on input x.

$$\begin{tikzcd}
 & & NPMV \circ FP^{NP} & \\
 & FP^{NP} \arrow[ur] \arrow[rr, leftrightarrow, "c"] & & NPMV_g \circ FP^{NP} \arrow[ul] \\
FP^{NP}_{\|} \arrow[ur] & & NPMV \arrow[ul, "c"] \arrow[ur, "c"'] \arrow[uu] & \\
FP^{NP[O(\log n)]} \arrow[u] & & & NPMV_g = (NPMV \circ FP^{NP})_g \arrow[ul] \arrow[uu] \\
 & FP \arrow[ul] \arrow[urr] & &
\end{tikzcd}$$

Figure 9: Function classes within FP^{NP} (adapted from Figure 1 of [Sel91]). Inclusion relations are denoted by unmarked arrows and refinement relations by arrows marked with c. Certain relationships that are not marked are possible; see main text.

1. The #-Hierarchy, #PH, whose kth level, $\#(F\Sigma_k^p)$, $k \geq 1$, is the class of functions $f(x)$ such that $f(x) = \#^N(x)$ for some $N \in F\Sigma_k^p$. This hierarchy is equivalent to that defined in [Val79b] on classes in PH instead of FPH.

2. The Span-Hierarchy, SpanPH [KST89], whose kth level, $\mathrm{Span}(F\Sigma_k^p)$, $k \geq 1$, is the class of functions $f(x)$ such that $f(x) = Span^N(x)$ for some $N \in F\Sigma_k^p$.

3. The function bounded #P query hierarchy, $FP^{\#P[f(n)]}$.

Let the first and second levels of #PH (SpanPH) be written #P and #NP (SpanP and SpanNP), and define hardness of functions in the classes of these hierarchies relative to metric reducibility. The following class relations are known:

Corollary 19 *The following hold:*

1. *#PH, SpanPH, and $FP^{\#P} \subseteq$ FPSPACE(poly).*

2. *$FPH \subseteq FP^{\#P[1]}$ [TW92, Theorem 5.1].*

3. *If either $\#P \subseteq FPH$ or $FPH \subseteq \#P$ then PH collapses to a finite level [TW92, Corollary 5.7, Part 1].*

4. *$\#PH \subseteq FP^{\#P[1]}$ [TW92, Theorem 4.1].*

5. *For $k \geq 1$, $\#(F\Sigma_k^p) \subseteq \mathrm{Span}(F\Sigma_k^p)$ [KST89, Generalization of Proposition 4.7]*

6. *For $k \geq 1$, $\mathrm{Span}(F\Sigma_k^p) \subseteq \#(F\Sigma_{k+1}^p)$ [KST89, Generalization of Proposition 4.8].*

7. *For $k \geq 1$. $\#(F\Sigma_k^p) = \mathrm{Span}(F\Sigma_k^p)$ if and only if $U\Sigma_k^p = \Sigma_k^p$ [KST89. Generalization of Theorem 4.9].*

8. *For $k \geq 1$. $\mathrm{Span}(F\Sigma_k^p) = \#(F\Sigma_{k+1}^p)$ if and only if $\Sigma_k^p = \Pi_k^p$ [KST89. Generalization of Theorem 4.11].*

9. *$\#PH = SpanPH$.*

Proof:

Proof of (1): Any polynomial-time NTM acceptor or transducer can be simulated in PSPACE [BDG88, Theorem 2.8(h)]; hence, by reserving space for constant $k+1$ such simulations, any $F\Sigma_k^p$ computation can be simulated in FPSPACE. A counter of accepting paths can be attached to any such simulation to calculate $\#^N(x)$; to calculate $\mathrm{Span}^N(x)$, count only those accepting paths whose output values have not been encountered before in the simulation i.e. resimulate N up to the current accepting path. As the output of any function in these hierarchies is polynomially bounded in the length of the input, these hierarchies are in FPSPACE(poly).

Proof of (5 – 8): The proof for (5) is a straightforward modification of that in [KST89]. Lemma 4.3 in [KST89] can be restated in terms of oracles $A, A' \in \Sigma_k^p$ if line 4 in the algorithm on page 367 is deleted, and the condition "or $(\exists i \langle q_i, z_i \rangle \notin B)$" is added to the definition of ORACLE on page 368. Using this lemma, it is easy to prove generalized versions of Proposition 4.5 and Corollary 4.6 in [KST89], from which (6), (7), and (8) follow. As $\Sigma_k^p = \Pi_k^p$ implies that $\#\mathrm{PH} = \#(F\Sigma_{k+1}^p)$, the leftwards portion of (8) implies the stronger result that Span($F\Sigma_k^p$) = #PH [Kob92].

Proof of (9): Follows from (5) and (6). ■

The counting functions of interest in this thesis are all in class Span($NPMV_g \circ FP^{NP}$), which is in the low end of SpanPH. The following class relations are known:

Corollary 20 *The following hold:*

1. *SpanP* $\subseteq$ Span($NPMV \circ FP^{NP}$).

2. Span($NPMV_g \circ FP^{NP}$) $\subseteq$ Span($NPMV \circ FP^{NP}$).

3. Span($NPMV \circ FP^{NP}$) $\subseteq$ *#NP.*

4. Span($NPMV_g \circ FP^{NP}$) $\subseteq$ *SpanP if and only if NP = co-NP.*

Proof:

Proofs of (1 - 2): By definition. As SpanP = Span($NPMV$), relation SpanP $\subseteq$ Span($NPMV_g \circ FP^{NP}$) is open in part because relation $NPMV \subseteq NPMV_g \circ FP^{NP}$ is open (see Section 4.1.1).

Proof of (3): Consider a NOTM M which computes a function $f \in NPMV \circ FP^{NP}$, and let M_{NPMV} be the machine in $NPMV$ invoked in the second phase of the computation of M. Define the following oracle on input x and output y for M_{NPMV}:

$\mathbf{A}(x, y)$ = {There is a computation path of M_{NPMV} on input x that produces output y}.

Oracle A is in NP. Consider the NOTM M' which computes a function g in $F\Sigma_2^p$: M' guesses an output y of M_{NPMV}, performs the initial FP^{NP} phase of the computation of M, formulates input x to M_{NPMV}, and uses a single call to oracle A to see if M_{NPMV} on input x outputs y. If the answer to oracle A is "yes", M' outputs y; else, M' rejects. Each distinct output of M is produced by M' exactly once; hence, Span(M) = #(M').

Proof of (4): The proof of the rightwards part is a variant of that for the rightwards part of [KST89, Theorem 4.11]. Let L be a language in NP. Define machine M in $NPMV_g \circ FP^{NP}$ which asks a single question to the oracle in NP for membership in L, and outputs " 1" on all computation paths if the oracle rejects i.e. input $x \notin L$, and otherwise has no accepting computation. Let f = Span(M); note that $f(x) > 0$ if and only if $x \notin L$. However, by hypothesis, f is also the Span function of some machine in $NPMV$. As this machine computes co-L, $L \in NP$ and NP = co-NP. To prove the leftwards part, note that if NP = co-NP, then SpanP = #NP by Part (8) of Corollary 19 above; the wanted result then follows from (1), (2), and (3).

▮

Note that by the results of Corollary 19, even though #P is contained in SpanP, the two are of equal computational hardness, i.e. every function in SpanP

$$\begin{tikzcd}
 & FPSPACE(poly) & \\
 & FP^{\#P} \arrow[u] & \\
 & FP^{\#P[2]} \arrow[u, dashed] & \\
 & FP^{\#P[1]} \arrow[u] & \\
FPH \arrow[r] \arrow[ur] & \#PH \arrow[r, leftrightarrow] & SpanPH \arrow[ul] \\
FP^{\Sigma_1^p} \arrow[u, dashed] \arrow[r] & \#NP \arrow[u, dashed] \arrow[r] & SpanNP \arrow[u, dashed] \arrow[ul] \\
FP \arrow[u] \arrow[r] & \#P \arrow[u] \arrow[r] & SpanP \arrow[u] \arrow[ul]
\end{tikzcd}$$

Figure 10: Function classes within FPSPACE(poly): counting classes.

metrically reduces to a function in #P; this parallels the relationship between OptP$[f(n)]$ and $FP^{NP[f(n)]}$.

4.2 Evaluation Functions

Much of the early work on evaluation problems focused on decision problems that approximate evaluation problems; see [WagK87, WagK88, WagK90] for a review of this work. Two approaches to directly determining the complexity of evaluation problems involve using paddability and the OptP hierarchy (see Section 4.1.1). Upper bounds on problem complexity within the function bounded NP query hierarchy are easily established using the OptP hierarchy. As many-one reductions often correspond to metric reductions, the same also holds for completeness results; indeed, Gasarch, Krentel, and Rappoport [GKR92, p. 1] conjecture that OptP[$f(n)$]-completeness is the normal behavior of evaluation problems corresponding to NP-complete decision problems. However, paddability is still useful in those cases when the transformation from many-one to metric reduction is not obvious.

Consider upper bounds on the complexity of the evaluation problems for the phylogenetic inference criteria examined in this thesis. By Corollaries 4, 5, and 8, all character compatibility problems and unweighted phylogenetic parsimony and distance matrix fitting problems have optimal costs that are polynomially bounded, and that all weighted problems have optimal costs that are exponentially bounded.

Corollary 21 *All character compatibility and unweighted phylogenetic parsimony and distance matrix fitting evaluation problems examined in this thesis are in OptP[$O(\log n)$]. All weighted phylogenetic parsimony and distance matrix fitting evaluation problems examined in this thesis are in OptP.*

By definition, weighted problems in which the magnitude of the largest weight is polynomially bounded are also in OptP[$O(\log n)$]; hence, let "unweighted" also refer to such problems. By results from [Kre88] cited above, one can read "OptP[$f(n)$]" as "$FP^{NP[f(n)]}$" in the remainder of this section.

Consider now completeness results, starting with the unweighted problems. MAX-CLIQUE and MIN-VERTEX COVER are both OptP[$O(\log n)$]-complete ([Kre88, Theorem 2.2]; [GKR92, Theorem 3.3]). Define MAX-X3C as the size of the largest non-overlapping, rather than exact, cover by a subset of the given 3-sets. As X3C is a generalization 3DM [GJ79, p. 53], MAX-X3C is a generalization of MAX-3DM; as MAX-3DM is OptP[$O(\log n)$]-complete [GKR92, Theorem 3.5], so is MAX-X3C. The reductions from these problems to character compatibility and unweighted phylogenetic parsimony and distance matrix fitting problems given in Sections 3.2.1, 3.2.2, and 3.2.3 give arithmetic relations between the costs of optimal solutions, and thus yield the following metric reductions:

- MIN-VERTEX COVER(x) = MIN-**X**(x) $-|E|$

  (**X** $\in$ UBCCS, UBQCS, UBW,UBGe)

- MIN-VERTEX COVER(x) = MIN-**X**(x) $-(3|V| + |E|)$

  (**X** $\in$ UBCDo, UBQDo, UCCI, UQCI)

- MAX-CLIQUE(x) = MAX-BCC(x)

- MAX-BCC(x) = MAX-BQC(x)

- MAX-X3C(x) = $((|E|-$MIN-FBUT2$[F_1]$(x))/3) $- 3|C|$

- MIN-FBUT2$[F_1]$(x) = MIN-FUDT$[F_1]$(x)

- MIN-FUDT$[F_1]$(x) = (MIN-FUDT$[F]$(x) $* \sum_{x,y \in S} d_{x,y}$)/100

- MIN-FBUT2$[F_1]$(x) = MIN-FBUT2$[F_2]$(x)

- MIN-FBUT2$[F_2]$(x) = MIN-FUDT$[F_2]$(x)

- MAX-X3C(x) = $((|E|-$MIN-FBUT2$[F_1, \geq]$(x))/3) $- 3|C|$

- MIN-FBUT2$[F_2, \geq]$(x) = MIN-FBUT2$[F_1, \geq]$(x)

- MIN-VERTEX COVER(x) = MIN-FUGT$[\geq]$(x) $-|E|$

Theorem 22 *All character compatibility and unweighted phylogenetic parsimony and distance-matrix fitting evaluation problems examined in this thesis are OptP[$O(\log n)$]-complete.*

Consider completeness results for the weighted problems. Ordinarily, a weighted evaluation problem is shown OptP-complete by a variant of the reductions used to show their unweighted versions to be OptP[$O(\log n)$]-complete [GKR92, p. 9]. However, the required modifications are not obvious for either phylogenetic parsimony or distance matrix fitting problems. For example, consider the weighted phylogenetic parsimony problems. Define weighted MIN-VERTEX COVER as the problem that associates weights with the vertices of a graph and returns the sum of the weights of the minimum weight vertex cover. This problem is OptP-complete [GKR92, Theorem 3.3]; however, the difficulty with modifying the many-one reductions to phylogenetic parsimony problems given in Section 3.2.1 is that optimal solutions to the reduced instance neither consistently minimize nor maximize the weights of the vertices of the candidate cover, but instead minimize the weight of the whole tree (see Figure 11). This complicates the extraction of the cost of the useful portions of the solution from the cost of the whole solution. This difficulty can be resolved in the same way as for weighted MIN-STEINER TREE IN GRAPHS [GKR92, Theorem 3.4], by including in the instance an explicit weighting function for all edges in the implicit graph. This version of each weighted phylogenetic parsimony problem is OptP-complete; however, it violates the spirit of the original biological problem, and thus will not be considered here further. Similar difficulties occur in attempts to modify the many-one reductions for distance matrix fitting problems given in Section 3.2.3.

By the restriction reductions from all unweighted to weighted phylogenetic parsimony and distance matrix fitting problems, the evaluation problems corresponding to the latter are OptP[$O(\log n)$]-hard. However, it is possible to do better using paddability:

Theorem 23 *The following hold:*

1. *MIN-WBCCS and MIN-WBQCS are paddable with respect to VERTEX COVER.*

2. *MIN-WBCDo and MIN-WBQDo are paddable with respect to VERTEX COVER.*

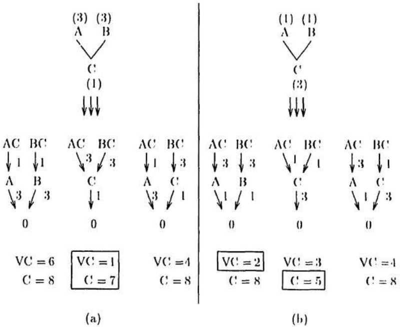


Figure 11: Difficulties with the reduction from weighted MIN-VERTEX COVER to weighted phylogenetic parsimony evaluation problems. Graphs are shown on top, and all possible trees for each graph under the reduction in Table 9 and the costs of these trees (C) and their corresponding vertex covers (VC) are given below. The numbers in parentheses in the graphs denote the weights associated with particular vertices. Note that for the graph in (a), the minimal tree in the reduced instance also yields a minimal vertex cover, which is not the case for the graph in (b).

3. *MIN-WBW is paddable with respect to VERTEX COVER.*

4. *MIN-WBCCI and MIN-WBQCI are paddable with respect to VERTEX COVER.*

5. *MIN-WBGe is paddable with respect to VERTEX COVER.*

Proof:

Proof of (1): Assume without loss of generality that all given instances $x_1, x_2, \ldots, x_k$ of VERTEX COVER have the same number of vertices, and can thus be mapped by the reduction f in Table 9 into k instances of UBCCS, each of which has d characters and m_i taxa, $1 \leq i \leq k$. Let $m^* = \max m_i$. Construct an instance x' of MIN-WBCCS on $d' = kd$ characters $c_1, \ldots, c_{d'}$ split into zones $z_i = c_{(i-1)*d+1}, \ldots, c_{i*d}$, $1 \leq i \leq k$, with each zone corresponding to one of the given instances of UBCCS. Let $S' = \bigcup_{i=1}^{k} S_i$, with each $s \in S_i$ being mapped into its appropriate zone as in $f(x_i)$, with zeroes in the characters of all other zones. Give each character in zone z_i weight $w_{z_i}(m^*d + 1)^{(i-1)}$. Note that the maximum weight in x' has a number of bits polynomial in k, m^*, and d. Hence, function h_1 is polynomial time.

No path in an optimal tree T for instance x' can include a vertex v such that v has characters with state 1 in two different zones. Suppose that such a path p exists, and assume without loss of generality that there are no vertices from S' on this path. Denote the two zones by z' and z'', the first two vertices surrounding v on p that have 1-states totally within one zone by x and y, and the number of 1-states in x and y by 1_x and 1_y. Suppose x and y are in the same zone; assume this zone is z'. Create a path p' by taking each vertex in p and retaining only those edges wholly in zone z' i.e. project path p onto the characters in zone z'. Path p' still connects x and y, and is shorter than p, which contradicts the optimality of T. Alternatively, suppose x and y are in zones z' and z'', respectively. Assume without loss of generality that there is a path from x to 0 in z'. Any path from x to y must contain at least $1_x + 1_y$ edges and have length at least $(1_x)w_{z'} + (1_y)w_{z''}$. Consider tree T'' that replaces the path p with a path from y to 0 of length $(1_y)w_{z''}$. Tree T'' is shorter than tree T', which is a contradiction. Hence, all edges in the optimal tree are between vertices in their own zones, and the cost of the optimal tree for x' corresponds to the summed costs of an optimal tree for each zone times the weight for that zone. Recall from Section 3.2.1 that an unweighted binary Camin-Sokal tree on m taxa and d characters has optimal length not greater than md; thus, the costs of optimal trees for each zone cannot overflow into the costs for trees in other zones, and the cost of the tree corresponding to any x_i can be easily extracted from the cost for x'. Hence, function h_2 is also polynomial time, establishing paddability.

Proof of (2): Given a set of instances $x_1, x_2, \ldots, x_k$ of VERTEX COVER, construct an instance x' as in (1) above with two additions: (1) there are $(k+1)$ zones, and the $(k+1)$-th zone of maximum weight is designated $z*$, and (2) S'' is augmented by $y_i, 1 \leq i \leq (k+1)d$, such that y_i has 1's in positions i to $(k+1)d$

and 0's everywhere else.

Consider an optimal tree T for x'. All edges $\{\{y_j, y_{(j+1)}\}, 1 \leq j < (k+1)d\} \cup \{y_{(k+1)d}, 0\}\}$ are in T by the reasoning given in [DJS86, Theorem 3], and by reasoning similar to that for (1) above, there are no paths in T between vertices in different zones. Moreover, there is no path p from any vertex u in a zone z' to any y_j. Suppose such a path p exists; project p onto characters in z', i.e. create a path from u to 0. As y_j and 0 are already connected, this yields a tree shorter than T, which is a contradiction. Hence, the cost of T is the cost of edges $\{\{y_j, y_{j+1}\}, 1 \leq j < (k+1)d\} \cup \{y_{(k+1)d}, 0\}\}$ plus the summed costs of an optimal tree for each zone times the weight of that zone. By reasoning similar to that for (1) above, functions h_1 and h_2 are polynomial time, establishing paddability.

Proofs of (3 - 5): The proofs for (3) and (4) are variants of those for (1) and (2), respectively. As any ordered phylogenetic parsimony problem can be simulated by an appropriately-structured instance of the Generalized parsimony problem, (5) can be proved by a variant on any of these other proofs. ∎

Corollary 24 *All weighted phylogenetic parsimony evaluation problems examined in this thesis are $FP_{\|}^{NP}$-hard.*

Similar results hold for several of the distance matrix fitting problems.

Theorem 25 *The following hold:*

1. *MIN-FUUT[F_1] is paddable with respect to X3C.*

2. *MIN-FUUT[$F_1, \geq$] is paddable with respect to X3C.*

3. *MIN-FUGT[$\geq$] is paddable with respect to VERTEX COVER.*

Proof:

Proof of (1): Assume without loss of generality that all given instances $x_1, x_2, \ldots, x_k$ of X3C have the same number of vertices, and can thus be mapped by the reduction f in Table 17 into k instances of FBUT[F_1], each of which has graphs G_i with c_i edges. Let $c^* = \max c_i$. Construct the instance x' of MIN-FUUT[F_1] as a distance matrix D' based on an underlying graph $G' = \bigcup_{i=1}^{k} G_i$ such that $d'_{i,j} = 0$ if $i = j$, $d'_{i,j} = (c^* + 1)^k - (c^* + 1)^{(m-1)}$ if $\{i, j\} \in E_m$, and $d'_{i,j} = (c^* + 1)^k$ otherwise. The maximum weight in x' has a number of bits polynomial in c^* and k. Hence, function h_1 is polynomial time.

No partition in an optimal ultrametric tree T for instance x' can join vertices from different component graphs. Assume that two such vertices u and v are joined at level l. Consider T' that instead joins u and v at level $(c^* + 1)^k$. As $d_{u,v} = (c^* + 1)^k$, T' is of lower cost than T, which is a contradiction. Hence, all partitions must join vertices within individual G_i. A partition of G_i into either three or four triangles in the manner described in Section 3.2.3 is optimal at level $(c^* + 1)^k - (c^* + 1)^{(i-1)}$. Moreover, there can be no other joining of vertices in G_i until level $(c^* + 1)^k$. Suppose there was a partition of G_i at level l, $(e^* + 1)^k - (e^* + 1)^{(i-1)} < l < (c^* + 1)^k$, which joined two previously separate groups of vertices X and Y. Let G_X, G_y, and $G_{X \bigcup Y}$ be the subgraphs of G_i

induced by the vertex-sets X, Y, and $X \bigcup Y$, respectively. Let e_p be the number of edges in G_X and G_Y, e_u be the number of edges in $G_{X \bigcup Y}$ less e_p , and e_t be $(|X|+|Y|-1)(|X|+|Y|)/2-(e_p+e_u)$. Note that e_p is the number of previously used edges, e_u is the number of unused edges, and e_t is the number of possible edges. Figure 12 shows that the partition at level l can exist if and only if $e_t \leq e_u$. As e_p is composed of complete subgraphs, $e_p = |X|(|X|-1)/2 + |Y|(|Y|-1)/2$, and $e_t = |X||Y| - e_u$; hence, this condition can be rewritten as $|X||Y|/2 < e_u$. Consider the following three cases for the simplest possible partitions at level l:

- *X and Y are single vertices* $\in \{x_{\alpha,1}, x_{\alpha,2}, x_{\alpha,3}\}$: As no edges join any two of these vertices in G_α, $e_u = 0$. Hence, the condition becomes $\frac{1}{2} < 0$, which is a contradiction.

- *X is a triangle in equation 7 and Y is a single vertex* $\in \{x_{\alpha,1}, x_{\alpha,2}, x_{\alpha,3}\}$: As there is at most one edge joining X and Y in G_α, $e_u = 1$. Hence, the condition becomes $\frac{3}{2} < 1$, which is a contradiction.

- *X and Y are triangles in either equation 6 or equation 7* : As there are at most two edges joining X and Y in G_α, $e_u = 2$. Hence, the condition becomes $\frac{9}{2} < 2$, which is a contradiction.

Using the argument above for the joining of two groups, the reader can verify that no joining of three or more groups at level l can occur in an optimal tree. Therefore, no partition at level l can exist in an optimal tree.

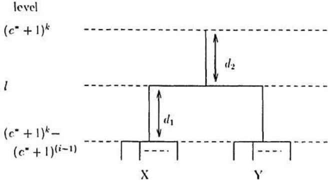


Figure 12: Conditions for multiple partition levels on subgraph G_u. If level l is not used in the optimal tree, the cost of groups X and Y is $c_u(d_1 + d_2)$; else, the cost is $c_ld_1 + c_ud_2$. Thus, level l can exist in an optimal tree only if $c_ld_1 + c_ud_2 \leq c_u(d_1 + d_2)$ i.e. $c_l \leq c_u$.

Hence, an optimal ultrametric tree T' for instance x' will consist of k nontrivial partitions at levels $(c^* + 1)^i, 0 \leq i \leq (k-1)$ corresponding to solutions to the k instances of X3C, and will have cost equal to the summed values of F_1 for these solutions. Recall from Section 3.2.3 that an optimal solution for x_i in T' will have weight $c_i(c^*+1)^{(i-1)} - 3X(c^*+1)^{(i-1)}$, where X is the number of triangle subgraphs in G_i that are induced by that solution; thus, the costs of optimal ultrametric trees for each x_i cannot overflow into the costs of optimal ultrametric trees for other x_j, and the cost of the solution for any x_i can be easily extracted from the cost for x'. Hence, function h_2 is polynomial time, establishing paddability.

Proof of (2): A variant of that given above for (1), made less complex by dominance.

Proof of (3): Assume without loss of generality that all given instances $x_1, x_2, \ldots, x_k$ of VERTEX COVER have the same number of vertices v and edges e. Construct the instance x' of MIN-FUGT[$\geq$] as $V' = \{*\} \cup \{\bigcup_{i=1}^{k} V_i - \{*\}\}$, $S' = \{*\} \cup \{\bigcup_{i=1}^{k} S_i - \{*\}\}$, and D' such that all distances between pairs of vertices in the same V_i are those given in the reduction in Table 18 multiplied by $(v + e + 1)^{(i-1)}$, and all distances between pairs of vertices in different instances are sums of the edges on the path between those vertices in a canonical tree for x'. The reader can verify that in an optimal tree for x', there will be no edges between vertices in different V_i, as these will be forbidden by the constraint of dominance. Hence, the cost of the optimal tree will be the sum of the weights of all edges in optimal trees for each V_i. Note that the sum of weights for each V_i will be less than $(v + e)(v + e + 1)^{(i-1)}$; hence, the costs of optimal trees for each x_i cannot overflow into the costs for optimal trees for other x_j, and the cost of the solution for any x_i can be easily extracted from the cost for x'. Hence, function h_2 is polynomial time, establishing paddability. ▮

It is unfortunate that most of the weighted distance matrix fitting evaluation problems do not yield to paddability proofs of the style above. The exponential increase in the length of the weights required to separate optimal solutions for each instance under the F_2 statistic complicates proofs for MIN-FUUT[F_2] and MIN-FUUT[$F_2, \geq$], and it is not obvious how one could show paddability for FUDT[F_1], FUDT[F], or FUDT[F_2].

Corollary 26 *The following hold:*

1. *MIN-FUUT$[F_1]$, MIN-FUUT$[F_1, \geq]$, and MIN-FUGT$[\geq]$ are $FP_{\|}^{NP}$-hard.*

2. *MIN-FUDT$[F_1]$, MIN-FUDT$[F]$, MIN-FUUT$[F_2]$, MIN-FUDT$[F_2]$, and MIN-FUUT$[F_2, \geq]$ are properly $FP^{NP[O(\log n)]}$-hard.*

4.3 Solution Functions

Solution problems have been studied indirectly via their approximation by decision problems [GJ79] and evaluation problems [GKR92, Kre88]. More recently, these problems have been studied directly using paddability [CT91, Gas86] and multivalued function classes such as $NPMV_g$ [Sel91]. The techniques developed in this latter work will be used in this section.

There are several types of solution functions.

1. A function that computes a single solution [GJ79, Chapter 5].

2. A function that computes but cannot enumerate all solutions i.e. a function in NPMV [Sel91].

3. An index-driven function $g(i, x)$ that computes the i-th solution for instance x under some polynomial-time ordering P on k-nary strings.

Definitions (1) and (2) will be discussed in this section; definition (3) describes the enumeration functions in Section 4.5 and will be discussed there.

Consider functions of the type in definition (1) above. Following [CT91], the focus will be on bounds on the complexity of SOL-$\mathbf{X}_f$, the class of single-valued functions that compute solutions to problem $\mathbf{X}$. Certain properties are known to imply upper bounds on SOL-$\mathbf{X}_f$: problems that have a polynomial number of feasible solutions for any instance are in $FP_{||}^{NP}$ [Sel91, Proposition 5], and problems that are polynomial-invertible in the sense of [WagK87] i.e. all solutions of cost k can be enumerated in polynomial time, are of complexity equivalent to their cost functions. However, none of the problems examined in this thesis exhibit either of these properties. Consider instead lower bounds. By the prefix-search technique, which builds an optimal solution bit by bit by consulting an NP solution-prefix oracle ([BDG88, p. 61]; [GJ79, Chapter 5]), every problem $\mathbf{X}$ has at least one member of SOL-$\mathbf{X}_f$ in FP^{NP}; hence, the lower bound can be no harder than FP^{NP}. As no optimal-cost solution function can be easier than its associated evaluation function, lower bounds can be derived from the complexity of the associated evaluation functions. Such bounds can be improved for phylogenetic inference problems by applying Theorem 16.

Theorem 27 *All single-valued functions solving all phylogenetic inference optimal-cost solution problems examined in this thesis are $FP_{||}^{NP}$-hard.*

Proof: As noted in Section 3.2.4, all reductions in Section 3.2 give algorithms for transforming optimal solutions for original and reduced instances into one another. Hence, the metric reductions from MAX-X3C, MAX-CLIQUE, and

MIN-VERTEX COVER to unweighted phylogenetic inference evaluation problems given in Section 4.2 can be modified to give metric reductions between the corresponding optimal-cost solution problems. SOL-MAX-CLIQUE is paddable [CT91, Theorem 4.2], and SOL-MAX-X3C and SOL-MIN-VERTEX COVER can be shown paddable via functions that simply combine the given instances into one instance without adding any new components. ∎

Consider now functions of the type in definition (2) above. All phylogenetic inference optimal-cost solution problems defined in this thesis are in $NPMV_g \circ FP^{NP}$, and all corresponding given-cost and given-limit solution problems are in $NPMV_g$. This definition is useful primarily for visualizing the set of solutions associated with particular instances of a problem, and highlighting the computational structures for different types of solution functions e.g. the two-phase nature of $NPMV_g \circ FP^{NP}$ computations (see Section 4.1.1). However, it is also possible to derive results using this definition, such as the following lower bound on the complexity of single-valued functions for the phylogenetic inference given-cost and given-limit solution problems.

Theorem 28 *All single-valued functions solving all phylogenetic inference given-cost and given-limit solution problems examined in this thesis are properly $FP^{NP[O(\log n)]}$-hard unless $P = NP$.*

Proof: Cook's generic reduction from decision problems in NP to SAT [GJ79, Section 2.6] (see Section 5.1) is a generic metric reduction from every solution problem in $NPMV_g$ to SOL-SAT. The reader can verify that the reductions from SAT to CLIQUE, VERTEX COVER, and X3C [GJ79, Section 3.1], and from these problems to the phylogenetic inference decision problems (see Section 3.2) are also metric reductions between the corresponding given-cost and given-limit solution problems. Hence, any single-valued function that solves any of the phylogenetic inference given-cost and given-limit solution problems can be used to construct a single-valued function that solves any problem in $NPMV_g$. To complete the proof, recall that $NPMV_g \subseteq_c FP^{NP[O(\log n)]}$ implies $P = NP$ [Sel91, Theorem 3]. ∎

4.4 Spanning Functions

Counting problems were first defined and studied in [Val79a, Val79b, SimJ77]. This early work considered the number of (not necessarily distinct) solutions encoded by a nondeterministic computation, and has led via threshold-acceptance mechanisms to the work on probabilistic computation [Joh90, Section 4]. There has been a recent resurgence of interest in counting for counting's sake [SchU90, Tor91, WagK86a, WagK86b], including the counting of distinct solutions [KST89], which will be the focus in this section.

All phylogenetic inference given-cost and given-limit problems examined in this thesis are in SpanP, and all corresponding optimal-cost spanning problems are in Span($NPMV_g \circ FP^{NP}$). At present, there are no lower bounds known on the complexities of any of these problems. Only the reduction from CLIQUE to BCC gives a one-to-one solution mapping, which yields a metric reduction between the corresponding optimal-cost, given-cost, and given-limit spanning problems. Hence, all character compatibility spanning problems are harder than the corresponding problems for CLIQUE. Unfortunately, none of these problems for CLIQUE are known to be be hard for either #P or SpanP. It is interesting that only the versions of CLIQUE and VERTEX COVER that count locally optimal solutions have been shown to be #P-complete [Val79a, Theorem 1].

Several trivial but intriguing bounds emerge for any spanning problem **X** by applying binary search arguments. The following hold for unweighted problems,

- SPAN-SOL-OPT-**X** $\in FP^{\text{SPAN-SOL-VAL.EQ-}\mathbf{X}}$

- SPAN-SOL-OPT-**X** $\in FP^{\text{SPAN-SOL-VAL.LE-}\mathbf{X}[O(\log n)]}$

- SPAN-SOL-VAL.LE-**X** $\in FP^{\text{SPAN-SOL-VAL.EQ-}\mathbf{X}}$

weighted problems,

- SPAN-SOL-OPT-**X** $\in FP^{\text{SPAN-SOL-VAL.LE-}\mathbf{X}}$

and for all problems:

- SPAN-SOL-VAL.EQ-**X** $\in FP^{\text{SPAN-SOL-VAL.LE-}\mathbf{X}[2]}$

Note that if either of the given-cost or given-limit spanning problems is in FPH, all three problems are in FPH; however, the optimal-cost spanning problem can be in FPH without implying anything about the complexity of the other two problems (see Section 4.7).

4.5 Enumeration Functions

All existing definitions of enumerability in complexity theory (see [HHSY91, Section 2] for a review) are concerned with enumerating languages rather than the ranges of functions for particular inputs. The enumeration problem considered here was defined more for the convenience of its users than theoretical tractability; however, it may still be of some use in pure complexity-theoretic investigations.

Though any function in FPH can be simulated in FPSPACE(poly), it is not obvious that any such function can be enumerated in FPSPACE(poly).

Theorem 29 *Given a problem* Π *in* $F\Sigma_k^p$ *and a polynomial-time ordering* P *on binary strings, the problem of computing the kth optimal solution under* P *for an instance* x *of* Π *is in FPSPACE(poly).*

Proof: Let $N \in F\Sigma_k^p$ be a polynomial-time NOTM transducer that computes the solutions of Π. Let $p(n)$ be the polynomial bounding the running time of N and assume that all solutions have length $p(n)$. Define the following function:

RANK(N,P,x,y) $= |\{w|$ w is a solution to N on input x and $w \leq y$ under ordering $P\}|$.

RANK can be computed in FPSPACE(poly) in the same way as functions in SpanPH (Corollary 19, Part (1)). Using RANK, a binary search can determine which of the $2^{p(n)}$ possible solutions has exactly $(i-1)$ solutions preceding it under P i.e. the i-th solution under P. Note that this binary search is conducted on the ordering of possible solution strings under P. This procedure is in $FP^{FPSPACE(poly)} = FPSPACE(poly)$. ▮

Theorem 30 *Given a problem* Π *in* $F\Sigma_k^p$ *and a polynomial-time ordering* P *on binary strings, the problem of computing the kth optimal solution under* P *for an instance* x *of* Π *is in* $FP^{\#P}$.

Proof: Modify NTM N in the preceding proof to take as additional input a binary string y and produces only those solutions w such that $w \leq y$ under ordering P. Proof follows by observing that RANK(N,P,x,y) = SPAN(N'(x,y)) and that SpanP $\subseteq FP^{\#P[1]}$ by Corollary 19. ▮

Corollary 31 *All phylogenetic inference optimal-cost, given-cost, and given-limit enumeration problems examined in this thesis are in* $FP^{\#P}$.

The only known lower bound for these problems is implied by the observation that, for a problem **X**, SPAN-**X** $\in FP^{ENUM-\mathbf{X}}$ i.e. no enumeration problem can be easier than its associated spanning problem.

4.6 Random Generation Functions

Though there are many papers on the random generation of particular types of graphs, general random-generation problems have been formulated and studied as a class only in [JVV86]. The results of the previous section suggest that random generation problems are in $FRP^{\#P}$; however, Jerrum, Valiant, and Vazirani have given a procedure of complexity $FRP^{\Sigma_2^p}$ which uses Stockmeyer's $FP^{\Sigma_2^p}$ approximation procedure for functions in #P (see Section 5.6) to generate outputs of any $NPMV_g$ machine at random under a uniform distribution. Recall that an NP query can be simulated by an appropriate Σ_2^p query.

Corollary 32 *All phylogenetic inference optimal-cost, given-cost, and given-limit random-generation problems examined in this thesis are in* $FRP^{\Sigma_2^p}$.

As any random-generation function is also a solution function, the lower bounds on the complexities of solution functions given in Section 4.3 also apply to random-generation functions.

These results have an irrevocably academic flavor because they depend on access to a source of truly random bits. Though it is impossible to obtain random bits by purely arithmetical methods, there are techniques for generating near random bit sequences and for expanding random "seed" sequences into longer random sequences. The interested reader is referred to [LV90b, Section 1] for an introduction to mathematical definitions of randomness, and [Riv90, Section 7] for a summary of methods for generating random sequences.

	Optimal-Cost		Given-Cost	Given-Limit
	Unweighted	Weighted		
Decision	-		NP-complete	
Evaluation	$FP^{NP[O(\log n)]}$-C	$FP_{\|\|}^{NP}$-hard †	-	
Solution	$FP_{\|\|}^{NP}$-hard, $\in NPMV_g \circ FP^{NP}$		properly $FP^{NP[O(\log n)]}$-hard, $\in NPMV_g$	
Spanning	$\in$ Span($NPMV_g \circ FP^{NP}$)		$\in$ SpanP	
Enumeration	$\in FP^{\#P}$			
Random Generation	$\in FRP^{\Sigma_2^p}$			

Table 20: Computational complexities of phylogenetic inference functions.

† Most weighted distance matrix fitting evaluation problems are only known to be properly $FP^{NP[O(\log n)]}$-hard (see Corollary 26).

4.7 Summary

All complexity results obtained in this section for phylogenetic inference problems are given in Table 20. Optimal-cost solution problems are provably harder than the corresponding given-cost and given-limit solution problems because of the NP queries allowed to optimal-cost problems. However, this difference seems to disappear for more complex versions of these problems. Though this difference may re-assert itself when completeness results are available, the relations between the spanning versions of these problems suggest otherwise (see Section 4.4). I conjecture that for problems more complex than computing solutions, optimal-cost problems are easier than their corresponding given-cost and given-limit problems.

Solution problems are of greatest interest to biologists, as these problems are concerned with the trees that define evolutionary hypotheses. Moreover, they are the only problems that have been investigated in the literature, albeit by assessing particular algorithms solving these problems [LP85, Pla89]. Several of the other problems treated above also have biological applications. For instance, spanning results give lower bounds on the running time of branch-and-bound algorithms that solve the corresponding solution problems [Sto85, Val79a]. Also, as each phylogeny incorporates a different hypothesis of character change, all such hypotheses should be considered to get an accurate idea of what is implied about phylogeny by a particular data set [Mad91, MRS92], which could be done by enumerating all phylogenies.

The results given in this section do not directly put upper or lower bounds on the time complexities of algorithms solving these problems; at present, it is only known that these problems, by virtue of being in FPSPACE, can be solved in exponential time. However, these results do give the relative hardnesses of these problems, and may suggest guidelines for algorithm designers about which approaches may *not* useful for solving these problems.

5 The Approximability of Phylogenetic Inference Functions

The results in previous sections suggest that polynomial-time algorithms providing exact solutions for phylogenetic inference optimal-cost problems probably do not exist. However, fast algorithms may exist if one is willing to settle for approximate solutions whose cost is within some fixed interval or ratio of the optimal cost. In this section, I will derive some limits on the types of approximations that are available to phylogenetic inference problems.

5.1 Types of Approximability

This section gives a brief overview of types of approximation algorithms and some class-based approaches to proving that various of these approximations cannot exist for a given problem. For in-depth reviews of topics in this section, see [BJY89, GJ79, HS78, Mot92].

Given a problem X, an instance I of X, and an approximation algorithm A_X for X, let $OPT_X(I)$ be the cost of the optimal solution for I, $A_X(I)$ be the cost of the solution for I found by A_X, and $MAX_X(I)$ be the largest of the costs of all solutions of I; further, let $\mathbf{Y} = OPT_X(I)$ if X is a minimization problem, and $\mathbf{Y} = A_X(I)$ if X is a maximization problem. There are several measures of the quality of an approximation [OM90, p. 6]:

- *Relative Error Measure*: $\mu_r(I) = \frac{|OPT_X(I) - A_X(I)|}{\mathbf{Y}}$

- *Absolute Error Measure*: $\mu_a(I) = |OPT_X(I) - A_X(I)|$

- *Normalized Relative Error Measure*: $\mu_r(I) = \frac{|OPT_X(I) - A_X(I)|}{MAX_X(I) - OPT_X(I)}$

The different **Y** are applied to map the error-measure values for minimization and maximization problems into the same interval, namely $[0, +\infty)$, for easier comparison [GJ79, p. 128]. There are several types of approximation algorithms defined by various bounds on the quality of the resulting approximations:

1. Absolute (Additive) Approximation

    - $|OPT_X(I) - A_X(I)| \leq f(|I|)$

2. Polynomial-Time Approximation Schemes

    - Polynomial-Time Approximation Scheme (PTAS):

      For all $k > 0$, there exists an algorithm A_X such that $|OPT_X(I) - A_X(I)| \leq \frac{1}{k}\mathbf{Y}$ and the runtime of A_X is polynomial in $|I|$ for each k.

    - Fully Polynomial-Time Approximation Scheme (FPTAS):

      For all $k > 0$, there exists an algorithm A_X such that $|OPT_X(I) - A_X(I)| \leq \frac{1}{k}\mathbf{Y}$ and the runtime of A_X is polynomial in $|I|$ and $\frac{1}{k}$.

The algorithm A_X can be either a single algorithm (*uniform PTAS*) or a family of algorithms (*non-uniform PTAS*).

3. Relative (Multiplicative) Approximation

- $|OPT_X(I) - A_X(I)| \leq c\mathbf{Y}, c > 0$

In the following, "a polynomial-time algorithm with a relative (an absolute) approximation c" will be abbreviated as "a relative (an absolute) approximation c". There are several variants on these definitions in the literature that are created by using different error measures or implying asymptotic rather than absolute error bounds. One such variant (indeed, the first [Joh74] and preferred notation) represents relative approximations as straightforward ratios $\frac{A_X(I)}{OPT_X(I)}$ (minimization problems) and $\frac{OPT_X(I)}{A_X(I)}$ (maximization problems). As some of these definitions are not equivalent and it is not always clear which definition is being used, the reader must exercise caution in comparing results from different sources. In this thesis, all approximability definitions and results will be phrased as above in terms of the absolute error measure, because (1) this measure unifies the three types of approximation algorithms described above, and (2) this measure is the formulation of choice in the proof techniques [ALMSS92, Kre88, PY91] used in this section.

Traditionally, the theory of approximation algorithms has been concerned with proofs that certain types of approximability did not exist for particular problems, with approximation-preserving reductions, and with necessary and sufficient conditions for the existence of various approximation algorithms for a given problem; see [BJY89, HS78, GJ79, Mot92] for a review of this work. Within the last four years, two approaches have emerged that are based on hierarchies of approximability classes:

1. *The Algorithmic Approximability Hierarchy [CP91, OM90]*: Define the class NPO of all NP optimization problems, and its subclasses FPTAS, PTAS, and APX consisting of all problems that have FPTAS, PTAS, and relative approximation algorithms, respectively. Orponen and Mannila [OM90] defined NPO, and showed that several problems are NPO-complete under a relative-approximation preserving reduction. Crecenzi and Panconesi [CP91] defined FPTAS, PTAS, and APX, and gave artificial problems that are complete for PTAS and APX. It is known that $FPTAS \subset PTAS \subset APX \subset NPO$ unless P = NP [CP91, Theorem 6], and that a problem that is hard for a particular class cannot have an approximation algorithm from a lower class unless P = NP.

2. *The Logical-Form Approximability Hierarchy [KT90, KT91, PR90, PY91]*: The algorithmic approach to defining approximability does not give insight into why problems are approximable [BJY89, p. 220]; moreover, it is not clear how one defines a notion of "approximate computation", let alone classes of such computations, using the Turing Machine encoding of problems [PY91, p. 426]. Building on the work of Fagin [Fag74], Papadimitriou and Yannakakis [PY91] initiated the study of approximability classes that do not involve computation – that is, classes of approximable prob-

> lems defined by the syntactic structure of the logic formulas that describe the solutions of those problems. Many classes have been defined in this framework [KT90, KT91, PR90]; only MAX NP and MAX SNP will be described here. Fagin showed that the class NP could be represented in logic as the class of problems whose solutions S can be expressed by formulas with structure $\exists S \forall x \exists y \phi(x, y, S)$ where ϕ is quantifier free. Given this formulation, Papadimitriou and Yannakakis defined MAX NP as the class of problems whose solutions have the form $\max_S |\{x | \exists y \phi(x, y, S)\}|$ that is, problems whose solutions S satisfy the maximum number of different x rather than all of them. Papdimitriou and Yannakakis also define subclass SNP of NP of the form $\exists S \forall x \phi(x, S)$ and subclass MAX SNP of MAX NP. The formulation of SAT, the boolean formula satisfiability problem, in each class is given in Tables 21 and 22.

Class MAX SNP will be important later in this section, as will the following reducibility.

Definition 33 ([PY91], p. 427) *Let* Π *and* Π' *be two optimization (maximization or minimization) problems. We say that* Π *L-reduces to* Π' *(*$\Pi \leq_L \Pi'$*) if there are two polynomial-time algorithms* f, g *and constants* $\alpha, \beta > 0$ *such that for each instance* I *of* Π*:*

(L1) *Algorithm* f *produces an instance* $I' = f(I)$ *of* Π'*, such that the optima of* I *and* I'*,* $OPT(I)$ *and* $OPT(I')$*, respectively, satisfy* $OPT(I') \leq \alpha OPT(I)$

SAT $\in$ NP

Instance: Boolean formula S in conjunctive normal form i.e. clauses composed of variables linked by disjunctions (logical OR), which are linked by conjunctions (logical AND).

Formula: $\exists T \forall c \exists x[(P(c,x) \wedge x \in T) \vee (N(c,x) \wedge \neg(x \in T))]$,

where P and N encode the instance S ($P(c,x)$ means that variable x appears unnegated in clause c of S; $N(c,x)$ means that variable x appears negated in clause c of S) and T is the set of true variables corresponding to a particular assignment for S.

MAX SAT $\in$ MAX NP

Instance: Boolean formula S in conjunctive normal form.

Formula: $\max_T |\{c | \exists x[(P(c,x) \wedge x \in T) \vee (N(c,x) \wedge \neg(x \in T))]\}|$

where P, N, and T are as defined for SAT.

Table 21: Formulations of SAT in first-order logic (adapted from [KT90, PY91]).

3SAT $\in$ SNP

Instance: Boolean formula S in conjunctive normal form, in which each clause has at most 3 variables.

Formula:

$$\begin{array}{ll}
\exists T \forall (x_1, x_2, x_3) \; [& ((x_1,x_2,x_3) \in C_0 \rightarrow x_1 \in T \vee x_2 \in T \vee x_3 \in T) \wedge \\
& ((x_1,x_2,x_3) \in C_1 \rightarrow x_1 \notin T \vee x_2 \in T \vee x_3 \in T) \wedge \\
& ((x_1,x_2,x_3) \in C_2 \rightarrow x_1 \notin T \vee x_2 \notin T \vee x_3 \in T) \wedge \\
& ((x_1,x_2,x_3) \in C_3 \rightarrow x_1 \notin T \vee x_2 \notin T \vee x_3 \notin T) \quad],
\end{array}$$

where C_0, C_1, C_2, and C_3 encode the instance S ($c = (x_1, x_2, x_3) \in C_j$ means that variables $x_1, \ldots, x_j$ are negated and variables $x_{j+1}, \ldots, x_3$ are unnegated in clause c of S) and T is the set of true variables corresponding to a particular assignment for S.

MAX-3SAT $\in$ MAX SNP

Instance: Boolean formula S in conjunctive normal form, in which each clause has at most 3 variables.

Formula:

$$\begin{array}{ll}
\max_T |\{(x_1, x_2, x_3)| \; [& ((x_1,x_2,x_3) \in C_0 \rightarrow x_1 \in T \vee x_2 \in T \vee x_3 \in T) \wedge \\
& ((x_1,x_2,x_3) \in C_1 \rightarrow x_1 \notin T \vee x_2 \in T \vee x_3 \in T) \wedge \\
& ((x_1,x_2,x_3) \in C_2 \rightarrow x_1 \notin T \vee x_2 \notin T \vee x_3 \in T) \wedge \\
& ((x_1,x_2,x_3) \in C_3 \rightarrow x_1 \notin T \vee x_2 \notin T \vee x_3 \notin T) \quad]\}|,
\end{array}$$

where C_0, C_1, C_2, C_3, and T are as defined for 3SAT.

Table 22: Formulations of SAT in first-order logic (cont'd from Table 21).

(L2) *Given any solution of I' with cost c', algorithm g produces a solution of I with cost c such that $|c - OPT(I)| \leq \beta|c' - OPT(I')|$.*

L-reducibility is closely related to most of the other defined approximation-preserving reducibilities [CP91, OM90]; indeed, a constrained L-reducibility applicable to pairs of maximization or minimization problems was defined independently by H. Simon [SimH89]. Following Simon, $r = \alpha\beta$ will be called the *expansion* of a given L-reduction.

Lemma 34 ([PY91], Proposition 1) *L-reductions compose.*

Lemma 35 ([PY91], Proposition 2) *If $\Pi \leq_L \Pi'$ with expansion r, and Π' has a relative approximation ϵ, then Π has a relative approximation $r\epsilon$.*

Corollary 36 *If $\Pi \leq_L \Pi'$ with expansion r, and "Π has a relative approximation ϵ" $\Rightarrow$ **X**, then "Π' has a relative approximation $\frac{\epsilon}{r}$" $\Rightarrow$ **X**.*

Note that restriction reductions are trivial L-reductions in which $\alpha = \beta = 1$.

Two following two theorems give bounds on approximability using the relationships among classes in the function and language bounded NP query hierarchies:

Theorem 37 ([WagK89], Corollary 16) *No evaluation problem A such that the corresponding decision problem A_{odd} i.e. "Is $OPT_A(X)$ odd?", is $P^{NP[O(\log n)]}$-hard can have an absolute approximation $\leq O(\log n)$ unless $PH = \Theta_3^p$.*

Theorem 38 ([Kre88], Theorem 4.3) *For every OptP[$f(n)$]-hard evaluation problem, $f(n)$ is smooth and $f(n) \in O(\log n)$, there exists $\epsilon \geq 0$ such that every absolute approximation must have value $\geq \frac{1}{2}2^{f(n^{\epsilon})}$ infinitely often.*

The proofs of each of these theorems note that an absolute approximation algorithm reduces the range in which the cost of the optimal solution lies, and that a sufficiently reduced range may be searched with fewer NP queries than are required to solve the evaluation problem. Krentel's theorem will be used in the following sections. This theorem implies almost all known connections between approximability and the function bounded NP query hierarchy.

5.2 Absolute Approximability

There are many elegant proofs which show that absolute approximations do not exist for particular problems [GJ79, HS78, WW86]. However, such results can also be derived for classes of problems.

Theorem 39 *The following hold:*

1. *No OptP[$c \log\log n + O(1)$]-hard evaluation problem can have an absolute approximation $c \leq o(\log n)$ unless $P = NP$.*

2. *No OptP[$O(\log n)$]-hard evaluation problem can have an absolute approximation $c \leq o(poly)$ unless $P = NP$.*

3. *No $FP^{NP}_{||}$-hard evaluation problem can have an absolute approximation $c \leq O(poly)$ unless $R = NP$ and $FewP = NP$.*

4. *No OptP-complete evaluation problem can have an absolute approximation $c \leq O(poly)$ unless $P = NP$.*

Proof:

Proofs of (1 – 2): Follows from Theorem 38.

Proof of (3): Follows from [Sel91, Theorem 12] and [Sel91, Corollary 4(ii)].

Proof of (4): Follows from [Kre88, Theorem 4.1]. ▮

Corollary 40 *The following hold:*

1. *No character compatibility, unweighted phylogenetic parsimony, or unweighted distance-matrix fitting optimal-cost solution problem examined in this thesis has an absolute approximation $c \leq o(poly)$ unless $P = NP$.*

2. *No weighted phylogenetic parsimony optimal-cost solution problem examined in this thesis has an absolute approximation $c \leq O(poly)$ unless $R = NP$ and $P = FewP$.*

3. *None of SOL-MIN-FUUT[F_1], SOL-MIN-FUUT[$F_1, \geq$], or SOL-MIN-FUGT[$\geq$] have an absolute approximation $c \leq O(poly)$ unless $R = NP$ and $P = FewP$.*

4. *None of SOL-MIN-FUDT[F_1], SOL-MIN-FUDT[F], SOL-MIN-FUUT[F_2], SOL-MIN-FUDT[F_2], or SOL-MIN-FUUT[$F_2, \geq$] have an absolute approximation $c \leq o(poly)$ unless $P = NP$.*

Note that results 2 and 3 in Corollary 40 can also be derived using the paddability of the associated evaluation problems and theorems from [Nig75] (see also [WW86, Section 9.1.2.1]).

5.3 Fully Polynomial and Polynomial Time Approximation Schemes

Consider fully polynomial time approximation schemes. Garey and Johnson derived sufficient conditions for FPTAS non-approximability using the notion of strong NP-completeness. An NP-complete decision problem is *strongly NP-complete* if it has an NP-complete subproblem in which all numbers are bounded by some polynomial of the instance length [GJ79, p. 95]. No solution problem whose corresponding decision problem is strongly NP-complete and whose optimal cost is polynomially bounded can have an FPTAS unless P = NP [GJ79, Theorem 6.8 and Corollary]. These conditions are satisfied by all unweighted phylogenetic inference optimal-cost solution problems examined in this thesis. Garey and Johnson also defined pseudo-polynomial reductions, which preserve strong NP-completeness [GJ79, p. 101]. The reader can verify that all reductions given in Section 3.2 from unweighted to weighted problems are also pseudo-polynomial reductions; hence, all weighted decision problems examined in this thesis are also strongly NP-complete. These same reductions also map into subproblems of the solution problems corresponding to these weighted problems whose optimal costs are polynomially bounded.

Theorem 41 *No phylogenetic inference optimal-cost solution problem examined in this thesis has an FPTAS unless $P = NP$.*

Consider polynomial time approximation schemes. The traditional approach to PTAS non-approximability involves proving that the given problem cannot have a relative approximation c, $c > 0$ unless P = NP [IS78, GJ79, WW86]. Such proofs typically derive contradictions by using polynomial-time graph expansions to "amplify" relative approximation algorithms such that cost-restricted NP-complete subproblems can be solved in polynomial time. However, few problems have the cost-restricted NP-complete subproblems required by this approach. A more widely-applicable technique has recently emerged from the study of interactive proof systems (see [Joh92] for an insightful review of the results described below). In what was initially thought to be an isolated result, Feige et al. [FGLSS91] showed that no constant relative approximation c, $c > 0$ exists for SOL-MAX-CLIQUE unless NP $\subset DTIME(n^{\log\log n})$. This result has been dramatically improved by Arora et al. [ALMSS92]:

Theorem 42 ([ALMSS92], Theorem 3) *If there is a PTAS for SOL-MAX-3SAT then $P = NP$.*

This result is significant because SOL-MAX-3SAT is complete for MAX SNP [PY91, Theorem 2], and many problems can be shown MAX SNP-hard via L-reductions.

Theorem 43 (Proposition 2, [ALMSS92]) *There does not exist a PTAS for any MAX SNP-hard problem unless $P = NP$.*

Several MAX SNP-hard problems of particular interest are:

- SOL-MIN-STEINER TREE IN GRAPHS [BP89, Theorem 4.2],
- SOL-MIN-VERTEX COVER-B, in which each vertex in the given graph has degree $\leq B$, and SOL-MIN-VERTEX COVER [PY91, Theorem 2(d)],
- SOL-MAX-CLIQUE [CFS91, Theorem 6], and
- SOL-MAX-X3C-B, in which each element in the given set occurs in $\leq B$ 3-sets, and SOL-MAX-X3C [Kan91, Corollary 4].

Using these problems, it is possible to show many of the problems examined in this thesis to be MAX SNP-hard. In the following, define a *canonical solution* for a problem **X** as a solution to an instance of **X** produced by the reductions given in Section 3.2, e.g. the canonical trees for FUGT[$\geq$].

Lemma 44 *The following hold:*

1. *Given a solution W of cost c to an instance of SOL-MIN-UBCCS or SOL-MIN-UBQCS derived by the reduction from VERTEX COVER given in*

[DJS86] (see Table 9), in polynomial time we can find a canonical solution W' with cost $c' \leq c$.

2. *Given a solution W of cost c to an instance of SOL-MIN-UBCDo or SOL-MIN-UBQDo derived by the reduction from VERTEX COVER given in [DJS86] (see Table 9), in polynomial time we can find a canonical solution W' with cost $c' \leq c$.*

3. *Given a solution W of cost c to an instance of SOL-MIN-UBCCI or SOL-MIN-UBQCI derived by the reduction from VERTEX COVER given in [DS87] (see Table 10), in polynomial time we can find a canonical solution W' with cost $c' \leq c$.*

4. *Given a solution W of cost c to an instance of SOL-MIN-FBUT2[F_1] derived by the reduction from X3C given in [KM86] (See Table 17), in polynomial time we can find a canonical solution W' with cost $c' \leq c$.*

5. *Given a solution W of cost c to an instance of SOL-MIN-FUDT[F_α] ($\alpha \in \{1, 2\}$) derived by the reductions from FBUT2[α] given in [Day87] (see Table 17), in polynomial time we can find a canonical solution W' of cost $c' \leq c$.*

Proof:

Proof of (1): If $c > 2|X|$, then replace W by the tree W' in which each member of $x \in X, x = \{v_i, v_j\}$ is connected to 0 by edges $\{\{v_i, v_j\}, \{v_i\}\}$ and $\{\{v_i\}, 0\}$; this tree is canonical, and has cost $c' < c$. Otherwise, create W' by trimming

W to remove all leaf vertices not in X, and applying the tree transformations in [DJS86, Lemma 1] to the leaf farthest from 0 until the tree is canonical. These tree transformations do not increase tree cost; moreover, as each such transformation removes at least one non-canonical vertex and there can be at most $e - (|X| + 1)$ such vertices, this algorithm is polynomial time.

Proofs of (2 – 3): Analogous to that for (1), using the tree transformations in [DJS86, Theorem 3] (Dollo) and [DS87, Lemma 2 and Theorem 3] (Chromosome Inversion).

Proof of (4): Create W' as follows: if there are partitions that group vertices not connected by edges in the created graph G, then break all such partitions into partitions that only group vertices connected by edges in G. For each group of vertices corresponding to a subgraph G_α, if the "corners" $\{x_{\alpha,1}, x_{\alpha,2}, x_{\alpha,3}\}$ are all included in partitions of G_α, then replace the set of partitions for the vertices of G_α with the four triangle-partitions in equation 6; else, replace with the three triangle-partitions in equation 7 and the appropriate subset of single-vertex partitions drawn from the set $\{x_{\alpha,1}, x_{\alpha,2}, x_{\alpha,3}\}$. These transformations do not increase the cost, as these triangle-partitions are optimal under the F_1 statistic (see Section 3.2.3); moreover, as there are only $|C|$ such groups to transform, the algorithm is polynomial time.

Proof of (5): Create tree W' by applying the tree transformations in [Day87, Proposition 3] to W to create a discretized additive tree W'' with an ultrametric subtree W''_U, and then applying the transformation in [Day87, Proposition 4] to reduce W''_U to an ultrametric subtree of height 2. These transformations do not increase the cost of the tree, and can be performed in polynomial time. ▮

Theorem 45 *The following hold:*

1. *SOL-MIN-VERTEX COVER-B $\leq_L$ SOL-MIN-UBCCS and SOL-MIN-UBQCS*

2. *SOL-MIN-VERTEX COVER-B $\leq_L$ SOL-MIN-UBCDo and SOL-MIN-UBQDo*

3. *SOL-MIN-VERTEX COVER-B $\leq_L$ UBW.*

4. *SOL-MIN-VERTEX COVER-B $\leq_L$ SOL-MIN-UBCCI and SOL-MIN-UBQCI.*

5. *SOL-MIN-VERTEX COVER-B $\leq_L$ SOL-MIN-UBGc.*

6. *SOL-MAX-CLIQUE $\leq_L$ SOL-MAX-BCC.*

7. *SOL-MAX-BCC $\leq_L$ SOL-MAX-BQC.*

8. *SOL-MIN-VERTEX COVER-B $\leq_L$ SOL-MIN-FUGT[$\geq$].*

9. *SOL-MAX-X3C-B $\leq_L$ SOL-MIN-FBUT2[F_1].*

10. *SOL-MIN-FBUT2[F_α] $\leq_L$ SOL-MIN-FUDT[F_α]* ($\alpha \in \{1, 2\}$).

Proof:

Proof of (1): Consider the reduction from VC to CS (= UBCCS and UBQCS) given in [DJS86] (see Table 9). As $OPT_{VC_B} \geq |E|/B$ and $OPT_{CS} \leq 2|E|$, then $OPT_{CS} \leq 2BOPT_{VC_B}$, satisfying condition (L1) with $\alpha = 2B$. As both problems are minimization problems, condition (L2) can be rewritten as $c_{VC_B} \leq OPT_{VC_B} + \beta(c_{CS} - OPT_{CS})$. For any canonical solution for UBCCS, $c_{VC_B} = c_{CS} - |E|$; moreover, such a solution is guaranteed by Part 1 of Lemma 44. Setting $\beta = 1$ makes condition (L2) equivalent to $c_{VC_B} \leq c_{CS} - |E|$. Hence, this reduction is an L-reduction.

Proof of (2): Consider the reduction from VC to Do (= UBCDo and UBQDo) given in [DJS86] (see Table 9). As $OPT_{VC_B} \geq |E|/B$, $OPT_{Do} \leq 3|V|+2|E|$, and $|V| \leq |E|$, then $OPT_{Do} \leq 5BOPT_{VC_B}$, satisfying condition (L1) with $\alpha = 5B$. The remainder of the proof follows that for (1), substituting the appropriate part of Lemma 44 to obtain $\beta = 1$.

Proofs of (3 - 5): The proof for (3) is identical to (1). The proof of (4) is a variant of that for (2) which uses the reduction given in Table 10 to yield an L-reduction with $\alpha = 5B$ and $\beta = 1$. As the Generalized parsimony criterion can simulate any ordered phylogenetic parsimony problem, (5) can be proved by a variant on any of the proofs for (1 - 4).

Proofs of (6 - 7): By the reductions given in Table 14, solutions to SOL-MAX-BCC (SOL-MAX-BQC) yield solutions to SOL-MAX-CLIQUE (SOL-MAX-BCC) of the same cost. Hence, these reductions yield L-reductions with $\alpha = \beta = 1$.

Proof of (8): The reduction given in [BP89] which shows the MAX SNP-hardness of SOL-MIN-STEINER TREE IN GRAPHS is actually from SOL-MIN-VERTEX COVER-B to SOL-MIN-STEINER(1,2), a version of SOL-MIN-STEINER TREE IN GRAPHS whose input is complete graphs with edge-lengths $\in \{1, 2\}$. However, SOL-MIN-STEINER(1,2) is a subproblem of SOL-MIN-FUGT[$\geq$]. As all solutions to any instance of SOL-MIN-STEINER(1,2) will satisfy the dominance condition, SOL-MIN-VERTEX COVER-B L-reduces to SOL-MIN-FUGT[$\geq$] with $\alpha = 2B$ and $\beta = 1$.

Proof of (9): Consider the reduction from X3C to FBUT2[F_1] given in [KM86] (see Table 17). Note that in a B-bounded instance of X3C, $3(B-1)+1$ is the maximum number of 3-sets that can share one of the values of a particular 3-set. Hence, the selection of any 3-set can prevent the selection of at most $3(B-1)$ other 3-sets in C; thus, $OPT_{X3C_B} \geq |C|/(3(B-1)+1)$. As $OPT_{FBUT2[F_1]} \leq |E|$, and $|E| = 21|C|$, then $OPT_{FBUT2[F_1]} \leq 21(3(B-1)+1)OPT_{X3C_B}$, satisfying condition (L1) with $\alpha = 21(3(B-1)+1)$. As X3C is a maximization problem and FBUT2[F_1] is a minimization problem, condition (L2) can be rewritten as $c_{X3C_B} \geq OPT_{X3C_B} - \beta(c_{FBUT2[F_1]} - OPT_{FBUT2[F_1]})$. For any canonical solution for FBUT2[F_1], $c_{X3C_B} = (|E| - c_{FBUT2[F_1]})/3 - 3|C|$; moreover, such a solution is guaranteed by Part 4 of Lemma 44. Setting $\beta = 1/3$ makes condition (L2) equivalent to $c_{X3C_B} \geq (|E| - c_{FBUT2[F_1]})/3 - 3|C|$. Hence, this reduction is an

L-reduction.

Proof of (10): Consider the reduction from FBUT2$[F_\alpha]$ to FUDT$[F_\alpha]$ $\alpha \in \{1,2\}$ given in [Day87] (see Table 17). As $OPT_{FBUT2[F_\alpha]} = OPT_{FUDT[F_\alpha]}$, condition (L1) is satisfied with $\alpha = 1$. As both problems are minimization problems, condition (L2) can be rewritten as $c_{FUBT2[F_\alpha]} \leq OPT_{FUBT2[F_\alpha]} + \beta(c_{FUDT[F_\alpha]} - OPT_{FUDT[F_\alpha]})$. For any canonical solution to FUDT$[F_\alpha]$, $c_{FUDT[F_\alpha]} = c_{FBUT2[F_\alpha]} + Y_\alpha + Z_\alpha$, where $Y_\alpha = 0$ and $Z_\alpha \geq 0$ [Day87, p. 465]; moreover, such a solution is guaranteed by Part 5 of Lemma 44. Setting $\beta = 1$ makes condition (L2) equivalent to $c_{FUBT2[F_\alpha]} \leq c_{FUDT[F_\alpha]}$. Hence, this reduction is an L-reduction. ∎

The arithmetic equivalence reductions from FBUT$[F_1]$ to FBUT2$[F_2]$ and FBUT$[F_1, \geq]$ to FBUT2$[F_2, \geq]$ given in Section 3.2.3 are L-reductions with $\alpha = \beta = 1$. However, the reduction from FUDT$[F_1]$ to FUDT$[F]$ does *not* seem to be an L-reduction; though condition (L1) is satisfied ($\alpha = 100$), condition (L2) does not hold under any constant β.

Corollary 46 *No phylogenetic inference optimal-cost solution problem examined in this thesis (excluding SOL-MIN-FUDT[F]) has a PTAS unless $P = NP$.*

Less dramatic but nonetheless intriguing PTAS non-approximability results can be derived using Theorem 38. A PTAS for an OptP$[f(n)]$-complete evaluation problem X implies that there exists for each c, $0 < c \leq 1$, and all instance I of X, a polynomial-time algorithm A such that $c \geq \frac{|A_X(I) - OPT_X(I)|}{OPT_X(I)}(\frac{|A_X(I) - OPT_X(I)|}{A_X(I)}) \geq$

$\frac{|A_X(I) - OPT_X(I)|}{2^{f(n)}}$. By Theorem 38, there exists an $\epsilon > 0$ such that $\frac{|A_X(I) - OPT_X(I)|}{2^{f(n)}} \geq \frac{1}{2}\frac{2^{f(n^\epsilon)}}{2^{f(n)}}$ infinitely often for each such A unless P = NP. For certain f this lower bound is a positive-valued function, which implies that polynomial-time algorithms for certain ϵ, and hence PTAS, do not exist for X.

Theorem 47 *If a smooth function $f(n) \in O(\log n)$ is such that $g(n) = \frac{2^{f(n^\epsilon)}}{2^{f(n)}}$ and $\lim_{n \to \infty} g(n) > 0$ for all $\epsilon > 0$, then no OptP[$f(n)$]-complete problem has a PTAS unless $P = NP$.*

Corollary 48 *No OptP[$c \log \log n + O(1)$]-complete problem has a PTAS unless $P = NP$.*

Though the relevant levels of the OptP hierarchy in these results are too low to be of consequence in this thesis, these are the first results which show that specific portions of the bounded NP query hierarchy are PTAS non-approximable.

5.4 Relative Approximability

Several of the phylogenetic inference problems examined in this thesis have relative approximations derived from approximation algorithms for related problems. STEINER TREE IN GRAPHS has a relative approximation of $1 - 2/|L|$, where $L \leq |S|$ is the number of leaves in the optimal tree [KMB81]. The algorithm guaranteeing this approximation is given in Table 23. Note that the two crucial operations in this algorithm (finding the length of, and producing, a shortest path between two given vertices) can be done in polynomial time using standard shortest-path algorithms [CLR91, Section 26] on a character-by-character basis in an implicit graph, provided there are no restrictions on character-state transitions.

Theorem 49 *All non-reticulate Wagner Linear, Wagner General, and Fitch phylogenetic parsimony optimal-cost solution problems examined in this thesis have relative approximations of* $1 - 2/|L|$.

The application of this algorithm to phylogenetic parsimony problems was discovered independently by Gusfield [Gus91, Theorem 2.1]. Indeed, the algorithm and result above also apply to the problem of constructing minimal-length trees on molecular sequences, as long as the function computing minimal evolutionary change (*edit distance*) between pairs of sequences is a metric [Gus93, Section 3]. Unfortunately, this algorithm does not seem applicable to other phylogenetic parsimony problems, as the proof that the ratio above holds depends on the existence of a path between each pair of vertices in X in the implicit graph [KMB81, Theorem 1]. All such paths may not exist in cladistic problems; moreover, it is not obvious how a character-ordering and orientation could be chosen for a qualitative problem in polynomial time such that all required paths existed, let alone how such a character-ordering or orientation could be enforced in subsequent stages of the algorithm. SOL-MAX-CLIQUE has a relative approximation of $O(\frac{n}{\log^2 n})$ [BH90] which, by the L-reductions in the last section, yields identical

Algorithm H:

Input: an undirected weighted graph $G = (V, E, d)$ and a set of vertices $S \subseteq V$.
Output: a Steiner tree, T_H, for G and S.

Steps:

1. Construct the complete undirected weighted graph $G_1 = (V_1, E_1, d_1)$ from G and S.

2. Find the minimal spanning tree, T_1, of G_1; if there are several minimal spanning trees, pick an arbitrary one.

3. Construct the subgraph, G_S, of G by replacing each edge in T_1 by its corresponding shortest path in G; if there are several shortest paths, pick an arbitrary one.

4. Find the minimal spanning tree, T_S, of G_S; if there are several minimal spanning trees, pick an arbitrary one.

5. Construct a Steiner tree, T_H, from T_S by deleting edges in T_S, if necessary, so that all leaves in T_H are in S.

Table 23: A polynomial-time relative approximation algorithm for STEINER TREE IN GRAPHS (adapted from [KMB81])

approximations for all character compatibility problems.

Theorem 50 *All character compatibility optimal-cost solution problems examined in this thesis have relative approximations of* $O(\frac{n}{\log^2 n})$.

No relative approximations are known for any of the distance matrix fitting problems examined in this thesis, though there are relative approximations for related clustering problems; see [Day92] for a review of these results.

Theorem 43 actually states that MAX-3SAT has no relative approximation ϵ for some $\epsilon > 0$ [ALMSS92, Footnote, p. 7]; thus, by Corollary 36, the L-reductions in the previous section imply bounds on relative approximability as well. Unfortunately, values of ϵ derived to date using the construction in Theorem 43 imply only trivial lower bounds [Joh92, pp. 519–520]. Other estimates for these bounds may be derived from the best known relative approximations on SOL-MIN-VERTEX COVER-B and SOL-MAX-X3C-B:

- SOL-MIN-VERTEX COVER-B has relative approximation $c = \{0.25, 0.25, 0.50, 0.56, 0.60, 0.64, 0.67, 0.691, 0.71\}$ for $3 \leq B \leq 11$, and $c \leq 2(\frac{B^2}{B^2+2B-1}) - 1$ for $B > 11$ [MS83].

- SOL-MAX-X3C-B has relative approximation $c = (B - 1)$, $B \geq 3$ [PR90, Theorem 3].

The only nontrivial lower bound on relative approximability is for the character compatibility problems, and is based on a result from [FGLSS91] as improved by Arora et al. [ALMSS92]:

Theorem 51 ([ALMSS92], Theorem 5) *There exists an $\epsilon > 0$ such that, if SOL-MAX-CLIQUE has a relative approximation of n^ϵ, then $P = NP$.*

Corollary 52 *There exists an $\epsilon > 0$ such that, if any character compatibility problem examined in this thesis has a relative approximation of n^ϵ, then $P = NP$.*

As a final note, relative approximations are known for certain counting problems. By Theorem 3.1 of [Sto85], all #P functions $f(x)$ have approximations $f_{app}(x)$ in $F\Delta_3^p$ such that $(1-\epsilon)f_{app}(x) < f(x) < (1+\epsilon)f_{app}(x)$ for all polynomials p, where $\epsilon = 1/p(|x|)$; Theorem 7.i of [KST89] extends this result to SpanP problems. Recall that an NP query can be simulated by an appropriate Σ_2^p query.

Theorem 53 *All phylogenetic inference optimal-cost, given-cost, and given limit spanning problems examined in this thesis have relative approximations of ϵ in $F\Delta_3^p$, where $\epsilon = 1/p(|I|)$ for any polynomial p.*

5.5 Approximability by Neural Networks

There has been much interest in recent years in computing approximate solutions to optimization problems using instance-specific neural networks [HT85a, HT85b]. In the discrete-time version of this model treated by Bruck and Goodman [BG90], a neural network is described by a set of two-state nodes V, a set of arcs with weights $W_{i,j}$ that specify the input from node i to node j, and a state-change threshold value T_i for each node. Let the state of node i at time t

be $V_i(t)$, and let the *state* of the network at time t be the vector $V(t)$. At each time t after initialization, the states of each vertex V_i in some subset $S \subseteq V$ are updated by

$$V_i(t+1) = \begin{cases} 1 & \text{if } \sum_{j=1}^{|V|} W_{ji}V_j(t) \geq T_j \\ -1 & \text{otherwise} \end{cases} \tag{9}$$

A state $V(t)$ is *stable* if $V(t) = V(t+1)$. Such networks are always guaranteed to get to a stable state [BG90, pp. 130–131] which corresponds to some solution to the associated problem. Consider the following restricted class of such neural networks that have symmetric weights and satisfy the following properties [BG90, p. 132].

- Each stable state corresponds to an optimal solution of the encoded instance I of the associated problem X, and that solution can be derived from this state in polynomial time.

- The network's description is of size polynomial in $|I|$.

A problem X is said to be solvable by a neural network if there exists an algorithm A_X which can, for any instance of X, generate the corresponding neural network in polynomial time. Note that such a network may potentially take exponential time to reach a stable state.

Theorem 54 ([BG90], Proposition 1) *If an NP-hard problem is solvable by a neural network then NP = co-NP.*

Corollary 55 *No phylogenetic inference optimal-cost, given-cost, or given-limit problem examined in this thesis can be solved by a neural network unless NP = co-NP.*

Alternatively, each stable state can correspond to an approximate, rather than an optimal, solution. Many traditional proofs of approximability [HS78, GJ79] can be trivially modified to show that certain approximations by neural networks for NP-hard problems are not possible unless NP = co-NP [BG90, Yao92]. Indeed, any proof in which an optimal solution can be derived in polynomial time using a given type of approximate solution can be so modified. By analogy with PTAS, define a Polynomial-Time Neural Approximation Scheme (PTNAS) for a problem X as an algorithm A which, given an instance I of X and an integer k, $k > 0$, produces in polynomial time a neural network that produces solutions whose cost is within a factor of k of optimal. The following results stated above can be rephrased in terms of approximability by neural networks:

Corollary 56 *The following hold:*

1. *If there is a PTNAS for SOL-MAX-3SAT then NP = co-NP.*

2. *There does not exist a PTNAS for any MAX SNP-hard problem unless NP = co-NP.*

3. *There exists an $\epsilon > 0$ such that, if SOL-MAX-CLIQUE has a relative approximation of n^{ϵ} by a neural network, then NP = co-NP.*

Proof: *Proofs of (1) and (3)* (sketch): Construct the exact-solution neural networks for every problem in NP from the assumed PTNAS for SOL-MAX-3SAT and SOL-MAX-CLIQUE by using the generic reductions from all languages in NP to MAX-3SAT and MAX-CLIQUE given in the original proofs in [ALMSS92, FGLSS91] as neural network description-encoding and solution-decoding functions. As NP-complete problems are by definition NP-hard, the results hold by Theorem 54.

Note that unlike the proofs given in [BG90, Yao92], these proofs do *not* involve deriving optimal-cost solution neural networks for SOL-MAX-3SAT and SOL-MAX-CLIQUE from their respective PTNAS.

Proof of (2): Note that L-reductions preserve PTNAS-approximability as well as PTAS-approximability. ∎

Corollary 57 *No phylogenetic inference optimal-cost solution problem examined in this thesis has a PTNAS unless NP = co-NP.*

A construction similar to that in Corollary 56 can also be used to show that no MAX SNP-hard problem has a randomized PTAS (RPTAS) [BS92, KL83], i.e. a PTAS which for each ϵ, $0 < \epsilon < 1$, guarantees a solution with the required cost with at least probability $(1 - \epsilon)$, unless R = NP [Joh92, p. 519].

The results in this section apply only to the restricted class of neural networks considered in [BG90]. Less constrained types of neural networks may exist for these problems; for example, Jagota [Jag92] has designed asymmetric-weighted neural networks for MAX-CLIQUE that perform extremely well on average.

5.6 Summary

The known theoretical and algorithmic lower limits on approximability for the phylogenetic inference problems examined in this thesis are given in Table 24. Though the logic-formulation of approximability has produced the most dramatic results, the various theorems derived using the work of [GJ79, Kre88] should not be dismissed, as these theorems establish a tentative connection between various types of approximability and the levels of the function bounded NP query hierarchy. Though the correspondence is not exact ([Kre88, p. 492]; [CP91, p. 243]), there is a pattern of approximability and non-approximability (see Table 25). This pattern may assume greater significance in the light of future discoveries of lower limits on approximability.

The results above imply that polynomial-time algorithms whose approximation bounds hold over all instances do not exist for any phylogenetic inference optimal-cost solution problem for any of the closest types of bounds (i.e. absolute approximation, FPTAS, PTAS). These results do not invalidate either existing phylogenetic inference approximation algorithms or phylogenies produced by these algorithms — other kinds of fast approximation may be possible (e.g. asymptotic approximations, whose bounds hold for all but finitely many

		Approximability	
		Theoretical Lower Limit	Algorithmic Lower Limit
Phylogenetic Parsimony	WL, WG, Fi	ϵ rel. app., $\epsilon > 0$	$2(1 - \frac{1}{\lvert S \rvert}) - 1$ rel. app.
	CS, Do, CI, Ge		-
Character Compatibility		n^ϵ rel. app., $\epsilon > 0$	$O(\frac{n}{\log^2 n})$ rel. app.
Distance Matrix Fitting	FUDT[F]	no $O(poly)$ abs. app., no FPTAS	-
	All Others	ϵ rel. app., $\epsilon > 0$	

Table 24: Approximability of phylogenetic inference optimal-cost solution functions.

	Absolute Approximations			FPTAS	PTAS
	$o(\log n)$	$o(poly)$	$O(poly)$		
$FP^{NP[c \log \log n + O(1)]}$	X	–	–	X	X †
$FP^{NP[O(\log n)]}$	X	X	?	X	√
$FP^{NP}_{\parallel}$	X	X	X	?	√
FP^{NP}	X	X	X	√	√

X	=	Whole class is non-approximable.
√	=	Members of class are approximable.
–	=	Approximability not relevant.

Table 25: Non-approximability of various levels of the Function Bounded NP Query Hierarchy.

† Applies only to complete problems.

instances; subexponential-time approximation), and phylogenies derived from instances encountered in practice may be among those that are close to optimal. However, in any application such as phylogenetic inference in which the degree of optimality of approximate solutions is important (see Section 1), no approximation algorithm or solution produced by such an algorithm should be trusted until analysis has shown exactly how good an approximation that algorithm gives.

6 Conclusion

In this thesis, I have established a framework that incorporates all phylogenetic inference decision problems studied to date. Within this framework, I have derived various bounds on the evaluation, solution, spanning, enumeration, and random-generation versions of the optimal-cost, given-cost, and given-limit phylogenetic inference problems. I have also derived lower bounds on the approximability of phylogenetic inference solution problems. These results are summarized in Table 20 and 24. These results show yet again that decision problems conceal many facets of the complexity of their underlying optimization problems. The complexity of more complex versions of optimization problems should be investigated not only to better assess the true difficulty of the underlying problems, but also because such complexities may have ramifications for how closely these problems can be approximated by fast algorithms.

Future directions for research are:

- Determining the precise complexity of phylogenetic inference evaluation and optimal-cost solution problems. If these problems are provably easier than FP^{NP}, more classes will need to be described between $FP^{NP[O(\log n)]}$ and FP^{NP}. Such a set of classes might belong to the function analogue of the hierarchy developed in [CS92].
- Determining the precise complexity of phylogenetic inference spanning and

enumeration problems. This may be possible using classes from the hierarchies of functions defined in [Kre92a, Lad89, WagK86a, WagK86b].

- Finding algorithms with guaranteed relative approximations for the distance matrix fitting problems and the remainder of the phylogenetic parsimony problems. The latter may be possible by recently-developed algorithms that improve on that given in [KMB81]; see [BR91] and references.

- Deriving approximability results for phylogenetic inference given-limit and given-cost problems, based not on algorithms that guarantee solutions of a particular cost but algorithms that are either polynomial-time or correct on all but some polynomially-bounded subset of their instances. Such a framework is described in [SchU86, Section 3] and [BDG90, Section 6]. This framework is also applicable to optimal-cost solution functions.

Results from the growing literature on computational learning theory [Kea90, LV90a, MIIJ89] and the computational complexity of local search heuristics [JFY88, Yan90] may also be applicable to the further analysis of phylogenetic inference problems.

References

[ABG91] Amir, A., Beigel, R., and Gasarch, W. I. *Some Connections between Bounded Query Classes and Non-Uniform Complexity.* Manuscript, 1991.

[ALMSS92] Arora, S., Lund, C., Motwani, R., Sudan, M., and Szegedy, M. *Proof Verification and Intractability of Approximation Problems.* Manuscript, 1992.

[ADS86] Ausiello, G., D'Atri, A., and Sacca, D. Minimal Representation of Directed Hypergraphs. *SIAM Journal on Computing*, 15(2), 419–431, 1986.

[ANI90] Ausiello, G., Nanni, U., and Italiano, G. F. Dynamic Maintenance of Directed Hypergraphs. *Theoretical Computer Science*, 72, 97 117, 1990.

[Ax87] Ax, P. *The Phylogenetic System: The Systematization of Organisms on the Basis of their Phylogenies.* Translated by R. P. S. Jeffries. John Wiley, New York, 1987.

[BDG88] Balcázar, J., Díaz, J., and Gabarró, J. *Structural Complexity I.* EATCS Monographs on Theoretical Computer Science no. 11. Springer Verlag, Berlin, 1988.

[BDG90] Balcázar, J., Díaz, J., and Gabarró, J. *Structural Complexity II.* EATCS Monographs on Theoretical Computer Science no. 22. Springer Verlag, Berlin, 1988.

[BFMY83] Beeri, C., Fagin, R., Maier, D., and Yannakakis, M. On the Desirability of Acyclic Database Schemes. *Journal of the Association for Computing Machinery*, 30(3), 479–513, 1983.

[Bei88] Beigel, R. *NP-hard Sets are p-superterse unless R=NP.* Technical Report 4, The Johns Hopkins University, Department of Computer Science, 1988.

[Bei91] Beigel, R. Bounded Queries to SAT and the Boolean hierarchy. *Theoretical Computer Science*, 84, 199–223, 1991.

[BS92] Berman, P. and Schnitger, G. On the Complexity of Approximating the Independent Set Problem. *Information and Computation*, 96, 77–94, 1992.

[BP89] Bern, M. and Plassmann, P. The Steiner Problem With Edge Lengths 1 and 2. *Information Processing Letters*, 32, 171–176, 1989.

[Ber73] Berge, C. *Graphs and Hypergraphs.* Translated by E. Minieka. North-Holland, Amsterdam, 1973.

[Ber85] Berge, C. *Graphs.* Second revised edition. North-Holland, Amsterdam, 1985.

[BR91] Berman, P. and Ramaiyer, V. *Improved Approximations for the Steiner Tree Problem.* Manuscript, 1991.

[BSS89] Blum, L, Shub, M., and Smale, S. On a Theory of Computation and Complexity over the Real Numbers: *NP*-completeness, Recursive Functions, and Universal Machines. *Bulletin of the American Mathematical Society (New Series)*, 21(1), 1–46, 1989.

[BH90] Boppana, R. and Halldórsson, M. M. Approximating Maximum Independent Sets by Excluding Subgraphs. In J. R. Gilbert and R. Karlsson (eds.) *SWAT' 90.* Lecture Notes in Computer Science no. 447, Springer-Verlag, Berlin, 1990. 13–25.

[BG90] Bruck, J., and Goodman, J. W. On the Power of Neural Networks for Solving Hard Problems. *Journal of Complexity*, 6, 129–135, 1990.

[BJY89] Bruschi, D., Joseph, D., and Young, P. A Structural Overview of NP Optimization Problems. In H. Djidjev (ed.) *Optimal Algorithms: Proceedings of the International Symposium*, Lecture Notes in Computer Science no. 401, Springer-Verlag, Berlin, 1989. 205–231.

[CS65] Camin, J. H., and Sokal, R. R. A Method for Deducing Branching Sequences in Phylogeny. *Evolution*, 19, 311–326, 1965.

[CS92] Castro, J., and Seara, C. Characterizations of Some Complexity Classes between Θ_2^p and Δ_2^p. In A. Finkel and M. Jantzen (eds.) *STACS '92: 9th Annual Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science no. 577, Springer-Verlag, Berlin, 1992. 305–318.

[CF87] Cavender, J. A. and Felsenstein, J. S. Invariants of Phylogenies in a Simple Case with Discrete States. *Journal of Classification*, 4, 57–71, 1987.

[CE67] Cavalli-Sforza, L. L. and Edwards, A. W. F. Phylogenetic Analysis: Models and Estimation Procedures. *American Journal of Human Genetics*, 19, 233–257, 1967; see also *Evolution*, 21, 550–570, 1967.

[CT91] Chen, Z.-Z. and Toda, S. *On the Complexity of Computing Optimal Solutions*. Manuscript, 1991.

[CLR91] Cormen, T. H., Leiserson, C. E., and Rivest, R. L. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1991.

[CFS91] Crescenzi, P., Fiorini, C., and Silvestri, R. A Note on the Approximation of the MAX CLIQUE Problem. *Information Processing Letters*, 40(1), 1–5, 1991.

[CP91] Crescenzi, P, and Panconesi, A. Completeness in Approximation Classes. *Information and Control*, 93, 241–262, 1991.

[Day83] Day, W. H. E. Computationally Difficult Parsimony Problems in Phylogenetic Systematics. *Journal of Theoretical Biology*, 103, 429–438, 1983.

[Day87] Day, W. H. E. Computational Complexity of Inferring Phylogenies from Dissimilarity Matrices. *Bulletin of Mathematical Biology*, 49(4), 461–467, 1987.

[Day88] Day, W. H. E. Class Notes, *Computer Science 6758: Special Topics in Computer Applications*, 1988.

[Day92] Day, W. H. E. Complexity Theory: An Introduction for Practitioners of Classification. In P. Arabie, G. de Soete, and L. J. Hubert (eds.) *Clustering and Classification*, World Scientific Publishing, Teaneck, NJ, 1992.

[DJS86] Day, W. H. E., Johnson, D. S., and Sankoff, D. The Computational Complexity of Inferring Rooted Phylogenies by Parsimony. *Mathematical Biosciences*, 81, 33–42, 1986.

[DS86] Day, W. H. E. and Sankoff , D. Computational Complexity of Inferring Phylogenies by Compatibility. *Systematic Zoology*, 35(2), 224–229, 1986.

[DS87] Day, W. H. E. and Sankoff, D. Computational Complexity of Inferring Phylogenies from Chromosome Inversion Data. *Journal of Theoretical Biology*, 124, 213–218, 1987.

[DT90] Díaz, J. and Torán, J. Classes of Bounded Nondeterminism. *Mathematical Systems Theory*, 23(1), 21–32, 1990.

[DRA92] Dorado, O., Rieseberg, L. H., and Arias, D. M. Chloroplast DNA Introgression in Southern California Sunflowers. *Evolution*, 46(2), 566–572, 1992.

[Duk85] Duke, R. Types of Cycles in Hypergraphs. *Annals of Discrete Mathematics*, 27, 399–418, 1985.

[EC80] Eldredge, N. and Cracraft, J. *Phylogenetic Patterns and the Evolutionary Process: Method and Theory in Comparative Biology.* Columbia University Press, New York, 1980.

[EJM76] Estabrook, G. F., Johnson, C. S., and McMorris, F. R. An Algebraic Analysis of Discrete Characters. *Discrete Mathematics*, 16, 141–147, 1976.

[EM77] Estabrook, G. F. and McMorris, F. R. When are Two Qualitative Taxonomic Characters Compatible? *Journal of Mathematical Biology*, 4, 195–200, 1977.

[EM80] Estabrook, G. F. and McMorris, F. R. When is One Estimate of Evolutionary Relationships a Refinement of Another? *Journal of Mathematical Biology*, 10, 367–374, 1980.

[Fag74] Fagin, R. Generalized First-order Spectra and Polynomial Time Recognizable Sets. In R. M. Karp (ed.) *Complexity of Computations*, SIAM-AMS Proceedings no. 7. American Mathematical Society, Providence, RI, 1974. 43–73.

[Fag83] Fagin, R. Degrees of Acyclicity for Hypergraphs and Relational Database Schemes. *Journal of the Association for Computing Machinery*, 30(3), 514–550, 1983.

[Far72] Farris, J. S. Estimating Phylogenetic Trees from Distance Matrices. *American Naturalist*, 106, 645–688, 1972.

[Far77] Farris, J. S. Phylogenetic Analysis under Dollo's Law. *Systematic Zoology*, 26, 77–88, 1977.

[Far78] Farris, J. S. Inferring Phylogenetic Trees from Chromosome Inversion Data. *Systematic Zoology*, 27, 275–284, 1978.

[Far83] Farris, J. S. The Logical Basis of Phylogenetic Analysis. In N. I. Platnick and V. A. Funk (eds.) *Advances in Cladistics, Volume 2: Proceedings of the Second Meeting of the Willi Hennig Society*, Columbia University Press, New York, 1983. 7–36.

[FGLSS91] Feige, U., Goldwasser, S., Lovasz, L., Safra, M., and Szegedy, M. Approximating Clique is Almost NP-Complete. In *Proceedings of the Thirty-Second Annual IEEE Symposium on the Foundations of Computer Science*, IEEE Computer Society Press, Washington, D. C., 1991. 2–14.

[Fel81] Felsenstein, J. S. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17, 368-376, 1981.

[Fel82] Felsenstein, J. S. How Can We Infer Geography and History from Gene Frequencies? *Journal of Theoretical Biology*, 96, 9–20, 1982.

[Fel88] Felsenstein, J. S. Phylogenies from Molecular Sequences: Inference and Reliability. *Annual Review of Genetics*, 22, 521–565, 1988.

[FHOS92] Fenner, S., Homer, S., Ogiwara, M., and Selman, A. L. *On Using Oracles That Compute Functions.* Manuscript, 1992. To appear in *STACS '93: 10th Annual Symposium on Theoretical Aspects of Computer Science.*

[Fit71] Fitch, W. M. Toward Defining the Course of Evolution: Minimal Change for a Specific Tree Topology. *Systematic Zoology*, 20, 406-41C, 1971.

[Fun85] Funk, V. A. Phylogenetic Patterns and Hybridization. *Annals of the Missouri Botanical Gardens*, 72, 681-715, 1985.

[FB90] Funk, V. A. and Brooks, D. R. *Phylogenetic Systematics as the Basis of Comparative Biology.* Smithsonian Institution Press, Washington, D. C., 1990.

[GW83] Galperin, H. and Wigderson, A. Succinct Representations of Graphs. *Information and Control*, 56, 183-198, 1983.

[GGJ77] Garey, M. R., Graham, R. L., and Johnson, D. S. The Complexity of Computing Steiner Minimal Trees. *SIAM Journal on Appled Mathematics*, 32(4), 835–859, 1977.

[GJ79] Garey, M. R. and Johnson, D. S. *Computers and Intractability.* W. H. Freeman, New York, 1979.

[Gas86] Gasarch, W. I. *The Complexity of Optimization Functions.* Technical Report no. 1652, University of Maryland, Department of Computer Science, 1986.

[Gas92] Gasarch, W. I. Personal communication, July 11, 1992.

[GKR92] Gasarch, W. I., Krentel, M. W., and Rappoport, K. J. *OptP as the Normal Behavior of NP-Complete Problems.* Manuscript, 1992. To appear in *Mathematical Systems Theory.*

[GF82] Graham, R. L. and Foulds, L. R. Unlikelihood That Minimal Phylogenies for a Realistic Biological Study Can Be Constructed in Reasonable Computational Time. *Mathematical Biosciences*, 60, 133–142, 1982.

[Gra81] Grant, V. *Plant Speciation.* Second edition. Columbia University Press, New York, 1981.

[Gus91] Gusfield, D. *The Steiner Tree Problem, Historical Reconstruction, and Phylogeny.* Manuscript, 1991. This is a revised version of Technical Report 332, Department of Computer Science, Yale University, 1984, by the same author.

[Gus93] Gusfield, D. Efficient Methods for Multiple Sequence Alignment with Guaranteed Error Bounds. *Bulletin of Mathematical Biology*, 55(1), 141–154, 1993.

[Hei90] Hein, J. Reconstructing Evolution of Sequences Subject to Recombination Using Parsimony. *Mathematical Biosciences*, 98, 185–200, 1990.

[HHSY91] Hemachandra, L., Hoene, A., Siefkes, D., and Young, P. On Sets Polynomially Enumerable by Iteration. *Theoretical Computer Science*, 80, 203–225, 1991.

[HP84] Hendy, M. D. and Penny, D. Cladograms Should Be Called Trees. *Systematic Zoology*, 33(2), 245–247, 1984.

[Hen66] Hennig, W. *Phylogenetic Systematics*. Translated by D. D. Davis and R. Zangerl. University of Illinois Press, Urbana, IL, 1966.

[HT85a] Hopfield, J. J., and Tank, D. W. Neural Computations of Decisions in Optimization Problems. *Biological Cybernetics*, 52, 141–152, 1985.

[HT85b] Hopfield, J. J., and Tank, D. W. Computing with Neural Circuits: A Model. *Science*, 233, 625–633, 1985.

[HS78] Horowitz, E. and Sahni, S. *Fundamentals of Computer Algorithms*. Computer Science Press, Rockille, MA, 1978.

[Hum83] Humphries, C. J. Primary Data in Hybrid Analysis. In N. I. Platnick and V. A. Funk (eds.) *Advances in Cladistics, Volume 2: Proceedings of the Second Meeting of the Willi Hennig Society*, Columbia University Press, New York, 1983. 89–103.

[Jag92] Jagota, A. *Efficiently Approximating MAX-CLIQUE in a Hopfield-style Network.* To appear in *International Joint Conference on Neural Networks*, IEEE Computer Society Press, Washington, D. .C, 1992.

[JS71] Jardine, N. and Sibson, R. *Mathematical Taxonomy.* John Wiley, London, 1971.

[JVV86] Jerrum, M. R., Valiant, L. G., and Vazirani, V. V. Random Generation of Combinatorial Structures from a Uniform Distribution. *Theoretical Computer Science*, 43, 169–188, 1986.

[Joh74] Johnson, D. S. Approximation Algorithms for Combinatorial Problems. *Journal of Computer and System Sciences*, 9, 256–278, 1974.

[Joh90] Johnson, D. S. A Catalog of Complexity Classes. In J. van Leeuwen (ed.) *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, MIT Press, Cambridge, MA, 1990. 69-161.

[Joh92] Johnson, D. S. The NP-completeness Column: An Ongoing Guide (23rd Edition). *Journal of Algorithms*, 13, 502–524, 1992.

[JPY88] Johnson, D. S., Papadimitriou, C. H., and Yannakakis, M. How Easy is Local Search? *Journal of Computer and System Sciences*, 37, 79–100, 1988.

[Kan91] Kann, V. Maximum Bounded 3-dimensional Matching is MAX SNP-complete. *Information Processing Letters*, 37, 27–35, 1991.

[Kar72] Karp, R. M. Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Thatcher (eds.) *Complexity of Computer Computations*, Plenum Press, New York, 1972. 85–103.

[KL83] Karp, R. M. and Luby, M. Monte-Carlo Algorithms for Enumeration and Reliability Problems. In *Proceedings of the Twenty-Fourth Annual IEEE Symposium on the Foundations of Computer Science*, IEEE Computer Society Press, Washington, D. C., 1983. 56–64.

[Kea90] Kearns, M. J. *The Computational Complexity of Machine Learning*. MIT Press, Cambridge, MA, 1990.

[KF69] Kluge, A. G. and Farris, J. S. Quantitative Phyletics and the Evolution of Anurans. *Systematic Zoology*, 18(1), 1–32, 1969.

[Ko91] Ko, K.-I. *Complexity Theory of Real Functions*. Birkhäuser, Boston, MA, 1991.

[Kob92] Köbler, J. Personal communication, July 23, 1992.

[KST89] Köbler, J., Schöning, U., and Torán, J. On Counting and Approximation. *Acta Informatica*, 26, 363–379, 1989.

[KT90] Kolaitis, P. G., and Thakur, M. N. *Logical Definability of NP Optimization Problems.* Technical report UCSC-CRL-90-48. University of California at Santa Cruz, Department of Computer and Information Sciences, 1990. To appear in *Information and Computation.*

[KT91] Kolaitis, P. G., and Thakur, M. N. Approximation Properties of NP Minimization Classes. In *Proceedings of the Sixth Annual Conference on Structure in Complexity Theory.* IEEE Computer Society Press, Washington, D. C., 1991. 353–366. To appear in *Journal of Computer and System Sciences.*

[KMB81] Kou, L., Markowsky, G., and Berman, L. A Fast Algorithm for Steiner Trees. *Acta Informatica*, 15, 141–145, 1981.

[Kre88] Krentel, M. W. The Complexity of Optimization Problems. *Journal of Computer and System Sciences*, 36(3), 490–509, 1988.

[Kre92a] Krentel, M. W. Generalizations of OptP to the Polynomial Hierarchy. *Theoretical Computer Science*, 97(2), 183–198, 1992.

[Kre92b] Krentel, M. W. Personal communication, July 13, 1992.

[Kri86] Křivánek, M. On the Computational Complexity of Clustering. In E. Diday, Y. Escoufier, L. Lebart, J. Pages, Y. Schektman, and R. Tomassone (eds.) *Data Analysis and Informatics IV: Proceedings of the Fourth International Symposium on Data Analysis and Informatics*, Elsevier Science (North-Holland), Amsterdam, 1986. 89–96.

[Kri88] Křivánek, M. The Complexity of Ultrametric Partitions on Graphs. *Information Processing Letters*, 27, 265–270, 1988.

[KM86] Křivánek, M. and Morávek, J. NP-Hard Problems in Hierarchical Tree Clustering. *Acta Informatica*, 23, 311–323, 1986.

[Lad89] Ladner, R. E. Polynomial Space Counting Problems. *SIAM Journal on Computing*, 18(6), 1087–1097, 1989.

[Lak87] Lake, J. A. A Rate-Independent Technique for Analysis of Nucleic Acid Sequences: Evolutionary Parsimony. *Molecular Biology and Evolution*, 4(2), 167-191, 1987.

[Lat82] Lathrop, G. M. Evolutionary Trees and Admixture: Phylogenetic Inference When Some Populations are Hybridized. *Annals of Human Genetics*, 46, 245–255, 1982.

[Lee88] Lee, A. R. BLUDGEON: A Blunt Instrument for the Analysis of Contamination in Textual Traditions. In Y. Choueka (ed.) *Computers in Linguistic and Literary Computing: Literary and Linguistic Computing 1988: Proceedings of the Fifteenth Annual Conference*, Champion-Slatkine, Paris, 1990. 261–292.

[LW92] Lengauer, T. and Wagner, K. W. The Correlation between the Complexities of the Nonhierarchical and Hierarchical Versions of Graph Problems. *Journal of Computer and System Sciences*, 44(1), 63–93, 1992.

[LV90a] Li, M. and Vitányi, P. M. B. Inductive Reasoning and Kolmogorov Complexity. *Journal of Computer and System Sciences*, 44, 343–384, 1992.

[LV90b] Li, M. and Vitányi, P. M. B. Kolmogorov Complexity and Its Applications. In J. van Leeuwen (ed.) *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, MIT Press, Cambridge, MA, 1990. 187–254.

[Lip92] Lipscomb, D. L. Parsimony, Homology, and the Analysis of Multistate Characters. *Cladistics*, 8(1), 45–65, 1992.

[LP85] Luckow, M. and Pimentel, R. A. An Empirical Comparison of Numerical Wagner Computer Programs. *Cladistics*, 1(4), 47–66, 1985.

[McD90] McDade, L. Hybrids and Phylogenetic Systematics I. Patterns of Character Expression in Hybrids and their Implications for Cladistic Analysis. *Evolution*, 44(6), 1685 1700, 1990.

[McM77] McMorris, F. R. On the Compatibility of Binary Qualitative Taxonomic Characters. *Bulletin of Mathematical Biology*, 39, 133 138, 1977.

[Mad91] Maddison, D. R. The Discovery and Importance of Multiple Islands of Most-Parsimonious Trees. *Systematic Zoology*, 40(3), 304 314, 1991.

[MRS92] Maddison, D. R., Ruvolo, M. and Swofford, D. L. Geographic Origins of Human Mitochondrial DNA: Phylogenetic Evidence from Control Region Sequences. *Systematic Biology*, 41(1), 111 124, 1992.

[ME85] Meacham, C. A. and Estabrook, G. F. Compatibility Methods in Systematics. *Annual Review of Ecology and Systematics*, 16, 431-446, 1985..

[MS72] Meyer, A. R. and Stockmeyer, L. J. The Equivalence Problem for Regular Expressions with Squaring Involves Exponential Space. In the *13th Annual IEEE Symposium on Switching and Automata*

Theory, IEEE Computer Society Press, Washington, D. C., 1972. 125–129.

[Mic82] Mickevich, M. F. Transformation Series Analysis. *Systematic Zoology*, 31(4), 461–478, 1982.

[MHJ89] Milosavljević, A., Haussler, D., and Jurka, J. Informed Parsimonious Inference of Prototypical Genetic Sequences. In R. Rivest, D. Haussler, and M. K. Warmuth (eds.) *COLT '89: Proceedings of the Second Annual Workshop on Computational Learning Theory*, Morgan Kaufmann Publishers, San Mateo, CA, 1989. 102–117.

[MS83] Monien, B. and Speckenmeyer, E. Some Further Approximation Algorithms for the Vertex Cover Problem. In G. Ausiello and M. Protasi (eds.) *CAAP '83*. Lecture Notes in Computer Science no. 159. Springer-Verlag, Berlin, 1983. 341–349.

[MH90] Moritz, C. and D. M. Hillis Molecular Systematics: Context and Controversies. In D. M. Hillis and C. Moritz (eds.) *Molecular Systematics*, Sinauer Associates, Sunderland, MA, 1990. 1–10.

[Mot92] Motwani, R. *Lecture Notes on Approximation Algorithms: Part I*. Manuscript, 1992.

[Nei87] Nei, M. *Molecular Evolutionary Genetics*. Columbia University Press, New York, 1987.

[Nel83] Nelson, G. Reticulation in Cladograms. In Platnick, N. I. and V. A. Funk (eds.) *Advances in Cladistics, Volume 2: Proceedings of the Second Meeting of the Willi Hennig Society*, Columbia University Press, New York, 1983. 105–111.

[NP81] Nelson, G. and Platnick, N. I. *Systematics and Biogeography: Cladistics and Vicariance*. Columbia University Press, New York, 1981.

[Nig75] Nigmatullin, R. G. Complexity of the Approximate Solution of Combinatorial Problems. *Doklady Akademii Nauk SSSR*, 224, 289–292, 1975 (in Russian). English translation (incorporating author's corrections) in *Soviet Mathematics Doklady*, 16, 1199–1203, 1975.

[OM90] Orponen, P., and Mannila, H. *On Approximation-Preserving Reductions: Complete Problems and Robust Measures.* Manuscript, 1990.

[PR90] Panconesi, A. and Ranjan, D. Quantifiers and Approximation (Extended Abstract). In *Proceedings of the 22nd ACM Symposium on Theory of Computing*, ACM Press, Washington, D. C., 1990. 446–456.

[PY86] Papadimitriou, C. H. and Yannakakis, M. A Note on Succinct Representations of Graphs. *Information and Control*, 71, 181–185, 1986.

[PY91] Papadimitriou, C. H. and Yannakakis, M. Optimization, Approximation, and Complexity Classes. *Journal of Computer and System Sciences*, 43, 425–440, 1991.

[PHS92] Penny, D., Hendy, M. D., and Stell, M. A. Progress with Methods for Constructing Evolutionary Trees. *Trends in Ecology and Evolution*, 7(3), 73–79, 1992.

[Phi84] Phipps, J. B. Problems of Hybridity in the Cladistics of *Crataegus* (Rosaceae). In W. F. Cerant (ed.) *Plant Biosystematics*. Academic Press, Toronto, 1984. 417–438.

[Pla89] Platnick, N. I. An Empirical Comparison of Microcomputer Parsimony Programs, II. *Cladistics*, 5(2), 145–161, 1989.

[PW76] Prager, E. M. and Wilson, A. C. Congruency of Phylogenies Derived from Different Proteins: A Molecular Analysis of the Phylogenetic Position of Cracid Birds. *Journal of Molecular Evolution*, 9, 45-57, 1976.

[Ric89] Richards, D. Fast Heuristic Algorithms for Rectilinear Steiner Trees. *Algorithmica*, 4, 191–207, 1989.

[Riv90] Rivest, R. L. Cryptography. In J. van Leeuwen (ed.) *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, MIT Press, Cambridge, MA, 1990. 717–755.

[SC83] Sankoff, D. D. and Cedergren, R. J. Simultaneous Comparison of Three or More Sequences Related by a Tree. In D. Sankoff and J. B. Kruskal (eds.) *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, Addison-Wesley, Reading, MA, 1983. 253–263.

[SchR86] Schoch, R. M. *Phylogeny Reconstruction in Paleontology*. Van Nostrand Reinhold, New York, 1986.

[SchU86] Schöning, U. *Complexity and Structure*. Lecture Notes in Computer Science no. 211, Springer-Verlag, Berlin, 1986.

[SchU90] Schöning, U. The Power of Counting. In A. L. Selman (ed.) *Complexity Theory Retrospective*, Springer-Verlag, Berlin, 1990. 204–223.

[Sel91] Selman, A. L. *A Taxonomy of Complexity Classes of Functions*. Technical report, State University of New York at Buffalo, Department of Computer Science, 1991. To appear in *Journal of Computer and System Sciences*. An abbreviated version appeared in *Bulletin of the European Association for Theoretical Computer Science*, 45, 114–130, 1991.

[Sel92] Selman, A. L. Personal communication, July 6, 1992.

[SimH89] Simon, H. U. Continuous Reductions Among Combinatorial Optimization Problems. *Acta Informatica*, 26, 771–785, 1989.

[SimJ77] Simon, J. On the Difference between One and Many. In A. Salomaa and M. Steiny (eds.) *Automata, Languages, and Programming - 4th Colloquim*, Lecture Notes in Computer Science no. 52, Spriner-Verlag, Berlin. 480–490.

[Sne75] Sneath, P. H. A. Cladistic Representation of Reticulate Evolution. *Systematic Zoology*, 24, 360–368, 1975.

[Sny92] Snyder, T. L. On the Exact Location of Steiner Points in General Dimension. *SIAM Journal on Computing*, 21(1), 163–180, 1992.

[StaC75] Stace, C. A. Hybridization. In C. A. Stace (ed.) *Hybridization and the Flora of the British Isles*, Academic Press, London, 1975. 1-90.

[StaT80] Standish, T. A. *Data Structure Techniques.* Addison-Wesley, Reading, MA, 1980.

[Sto77] Stockmeyer, L. J. The Polynomial Hierarchy. *Theoretical Computer Science*, 3, 1–22, 1977.

[Sto85] Stockmeyer, L. J. On Approximation Algorithms for #P. *SIAM Journal on Computing*, 14, 849–861, 1985.

[SSV92] Stoneking, M., Sherry, S. T., and Vigilant, L. Geographic Origin of Human Mitochondrial DNA Revisited. *Systematic Biology*, 41(3), 384–391, 1992.

[SO90] Swofford, D. L. and Olsen, G. J. Phylogeny Reconstruction. In D. M. Hillis and C. Moritz (eds.) *Molecular Systematics*, Sinauer Associates, Sunderland, MA, 1990. 411-501.

[Tho82] Thorpe, R. S. Reticulate Evolution and Cladism: Tests for the Direction of Evolution. *Experientia*, 38, 1242-1244, 1982.

[TW92] Toda, S. and Watanabe, O. Polynomial-time 1-Turing reductions from #PH to #P. *Theoretical Computer Science*, 100(1), 205 221, 1992.

[Tor91] Torán, J. Complexity Classes Defined by Counting Quantifiers. *Journal of the Association for Computing Machinery*, 38(3), 753 774, 1991.

[Val76] Valiant, L. G. The Relative Complexity of Checking and Evaluating. *Information Processing Letters*, 5, 20–23, 1976.

[Val79a] Valiant, L. G. The Complexity of Enumeration and Reliability Problems. *SIAM Journal on Computing*, 8, 410–421, 1979.

[Val79b] Valiant, L. G. The Complexity of Computing the Permanent. *Theoretical Computer Science*, 8, 189–201, 1979.

[WagK86a] Wagner, K. W. The Complexity of Combinatorial Problems with Succinct Input Representations. *Acta Informatica*, 23, 325–356, 1986.

[WagK86b] Wagner, K. W. Some Observations on the Connection between Counting and Recursion. *Theoretical Computer Science*, 47, 131–147, 1986.

[WagK87] Wagner, K. W. More Complicated Questions about Maxima and Minima, and Some Closures of NP. *Theoretical Computer Science*, 51, 53–80, 1987.

[WagK88] Wagner, K. W. Bounded Query Computation. In the *Proceedings of the Third Annual Conference on Structure in Complexity Theory*, IEEE Computer Society Press, Washington, D. C., 1988. 260–277.

[WagK89] Wagner, K. W. *Number-of-Query Hierarchies.* Technical Report no. 4, Instut für Informatick, Bayerische Julius-Maximilians-Universität, Würzburg, 1989.

[WagK90] Wagner, K. W. Bounded Query Classes. *SIAM Journal on Computing*, 19(5), 833-846, 1990.

[WW86] Wagner, K. and Wechsung, G. *Computational Complexity Theory.* D. Reidel, Dordrecht, 1986.

[WagW68] Wagner Jr., W. H. Hybridization, Taxonomy, and Evolution. In V. H. Heywood (ed.) *Modern Methods in Plant Taxonomy.* Botanical Society of the British Isles Conference Report no. 10. Academic Press, London, 1968. 113-138.

[WagW80] Wagner Jr., W. H. Origin and Philosophy of the Groundplan-divergence Method of Cladistics. *Systematic Botany*, 5(2), 173-193, 1980.

[Wil81] Wiley, E. O. *Phylogenetics: The Theory and Practice of Phylogenetic Systematics.* John Wiley, New York, 1981.

[WSBF91] Wiley, E. O., Seigel-Causey, D., Brooks, D. R., and Funk, V. A. *The Compleat Cladist: A Primer of Phylogenetic Procedures.* Special Publication no. 19. University of Kansas Museum of Natural History, Lawrence, Kansas, 1991.

[Win87] Winter, P. Steiner Problems in Networks: A Survey. *Networks*, 17, 129–167, 1987.

[Wra77] Wrathall, C. Complete Sets and the Polynomial-time Hierarchy. *Theoretical Computer Science*, 3, 23–33, 1977.

[Yan90] Yannakakis, M. The Analysis of Local Search Problems and their Heuristics. In C. Choffrut and T. Lengauer (eds.) *STACS '90: 7th Annual Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science no. 415, Springer-Verlag, Berlin, 1990. 298–311.

[Yao92] Yao, X. Finding Approximate Solutions to NP-hard problems by Neural Networks is Hard. *Information Processing Letters*, 41(2), 93–98, 1992.

A Phylogenetic Systematics and the Inference of Reticulation

In this appendix, I will give a short review of various approaches to inferring reticulation, followed by a justification of the reticulate phylogenetic parsimony problem schemata defined in Section 3.2.1. For in-depth reviews of the topics in this appendix, see [Fun85, Gra81, SchR86, StaC75].

Reticulate events as described in Section 2.1 are part of biological evolution. Hybridization has occurred frequently in many groups of plants and less frequently among animals, notably in birds and fishes [Gra81, pp. 202 204], and introgression, a form of recombination in which characteristics are passed via hybrids from one species to another, seems to occur with greater frequency than previously thought, especially among the cytoplasmic and nuclear genes in plants and animals (see [DRA92] and references). The evolutionary significance of such reticulation has been debated for decades; for instance, hybridization has been viewed as mere noise on the underlying substrate of dichotomous evolution [WagW68], as an important force in particular groups at particular times [Gra81, pp. 179–189], and as the dominant force in plant evolution [StaC75, Lotsy (1916) quoted on p. 24]. Regardless of such debate, reticulation and its inference is crucial to many investigators.

There are many traditional biological heuristics for recognizing individual instances of hybridization and recombination, based on the intermediate nature of the produced character-states and various attributes of the proposed hybrid and its parent species e.g. geographical distribution, parental interfertility, experimental re-creation of hybrid [Gra81, StaC75]; some of these heuristics have been coded as numerical measures (*hybrid indices*) ([Gra81, pp. 207–210]; [StaC75, pp. 74–82]). Recently, algorithmic methods have been proposed for inferring hybridization under the compatibility [Sne75], maximum likelihood [Fel82, Lat82], and phylogenetic parsimony [Fun85, Hei90, Lee88, Nel83, Phi84, Tho82, WagW80] criteria. The focus in this appendix will be on those methods based on the phylogenetic parsimony criterion.

All known parsimony-based methods infer hybridization using the character conflict induced by hybrids. In a phylogenetic parsimony analysis, the theoretical lower limit on cost is that each character-state transition event occurs only once in a tree; the portion of a tree's cost above this theoretical minimum consists of additional hypotheses of character-state transition (*homoplasy*) which are required to explain character states that did not arise only once in that tree. The phylogenetic parsimony criterion, in preferring trees of minimum cost, minimizes homoplasy. When the possibility of error in character analysis has been ruled out, homoplasy is a sign that evolutionary processes not belonging to the single-transition, dichotomous-speciation model have occurred. Reticulation as defined in this thesis comprises one such set of processes.

Following [Fun85], all parsimony-based methods for inferring hybridization can be classified into three approaches, depending on how they deal with homoplasy.

1. Include reticulation implicitly via the homoplasy in the most parsimonious tree [NP81].

2. Identify and remove hybrid taxa before phylogenetic analysis, and introduce reticulation after phylogenetic analysis to accommodate these taxa on the basis of homoplasy in the most parsimonious tree [WagW80].

3. Include all taxa in the phylogenetic analysis, and introduce reticulation and hybrid taxa as necessary either during [Phi84] or after [Fun85, Lee88, Nel83, Tho82] analysis, on the basis of homoplasy.

Each of these approaches has intrinsic difficulties because reticulation can be characterized by a wide variety of character-state patterns, both within the produced taxa and within any non-reticulate tree including these taxa [Fun85, Hum83, McD90, StaC75]. Moreover, these approaches are not satisfactory for defining criterion-based problems because they are based on specific algorithms and heuristics (see Section 1).

There are no general difficulties with inferring reticulation using the parsimony criterion: reticulations remove homoplasy by unifying the occurrence of seemingly incompatible character-states into a single event, and can thus be ranked (as are trees) by the decrease in homoplasy that they induce. However, there are several specific difficulties.

1. As appropriate reticulation can represent any number of character-state transitions in one event, unbounded reticulation renders dichotomous speciation irrelevant and the phylogenetic parsimony criterion meaningless [NP81, pp. 217-218].

2. As homoplasy is used to justify the addition of reticulation, it is no longer possible to use homoplasy as a sign of possible error in character analysis and coding (the "self-illuminating" property of phylogenetic parsimony analysis [Wil81, p. 130]).

3. It is much more difficult to infer phylogenies by hand using Hennigian argumentation [Wil81, WSBF91] or by algorithm when reticulation is allowed; moreover, the produced phylogenies cannot be readily used as the basis for hierarchical Linnean classifications of species.

The first two of these difficulties are actually guidelines for the formulation of useful computational problems. By (1), a problem should only be able to infer a limited amount of well-defined reticulation for a given instance, and this limit should be under the control of the investigator. By (2), such a problem should only be invoked after a non-reticulate analysis has been performed to detect possible errors in character coding, and to determine if there is any homoplasy that can be explained by reticulation. The difficulties in (3) are consequences of searching for phylogenetic trees in a richer hypothesis-space, and must be accepted if reticulate hypotheses are desirable.

The reticulate problem schemata defined in Section 3.2.1 satisfy the first condition above, and a procedure patterned after that given in [Nel83], in which reticulations are added one at a time to the most parsimonious tree such that the homoplasy removed with each insertion is maximized, will satisfy the second. Such a procedure using these schemata would differ from that in [Nel83] in that it would be able to search over the whole space of available reticulate phylogenies, not just those that can be reached by additions of reticulation to the most parsimonious non-reticulate phylogeny, and may thus be able to find less obvious but equally valid solutions. This procedure is not immune to the problems discussed above of recognizing the patterns of homoplasy that imply reticulation, or the possibility that the observed homoplasy may have other causes e.g. multiple speciation, ecological convergence, or the inclusion of ancestral taxa in the given taxa [NP81, p. 265]. Moreover, this procedure is not so much a method for producing phylogenetic trees as an aid for exploring the space of phylogenetic hypotheses. However, this is consistent with the viewpoint that systematics does not so much derive evolutionary history as obtain successively better approximations to it.

The beauty and power of these reticulate problem schemata is that they do not depend on the precise structure of the permitted reticulation events. This allows investigators to define reticulation events appropriate to their needs, and renders the corresponding NP-completeness proofs for such problems trivial. Other schemata may be defined by allowing weighted reticulations or polynomially-bounded sets of forbidden reticulations.

The hypergraph formalism given in this thesis should be adopted to describe reticulate events in phylogenetic systematics. Hyperarcs provide unified representations of complex evolutionary phenomena. Moreover, such a formalism will make the recognition and transfer of relevant results from other fields easier. The NP-completeness results given in this thesis are one example. Of perhaps more practical use would be the application of work done in database design [ADS86, ANI90, BFMY83, Fag83] to algorithms for constructing reticulate phylogenetic trees.

B The Computational Complexity of Phylogenetic Parsimony Problems Incorporating Explicit Graphs

Consider the following decision problem:

UNWEIGHTED BINARY WAGNER PARSIMONY WITH GRAPH (UBW_G)

Instance: Positive integer d; graph $G = (V, E)$, where $V = \{0,1\}^d$ and $E = \{\{u,v\}: u, v \in V$ and u and v differ in exactly one position$\}$; a subset S of $\{0,1\}^d$; and a positive integer B.

Question: Is there a phylogeny satisfying the Wagner phylogenetic parsimony criterion that includes S and has length at most B?

This problem differs from problem UBW defined in Section 3.2.1 by including the d-dimensional graph explicitly in its instance. Both of these problems are in NP; however, UBW has been shown NP-complete [DJS86, GF82], and the complexity of UBW_G is unknown. The complexity of UBW_G is of interest not only because it has been used in proofs of NP-completeness [Day83], but also because it would be interesting to know by exactly how much the exponential padding of the input instance with G reduces the complexity of UBW.

To this end, consider the following restrictions on a phylogenetic parsimony problem Π: let $\Pi^{O(poly)}$ be the subproblem Π restricted to instances such that $|S| \leq p(d)$ for some polynomial p, and $\Pi^{\Omega(exp)}$ be the subproblem of Π restricted to instances such that $c2^d \leq |S|$ for some constant $c, c > 0$. The former restriction highlights, and the latter isolates, the complexity introduced by padding. If the complexity of UBW_G cannot be determined directly, these restricted subproblems may still give lower bounds.[3]

Consider the complexities of UBW and UBW_G. As mentioned already, UBW is NP-complete. If UBW_G is NP-complete, then UBW $\leq_m^p$ UBW_G; such a reduction is difficult to visualize, because it implies that a problem on dimension d can be mapped onto an equivalent problem of dimension $O(\log d)$. Alternatively, the padding introduced by G might yield polynomial algorithms via algorithms that solve the problem STEINER TREE IN GRAPHS (see Section 3.2.1). However, all known STG algorithms, including those restricted to d-dimensional graphs, are linear in $|G|$ and exponential in $|S|$ [Sny92, Win87].

Consider now the complexities of $UBW^{\Omega(exp)}$ and $UBW_G^{\Omega(exp)}$. These problems are computationally equivalent i.e. $UBW_G^{\Omega(exp)} \leq_m^p UBW^{\Omega(exp)}$ (discard G), and $UBW^{\Omega(exp)} \leq_m^p UBW_G^{\Omega(exp)}$ (add G, which can be constructed in time linear in the size of an instance of $UBW^{\Omega(exp)}$). Both of these problems reduce to UBW_G; however, for reasons similar to those given above, it is not obvious that they are either NP-complete or in P.

The complexity of the third pair of problems, $UBW^{O(poly)}$ and $UBW_G^{O(poly)}$, is the most interesting. The reduction from VERTEX COVER given in [DJS86]

[3] Since this thesis was submitted to the referees, I have found out that the weighted version of UBW_G (actually, the weighted version of $UBW_G^{\Omega(exp)}$) has been shown to be NP-complete [Gus91, Section 6]. While this does not immediately affect the problems examined in this section, it may be a stimulus for further research.

is actually to $UBW^{O(poly)}$, not UBW; hence, $UBW^{O(poly)}$is NP-complete.

Theorem 58 *$UBW^{O(poly)} \leq^p_m UBW_G^{O(poly)}$ if and only if $P = NP$.*

Proof: The implication from right to left is trivial. The implication from left to right follows by this construction: A reduction from an instance of $UBW^{O(poly)}$ to $UBW_G^{O(poly)}$ must map a polynomial number of vertices in a graph of dimension d into a polynomial number of vertices in a graph of logarithmically lower dimension. However, as $UBW_G^{O(poly)} \leq^p_m UBW^{O(poly)}$, this reduced instance is also an instance of $UBW^{O(poly)}$. Repeat this process a polynomial number of times to produce an instance of dimension $O(1)$, which can be solved in constant time. This yields a polynomial algorithm for $UBW^{O(poly)}$, which implies that P = NP. ▮

Corollary 59 *If $P \neq NP$ then $UBW_G^{O(poly)}$ is not NP-complete*

All optimal solutions to instances of $UBW_G^{O(poly)}$ are of size polynomial in d (see Section 3.2.1), and hence of size polylogarithmic in the instance of $UBW_G^{O(poly)}$. Thus, $UBW_G^{O(poly)}$ is in $\beta_{polylog}$, the class of decision problems requiring only polylogarithmic nondeterminism, which is probably strictly contained between P (= $\beta_{\log n}$) and NP (= $\bigcup_{k\geq 1} \beta_{n^k}$) [DT90, p. 22]. Moreover, by the Dreyfus-Wagner STG algorithm ([Sny92, Section 2] [Win87, Section 4.2]), $UBW_G^{O(poly)}$ is in $O(n^{O(\log n)})$.

There is even circumstantial evidence that $UBW_G^{O(poly)}$ is in P. An encoding of a graph $G = (V, E)$ is *succinct* if it is of size polylogarithmic in $|V|$ [GW83]; an example of such an encoding is a polylogarithmically sized circuit that computes the adjacency matrix for G. Though $UBW^{O(poly)}$ does not explicitly incorporate an encoding of G, its problem instances will always be of size polylogarithmic in G; hence, $UBW^{O(poly)}$ can be considered as the succinctly encoded version of $UBW_G^{O(poly)}$. In general (cf. [LW92]), succinct encodings *precisely* exponentiate the time complexity of graph problems e.g. the succinct version of the trivial graph property *existence of a triangle* is NP-hard [GW83, Theorem 2.1], and the succinct version of the NP-complete problem 3-COLORABILITY is NEXP-complete [PY86, Corollary]. If a problem Π is P-hard via a certain type of reduction called a *projection* from the Circuit Value Problem, then the succinct encoding version of Π is EXP-hard [PY86, p. 184]; if, in turn, the succinct encoding version of Π is NP-complete, then $P \neq NP$.

Corollary 60 *If $UBW_G^{O(poly)}$ is P-hard via a projection from the Circuit Value Problem, then $P \neq NP$.*

Many classical polynomial-time reductions can be easily made into projections [PY86, p. 182]; this may also be true of the log-time reductions used to establish P-hardness. As $P \neq NP$ probably cannot be proved in our standard system of logic [GJ79, p. 186], it is unlikely that $UBW_G^{O(poly)}$ can be proved to be P-hard, and likely that it is in P.

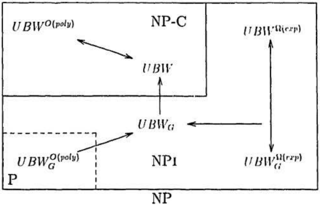


Figure 13: Reductions among implicit and explicit graph Unweighted Binary Wagner parsimony decision problems. Reductions $\Pi \leq_m^r \Pi'$ are denoted by arrows from Π to Π'. The abbreviations NP-C and NPI stand for the classes NP-complete and NP-intermediate (= NP − (P $\bigcup$ NP-C)), respectively.

The known relations among problems examined in this section are summarized in Figure 13. I conjecture that $UBW_G^{O(poly)}$ is in P and that $UBW^{\Omega(exp)}$, $UBW_G^{\Omega(exp)}$, and UBW_G are all strictly contained between P and NP-complete. To my knowledge, $UBW_G^{O(poly)}$ and $UBW^{O(poly)}$ are the only problem-pair such that the complexity of the succinct encoding version is known but the complexity of the full graph version is unknown. This in itself makes them candidates for further research.