The Dissertation Committee for Luay Nakhleh
certifies that this is the approved version of the following dissertation:

Phylogenetic Networks

Committee:

Tandy Warnow, Supervisor

Vladimir Lifschitz

C. Randal Linder

Daniel Miranker

Don Ringe

Phylogenetic Networks

by

Luay Nakhleh, B.S., MCS

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2004

To my father and mother

In loving memory of my grandfather Elias

Acknowledgments

This work would not have been possible without the help, support, and guidance of many people. First and foremost, I would like to acknowledge my advisor, Tandy Warnow. I joined Tandy's research group at a time when my interest in graduate studies was fading away (due to reasons beyond my control). Even worse, trusting my own research abilities was vanishing at the time. But then, Tandy was the ideal advisor at the right time. I doubt I would have continued my research and obtained my PhD degree if it were not for her support and guidance. No matter what I write here, I will fail to fully acknowledge Tandy's help, support and guidance. Every single word in this dissertation, and in all technical papers I have written, bears her marks. Tandy opens a whole world of opportunities to her students, be it in research ideas or wonderful people to collaborate with. Tandy's insistence on preparing and giving the best possible presentation, writing the best possible paper, and being the best possible student, will most definitely make it much easier for me as I embark on my new career. Equally important, Tandy has an amazing personality, which complements her technical strengths, making her really the ideal advisor.

I am greatly indebted to my committee members as well. Besides the helpful and interesting discussions I have had with them, I have also had the honor and privilege to co-author papers with all of them. I have had many interesting discussions with Vladimir Lifschitz about the historical linguistics part of this dissertation. I have met on a regular basis with Randy Linder, and learned biology from him. Randy has played a vital role in providing me with the necessary background in biology to pursue this challenging research area. Dan Miranker has truly been a mentor, colleague and friend—I learned a lot from him. I found historical linguistics to be such an intriguing and interesting area, but I seriously doubt that would have been my view had it not been for Don Ringe. Don always gives the most thorough answers—never have I seen someone who writes such self-contained and self-explanatory email messages.

I owe special thanks to Bernard Moret. Bernard has been an advisor, a mentor, and a role model. His vast knowledge, priceless advice, and relentless support have made all this work possible. All work on phylogenetic networks in biology that is described in this dissertation was done in close collaboration with Bernard.

I would like to thank Elaine Rich, Mary Eberlein, and Don Baker for giving me the opportunity to be involved in the Automata Theory course. I really enjoyed teaching the course and interacting with the students—that experience has tremendously improved my presentation skills.

My friends and colleagues in Austin have been a huge help to me. Jerry Sun has been a wonderful friend, colleague, and person to know. Much of the work in this dissertation would not have been done in a timely fashion without his relentless work and help. Alper Sen has been an ideal friend. I have known Alper since I came to UT, and we have been very close friends ever since, having lunch together almost daily, drinking together almost weekly (and when research problems became more challenging, the frequency of drinking increased). Most importantly, Alper and I disagree on everything, which makes it a great friendship. My colleagues in the Phylo lab have been a pleasure to have around and discuss research and other topics with: Li-San Wang, Usman Roshan, Ganesh Ganapathy, Allen Clement, and Cara Stockham. The help that Laurie Alvarez and Gem Naiver offer to students and faculty alike is beyond expression—they are wonderful, and I will never forget all the favors they did to me. Anna Tholse and Tiffani Williams, both Bernard's students, have been great friends and colleagues. Ruth Timme and Anneke Padolina, both Randy's students, have also been wonderful and very helpful.

On the more personal side, I owe everything in my life to my parents. They do not know much about computer science; however, they have always encouraged me to pursue whatever I was interested in. They always believed I was doing what's best for me. Their unwavering support made it all possible. My brother Haytham has supported me greatly as well. We talk on a regular basis, remember old stories from back home, and laugh. My grandmother, Saa'da—still young at the age of 85—has been an enormous source of support, love, and inspiration. My friends, aunts, uncles and their families in Israel have also been wonderful, and I enjoy their company every time I visit in Israel. Ever since his death in 1990, the memory of my grandfather Elias has never left my mind. My pride in the man and his achievements is beyond expression;

he will always be a role model for me.

My research started flourishing at exactly the same time I met my girlfriend, Mika; and that was not a coincidence. Her unconditional love and unwavering support have made my life more pleasant and enjoyable. Her genuine understanding allowed me to focus on my work and finish my dissertation.

Work on a PhD dissertation comes with a daily dose of stress and uncertainty. To alleviate the inconvenience, I listen to Om Kulthoum daily. Om Kulthoum, the legend of Arabic singing and the divine voice, died nine months after I was born; that did not prevent me from becoming addicted to her voice. Ever since I was introduced to her music, never has her voice left my ears, heart and mind. I cannot explain the influence her voice has on me; all I can say is: Thank You Om Kulthoum.

Phylogenetic Networks

Publication No. ____________________

Luay Nakhleh, Ph.D.
The University of Texas at Austin, 2004

Supervisor: Tandy Warnow

Phylogenies, i.e., evolutionary histories, play a major role in representing the relationships among groups of entities, such as species, genes, and languages. Their pervasiveness has led scientists, mainly biologists, mathematicians, and computer scientists, to develop methods and tools for their accurate reconstruction. Most of these tools, however, assume that the underlying model of speciation is a tree. While a good first approximation, trees fail to model the evolutionary histories in the presence of complex evolutionary events, such as lateral gene transfer and hybrid speciation among biological entities, and borrowing of linguistic features among natural languages. These events lead to “networks”, rather than trees, of relationships.

In this dissertation, we present two methodologies for reconstructing phylogenetic networks. In the biological context, our method is based on the observation that contained within the branches of a (species) phylogenetic networks are phylogenetic trees that model the evolution of individual genes.

To study the accuracy of our new method, as well as existing methods, we have developed a suite of simulation tools and error measures. Our simulation studies show a clear outperformance of existing methods.

In historical linguistics, we extend the Ringe-Warnow model of language evolution, to incorporate non-treelike evolutionary events; our new methodology is called "perfect phylogenetic networks". We have implemented a reconstruction method, based on the new methodology, and analyzed a dataset of Indo-European languages.

Table of Contents

Acknowledgments **v**

Abstract **ix**

List of Tables **xvi**

List of Figures **xvii**

Part I INTRODUCTION AND BACKGROUND **1**

Chapter 1. Introduction **2**

1.1 Phylogeny and Its Centrality to Biology 2

1.2 Non-treelike Evolution in Biology 4

1.3 Phylogenies of Languages 5

1.4 Non-treelike Evolution of Languages 6

1.5 Challenges and Our Contributions 7

1.6 Organization of the Dissertation 9

Chapter 2. Phylogenetic Tree Reconstruction: A Background **11**

2.1 The Phylogeny Problem 11

2.1.1 Input Data 12

2.1.1.1 Character Data 13

2.1.1.2 Distance Data 13

2.2 Assessing the Quality of Methods via Simulation Studies . . . 14

2.2.1 Steps of a Simulation Study 15

2.2.2 Stochastic Models of Sequence Evolution 16

2.2.3 Simulation Tools for Studying the Performance of Tree Reconstruction Methods 19

2.2.3.1 r8s 19
2.2.3.2 Seq-Gen 20
2.2.4 Measuring the Topological Accuracy of Tree Reconstruction Methods 24
2.3 Phylogenetic Reconstruction Methods 25
2.3.1 Distance-based Methods 26
2.3.2 Maximum Parsimony Heuristics 26
2.3.2.1 Inconsistency of Parsimony 27
2.3.3 Maximum Likelihood Heuristics 28
2.3.4 Perfect Phylogeny: Homoplasy-free Evolution 29

Part II PHYLOGENETIC NETWORKS IN BIOLOGY **32**

Chapter 3. Biology of Reticulate Evolution **33**
3.1 Reticulation Between Chromosome Pairs: Meiotic Recombination 34
3.2 Reticulation Within a Lineage: Sexual Recombination 35
3.3 Reticulation Among Lineages: Horizontal Gene Transfer and Hybrid Speciation 35
3.4 Types of Hybrids and Gene Flow 40

Chapter 4. Mathematical Models of Reticulate Evolution **45**
4.1 Population Level 45
4.2 Species Level 46
4.2.1 Model networks 46
4.2.2 Reconstructible Networks 51
4.2.3 *k*-GT-Networks 53

Chapter 5. Simulation Tools for Studying Phylogenetic Network Reconstruction Methods **58**
5.1 Network File Format 58
5.2 GenNet: A Tool for Generating Random Networks 62
5.2.1 Description of the Tool 62
5.2.2 Running the Tool 64

5.3 NetSeqGen: A Tool for Simulating Sequence Evolution on Networks . . . 68
5.3.1 Description of the Tool . . . 68
5.3.2 Running the Tool . . . 68

Chapter 6. Quality Measures for Phylogenetic Networks 71
6.1 A Parsimony Criterion for Phylogenetic Networks . . . 74
6.1.1 Parsimony on Trees and Extensions to Networks . . . 74
6.1.2 Fixed-parameter Network Parsimony . . . 75
6.2 Topological Accuracy of Networks . . . 81
6.2.1 Networks and Tripartitions . . . 82
6.2.2 Networks and Trees . . . 84
6.2.3 Networks and Splits . . . 85
6.2.4 Illustrating the Three Measures . . . 85
6.3 Tripartitions and Their Induced Metric . . . 89
6.4 k-gt-Networks and the Three Measures: Metric Spaces . . . 95
6.5 Computing the Splits of gt-Networks . . . 99
6.6 The Relationship Between Splits and Tripartitions . . . 101

Chapter 7. Genes and Species: Trees Within Trees and Trees Within Networks 104
7.1 Gene Tree Incongruence . . . 105
7.2 Combined vs. Separate Analysis Approaches . . . 107
7.3 The rSPR Operation . . . 109
7.4 Characterizing Trees Inside Networks . . . 110
7.5 Ramifications of Missing Taxa on Network Reconstruction . . . 114
7.5.1 Identifiability of Reticulation Events . . . 114
7.5.2 Reconstructing Missing Taxa . . . 116

Chapter 8. Existing Methods for Detecting and Reconstructing Reticulate Evolution 120
8.1 Detecting Reticulate Evolution . . . 121
8.1.1 Detecting Recombination . . . 121
8.1.2 Detecting Horizontal Gene Transfer . . . 122

8.1.3 Detecting Hybrid Speciation 125
8.2 Reconstructing Reticulate Evolution 126
8.2.1 Combined Analysis Methods 126
8.2.2 Separate Analysis Methods 127
8.3 Software Related to Network Reconstruction 128

Chapter 9. Reconstructing Species Networks: Our New Method 137
9.1 Background 138
9.1.1 Graph-theoretic Definitions 138
9.1.2 Strict Consensus and Compatibility Trees 139
9.2 Reconstructing gt-Networks When Gene Tree Estimates are Accurate 140
9.2.1 Efficient Reconstruction of gt-Networks from Gene Trees 141
9.3 Reconstructing gt-Networks When Gene Tree Estimates are Inaccurate 146
9.3.1 Compatible Consensus Trees 148
9.3.2 Incompatible Consensus Trees 150
9.4 SpNet: Our Technique for Inferring gt-Networks 153

Chapter 10. Experimental Studies 155
10.1 Experimental Settings 156
10.1.1 Methods 156
10.1.2 Tools and Measures 158
10.1.3 Simulation Parameters 158
10.2 Results and Analyses 161
10.2.1 Detection Quality 161
10.2.1.1 SpNet: ML vs. MP Trees 163
10.2.1.2 Maddison's Method: ML vs. MP Trees 165
10.2.1.3 Maddison's Method vs. SpNet 167
10.2.2 Reconstruction Quality 169
10.2.3 Major Challenge: Detection 170

Part III Phylogenetic Networks in Historical Linguistics **174**

Chapter 11. Language Evolution **175**
11.1 Introduction 175
11.2 Historical Linguistics Data and Methods 178
11.2.1 Characters 178
11.2.2 The Comparative Method 179
11.2.3 The Warnow-Ringe Model for Trees 180
11.3 Character Compatibility and Perfect Phylogeny 182
11.4 Non-treelike Evolution of the IE Languages and the Need for Phylogenetic Networks 184

Chapter 12. Perfect Phylogenetic Networks **186**
12.1 The Mathematical Model 194
12.2 The PPN Reconstruction Problems 203
12.3 Related Work 206
12.3.1 The Answer-Set Programming Approach 206
12.3.2 Perfect Phylogenetic Networks with Recombination . . . 207

Chapter 13. Case Study: Analysis of an Indo-European Dataset **210**
13.1 The IE Dataset 210
13.2 Phylogenetic Analysis 216
13.2.1 Overview 216
13.2.2 Our Candidate Trees 218
13.2.3 Constructing the Perfect Phylogenetic Networks 220
13.2.4 The Set of PPNs We Obtained 223
13.2.5 Comparison of the PPNs 225
13.2.6 Comparison of All 23 PPNs With Respect to Mathematical Criteria 238
13.2.7 Discussion 239
13.3 Our Proposed Solution to IE Evolution 239

Bibliography **241**

Vita **268**

List of Tables

6.1 The tripartitions induced by the edges of networks N_1 and N_2 of Figure 6.4. 86

6.2 The splits induced by the networks N_1 and N_2 of Figure 6.4. . . . 88

13.1 The 24 IE languages analyzed. 211

13.2 Summary of the results of the phylogenetic analysis of the five IE trees described in Figures13.1–13.5. 225

13.3 The 16 3-edge solutions found on Tree A; the highlighted rows correspond to the three feasible solutions (1, 3, and 5). Each solution is described in terms of the three lateral edges added to Tree A to produce a PPN. The abbreviations are explained in Table 13.4. The rightmost column summarizes the minimum number of borrowing events needed on each of the 16 PPNs in order to make all characters compatible. 227

13.4 Abbreviations for proto languages. 228

13.5 The 4 3-edge solutions found on Tree B; the two feasible solutions highlighted. Each solution is described in terms of the three lateral edges added to Tree B to produce a PPN. The abbreviations are explained in Table 13.4. The rightmost column summarizes the minimum number of borrowing events needed on each of the 4 PPNs in order to make all characters compatible. 234

List of Figures

2.1 (a) A four-taxon tree that contains two long edges (adjacent to leaves), with all other edges being very short. (b) An incorrect tree inferred by maximum parsimony. 28

3.1 Reticulation at various levels: (a) species level, (b) population level, and (c) individual (chromosomal) level. 34

3.2 Horizontal gene transfer: the species network in (a) and the two possible gene trees in (b) and (c). 36

3.3 Hybrid speciation: the species network in (a) and the two possible gene trees in (b) and (c). 37

3.4 An example of allopolyploid hybrid speciation. 42

3.5 An example of autopolyploid hybrid speciation. 43

3.6 An example of an introgression event. 44

4.1 An example of a pedigree. 46

4.2 A phylogenetic network N on 6 species. The solid circles denote the tree nodes, and the solid squares denote the network nodes. The solid lines denote the tree edges, and the dashed lines denote the network edges. 48

4.3 A scenario illustrating two nodes X and Y that cannot co-exist in time. 50

4.4 (a) A phylogenetic network N on 5 species, where species B had gone extinct. (b) the same network under the reconstructible network model. 52

4.5 (a) A gall Q whose coalescent and reticulation nodes are w and x respectively. (b) and (c) show the two possible ways of "breaking" the gall Q to induce trees T_1 and T_2, respectively. The marked edges in T_1 and T_2 form $RP^Q(T_1)$ and $RP^Q(T_2)$, respectively. 54

4.6 An example of a 3-gt-network. 57

5.1 The topology of a phylogenetic tree in (a), and its representation in the Newick format in (b). 59

6.1 Fitch's algorithm for computing the parsimony score of a tree. . . 74

6.2 Algorithm for computing the parsimony score of a gt-network: the bottom-up stage. 78

6.3 Algorithm for computing the parsimony score of a gt-network: the top-down stage. 79

6.4 Two networks N_1 and N_2 used to illustrate the error rates computed by the three measures. 86

6.5 The two trees T_1^1 and T_2^1 induced by the network N_1 of Figure 6.4. 87

6.6 The weighted complete bipartite graph G^{N_1,N_2} that is constructed to compute $m^{tree}(N_1, N_2)$ for the two networks of Figure 6.4. The thick edges correspond to the minimum-weight edge-cover, and the sum of their weights is 2/3. 87

6.7 The two trees T_1^2 and T_2^2 induced by the network N_2 of Figure 6.4. 88

6.8 Two phylogenetic networks N_1 and N_2. We have $m^{tri}(N_1, N_2) \neq 0$, $m^{tree}(N_1, N_2) = 0$, and $m^{sp}(N_1, N_2) = 0$ although $N_1 \neq N_2$. 89

6.9 Algorithm for computing the splits of a gt-network. 103

7.1 Gene trees, a species tree, and a species network for three species A, B, and C. The gene trees may be discordant due to lineage sorting or due to hybrid speciation. Further, the incongruence between the two gene trees may be similar in both cases. (a) Gene tree that disagrees with the (species) phylogeny due to lineage sorting. (b) Gene tree that agrees with (species) phylogeny. (c) A species phylogeny that contains one hybrid speciation event. (d) One of two possible gene trees inside the phylogeny in (c). (e) The second possible gene tree inside the phylogeny in (c). 106

7.2 (a) A phylogenetic network N on 5 species, where species B had gone extinct. The network N induces the two trees shown in (b) and (c). The two networks N' and N'' in (d) and (e) represent different evolutionary histories (and, in particular, contains different reticulation events), yet each induces the two trees in (b) and (c). The network in (f) is an "augmentation" of (d) designed to satisfy the time coexistence property. 115

7.3 (a) A phylogenetic network N on 5 species, where species B and D are missing. The network N induces the two trees shown in (b) and (c). (d) A network N' that reconciles the two trees in (b) and (c). The networks in (e), (f) and (g) are augmentations of N' to satisfy time coexistence. 117

7.4 (a) A phylogenetic network N on 5 species, where species B and D are missing. The network N "induces" two trees shown in (b) and (c). (d) A network N' that induces the two trees in (b) and (c), but fails to capture the two missing species B and D. (e) A second network N'' that induces the two trees in (b) and (c) and captures the two missing species. 119

10.1 SpNet(1): SpNet run on strict consensus of ML trees, where the strict consensus trees are missing one edge each. SpNet(OPTSC): SpNet run on strict consensus of all optimal MP trees. Settings: 20-taxon networks, 0.1 scaling factor. The rate of correct predictions is the number of times the method predicted the correct number of reticulation events, normalized by 25—the total number of runs. . 164

10.2 A comparison of Maddison's approach (SpNet(0)) on the single best ML trees, to an extension of Maddison's approach where we find all best MP trees and analyze them pairwise (SpNet(OPTPW)), and to SpNet run on the strict consensus of all the best MP trees (SpNet(OPTSC)). Settings: 20-taxon networks, 0.1 scaling factor. The rate of correct predictions is the number of times the method predicted the correct number of reticulation events, normalized by 25—the total number of runs. 166

10.3 SpNet run on the strict consensus of trees obtained in a maximum likelihood (ML) analysis. SpNet(i) indicates that i edges are missing from the strict consensus tree; hence SpNet(0) is Maddison's method. Settings: 20-taxon networks, 0.1 scaling factor. The rate of correct predictions is the number of times the method predicted the correct number of reticulation events, normalized by 25—the total number of runs. 168

10.4 FN and FP error rates of NeighborNet (NNet) and SpNet on 20-taxon networks, with 0.1 scaling factor, with tree model phylogeny in (a), and 1-hybrid network in (b). The graph in (c) shows the results without the FP rate of NNet. SpNet was run on strict consensus trees of ML trees such that the strict consensus trees were missing one edge. 171

10.5 FN and FP error rates of NeighborNet (NNet) and SpNet on 20-taxon networks, with 0.1 scaling factor, with tree model phylogeny in (a), and 1-hybrid network in (b). The graph in (c) shows the results without the FP rate of NNet. SpNet was run on strict consensus trees of ML trees such that the strict consensus trees were missing two edges. 172

12.1 (a) Five languages $L_1, \ldots, L_5$, with two characters c_1, and c_2. (b) The same five languages with a third character c_3. There exists a perfect phylogenetic tree for the languages and characters in (a); however, there does not exist a perfect phylogenetic tree for the data in (b). 187

12.2 A perfect phylogeny T for the languages and character states of Figure 12.1(a). 187

12.3 A perfect phylogenetic network N for the languages and character states of Figure 12.1(b). 188

12.4 The three trees contained inside the network in Figure 12.3. While the network "reconciles" the evolutionary history of all three characters, each one of those characters actually evolved down exactly one of the three trees. 189

12.5 (a) A "YES-instance" of the Disjoint Connecting Paths problem: there are two vertex-disjoint paths, one connecting s_1 with t_1, and the other connecting s_2 with t_2. (b) A "NO-instance" of the Disjoint Connecting Paths problem: there do not exist two vertex-disjoint paths, such that one connects s_1 and t_1 and the other connects s_2 and t_2. 201

12.6 (a) The perfect phylogenetic network that is the output of the reduction in the proof of Theorem 18 for the graph in Figure 12.5(a). (b) A tree inside the network and on which the character is compatible. The internal nodes are labeled so as to make the character compatible. The dashed lines form the set E_0 of tree edges, while the solid lines denote the bidirectional contact edges. 202

12.7 The phylogenetic network that is the output of the reduction in the proof of Theorem 18 for the graph in Figure 12.5(b). Clearly the character is not compatible on this network. 202

12.8 A counterexample to Theorem 5 in [186]. 209

13.1 Tree A. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome. 220

13.2 Tree B. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome. 221

13.3 Tree C. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome. 222

13.4 Tree D. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome. 223

13.5 Tree E. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome. 224

13.6 The first feasible perfect phylogenetic network based on Tree A (corresponding to Solution 1 in Table 13.3). The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional. 230

13.7 The second feasible perfect phylogenetic network based on Tree A (corresponding to Solution 3 in Table 13.3). The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional. 231

13.8 The third feasible perfect phylogenetic network based on Tree A (corresponding to Solution 5 in Table 13.3). The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional. 232

13.9 The first perfect phylogenetic network on Tree B. The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional. . . . 235

13.10 The second perfect phylogenetic network on Tree B. The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional. . . . 236

Part I

Introduction and Background

Chapter 1

Introduction

1.1 Phylogeny and Its Centrality to Biology

Phylogenies are the main tool for representing evolutionary relationships among biological entities at the level of species and above. Since the evolutionary history is at best partially known, biologists, mathematicians, and computer scientists have designed a variety of criteria and methods for their accurate reconstruction. Over the last 30 years, biologists have come to embrace reconstruction of phylogenetic trees as a major research goal [49, 73, 85], with the number of phylogeny-related publications exponentially increasing.

One of science's great challenges is reconstructing the "Tree of Life", which is the evolutionary history of all organisms on Earth. The National Science Foundation, for example, has allocated more than $30 million to support this endeavor. Further, the NSF has recently awarded our group a $12 million grant for the development of methodologies and infrastructure for assembling the Tree of Life (see `http://www.phylo.org`). The implication of this grand phylogeny—that all living things on Earth today (from bacteria, to seaweeds, to mushrooms, to humans) are related—has forever changed our perception of the world around us.

Phylogenies play a major role in the interpretation of information on all characteristics of organisms, from structure and physiology to genomics. With the technological advances and the increasing availability of molecular date, their accurate reconstruction seems more attainable than ever.

Phylogeny reflects the history of transmission of life's genetic information, and hence organizes our knowledge of diverse organisms, genomes, and molecules. A reconstructed phylogeny provides invaluable information for resolving various issues. At the species level, a phylogeny

1. Provides hypotheses about the lineages in which traits arose and under what circumstances, thus playing a vital role in studies of adaptation and evolutionary constraints [48, 100, 104, 112, 121].
2. Informs the dynamics of speciation and, to some extent, extinction—the two forces that generate and reduce biodiversity [23, 55].

Further, pharmaceutical companies are using phylogenies to:

1. Detect the functional demands to which gene families have been subjected, so that phylogenetic analyses can elucidate functional relationships within living cells [56, 61, 200].
2. Make functional predictions from sequence databases of gene families [9].
3. Predict ligands [24].

4. Help in the development of vaccines [64] and antimicrobials and herbicides [19, 134].

Because phylogenies are such an important part of biological investigations, many methods exist for reconstructing phylogenies. Many of these methods work on aligned biomolecular sequences (i.e., DNA, RNA, or amino-acid sequences), and use these sequences to infer the evolutionary history of the sequences.

1.2 Non-treelike Evolution in Biology

For the most part, phylogenetic methods assume that the phylogeny underlying the data is a tree. However, such is not always the case: for many organisms, a significant level of genetic exchange occurs between lineages, and for some groups, lineages can combine to produce new independent lineages. These exchanges and combinations transform a tree into a network. Ford Doolittle [37] famously wrote

> Molecular phylogeneticists will have failed to find the "true tree," not because their methods are inadequate or because they have chosen the wrong genes, but because the history of life cannot properly be represented as a tree.

Indeed, events such as meiotic and sexual recombination, horizontal gene transfer and hybrid speciation cannot be modeled by bifurcating trees. Meiotic recombination occurs in every generation at the level of individual chromosomes;

sexual recombination commonly acts at the population level and recombines the evolutionary histories of genomes. Hybrid speciation is very common in some very large groups of organisms: plants, fish, frogs, and many lineages of invertebrates, and horizontal gene transfer is ubiquitous in bacteria. Although the mixing (reticulation) of evolutionary histories has long been widely appreciated and acknowledged, there has been comparatively little work on computational methods for studying and estimating reticulate evolution, especially at the species level.

1.3 Phylogenies of Languages

Languages differentiate and divide into new languages via a process similar to how biological species divide into new species: communities separate (typically geographically), the language changes differently in each of the new communities, and in time people from separate communities can no longer understand each other. While this is not the only means by which languages change, it is this process which is referred to when we say, for example, "French and Italian are both descendants of Latin." The evolution of related languages is mathematically modeled as a rooted tree in which internal nodes represent the ancestral languages and the leaves represent the extant languages.

Reconstructing this process for various language families is a major endeavor within the historical linguistics community, but is also of interest to archaeologists, human geneticists, and physical anthropologists, for example, because an accurate reconstruction of how certain languages evolved can help answer questions about human migrations, the time that certain artifacts were developed, when ancient people began to use horses in agriculture, the identity of physically European mummies found in China, etc. (see in particular [107, 110, 160, 191]).

In historical linguistics, one of the major factors that adds to the difficulties in reconstructing the evolutionary history of a set of languages is the observation that some groups of languages contain closely related dialects which have evolved in close contact. For these groups, a tree is not an appropriate model of evolution.

1.4 Non-treelike Evolution of Languages

Various researchers [35, 36, 58, 182] have noted that if communities are sufficiently separated after they diverge, then the phylogeny (i.e., evolutionary tree) for the languages can be inferred by comparing the characteristics of the languages (grammatical features, regular sound changes, and cognate classes for different basic meanings), and searching for "perfect phylogenies." However, the problem of determining if a perfect phylogeny exists, and then computing it, is NP-hard [17]. Consequently, efficient techniques for the inference of evolutionary trees for language families were not easily obtained.

In the 1990's, various fixed-parameter approaches for the perfect phylogeny problem were developed (although inspired by the biological context rather than the linguistic one; see [47]). Subsequently, Ringe and Warnow worked together to fully develop the methodology (character encoding and algorithmic techniques) needed to apply these algorithms to the Indo-European (IE) language family. Their initial test of the methodology largely supported the claim that a perfect phylogeny should exist. Largely, but not entirely, because of the "Problem with Germanic". The Germanic family seemed to introduce non-treelike behavior, evidently acquiring some of its characteristics from its neighbors rather than (only) from its direct ancestors.

1.5 Challenges and Our Contributions

While biologists have long recognized reticulate evolution and the challenges incurred by their presence, almost all existing phylogenetic methods assume the underlying evolutionary history is treelike. The main challenge is to develop accurate methods for reconstructing phylogenetic networks. This task entails developing a suite of simulation tools and accuracy measures to study the performance of existing methods, and facilitate devising new accurate ones. Further, commonly used criteria for reconstructing trees, most notably maximum parsimony and maximum likelihood [50], need to be extended to networks. The contributions described in this dissertation include:

1. a mathematical model of phylogenetic networks;
2. a suite of tools for generating biologically-realistic phylogenetic networks, as well as simulating the evolution of DNA sequences on networks;
3. the first error metric for quantifying topological accuracy of network reconstruction methods;

4. extension of a parsimony criterion to networks, and an efficient algorithm for scoring the parsimony of special-case networks;
5. an efficient and accurate method for reconstructing (species) networks from their constituent (gene) trees.

We have also undertaken similar tasks in the reconstruction of evolutionary histories of natural languages. While the Ringe-Warnow methodology seemed very clearly heading in the right direction, and even seemed to potentially answer many of the controversial problems in IE evolution (see [156, 159, 182, 188, 189]), it became necessary to extend the model to address the problem of how characters evolve when the language communities remain in significant contact. In this dissertation, we present:

1. the *perfect phylogenetic network* model of non-treelike evolutionary histories;
2. NP-hardness and FPT results for testing the compatibility of characters on phylogenetic networks;
3. two reconstruction criteria:
   (a) reconstructing a network directly from a set of sequences;
   (b) completing a given tree into a perfect phylogenetic network.
4. a new analysis of a dataset of Indo-European languages, using the new methodology.

1.6 Organization of the Dissertation

This dissertation is organized in three parts, the first of which introduces the motivations and contributions and reviews the necessary background material. In Part II we describe our work on phylogenetic networks in biology, and in Part III we describe our work on phylogenetic networks in historical linguistics.

Part II We start in Chapter 3 by briefly describing the biological context of reticulate evolution, and distinguishing among the various levels of reticulation. Focusing on reticulation at the species level, we introduce a mathematical model of phylogenetic networks in Chapter 4. Although a large array of tools exists for simulation studies of tree reconstruction methods, none exists for studying network reconstruction methods. In Chapter 5 we describe the new suite of simulation tools that we have developed. In Chapter 6 we describe our novel error metric that we have developed for comparing phylogenetic networks and quantifying error rates of methods. Further, we describe an extension of the parsimony criterion to phylogenetic networks, and give an efficient algorithm for scoring the parsimony of special-case networks. In Chapter 7 we describe the relationship between gene trees and species phylogenies (trees as well as networks); discordance of gene trees forms the basis of our new method for reconstructing (species) phylogenetic networks. In Chapter 8 we briefly describe existing methods aimed at detection and reconstruction of reticulate evolution, at the various levels. We introduce our new method for network reconstruction in Chapter 9, and conclude this part of the dissertation by describing in Chapter 10 simulation studies of our new method as well as existing methods.

Part III In this part, we describe our new methodology for reconstructing phylogenetic networks of natural languages. We start in Chapter 11 by describing language evolution, including the Ringe-Warnow model. In Chapter 12 we give a mathematical definition of perfect phylogenetic networks, and two network reconstruction criteria. We have implemented a method, based on the new methodology, and analyzed a dataset of Indo-European languages; we present our findings in Chapter 13.

Chapter 2

Phylogenetic Tree Reconstruction: A Background

In this chapter we review background material relevant to phylogeny reconstruction, when trees are the appropriate model.

2.1 The Phylogeny Problem

A *phylogenetic tree* (or, simply a *phylogeny*) is a leaf-labeled binary tree that models the evolution of a set of a taxa (species, genes, languages, etc.) from a common ancestor. The internal nodes of a phylogeny represent the (hypothetical) ancestral taxa, and the leaves represent the extant taxa. The *phylogeny problem* is to reconstruct the evolutionary history of a set S of taxa.

Definition 2.1.1. (The Phylogeny Problem)

- **Input:** A set S of taxa.
- **Output:** A tree T leaf-labeled by S.

Although true evolutionary histories are often best represented by a rooted tree, locating the root of the tree is usually hard to achieve with any degree of accuracy, due to statistical or data-ralated issues. Nevertheless, systematists often reconstruct rooted trees, and the technique that they generally employ is using an *outgroup*, which is a taxon known to have branched off before all other taxa under consideration.

Given a set S of n taxa, the number of rooted binary trees on S is

$$(2n-3)!! = (2n-3) \times (2n-5) \times \cdots \times 3 \times 1$$

and the number of unrooted binary trees on S is

$$(2n-5)!! = (2n-5) \times (2n-7) \times \cdots \times 3 \times 1.$$

This shows that the number of possible phylogenies on a set of taxa is exponential in the number of taxa in the set. The fundamental problem is how to choose one phylogeny over another. To solve this problem, various measures of accuracy have been defined to assess how well a given phylogeny fits a set of data. Various phylogenetic reconstruction methods have been introduced based upon the measures that are used for assessing the quality of the phylogeny. Before we discuss the various phylogenetic reconstruction methods, we first briefly describe two types of data (see below) that are used to represent the input to those methods, and models of sequence evolution.

2.1.1 Input Data

Different types of data can be used as input to methods of tree reconstruction. Character data and distance data are the most common types that are used in phylogenetic analysis.

2.1.1.1 Character Data

Qualitative characters in biomolecular sequences are the single positions within multiple alignments, and they have a fixed number of states (4 for DNA and RNA, 20 for amino-acids). Qualitative characters for languages are grammatical features, unusual sound changes, and cognate classes for different meanings.

We now formalize the concept of a qualitative character mathematically. Suppose T is a tree describing the evolutionary history of a set S of taxa (e.g., languages or species), and that a qualitative character α is defined for each of the taxa in S. Thus, α is a function such that $\alpha : S \to Z$, where Z denotes the set of integers (i.e., each integer represents a possible state for α). We call $\alpha(s)$ the state of character α on taxon $s \in S$.

Phylogenetic reconstruction methods based on character data take as input a matrix $M_{n \times k}$ of n species and k characters, so that each species $s \in S$ is represented by a vector in Z^k. Thus, M_{ij} is the state of character j for species s_i, where $S = \{s_1, \ldots, s_n\}$.

The output of the method is a phylogeny leaf-labeled by S, and whose internal nodes are also labeled by vectors in Z^k.

2.1.1.2 Distance Data

Some phylogenetic reconstruction methods take "distance" matrices [99] as a representation of the input. A distance matrix $M_{n \times n}$ is a symmetric matrix that indicates the pairwise distances between taxa; M_{ij} represents the distance between the two taxa i and j. Distance matrices are usually computed from the character data of the input. Those matrices have $M_{ii} = 0$ for every $1 \leq i \leq n$, but do not necessarily satisfy the triangle inequality.

2.2 Assessing the Quality of Methods via Simulation Studies

Phylogenetic reconstruction methods are evaluated according to two basic types of criteria: *statistical performance*, which addresses the accuracy of the method under a specified stochastic model of evolution, and *computational performance*, which addresses the computational requirements of the method. A method is said to be *statistically consistent* with respect to a specific model of evolution if it is guaranteed to recover the true tree with probability going to 1 as the amount of data (i.e., sequence length) goes to infinity.

Accuracy in a phylogenetic reconstruction method is determined primarily by comparing the unrooted leaf-labelled tree obtained by the method to the "true" tree. Since the true tree is usually unknown, accuracy is addressed either theoretically, with reference to a fixed but unknown tree in some model of evolution, or through simulation studies.

Simulation studies as a means for studying the accuracy of phylogenetic reconstruction methods are advantageous in more ways than one:

- In real-life applications of phylogenetic methods, the true phylogenetic tree is usually not known, whereas it is known in simulation studies. This allows the developer as well as the user of phylogenetic methods to assess the quality of those methods by comparing the tree inferred by the methods against the true phylogenetic tree.

- Theoretical reasoning about phylogenetic reconstruction methods is very difficult in general, due to the complex stochastic processes underlying the evolutionary models, and the complicated mathematics involved in phylogenetic studies. Instead, simulation studies offer an alternative approach by providing empirical results about the performance of methods.

- Many of the theoretical results that have been established regarding phylogenetic methods, often provide loose bounds on the actual results. Simulation studies, instead, provide actual results.

2.2.1 Steps of a Simulation Study

To study the performance of a phylogenetic reconstruction method M, the following steps of a simulation study are followed:

Step 1: A random phylogeny T is generated. Associated with the edges of the topology are parameters (e.g., substitution matrices, distribution shape parameters, etc.) that control the evolution of sequences along those edges. The topology as well as the parameters are drawn from biologically-realistic distributions.

Step 2: Sequence evolution is simulated by placing a sequence (or a set of sequences) at the root of T, and then evolving the sequence down the phylogeny using the parameters associated with the edges, as well as a specified model of evolution. The result of this step is a sequence (or a set of sequences) associated with each node of the phylogeny.

Step 3: The sequences obtained at the leaves of T in Step 2 are fed to a phylogenetic reconstruction method, M, which returns a phylogeny T'.

Step 4: The inferred phylogeny T' is then compared against the model phylogeny T, and error (usually topological) is quantified, using some measure.

Simulation studies show that the topological accuracy of a method is affected by the number of leaves in the tree, the maximum evolutionary diameter, the deviation from a molecular clock, as well as tree shape [74, 76, 122, 124–126], and that different methods are affected differently by these aspects of the model tree. Other simulation studies show that different heuristics for maximum parsimony or maximum likelihood also perform differently [163, 193].

2.2.2 Stochastic Models of Sequence Evolution

Both approaches (mathematics and simulation) to evaluating performance of reconstruction methods require that the stochastic model of evolution be specified. Many stochastic models of sequence evolution have been proposed; most assume that the sites (positions within the sequences) evolve down a tree via nucleotide substitutions under a Markov process. The stochastic process under which an individual site evolves down an edge within the tree can therefore be given by a 4×4 substitution matrix M_e for that edge e, dictating the probability of each of the possible states at the "bottom" of the edge as a function of the state at the "top" of the edge. Thus, the evolution of a single site down the tree is given by a rooted tree with edges annotated with substitution matrices.

The simplest of these models for DNA sequences is the Jukes-Cantor [90] model, which assumes:

1. the sites evolve identically and independently (the *i.i.d.* assumption),
2. the state of each site at the root is randomly selected, and
3. if a site changes state on an edge, it changes with equal probability to each of the remaining three states.

The Jukes-Cantor model thus has only one free parameter on each edge.

Other models of site substitution with more free parameters are considered to be biologically more realistic. The simplest of these is the Kimura 2-parameter (K2P) model [93]; the most general is the General Markov (GM) model, which allows the substitution matrices to be quite general. Many (but not all) phylogenetic reconstruction methods are guaranteed to be *statistically consistent* under the General Markov model (and hence under all the models it contains)—that is, these methods can infer the true tree with high probability when given long enough sequences. Statistical consistency is still possible under models where these conditions are relaxed in a controlled fashion—for example, by allowing the sites to have rates of evolution that vary with the sites and that are drawn from a known distribution. However, if the rates are drawn from an unknown distribution (for example, if an unknown proportion of sites are constant), then it may not be possible to identify the true tree, even from infinite data. See [50, 92, 98] for a further discussion of these issues.

The *i.i.d.* models of DNA site substitution are clearly unreasonable, yet most enhanced models simply allow rates to vary across sites, which by itself does not alter the *i.i.d.* assumption. Bruno and Halpern [20] have developed a model for protein-coding DNA sequences that addresses most of the weaknesses. In particular, they use the GM model to generate changes, but then have these changes fixed or lost according to the equilibrium frequencies of the amino acid that would result from the changes. The model is extensible to other types of sequences.

In our experimental studies (to be described later in the dissertation), we used the General Time Reversible (GTR) model, with the Γ-distributed rate variation among sites. Further, it is often biologically realistic to assume that a certain number of sites are invariant, i.e., has zero rate of change. Hence, we use the GTR+Γ+I model, where *I* stands for "invariant" sites.

Similar, but necessarily more complex, models can be built from larger sequence units: from codons, for instance, one can build a mode of amino-acid evolution, using, among other things, a 20×20 substitution matrix.

2.2.3 Simulation Tools for Studying the Performance of Tree Reconstruction Methods

Various simulation tools have been designed and implemented for studying the performance of tree reconstruction methods; see [1] for an extensive list of such tools. Of the many existing tools, two are directly related to the tools we have developed for studying the quality of network reconstruction methods (described in Chapter 5):

- r8s [168]: a tool for generating random birth-death trees.
- Seq-Gen [144]: a tool for simulating DNA sequence evolution on trees.

We now briefly describe each of the two tools.

2.2.3.1 r8s

Model trees are typically taken from some underlying distribution on all rooted binary trees with n leaves; commonly used distributions include the uniform distribution and the Yule-Harding distribution [66, 199]. The tool r8s [168] generates random trees based on the widely used model of birth-death evolution, which we now briefly review.

To generate a random birth-death tree on n leaves, we view speciation and extinction events as occurring over a continuous interval. During a short time interval, Δt, since the last event, a species can split into two with probability $b(t)\Delta t$ or become extinct with probability $d(t)\Delta t$. To generate a tree with n taxa, we begin this process with a single node and continue until we have a tree with n leaves. (With some nonzero probability some processes will not produce a tree of the desired size, since all nodes could go "extinct" before n species are generated; we then repeat the process until a tree of the desired size is generated.) Under this distribution, a natural length is associated with each edge, namely the time elapsed between the speciation event that gave rise to that edge and the (speciation or extinction) event that ended that edge. Thus birth-death trees are inherently ultrametric, that is, the branch lengths are proportional to time.

2.2.3.2 Seq-Gen

Of the many tools for simulating the evolution of biomolecular sequences on trees, Seq-Gen [144] is one of the most popular. This tool takes a tree in the Newick format [132] and simulates the evolution of sequences along that tree, using various models of molecular evolution, including the Hasegawa-Kishino-Yano [69], F84 [51], and GTR [197] models (including all the models that are special cases of the GTR model, such as JC and K2P). Seq-Gen simulates the evolution of sequences on a tree by placing a random sequence of the desired length at the root of the tree and then evolving it down the tree using the specified model of evolution and other parameters, including scaling factor, codon-specific corrections, gamma rate heterogeneity, etc.

We briefly review the options that the user can specify in Seq-Gen (the description of the options is taken from [144]):

- **-m *MODEL***

  This option sets the model of nucleotide substitution with a choice of either F84, HKY, or REV (Markov general reversible model). Other models such as K2P, F81, and JC are special cases of HKY and can be obtained by setting the nucleotide frequency equal (for K2P), the transition/transversion ratio to 1.0 (for F81), or both (for JC).

- **-l *SEQUENCE_LENGTH***

  This option allows the user to specify the length of the sequences (in nucleotides). *SEQUENCE_LENGTH* is a natural number.

- **-n *NUMBDER_OF_DATASETS***

  This option allows the user to specify how many separate datasets should be simulated. *NUMBER_OF_DATASETS* is a natural number.

- **-s *SCALE***

  This option allows the user to set a value with which to scale the branch lengths in order to make them equal the expected number of substitutions per site for each branch. Basically, NetSeqGen multiplies each branch length by this value. *SCALE* is a real number greater than zero.

- **-c *CODON_POSITION_RATES***

  This option allows the user to specify the relative rates for each codon position. This allows codon-specific rate heterogeneity to be simulated.

The default is no site-specific rate heterogeneity.

CODON_POSITION_RATES is three decimal numbers that specify the relative rates of substitution at each codon position, separated by commas or spaces.

- **-a** ***ALPHA***

  This option allows the user to specify a shape for the gamma rate heterogeneity called alpha. The default is no site-specific rate heterogeneity. When *ALPHA* is a real number greater than 0 that specifies the shape of the gamma distribution to use with gamma rate heterogeneity. If this is used with the **-g** option, below, then a discrete model is used, otherwise a continuous one.

- **-g** ***NUM_CATEGORIES***

  This option allows the user to specify the number of categories for the discrete gamma rate heterogeneity model. The default is no site-specific rate heterogeneity (or the continuous model if only the **-a** option is specified). *NUM_CATEGORIES* is an integer between 2 and 32 that specifies the number of categories to use with the discrete gamma rate heterogeneity model.

- **-i** ***PROPORTION_INVARIABLE***

  This option allows the user to specify the proportion of sites that should be invariable. These sites will be chosen randomly with this expected frequency. The default is no invariable sites. Invariable sites are sites that cannot change as opposed to sites which do not exhibit any changes due to chance (and perhaps a low rate). *PROPORTION_INVARIABLE* is a real number greater than or equal to 0 and less than 1 that specifies the proportion of invariable sites.

- **-f *NUCLEOTIDE_FREQUENCIES***

  This option allows the user to specify the relative frequencies of the four nucleotides. By default, NetSeqGen will assume these to be equal. If the given values do not sum to 1, then they will be scaled so that they do. *NUCLEOTIDE_FREQUENCIES* are four decimal numbers for the frequencies of A, C, G, and T, respectively, separated by spaces or commas.

- **-t *TSTV_RATIO***

  This option allows the user to specify the value of the transition/transversion ration (TS/TV). This is only valid when either the HKY or F84 model has been selected. *TSTV_RATIO* is a decimal number greater than 0.

- **-r *RATE_MATRIX_VALUES***

  This option allows the user to set 6 values for the general time reversible model's rate matrix. This is only valid when the REV model has been selected. *RATE_MATRIX_VALUES* are 6 decimal numbers for the rates of transition from A to C, A to G, A to T, C to G, C to T, and G to T, respectively, separated by commas or spaces. The matrix is symmetric; therefore only six values need be set. These values will be scaled such that the last value (G to T) is 1 and the others are set relative to this.

- **-q**

  This option prevents any output except for the sequences and any error messages.

- **-h**

  This option prints a help message describing the options and then quits.

2.2.4 Measuring the Topological Accuracy of Tree Reconstruction Methods

Of the several measures available for quantifying the topological accuracy of tree reconstruction methods, the most commonly used is the Robinson-Foulds (RF) measure [161], which we now review.

The Robinson-Foulds measure Every edge e in a leaf-labeled tree T defines a bipartition π_e on the leaves (induced by the deletion of e), so that we can define the set $C(T) = \{\pi_e \colon e \in E(T)\}$, where $E(T)$ is the set of all internal edges of T; this is called the *character encoding* of the tree T. If T is a model tree and T' is the tree inferred by a phylogenetic reconstruction method, we define the *false positives* to be the edges of the set $C(T') - C(T)$ and the *false negatives* to be those of the set $C(T) - C(T')$. We can then compute the error *rates* by normalizing these values by the number of internal edges in the model tree; since we assume that model trees are binary, this is $n-3$ for n-leaf trees. We obtain:

- The *false positive rate (FP)* is $\frac{|C(T') - C(T)|}{n-3}$.
- The *false negative rate (FN)* is $\frac{|C(T) - C(T')|}{n-3}$.

When both trees are binary, we have $FP = FN$; in general, we have $FP \leq FN$. Since n is the number of internal edges of a rooted binary tree on n leaves, the false positive and false negative rates are values in the range $[0, 1]$. The RF distance between T and T' is simply the average of these two rates, i.e., $\frac{FN+FP}{2}$. (An equivalent formulation is $RF(T, T') = \frac{1}{2}|C(T) \triangle C(T')|$, where $\triangle$ denotes the symmetric difference.) Error rates below 10% are considered not too bad, but systematists prefer to see error rates below 5%.

2.3 Phylogenetic Reconstruction Methods

There are three basic types of phylogenetic reconstruction methods in common use: distance-based methods, maximum parsimony heuristics, and maximum likelihood heuristics. Generally, only the distance-based methods operate in polynomial time, since the other methods attempt to solve NP-hard optimization problems. (Distance-based methods may also attempt to solve NP-hard optimization problems, but there are polynomial-time distance-based methods, such as *neighbor joining (NJ)* [165], that perform very well in practice and that do not attempt to solve NP-hard optimization problems.)

2.3.1 Distance-based Methods

Distance-based methods operate by first estimating pairwise distances and then computing an edge-weighted tree using those distances. Such methods are guaranteed to reconstruct the true tree if their estimates of pairwise distances are sufficiently close to the number of evolutionary events between pairs of taxa [92]. For many models of biomolecular sequence evolution, estimation of sufficiently accurate pairwise distances is possible [98]; for example, log-det [177] distances are statistically consistent estimators for the General Markov model of evolution. Therefore, distance-based methods are statistically consistent for the General Markov model of DNA sequence evolution, and hence for its constituent submodels. Distance-based methods are typically very fast: most run in $O(n^3)$ time, where n is the number of taxa. Of the various distance-based methods in use, neighbor joining [165] is certainly the most popular.

2.3.2 Maximum Parsimony Heuristics

Parsimony is one of the most popular methods used for tree reconstruction, although its applicability is highly debatable in the systematic biology community. Roughly, this method is based on the assumption that "evolution is parsimonious", i.e., the best evolutionary trees are the ones that minimize the number of changes along the edges of the tree. To formalize this concept, we begin with the following definitions.

Definition 2.3.1. The Hamming distance between two equal-length sequences

x and y is the number of positions j such that $x_j \neq y_j$, and is denoted by $H(x,y)$.

Given a fully-labeled tree T, i.e., a tree in which each node v is labeled by a sequence s_v over some alphabet Σ, we define the Hamming distance of an edge $e \in E(T)$, denoted by $H(e)$, to be $H(s_u, s_v)$, where u and v are the two endpoints of e. We now define the parsimony score of a tree T.

Definition 2.3.2. The parsimony score of a fully-labeled tree T, is $\sum_{e \in E(T)} H(e)$. Given a set S of sequences, a maximum parsimony tree for S is a tree leaf-labeled by S and assigned labels for the internal nodes, of minimum parsimony score.

Given a set S of sequences, the parsimony problem is to find a maximum parsimony tree T for the set S. Unfortunately, this problem is NP-hard, even when the sequences are binary [29, 53]. One approach that is used in practice is to look at as many leaf-labeled trees as possible, and choose one with a minimum parsimony score. The problem of computing the parsimony score of a fixed leaf-labeled tree is solvable in polynomial time [52, 68].

2.3.2.1 Inconsistency of Parsimony

The parsimony method may be statistically inconsistent; in other words, parsimony may infer the wrong topology of a tree even when given sequences of infinite length [46]. An example of the inconsistency of maximum parsimony (taken from [75]) is illustrated in Figure 2.1. The four-taxon tree in

Figure 2.1(a), where the lengths of the edges indicate the relative amount of evolutionary change along each edge, is an example of the Felsenstein zone. Suppose that the short edges of the tree (edges 3, 4, and 5) are so short that there are essentially no changes along the lineages leading to taxa B and C (edges 3 and 4, respectively). Consequently, B and C will have the same sequences, and will be grouped together under the parsimony criterion, as reflected by the tree in Figure 2.1(b).

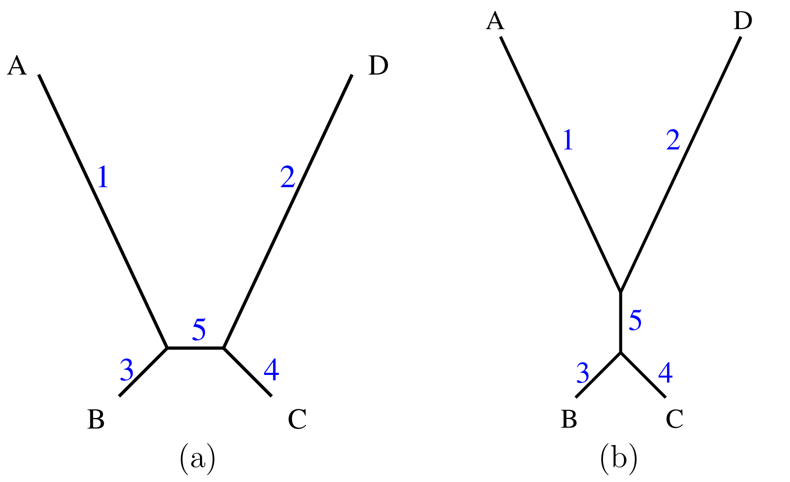


Figure 2.1: (a) A four-taxon tree that contains two long edges (adjacent to leaves), with all other edges being very short. (b) An incorrect tree inferred by maximum parsimony.

2.3.3 Maximum Likelihood Heuristics

Another major criterion for phylogeny reconstruction from molecular sequences is *Maximum Likelihood* (ML) [50]. In the ML problem, we seek the tree T and its associated parameters (such as edge-lengths, rates of evolution for each site, etc.) that maximize the probability of generating the given set of sequences. The general idea behind maximum likelihood estimators is the observation that

$$P(Model|Data) = \frac{P(Model\ and\ Data)}{P(Data)} = \frac{P(Data|Model)P(Model)}{P(Data)}.$$

In this formulation, $P(Model|Data)$ is proportional to $P(Data|Model)$; therefore, we can justify estimating a model by finding the model that maximizes the conditional probability $P(Data|Model)$, which is also called the *likelihood* of the data.

From a practical standpoint, the ML problem is difficult on two levels: first finding the best leaf-labelled tree and then finding the edge parameters for that tree. In practice, hill climbing heuristics are used for both optimization problems (setting edge parameters, and finding the best tree), and heuristics for ML are generally slower than heuristics for MP. If solved exactly, however, ML is statistically consistent under the General Markov model of evolution, and therefore under all of its submodels. Bayesian approaches are also used; these seeks to estimate parameters in much the same way, but the goal is the recovery of the conditional posterior probabilities, using some initial estimate (or guess) for the *a priori* probability.

2.3.4 Perfect Phylogeny: Homoplasy-free Evolution

Some types of qualitative characters, such as morphological features, exhibit properties that cannot be adequately described by parsimony. For example, the qualitative character *vertebrate-invertebrate* imposes a very precise constraint on the topology of the evolutionary tree, and that is that there should be an edge in the tree whose removal separates the vertebrates from the invertebrates.

The assumption of the historical linguistic community is that qualitative characters evolve in such a way that there is no backmutation or parallel evolution. What this means is that when the state of the qualitative character changes in the evolutionary history of the set of languages, it changes to a state which does not exist anywhere else on earth at that time, nor has it appeared earlier.

Definition 2.3.3. Suppose T is a rooted binary tree describing the evolution of a set S of taxa, and that a qualitative character α is defined for each of the taxa in S. We say that α is compatible (or "convex") on T if we can extend α to every internal node of the tree T, thus defining a qualitative character α', so that for every state, the nodes having that state form a connected subgraph of T. In other words, $\forall z \in Z, \{v \in V(T) : \alpha'(v) = z\}$ is connected.

The compatibility problem is defined based on this property. Given a set S of taxa defined by a set C of characters, the maximum compatibility problem is to find a tree T on which a maximum number of characters in C are compatible. Unfortunately, the maximum compatibility problem is NP-hard, even for binary characters [31, 139]. If all characters are compatible on a tree T, we call it a perfect phylogeny.

Definition 2.3.4. Let C be a set of qualitative characters defined on a set

S of taxa. A tree T is a perfect phylogeny for C and S if every qualitative character in C is compatible on T.

The perfect phylogeny problem is defined using this property. Given a set S of taxa defined by a set C of characters, the perfect phylogeny problem is to decide whether a tree T on which all character in C are compatible exists, and if so, to construct it.

Unfortunately, the perfect phylogeny problem is NP-Complete. However, the problem is solvable in polynomial time if any of the three parameters to the problem (the number of taxa, the number of qualitative characters, and the number of states per qualitative character) is bounded by a constant. See [5, 91, 139] for these results.

Part II

Phylogenetic Networks in Biology

Chapter 3

Biology of Reticulate Evolution

Reticulation in biology refers to the lack of independence between two evolutionary lineages. In effect, when reticulation occurs, two or more independent evolutionary lineages are combined at some level of biological organization. Because life is organized hierarchically, reticulation can occur at different levels: chromosomes, genomes and species.

At the species level, events such as hybrid speciation (by which two lineages recombine to create a new one) and horizontal gene transfer (by which genes are transferred across species) are the main causes of reticulate evolution. Within each lineage, at the population level, sexual recombination causes evolution to be reticulate, whereas meiotic recombination causes the shuffling of genes at the chromosomal level. Figure 3.1 illustrates these three scenarios. The phylogeny in Figure 3.1(a) depicts a hybrid speciation scenario, in which species C is the product of hybridization between B and D. Zooming in on a lineage of the phylogeny gives a picture of reticulate evolution at the population level, as in Figure 3.1(b). Finally, zooming in on an individual in each population, reticulation between chromosomes can be viewed, as in Figure 3.1(c). Looking through a macroevolution lens (evolution among lineages),

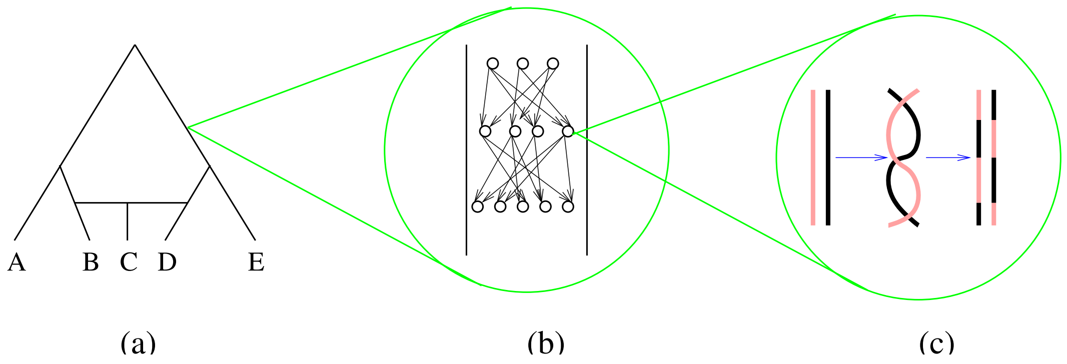


Figure 3.1: Reticulation at various levels: (a) species level, (b) population level, and (c) individual (chromosomal) level.

only reticulate events at the species level fail to be modeled by a bifurcating tree. However, looking through a microevolution lens (evolution within a lineage), sexual and meiotic recombination fail to be modeled by a bifurcating tree. Since phylogenies are usually constructed at either the population or the species level, meiotic recombination does not cause a species-level reticulate evolutionary history, but it can confound species-level inference of reticulation by producing patterns that have the appearance of species-level reticulation. We now briefly explain reticulation at the chromosomal, population and species levels.

3.1 Reticulation Between Chromosome Pairs: Meiotic Recombination

During each round of sexual reproduction, the total number of chromosomes must be halved to produce the gametes. The process is called *meiosis* and during one phase of it the chromosome pairs (sister chromatids) exchange pieces in a precise fashion known as *meiotic recombination.* The net result is chromatids that have two or more evolutionary histories on them. Blocks of chromosomes that share a single evolutionary history are referred to as haplotype blocks.

3.2 Reticulation Within a Lineage: Sexual Recombination

For sexually reproducing organisms, there is recombination of nuclear genomes during each bout of reproduction. Each parent contributes half of its original nuclear genome—one sister chromatid from each chromosome—and each of these chromosomes have themselves undergone meiotic recombination during the process of producing the haploid gametes (sex cells). Because different parts of each parent's contribution to the genome of the next generation may have a different evolutionary history from that of the other parent's contribution, sexual recombination is a form of population-level reticulation. Organellar genomes (mitochondria and chloroplasts) are usually inherited uniparentally so they do not usually undergo any sort of sexual recombination.

3.3 Reticulation Among Lineages: Horizontal Gene Transfer and Hybrid Speciation

In horizontal (also called lateral) gene transfer (HGT for short), genetic material is transferred from one lineage to another; see Figure 3.2(a). In an evolutionary scenario involving horizontal transfer, certain sites (specified by a specific substring within the DNA sequence of the species into which the horizontally transferred DNA was inserted) are inherited through horizontal transfer from another species (as in Figure 3.2(b)), while all others are inherited from the parent (as in Figure 3.2(c)). Thus, *each site evolves down one of the trees contained inside the network*.

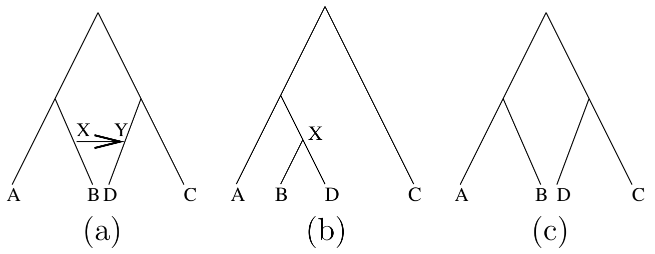


Figure 3.2: Horizontal gene transfer: the species network in (a) and the two possible gene trees in (b) and (c).

Horizontal gene transfer is of paramount importance in the study of the evolution of prokaryotes and, to a lesser extent, the eukaryotic lineages that have interspecific hybridization. Describing the ubiquity of HGT, de la Cruz and Davies [32] wrote

> It is clear that genes have flowed through the biosphere, as in a global organism. Horizontal gene transfer, once solely of interest for practical applications in classical genetics and biotechnology, has now become the substance of evolution.

Horizontal transfers are believed to be ubiquitous among bacteria and still quite common in other branches of the "tree"—although this view has recently been challenged [40, 94, 166, 167]. Three mechanisms of HGT in the Archaea and Bacteria are *transformation* (uptake of naked DNA from the environment), *conjugation* (transfer of DNA by direct physical interaction between a donor and a recipient), and *transduction* (transfer of DNA by phage infection) [141].

The second non-treelike event acting at the species level is hybrid speciation. In hybrid speciation, two lineages recombine to create a new species; see Figure 3.3(a) for a visual representation of this scenario. The new species may

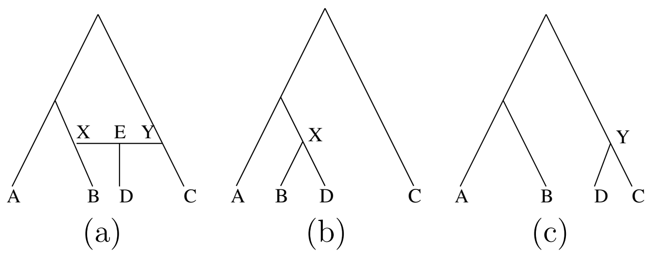


Figure 3.3: Hybrid speciation: the species network in (a) and the two possible gene trees in (b) and (c).

have the same number of chromosomes as its parent (*diploid hybridization*) or the sum of the number of its parents (*polyploid hybridization*).

Hybrid speciation comes about in at least three ways: allopolyploidization, autopolyploidization, and diploid or homoploid hybrid speciation. Autopolyploidization is probably more properly considered a specialized form of normal (bifurcating) speciation since only a single parental species is involved in its production. Allopolyploidization is hybrid speciation between two species resulting in an offspring that has the complete diploid chromosome complement of both its parents. Each parent need not have the same number of chromosomes. Allopolyploidization results in instantaneous speciation because any backcrossing to the diploid parents would produce inviable or sterile triploid offspring. Diploid hybrid speciation is a normal sexual event where each gamete has a haploid complement of the chromosomes from its parent, but the gametes that form the zygote come from different species. In nearly all cases, both parents must have the same number of chromosomes. In this case, successful backcrossing to the parents is possible, so it is thought that the hybrids have to be isolated from the parents by undergoing selection for life in a novel environment [149]. Not surprisingly, diploid hybrid speciation is much rarer than polyploidization.

Consider how an individual site evolves down a network modelling hybrid speciation. For normal diploid organisms, each chromosome consists of a pair of homologs. In a diploid hybridization event, the hybrid inherits one of the two homologs for each chromosome from each of its two parents. Since homologs assort at random into the gametes (sex cells), each has an equal probability of ending up in the hybrid. In polyploid hybridization, both homologs from both parents are contributed to the hybrid. Prior to the hybridization event, each site on the homolog has evolved in a tree-like fashion, although due to meiotic recombination (exchanges between the parental homologs during production of the gametes), different strings of sites may have different histories. Thus, each site in the homologs of the parents of the hybrid evolved in a tree-like fashion on one of the trees contained inside the network representing the hybridization event (see Figures 3.3(b) and 3.3(c)). As in the case of HGT, *each site evolves down one of the trees contained inside the network.*

Hybrid speciation is very common in some groups of organisms (plants, fish, amphibians, some groups of invertebrates, and possibly fungi) and is virtually absent in others, notably mammals and most arthropods. These latter groups do produce very occasional hybrids, but they are usually triploid and are only able to survive by asexual reproduction. Odd ploidy levels cannot reproduce sexually because the odd number of chromosome sets does not allow correct pairing during meiosis. The resulting gametes have unequal numbers of chromosomes and are nearly always inviable. These asexual lineages are considered evolutionary dead ends and are expected to be short lived. The reasons some groups can successfully speciate via hybridization while others cannot is not understood (see [133] for a review). It is thought that developmental complexity may play an important role: animals with complex developmental programs may simply be unable to produce viable offspring after interspecific mating has taken place. It has also been argued that species with chromosomal sex determination (species with distinct sex chromosomes) are much less likely to produce hybrid species. However, the true reasons for the barrier(s) to successful hybrid speciation have yet to be demonstrated.

3.4 Types of Hybrids and Gene Flow

We define the terms relating to types of hybrids and gene flow that we consider.

Haploid. An individual with only a single set of chromosomes (N). Bacteria are like this throughout their entire lives. In most higher plants and animals, the genome is only haploid in the gametes. In lower plants there are significant parts of the life cycle that are haploid.

Diploid. An individual with two sets of chromosomes (2N), one from each of the parents. During ordinary cell division (mitosis), each daughter cell inherits a full set of chromosomes (2N). During the production of gametes (meiosis) the two sets of chromosomes found in each cell are reduced to a single set (N) per cell. For this process to operate correctly, each chromosome must consist of two homologues. When this does not occur (aneuploidy), the homologues do not assort correctly and some gametes will lack some chromosomes.

Aneuploid. A condition in a normally diploid (or other even numbered ploidy level), where not all the chromosomes occur in pairs. This is almost always a pathological condition because it creates a cell where the chromosomes cannot assort properly during meiosis (the production of haploid gamete cells). Chromosomes that are not paired assort at random, creating cells with either incomplete sets of chromosomes or ones with too many of some chromosomes. Most often, aneuploid individuals either produce non-viable gametes or cannot produce viable offsprings. The condition occurs most often when two species that have different numbers of chromosomes or whose chromosomes are sufficiently rearranged with respect to each other hybridize. It almost never leads to a stable hybrid and can be ignored in our models of hybrid speciation.

Homoploid (Diploid) hybridization. Hybridization resulting in new species that has the same number of chromosomes as its parents, e.g., 2N if its parents are 2N. There is no summing of parental chromosome numbers as with polyploidization. Homoploid hybridization is rarer than polyploid hybridization. Usually, a homoploid hybrid will have diploid parents, but it could have parents of other ploidy levels, e.g., two tetraploid parents giving rise to a tetraploid hybrid species. In order for homoploid hybridization to be successful, both parents will usually have to have the same number of chromosomes; otherwise, the hybrid will be aneuploid (see definition for diploid).

Polyploid. A hybrid species with more than two sets of chromosomes (3N, 4N, 5N, 6N,...), relative to their parents. A polyploid individual receives the entire (unreduced) set of chromosomes from each of its parents (e.g., 2N+2N=4N) instead of just half. Because a polyploid individual has a complete set of chromosomes from each of its parents, the parents need not have the exact same number of chromosomes. Aneuploidy does not occur because each chromosome consists of a pair of homologues when it is inherited from the parents. It is possible in some cases to have organisms with odd ploidy levels (e.g., 3N,5N), but they are almost always sterile or only produce the occasional viable gamete (see aneuploid). They usually only survive by reproducing asexually. A good example of this can be found in some types of potatoes that are grown in Peru, which are 3N and 5N. Generally, even-numbered ploidy levels are produced by even-numbered ploidy parents. For example, 2N+2N=4N, but not 1N+3N=4N. (There are rare exceptions that we will not consider for the time being.)

Allopolyploid. A polyploid with parents of two species. In Figure 3.4, Y is an allopolyploid hybrid whose two parents are of two different species, namely X and Z.

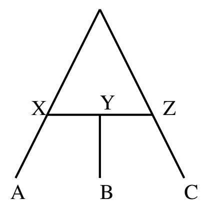


Figure 3.4: An example of allopolyploid hybrid speciation.

Autopolyploid. A polyploid with parents of the same species. (Results in tree-like evolution, except that one of the offsprings has a different ploidy level than its parent.) In Figure 3.5, B is an autopolyploid hybrid whose two parents are of the same species, namely X.

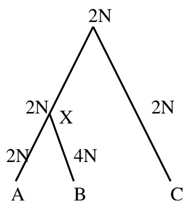


Figure 3.5: An example of autopolyploid hybrid speciation.

Tetraploid. A polyploid with four sets of chromosomes (4N).

Hexaploid. A polyploid with six sets of chromosomes (6N).

Octaploid. A polyploid with eight sets of chromosomes (8N).

Back-crossing. Mating between a hybrid and one of its parents. Back-crossing between homoploid hybrids and their parents is much more common than back-crossing between polyploids and their parents. However, some gene flow can occur between some polyploids and their parents.

Introgression. Incorporation of genes from one species into the genome of another species via hybridization and subsequent back-crossing. Movement of genes from one diploid species (P_1) to another (P_2) always begins with a hybridization event. The two parental species hybridize to produce an F_1 (first generation) hybrid. The hybrid then mates with one of its parents (back-crossing) rather than with another hybrid. Repeated rounds of back-crossing to the same parent rapidly restores the vast majority of the parental genome, but by chance or selection, some of the

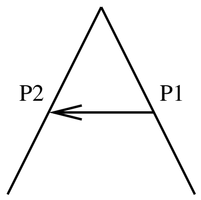


Figure 3.6: An example of an introgression event.

DNA regions from the other parent will remain in the other parent's genome, e.g., in Figure 3.6, a small stretch of P_1's genome may remain in P_2's genome.

Chapter 4

Mathematical Models of Reticulate Evolution

4.1 Population Level

Strimmer *et al.* [179] proposed DAGs (directed acyclic graphs) as a model for describing the evolutionary history of a set of sequences under recombination events. They also described a set of properties that a DAG must possess in order to provide a realistic model of recombination. Strimmer *et al.* [180] proposed adopting *ancestral recombination graphs* (ARGs), due to Hudson [82] and Griffiths and Marjoram [60] as a more appropriate model of phylogenetic networks. ARGs are rooted graphs that provide a way to represent linked collections of clock-like trees by a single network. Another network-like model is pedigrees, designed to represent the parentage of individual organisms—so that the indegree of each internal node of a pedigree is either 0 or 2, thereby not allowing for tree nodes [140]; Figure 4.1 gives an example of a pedigree, where squares represent males, and circles represent females. Nodes of indegree 2 correspond to recombination events.

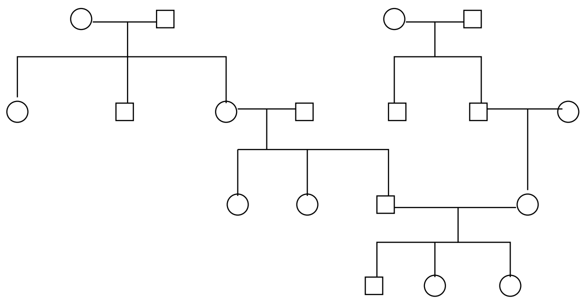


Figure 4.1: An example of a pedigree.

4.2 Species Level

Like Strimmer *et al.* [179], we use DAGs to describe the topology of our phylogenetic networks and, like Hallett and Lagergren [65], we add a set of (mostly simpler) conditions to ensure that the resulting DAGs reflect the properties of hybridization. We now define the mathematical model of phylogenetic networks, and distinguish between model networks and reconstructible ones based on the properties they possess [102].

4.2.1 Model networks

A phylogenetic network $N = (V, E)$ is a rooted DAG obeying the following constraints. The set V of nodes is partitioned into two sets:

- V_T: the set of **tree nodes**. A node $v \in V$ is a tree node if and only if one of these three conditions holds:
  - $indegree(v) = 0$ and $outdegree(v) = 2$: v is the root,

    - $indegree(v) = 1$ and $outdegree(v) = 0$: v is a leaf, or
    - $indegree(v) = 1$ and $outdegree(v) = 2$: v is an internal node.

- V_N: the set of **network nodes**. A node $v \in V$ is a network node if and only if $indegree(v) = 2$ and $outdegree(v) = 1$.

Tree nodes correspond to regular speciation or extinction events, whereas network nodes correspond to reticulation events (such as hybrid speciation and lateral gene transfer). We clearly have $V_T \cap V_N = \emptyset$ and can easily verify that we have $V_T \cup V_N = V$.

The set E of edges is partitioned into two sets:

- E_T: the set of **tree edges**. An edge $e = (u, v) \in E$ is a tree edge if and only if v is a tree node.
- E_N: the set of **network edges**. An edge $e = (u, v) \in E$ is a network node if and only if v is a network node.

The tree edges are directed from the root of the network towards the leaves and the network edges are directed from their tree-node endpoint towards their network-node endpoint. For any pair of nodes u and v in V, if (u, v) is an edge in E, then at least one of u or v is a tree node. Figure 4.2 shows an example of a phylogenetic network in which the species at node Z is the product of (homoploid or allo-polyploid) hybridization. Dashed edges denote network edges and solid edges denote tree edges.

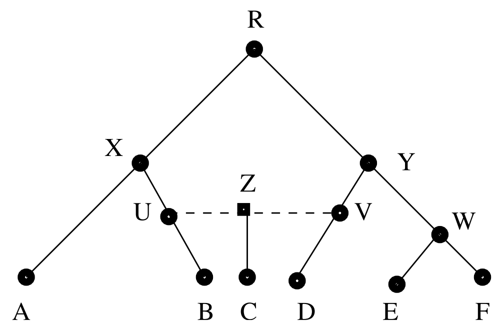


Figure 4.2: A phylogenetic network N on 6 species. The solid circles denote the tree nodes, and the solid squares denote the network nodes. The solid lines denote the tree edges, and the dashed lines denote the network edges.

In such a network, a species appears as a directed path p that does not contain any network edge (since a network edge connects between two existing species or between an existing species and a newly-created one). If p_1 and p_2 are two directed paths that define two distinct species, then p_1 and p_2 must be edge-disjoint, that is, the two paths cannot share edges. For example, the directed path p from node X to node B in Figure 4.2 could define species B (as could the path from R to B), whereas the directed path from node X to node C does not define a species, since it contains a network edge.

A phylogenetic network $N = (V, E)$ defines a partial order on the set V of nodes. Based on this partial order, we assign times to the nodes of N, associating time $t(u)$ with node u. If there is a directed path p from node u to node v, such that p contains at least one tree edge, then we must have $t(u) < t(v)$ (in order to respect the time flow). If $e = (u, v)$ is a network edge, then we must have $t(u) = t(v)$ (because a reticulation event is, at the scale of evolution, an instantaneous process). Because of that property, the orientation of a network edge is irrelevant when examining time flows.

Given a network N, we say that p is a *positive-time directed path* from u to v, if p is a directed path from u to v, and p contains at least one tree edge. Given a network N, two nodes u and v cannot co-exist in time if the following condition holds:

- there exists a sequence $P = \langle p_1, p_2, \ldots, p_k \rangle$ of paths such that:
  - p_i is a positive-time directed path, for every $1 \leq i \leq k$,
  - u is the tail of p_1, and v is the head of p_k, and
  - for every $1 \leq i \leq k-1$, there exists a network node whose two parents are the head of p_i and the tail of p_{i+1}.

Notice that the simple condition of having a positive-time directed path from u to v is, by itself, not sufficient to capture all temporal constraints imposed by reticulation events; Figure 4.3 illustrates this point. In Figure 4.3, $t(Y) = t1$ and $t(X) = t4$; further, reticulation events $H1$ and $H2$ occur at times $t2$ and $t3$, respectively. It is obvious that the two reticulation events imply $t1 < t2 < t3 < t4$, which, in turn, implies that X and Y cannot co-exist in time, and hence cannot be the "parents" of a reticulation event. Obviously, there does not exist a positive-time directed path from Y to X. Yet, there exists a sequence $P = \langle p_1, p_2, p_3 \rangle$ of positive-time directed paths, where p_1 is the directed path from Y to A, p_2 is the directed path from B to C, and p_3 is the directed path from D to X; further, $H1$ is a network node whose two parents are A and B, and $H2$ is a network node whose two parents are C and D. Hence, according to our definition, X and Y cannot co-exist in time.

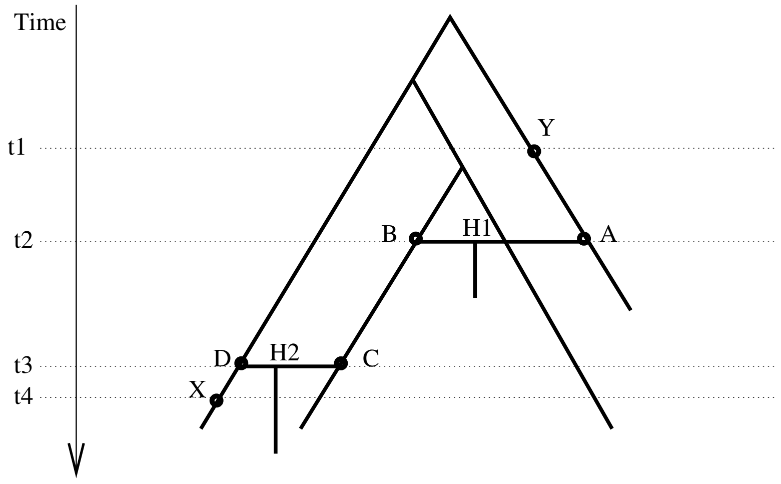


Figure 4.3: A scenario illustrating two nodes X and Y that cannot co-exist in time.

Since reticulation events (such as hybrid speciation and lateral gene transfer) occur between two lineages (nodes in the network) that co-exist in time [106, 137], a phylogenetic network N must satisfy the following property:

- If two nodes x and y cannot co-exist in time, then a reticulation event between them is impossible.

For example, if x and y cannot co-exist in time, then they cannot be the parents of a hybrid, nor can a lateral gene transfer event occur between them.

Graph-theoretically, in the case of hybrid speciation, there cannot exist a node v with the two edges (x, v) and (y, v) both in E, and, in the case of lateral gene transfer, neither (x, y) nor (y, x) is an edge in E.

4.2.2 Reconstructible Networks

The topology of the reconstructed phylogeny (tree or network) can differ from the topology of model phylogeny due to various factors:

- **No-change edges.** If the amount of change on an edge $e = (u, v)$ in the model phylogeny is very small (possibly 0), the edge e may not be reconstructible. This scenario may lead to a polytomy (node of outdegree greater than 2) in the reconstructed phylogeny. Also, a lineage may undergo two (or more) simultaneous reticulation events, which may also lead to nodes of outdegrees higher than 2.

- **Extinction.** If a certain lineage becomes extinct, that lineage may not be reconstructible. In the case of phylogenetic trees, the topology would still be modeled as a bifurcating tree (after elimination of nodes of indegree and outdegree 1).

- **Taxon sampling.** When only a subset S' of the taxa is sampled and a phylogeny is reconstructed on S', the resulting network need not be compatible with the model.

Because of these three problems, in fact, the topology of the reconstructed network may violate the conditions and properties of the model topology described in Section 4.2.1. We thus relax the model described in Section 4.2.1 to obtain a model of *reconstructible* networks, by allowing the following:

- $outdegree(v) \geq 2$ for any internal tree node.
- $indegree(v) = 2$ and $outdegree(v) \geq 1$ for any network node v.
- A tree edge may be incident into a network node.
- The network may violate the time co-existence property.

In Figure 4.4(a) we show a network N on a set of 5 species, where species B

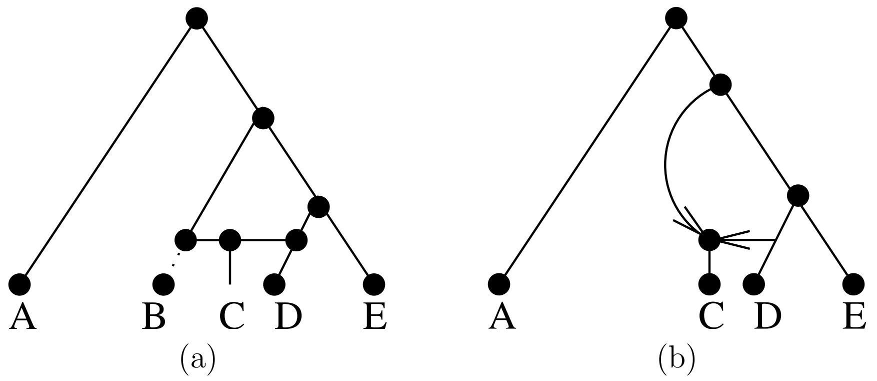


Figure 4.4: (a) A phylogenetic network N on 5 species, where species B had gone extinct. (b) the same network under the reconstructible network model.

had become extinct (or was not sampled). Figure 4.4(b) shows the topology of the same network under the reconstructible network model. The network in Figure 4.4(b) violates time co-existence, and contains a pair of edges, one of which is a tree edge and the other a network edge, both incident into a network node (a scenario that is not allowed under the model described in Section 4.2.1).

4.2.3 k-GT-Networks

We consider a biologically-motivated restricted class of phylogenetic networks, called *gt-networks*, proposed by Wang *et al.* [186] and Gusfield *et al.* [63] (where these networks are called "galled trees").

Definition 4.2.1. In a phylogenetic network N, let w be a node that has two directed paths out of it that meet at a reticulation node x. Those two directed paths together define a "reticulation cycle" Q. Node w is called the "coalescent node" of Q, and x is the "reticulation node" of Q.

Definition 4.2.2. A reticulation cycle in a phylogenetic network that shares no nodes with any other reticulation cycle is called a "gall".

We denote by Q_x^w a gall whose coalescent node is w and whose reticulation node is x. We denote by $E(Q_x^w)$ the set of all edges on gall Q; formally, $E(Q_x^w) = \{e : e \text{ is an edge on a directed path from } w \text{ to } x\}$. The set $RE(Q_x^w)$ (for "reticulation edges") denotes the edges whose head is x, i.e., the edges incident into x. When the context is clear, we simply write Q for a gall, without explicitly naming the coalescent and reticulation nodes.

Definition 4.2.3. A phylogenetic network N is called a "gt-network" if every reticulation cycle is a gall.

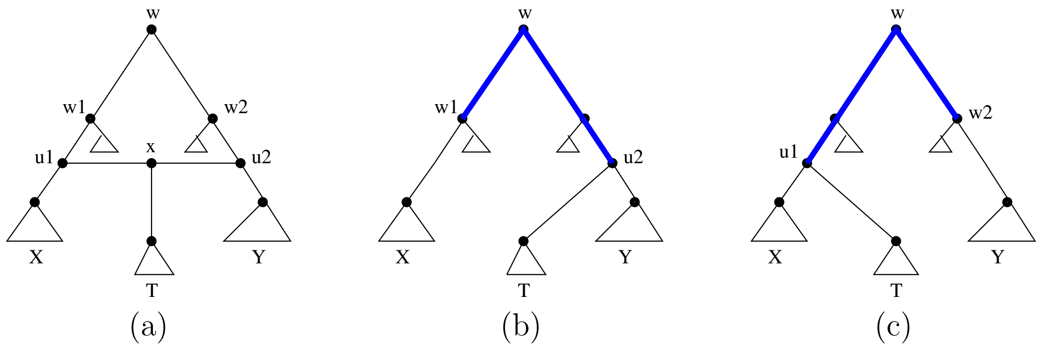


Figure 4.5: (a) A gall Q whose coalescent and reticulation nodes are w and x respectively. (b) and (c) show the two possible ways of "breaking" the gall Q to induce trees T_1 and T_2, respectively. The marked edges in T_1 and T_2 form $RP^Q(T_1)$ and $RP^Q(T_2)$, respectively.

Figure 4.5(a) shows a gt-network N with a gall Q_x^w. The set $E(Q_x^w)$ contains the edges (w, w_1), (w_1, u_1), (w, w_2), (w_2, u_2), (u_1, x), and (u_2, x). The set $RE(Q_x^w)$ contains the two edges (u_1, x) and (u_2, x). Obviously, gt-networks satisfy the synchronization property. In this paper, we assume that there is at least one tree node on each of the two paths from w to x in a gall Q_x^w (otherwise, the network would violate the synchronization property).

We *break* a gall Q_x^w by removing exactly one of the edges in the set $RE(Q_x^w)$.

Definition 4.2.4. A tree T is **induced** by a gt-network N if T can be obtained from N through one of the possible ways of breaking all the galls in N, followed by forced contraction operations on all nodes of indegree and outdegree 1.

Figures 4.5(b) and 4.5(c) show the two possible trees induced by the gt-network N in Figure 4.5(a). To obtain the tree in Figure 4.5(b), the gall was broken by removing edge (u_1, x) and applying forced contraction to node x; to obtain the tree in Figure 4.5(c), the gall was broken by removing edge (u_2, x) and applying forced contraction to node x. In general, given a network N with p reticulation nodes, we say that a tree T is *induced* by N if T can be obtained by removing exactly one of the two edges incoming into each of the p reticulation nodes in N.

Definition 4.2.5. Let Q_x^w be gall in a gt-network N, with $RE(Q) = \{e_1 = (u_1, x), e_2 = (u_2, x)\}$. Further, let w_1 be the parent of u_1, and w_2 be the parent of u_2. Assume tree T_1 is obtained from N by removing edge e_1, and tree T_2 is obtained from N by removing edge e_2. The two directed paths $w \rightsquigarrow w_1$ and $w \rightsquigarrow u_2$ together define a "reticulation path" in T_1, and the two directed paths $w \rightsquigarrow w_2$ and $w \rightsquigarrow u_1$ together define a "reticulation path" in T_2.

Given a gt-network with m galls, there are 2^m possible ways of breaking the m galls, and thus inducing a tree. There is a direct correspondence between the edges and nodes of a gt-network N and a tree T induced by N, and hence we talk about a node or edge of T in N, or a node or edge of N in T (excluding the edges in $RE(Q)$ and the nodes removed by forced contraction). We denote by $RP^Q(T)$ the "reticulation path" in T that results from breaking gall Q. The marked edges in tree T_1 of Figure 4.5(b) form the reticulation path $RP^Q(T_1)$, and the marked edges in tree T_2 of Figure 4.5(c) form the reticulation path $RP^Q(T_2)$ (we also use $RP^Q(T)$ to denote the edges on the reticulation path in T).

The definition of galls precludes their being nested, yet nested reticulation cycles do occur in biological data and offer interesting generalizations of galls.

Definition 4.2.6. A *k-gall* is defined recursively as follows.

- A *0-gall* is any single node.
- A *1-gall* is a gall.
- If nodes w and x are the coalescent, resp. reticulation, nodes of a reticulation cycle, and nodes along the two paths belong to i-galls, for $i < k$, with at least one belonging to a $(k-1)$-gall, then the reticulation cycle forms a k-gall.

In other words, a k-gall contains $k-1$ levels of nested galls. A *k-gt-network* is then a network in which every reticulation cycle is an i-gall, for $i \leq k$, and at least one is a k-gall. Figure 4.6 shows an example of a 3-gt-network.

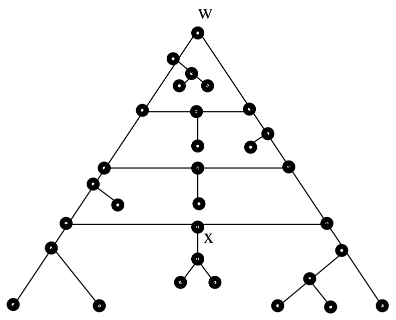


Figure 4.6: An example of a 3-gt-network.

Chapter 5

Simulation Tools for Studying Phylogenetic Network Reconstruction Methods

Simulations are the standard methodology for studying phylogenetic reconstruction methods and assessing their accuracy. Almost all existing simulation tools, however, were developed for tree reconstruction methods. In [127] we reported on the first suite of simulation tools for studying network reconstruction methods. In this chapter we review those tools.

5.1 Network File Format

In most phylogenetic tools, trees are represented using the Newick format [132], which is a form of post-order traversal of trees. Figure 5.1(b) shows the Newick format representation of the tree given in Figure 5.1(a).

However, this representation is not suitable for networks, due to the presence of nodes with indegree larger than one. We adopt a format for network representation which is a natural extension of the DIMACS network flow file format [3].

The following information makes up a network input file:

- comment lines

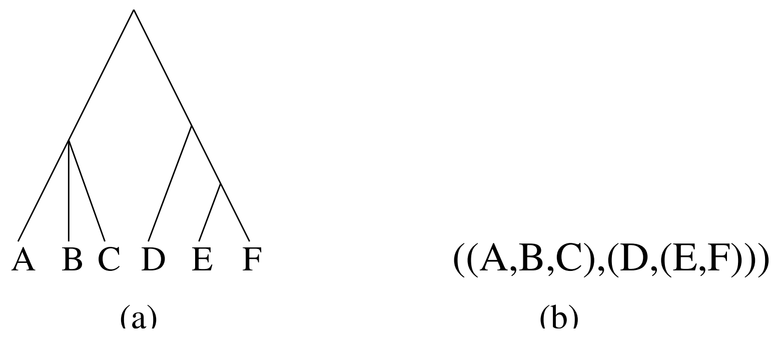


Figure 5.1: The topology of a phylogenetic tree in (a), and its representation in the Newick format in (b).

- problem line
- internal node descriptors
- leaf node descriptors
- arc descriptors

As noted above, information is collected into *lines*, which begin with one character designators. We describe each type of information line in turn.

Comment Lines Those lines give human-readable information about the file and are ignored in programs. Comment lines can appear anywhere in the file. Each comment line begins with a lower-case character **c**. Examples of essential information that can be printed on these lines are the various parameters that were used to generate the network, such as the speciation/extinction rates, number of taxa, etc.

c This Is A Comment Line.

Problem Line There is a single problem line per network file. The problem begins with a lower-case character **p**, and the line must appear before any node or arc descriptor lines. The problem line has the following format:

p NODES ARCS LEAVES HYBRIDS OUTGROUP HEIGHT

The **NODES** field contains an integer value specifying n – the number of nodes. The **ARCS** field contains an integer value specifying the number of arcs. The **LEAVES** field contains an integer value specifying the number of leaves in the network. The **HYBRIDS** field contains an integer value specifying the number of nodes with indegree two in the network. The **OUTGROUP** is either -1 or an integer between 1 and n. If **OUTGROUP=-1**, that implies that the network does not have an outgroup taxon; otherwise, the **OUTGROUP** value indicates the *id* of the outgroup node. The **HEIGHT** field is a non-negative real number that indicates the longest root-to-leaf path in the network.

Internal Node Descriptors All internal node descriptors must appear before all arc descriptor lines. The internal node descriptor lines specify the *id*'s and types of the nodes. There is one internal node descriptor line for each such node, with the following format.

n ID TYPE

The lower-case character **n** signifies that this is an internal node descriptor line. The **ID** field gives a node identification number, an integer between 1 and n, and the **TYPE** field contains an integer value specifying the type of the node (0: tree-node, 1: diploid hybrid, 2: allo-tetraploid hybrid, 3: allo-hexaploid hybrid, 4: allo-octaploid hybrid, 5: auto-tetraploid hybrid, 6: auto-octaploid hybrid). We treat leaves as tree nodes.

Leaf Node Descriptors All leaf node descriptor lines must appear before all arc descriptor lines. The leaf node descriptor lines specify the *id*'s, names, and types of the leaves. There is one leaf node descriptor line for each such node, with the following format.

t ID NAME TYPE

The lower-case character **t** signifies that this is a leaf node descriptor line. The **ID** field gives a node identification number, an integer between 1 and n, the **NAME** field gives the name of the node, and the **TYPE** field contains an integer value specifying the type of the leaf (the same as the types of internal nodes).

Arc Descriptors There is one arc descriptor line for each arc in the network. Arc descriptor lines are of the following format.

a SRC DIST WEIGHT

The lower-case character **a** signifies that this is an arc descriptor line. For a directed arc (u, v) the **SRC** field gives the identification number for the source vertex u, and the **DST** field gives the destination vertex v. Identification numbers are integers between 1 and n. The **WEIGHT** field contains a real number that specifies the expected number of changes on the arc.

5.2 GenNet: A Tool for Generating Random Networks

5.2.1 Description of the Tool

We have designed and implemented GenNet [127], a tool for generating random networks. The networks are generated based on a birth-death model. A birth event in trees represents regular speciation; in networks, a birth event can be either regular speciation (by which one lineage splits into two) or hybrid speciation (by which two lineages recombine to create a new one).

Our model of network generation incorporates three types of events: regular speciation, hybrid speciation, and extinction. (Lateral gene transfer events can easily be added as well.) Hybrid speciation events can be of any of the hybrid types described in Section 3.4.

To generate a random network, we start with one node (the root), and initiate a regular speciation event, thus creating two new lineages. Each lineage is represented by an edge in the network. An edge e is defined by its two endpoint nodes u and v. Associated with each node u is a time-stamp, $t(u)$, and associated with each edge e is a positive real number, $w(e)$, indicating the expected number of changes along that edge.

At any time t, we consider all of the lineages that exist at that time. For each such lineage l, and based on certain probabilities, either nothing happens, which means the lineage l is continued without changes, or one of three mutually-exclusive events occurs:

Extinction: Lineage l becomes extinct and a leaf u is created with stamp t.

Regular speciation: A node u is created with stamp t and two new lineages are started from u.

Hybrid speciation: Let H be the set of all lineages at time t and choose a lineage $l' \in H$ to hybridize with l. The choice of l depends on the ploidy level and number of chromosomes of l', as well as the evolutionary distance between l and l'. When l and l' hybridize, the two lineages are continued and a new, third lineage arises from the hybrid speciation event. We allow each lineage to hybridize only once at each point in time.

This process may generate a network with fewer than the desired number of leaves, since all lineages might go extinct before enough lineages are created. In such a case, we can repeat the process until we obtain a network of the desired size or we can conduct longitudinal studies in a population of networks of diverse sizes.

The networks thus generated are ultrametric; for each edge, we use a uniform variate x in the range $[-\ln d, \ln d]$ and multiply the edge length by e^x

to deviate the networks from ultrametricity. We call d the *deviation factor*. Setting $d = 1$ maintains ultrametricity.

As stated above, the probability of a hybrid speciation depends on the distance between the two species under considerations. Two possible distances that can be used are the following:

- $d_1(u, v) = \sum_{e \in P(u,v)} length(e)$, where $P(u, v)$ denotes the path between nodes u and v.
- $d_2(u, v) = dist(s_u, s_v)$, where s_x denotes the sequence(s) stored at node x, and $dist(s_1, s_2)$ is a distance function defined between two sequences.

Our current implementation computes the distances using $d_1(u, v)$.

5.2.2 Running the Tool

GenNet is a command-line tool, and can be run with the command:

gennet -i *infile* -o *outfile*

where *infile* is the name of the input file, and *outfile* is the name of the output file. The output file will contain the description of the generated network, in the format described in Section 5.1.

The input file contains the values of the parameters that control the network generation process. The input file starts with

begin;

and ends with

end;

Enclosed within these two commands, the user can specify the values of various parameters, each on a separate line, as follows:

- **ntax** m

  where m is an integer (greater than 0) denoting the number of taxa (leaves) in the network.

- **outgroup**

  the presence of this line instructs the program to generate an outgroup along with the network. If this command is absent, the network will be generated without an outgroup.

- **deviation** d

  where d is a real number denoting the value of the deviation factor (defined above).

- **threshold** t

  where t is a positive real number denoting the value of the distance threshold. Biologically, it may be the case that lineages whose evolutionary distance is beyond a certain threshold may not be able to successfully hybridize. The threshold value instructs the program to exclude hybrid speciation among such lineages.

- **sp_rate** *sp*

  where *sp* is a nonnegative real number that denotes the (regular as well as hybrid) speciation rate.

- **ex_rate** *ex*

  where *ex* is a nonnegative real number that denotes the extinction rate. The sum of the speciation and extinction rates must be at most one.

- **reg_rate** x

  where x is a nonnegative real number that denotes the regular speciation rate.

- **dip_rate** x

  where x is a nonnegative real number that denotes the diploid hybridization rate.

- **allotetra_rate** x

  where x is a nonnegative real number that denotes the allo-tetraploid hybridization rate.

- **allohexa_rate** x

  where x is a nonnegative real number that denotes the allo-hexaploid hybridization rate.

- **allocta_rate** x

  where x is a nonnegative real number that denotes the allo-octaploid hybridization rate.

- **autotetra_rate** x

  where x is a nonnegative real number that denotes the auto-tetraploid hybridization rate.

- **autoocta_rate** x

  where x is a nonnegative real number that denotes the auto-octaploid hybridization rate.

The following is an example input file:

```
begin;
ntax 20
deviation 16
sp_rate 0.4
ex_rate 0.2
reg_rate 0.4
dip_rate 0.2
alloocta_rate 0.05
allotetra_rate 0.1
allohexa_rate 0.1
autotetra_rate 0.1
autoocta_rate 0.05
threshold 0.2
end;
```

5.3 NetSeqGen: A Tool for Simulating Sequence Evolution on Networks

5.3.1 Description of the Tool

NetSeqGen [127] takes a network as an input; it assumes diploid species at the root, so it starts by placing a pair of sequences at the root, each of which representing one of the two homologs of a chromosome. (It is straightforward to modify the tool to allow for species with different ploidy levels at the root.) The two sequences are then evolved independently down the network, using the specified model of evolution and the other parameters.

In the case of tree nodes, all sequences at a node evolve in exactly the same manner as they would evolve down a tree. In the case of a network node, the node inherits sequences from both of its parents. In the case of diploid hybridization, the hybrid inherits one sequence from each parent. In the case of polyploid hybridization, the hybrid inherits all sequences from both parents. There is no evolution on the edges between the parents of the hybrid and the network node, since, at the scale of evolution, hybridization is essentially instantaneous. The output of NetSeqGen is a list of sets of sequences at the leaves.

5.3.2 Running the Tool

The command-line for running NetSeqGen is:

netseqgen [parameters] < ***infile*** > ***outfile***

where *infile* is the name of the input file that contains the description of the (rooted) network on which the evolution of sequences will be simulated. The program prints the sequences to the standard output, and therefore, to store those sequences in a file, the user needs to redirect the output to the output file *outfile*.

The sequences are printed in the Phylip format. The first line of the output file consists of two numbers, in the form:

- n k

where n is the number of taxa, and k is the length of each sequence. Following this first line, the name of each taxon is given, followed by the set of sequences that were generated for that taxon (each sequence on a separate line). The following is an example of an output file of NetSeqGen, which contains the sequences for 8 taxa, each sequence of length 20. Notice that the numbers of sequences per taxon differ, due to possible polyploidization.

```
8 20
s3      AAGCGCCAGCGTGACCACCC
        TAGCGCCAGCGTGACTACCC
s8      AAGCGCCAGCGTGACCACCC
        AACCGCCAGCGCGACCACCC
s9      AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGCGACCACCC
```

```
s10     AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
s11     AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
s12     AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACTC
s14     AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
s15     AAGCGCCAGCGTGACCACCC
        AAGCGCCAGCGTGACCACCC
```

The parameters that control the simulation process are the same as for Seq-Gen (reviewed in Chapter 2).

For example, to run NetSeqGen on a network in file *infile* to simulate 3 sets of sequences 50 nucleotide long using the HKY model and a TS/TV ratio of 3.0, and store the output in file *outfile*, the command is:

```
netseqgen -mHKY -t3.0 -l40 -n3 -q < infile > outfile
```

Chapter 6

Quality Measures for Phylogenetic Networks

Due to the pervasiveness of phylogenies, many different methods have been proposed for their reconstruction. Such methods can be assessed in terms of the quality of the reconstructions they provide (as well as, of course, in terms of the speed at which they provide such reconstructions). However, such an assessment requires knowledge of the true phylogeny, something that is normally lacking (except in simulations); thus surrogate criteria have been devised, such as the maximum likelihood criterion and the maximum parsimony criterion.

Maximum parsimony has been studied and used extensively for phylogenetic trees [169]. It is based on a minimum-information principle (similar to Occam's razor): in absence of information to the contrary, the best explanation for the observed data is that involving the smallest number of manipulations. In the case of evolutionary histories, that phylogeny involving the fewest evolutionary events is sought. As pointed out by Hein [70, 71], maximum parsimony can be extended to phylogenetic networks: he observed that each individual site in a set of sequences labeling a network evolves down a tree contained in the network (i.e., a tree whose edges are edges of the network). In consequence, the obvious extension is to define the parsimony score of a network as the sum, over all sites, of the parsimony score of the best possible tree contained within the network for each site.

A member of our group [185] pointed out that this generalization suffers from a major flaw: adding reticulation events (in the form of additional edges) to the network can only lower the parsimony score, since introducing a new edge cannot increase the score of any site, but may help lower the score of some. In consequence, this criterion leads to a gross overestimation of the number of reticulation events needed to explain the data. Many ways can be devised to remedy this problem; in this paper, we use perhaps the simplest such: we fix the number of reticulation events and find the best network with that number of events.

Whereas computing the parsimony score of a given phylogenetic tree can be done in time linear in the number of tree nodes using Fitch's algorithm [52, 68], computing the parsimony score of phylogenetic networks is a hard problem (possibly NP-hard). However, we show that the problem is fixed-parameter tractable (FPT) [38]. Indeed, we show that the problem is FPT for two unrelated parameters when restricted to a particular class of networks, which we call *k-gt-networks*, an extension of the gt-networks proposed by Wang *et al.* [186] and studied by Gusfield *et al.* [63]

The standard methodology for assessing the performance of phylogenetic reconstruction methods uses simulation studies (the details of such studies are described in Chapter 2). While the same methodology is directly applicable to phylogenetic networks, it has not yet been applied, except in our own work [102, 127, 128].

The topology of a phylogenetic network can be characterized directly (using its edges) or in terms of its constituent trees (trees with the same set of leaves and a subset of the edges of the network). In the first category falls a measure we define [102, 127]. We extended the RF measure to networks by observing that each edge of a phylogenetic network induces the following tripartition of the leaves: those that are reachable from the root only via that edge, those that are reachable from the root via that edge and also via a path that does not use that edge, and those that are not reachable from the root via that edge. We showed that our measure induced a metric on the space of all phylogenetic networks (on the same set of leaves) [102], and also that it exhibited desirable properties in experimental studies [127, 185]. When we view the network as the collection of its induced trees, we can define measures of topological accuracy based on trees. We defined such a measure in earlier work [127]; we review and characterize it here. We also review and characterize another tree-based measure, which can be viewed as an alternate extension of the RF measure, but also as one based on the notion of *splits*: simply consider all bipartitions of the leaves of the network induced by the edges of its constituent trees. Such a measure has already been used [21, 128], but we provide new characterizations and relate it to the other two measures. We also show that all three measures form metrics on the space of k-gt-networks.

FITCH'S ALGORITHM

Input: A tree T leaf-labeled by a set S of sequences of length k over the alphabet Σ.

Output: TScore(N, S).

Bottom-up stage: For node v, site i, and $\sigma \in \Sigma$,

$$\text{score}_i(v, \sigma) = \min_{\alpha,\beta \in \Sigma}\{\text{score}_i(u, \alpha) + d(\sigma, \alpha) + \text{score}_i(x, \beta) + d(\sigma, \beta)\},$$

where u and x are the two children of v.
Top-down stage: For node v and site i, label v_i by σ, where $\text{score}_i(v, \sigma) = \min_{\alpha \in \Sigma}\{\text{score}_i(v, \alpha)\}$.
Computing the score:

$$\text{TScore}(T, S) = \sum_{1 \le i \le k} \min_{\sigma \in \Sigma}\{\text{score}_i(v, \sigma)\},$$

where v is the root of T.

Figure 6.1: Fitch's algorithm for computing the parsimony score of a tree.

6.1 A Parsimony Criterion for Phylogenetic Networks

6.1.1 Parsimony on Trees and Extensions to Networks

Recall the maximum parsimony problem defined in Chapter 2. Unfortunately, finding a most parsimonious tree for a set of sequences (the MP problem) is NP-hard, even when the sequences are binary [29, 53]. Fortunately, computing the parsimony score of a fixed leaf-labeled tree is solvable in linear time using Fitch's algorithm [52, 68], outlined in Figure 6.1.

We write $\text{score}_i(x, \sigma)$ to denote the parsimony score of site i on the subtree rooted at x when site i at node x is labeled with σ. For every two alphabet symbols α and β in Σ, we have $d(\alpha, \beta) = 0$ if $\alpha = \beta$, and $d(\alpha, \beta) = 1$ if $\alpha \neq \beta$. We write $\text{TScore}(T, i)$ to denote the score of site i on tree T, as computed by Fitch's algorithm; in other words, we set $\text{TScore}(T, i) = \min_{\sigma \in \Sigma}\{\text{score}_i(v, \sigma)\}$, where v is the root of T.

Using the same notation, Hein's natural extension of parsimony scores to networks can be formalized as follows.

Definition 6.1.1. The parsimony score of a network N leaf-labeled by a set S of sequences, each of length k, is

$$\text{NScore}(N, S) \ := \ \textstyle\sum_{1 \leq i \leq k} \text{NScore}(N, i)$$

where $\text{NScore}(N, i) = \min_{T \in \mathcal{T}(N)} \text{TScore}(T, i)$.

6.1.2 Fixed-parameter Network Parsimony

Definition 6.1.1 gives a natural optimization criterion for constructing networks: given a set S of sequences, construct a phylogenetic network N leaf-labeled by S, such that $\text{NScore}(N, S)$ is minimized. However, as mentioned earlier, this generalization has a major flaw, namely grossly overestimating the number of reticulation events (in the form of additional edges). This is due to the fact that adding additional edges to the network either lowers its parsimony score or keeps it unchanged (never increases it). This flaw can be remedied in many ways. Tholse [185] suggested minimizing the *average* parsimony score over all induced trees in the network—clearly an NP-hard problem, since constructing a single most parsimonious tree is already NP-hard [29, 53]. We consider here the problem of constructing networks with a *fixed* number of reticulation events. Naturally, this approach requires an accurate estimate of the number of reticulation events—a problem we do not address here, but that has been tackled by several researchers [81, 123, 174].

Definition 6.1.2. FIXED-PARAMETER MAXIMUM-PARSIMONY (FPMP) FOR NETWORKS

- *Input:* set S of aligned biomolecular sequences and non-negative integer m.
- *Output:* phylogenetic network N leaf-labeled by S, with m reticulation events, that minimizes NScore(N, S).

The FPMP problem is NP-hard: for $m = 0$, it is the classical MP problem for trees.

Standard heuristics for MP all require a fast *point estimation* routine—compute the score of the current solution to compare it with the best one found so far. What of the parsimony score of a phylogenetic network?

Definition 6.1.3. PARSIMONY SCORE OF PHYLOGENETIC NETWORKS (PSPN):

- *Input:* set S of aligned biomolecular sequences and phylogenetic network N leaf-labeled by S.
- *Output:* NScore(N, S).

This problem appears to be quite hard in general—we conjecture that it is NP-hard. However, it is *fixed-parameter tractable* [38], as we now show.

Lemma 1. *The PSPN problem is fixed-parameter tractable, where the parameter is the number of reticulation nodes.*

Proof. Let N be a phylogenetic network leaf-labeled by a set S of n sequences of length n over an alphabet Σ and let m be the number of reticulation nodes in N. The number of trees induced by N is $O(2^m)$. Computing the parsimony score of a site i on a tree T with n leaves takes $O(|\Sigma|n)$ time using Fitch's algorithm. Therefore, computing the parsimony score of N on the set S of sequences takes $O(2^m|\Sigma|nk)$ time. □

We now show that the PSPN problem is solvable in $O(|\Sigma|nl)$ time for gt-networks, for sequences of length l and any number m of reticulation events; for k-gt-networks, the running time is simply $O(2^k|\Sigma|nl)$, unchanged in asymptotic terms for any fixed k. Algorithm **ALG1** (which proceeds in two stages: a bottom-up stage, shown in Figure 6.2, and a top-down stage, shown in Figure 6.3) is an extension of Fitch's algorithm and computes the parsimony score of a gt-network in linear-time.

(In the interest of clarity, we show the algorithm for 1-gt-networks only; it extends to k-gt-networks in the obvious way.) We write NScore(N, i) to denote the score of site i on network N, as computed by algorithm **ALG1**. That is, we have

$$\text{NScore}(N, i) = \min_{\sigma \in \Sigma}\{\text{score}_i(v, \sigma)\}$$

ALG1

Input: A gt-network N leaf-labeled by a set S of sequences of length l over the alphabet Σ.

Output: NScore(N, S).

Bottom-up stage: For node v, site i, and $\sigma \in \Sigma$, compute $\text{score}_i(v, \sigma)$ as follows:

- For a node v that is not on any gall in N:
  - If v is a leaf: $\text{score}_i(v, \sigma) = d(\sigma, v_i)$.
  - If v is an internal node:

    $$\text{score}_i(v, \sigma) = \min_{\alpha,\beta \in \Sigma} \{\text{score}_i(u, \alpha) + d(\sigma, \alpha) + \text{score}_i(y, \beta) + d(\sigma, \beta)\},$$

    where u and y are the two children of v.
- For a node v that is on some gall Q_x^w, where $RE(Q) = \{e_1, e_2\}$:
  - if $v = x$: $\text{score}_i(v, \sigma) = \min_{\alpha \in \Sigma}\{\text{score}_i(u, \alpha) + d(\sigma, \alpha)\}$, where u is the child of v.
  - if $v = w$:

    $$\text{score}_i(v, \sigma) = \min_{\alpha,\beta \in \Sigma, t \in \{1,2\}} \{\text{score}_i^t(u, \alpha) + d(\sigma, \alpha) + \text{score}_i^t(y, \beta) + d(\sigma, \beta)\},$$

    where u and y are children of w. Further, for a gall Q, maintain in $t_\sigma(Q)$ the value of t that yielded $\text{score}_i(v, \sigma)$.
  - if $v \neq x$ and $v \neq w$, maintain two scores, $\text{score}_i^t(v, \sigma), t \in \{1, 2\}$, that correspond to the cases when e_1 is removed and e_2 is removed, respectively.

    $$\text{score}_i^t(v, \sigma) = \min_{\alpha,\beta \in \Sigma} \{\text{score}_i^t(u, \alpha) + d(\sigma, \alpha) + \text{score}_i(y, \beta) + d(\sigma, \beta)\},$$

    where u and y are v's children, u is a gall node, and $t \in \{1, 2\}$.

Figure 6.2: Algorithm for computing the parsimony score of a gt-network: the bottom-up stage.

Top-down stage: For node v and site i, label v_i as follows:

- if v is a coalescent or reticulation node, or is not a gall node: $v_i = \sigma$ where

$$\text{score}_i(v, \sigma) = \min_{\alpha \in \Sigma}\{\text{score}_i(v, \alpha)\}.$$

- if v is a gall node, for some gall Q: $v_i = \sigma$ where

$$\text{score}_i^{t_\sigma(Q)}(v, \sigma) = \min_{\alpha \in \Sigma}\{\text{score}_i^{t_\alpha(Q)}(v, \alpha)\}.$$

Computing the score:

$$\text{NScore}(N, S) = \sum_{1 \le i \le l} \min_{\sigma \in \Sigma}\{\text{score}_i(v, \sigma)\},$$

where v is the root of N.

Figure 6.3: Algorithm for computing the parsimony score of a gt-network: the top-down stage.

where v is the root of N.

Theorem 1. *Given a gt-network N leaf-labeled by set S of sequences, Algorithm* **ALG1** *computes NScore(N, S).*

Proof. We prove that, upon termination of **ALG1**, we have, for every site i, $1 \le i \le l$,

$$\text{NScore}(N, i) = min_{T \in \mathcal{T}(N)}\{\text{TScore}(T, i)\}.$$

We prove this by induction on m—the number of galls in N. For the base case, $m = 0$, **ALG1** reduces to Fitch's algorithm and the theorem holds. Assume then that the theorem holds for gt-networks with m galls and let N be a gt-network with $m + 1$ galls. Let Q_x^w be a gall in N, with $RE(Q) = \{e_1, e_2\}$.

Breaking Q^w_x, by removing e_1 or e_2, produces gt-networks N_1, respectively N_2, each with m galls. By inductive hypothesis, we thus have, for each site i, $1 \leq i \leq l$,

$$\text{NScore}(N_1, i) = \min_{T \in \mathcal{T}(N_1)} \{\text{TScore}(T, i)\},$$

and

$$\text{NScore}(N_2, i) = \min_{T \in \mathcal{T}(N_2)} \{\text{TScore}(T, i)\}.$$

Let $T_{opt} \in \mathcal{T}(N)$ be the tree obeying

$$\text{TScore}(T_{opt}, i) = \min_{T \in \mathcal{T}(N)} \{\text{TScore}(T, i)\}.$$

Tree T_{opt} is induced by either N_1 or N_2; without loss of generality, assume $T_{opt} \in \mathcal{T}(N_1)$. Thus we need only prove $\text{NScore}(N, i) = \text{NScore}(N_1, i)$. In N_1, we have

$$\text{score}_i(w, \sigma) = \min_{\alpha, \beta \in \Sigma} \{\text{score}_i(u, \alpha) + d(\sigma, \alpha) + \text{score}_i(y, \beta) + d(\sigma, \beta)\}$$

where u and y are the two children of w; in N, we have

$$\text{score}_i(w, \sigma) = \min_{\alpha, \beta \in \Sigma, t \in \{1,2\}} \{\text{score}^t_i(u, \alpha) + d(\sigma, \alpha) + \text{score}^t_i(y, \beta) + d(\sigma, \beta)\}$$

Since T_{opt} is induced by N_1, $\text{score}_i(w, \sigma)$ in N is equal to $\text{score}_i(w, \sigma)$ in N_1.

In the bottom-up stage of **ALG1**, the algorithm runs identically on nodes above w in both N and N_1. In the top-down stage, the algorithm labels site i of node w correctly, since it stores the values $t_\sigma(Q)$, which indicates which of the two edges in $RE(Q)$ was removed to obtain the minimum score. $\square$

Algorithm **ALG1** computes the parsimony score of a gt-network (or k-gt-network, with the obvious extension) N on n leaves in $O(|\Sigma|nl)$, where l is the length of the sequences and Σ is the alphabet: it does the same amount of work as Fitch's algorithm on nodes that are not part of a gall and twice that amount on nodes that are part of a gall.

6.2 Topological Accuracy of Networks

Definition 6.2.1. We say two networks $N_1 = (V_1, E_1)$ and $N_2 = (V_2, E_2)$, leaf-labeled by the set L of taxa, are *isomorphic* (denoted $N_1 = N_2$) if there exists a bijection $f : V_1 \rightarrow V_2$ such that

1. $(u, v) \in E_1$ if and only if $(f(u), f(v)) \in E_2$, and
2. if leaf $v_1 \in V_1$ is labeled by l, for some $l \in L$, then $f(v_1)$ is also labeled by l.

We want a measure $m(N_1, N_2)$ of the distance between two networks N_1 and N_2 on the same set of taxa, such that m is symmetric and nonnegative and satisfies the following three conditions:

C1 If N_1 and N_2 are two trees, then we have $m(N_1, N_2) = RF(N_1, N_2)$, where $RF(N_1, N_2)$ denotes the Robinson-Foulds distance between trees.

C2 If N_1 and N_2 are isomorphic (with respect to the leaf labels), then we have $m(N_1, N_2) = 0$.

C3 If we have $m(N_1, N_2) = 0$, then N_1 and N_2 are isomorphic.

A phylogenetic network can be characterized in various ways, each of which gives rise to an associated measure.

6.2.1 Networks and Tripartitions

We begin with our tripartition measure [102], which extends the Robinson-Foulds measure by considering the tripartitions that edges naturally induce in networks (instead of the bipartitions that they induce in trees). Let N be a phylogenetic network, leaf-labeled by set S, and let $e = (u, v)$ be an edge of N. Edge e induces a tripartition of S, defined by the sets

- $A(e) = \{s \in S \mid s \text{ is reachable from the root of } N \text{ only via } v\}$.
- $B(e) = \{s \in S \mid s \text{ is reachable from the root of } N \text{ via at least one path passing through } v \text{ and one path not passing through } v\}$.
- $C(e) = \{s \in S \mid s \text{ is not reachable from the root of } N \text{ via } v\}$.

We denote by $\psi(e)$ the tripartition of S induced by edge e, and write $\psi(e) = \langle A(e), B(e), C(e)\rangle$. Two tripartitions $\psi(e_1)$ and $\psi(e_2)$ are equivalent, denoted by $\psi(e_1) \equiv \psi(e_2)$, whenever we have $A(e_1) = A(e_2)$, $B(e_1) = B(e_2)$, and $C(e_1) = C(e_2)$. We denote by $\Psi(N)$ the set of all tripartitions induced by the edges of N. Table 6.1 shows $\Psi(N_1)$ and $\Psi(N_2)$ for the two networks N_1 and N_2 in Figure 6.4.

Let x be a network node with parents x_1 and x_2. The *reticulation scenario* of x, denoted $RS(x)$, is the set $\{X_1, X_2\}$, where X_i is the set of leaves under x_i (that is, if x_i is the head of edge e, then $X_i = A(e) \cup B(e)$). Intuitively, a reticulation scenario of a network node x denotes the two groups of taxa whose common ancestors were involved in the reticulation event. Let $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two edges; we say that e_1 and e_2 are *compatible*, denoted by $e_1 \equiv e_2$, if and only if

- both are tree edges and $\psi(e_1) \equiv \psi(e_2)$; or
- both are network edges and $\psi(e_1) \equiv \psi(e_2)$ and $RS(v_1) = RS(v_2)$.

We can now define the *false negative rate* (FN) and *false positive rate* (FP) between two networks N_1 and N_2 in the usual way as follows:

- $FN(N_1, N_2) = \dfrac{|\{e_1 \in E(N_1) \mid \nexists e_2 \in E(N_2),\, e_1 \equiv e_2\}|}{|E(N_1)|}$.
- $FP(N_1, N_2) = \dfrac{|\{e_2 \in E(N_2) \mid \nexists e_1 \in E(N_1),\, e_1 \equiv e_2\}|}{|E(N_2)|}$.

Definition 6.2.2. The error rate between N_1 and N_2, denoted $m^{tri}(N_1, N_2)$, is $(FN(N_1, N_2) + FP(N_1, N_2))/2$.

This error measure can be computed in time polynomial in the size of the two networks. We implemented an early version of this measure and reported on experimental results [127].

6.2.2 Networks and Trees

As discussed earlier, a phylogenetic network induces a set of trees. Each site in the homologs of the parents of a hybrid evolves in a tree-like fashion on one of the trees induced by the network representing the hybridization event. We can then characterize the similarity between the sets of trees induced by two networks. Let N_1 and N_2 be two networks and let $\mathcal{T}(N_1)$ and $\mathcal{T}(N_2)$ be the corresponding sets of induced trees. Define the complete bipartite graph $G^{N_1,N_2} = (\mathcal{T}(N_1) \cup \mathcal{T}(N_2), E)$, which has an edge $e = (u, v)$ between every $u \in \mathcal{T}(N_1)$ and $v \in \mathcal{T}(N_2)$, and assign to each edge $e = (u, v)$ a weight $w(e)$ equal to the RF distance between the two trees corresponding to nodes u and v. Recall that the minimum-weight edge cover of a graph $G = (V, E)$ is a subset $E' \subseteq E$ such that E' covers V (i.e., every node $v \in V$ is the endpoint of an edge $e \in E'$) and the sum of edge weights, $\sum_{w(e)\in E'} w(e)$, is minimum.

Definition 6.2.3. The error rate between N_1 and N_2, denoted $m^{tree}(N_1, N_2)$, is the weight of the minimum-weight edge cover of G^{N_1,N_2}.

Although the minimum-weight edge-cover problem is solvable in polynomial time, the size of the bipartite graph is exponential (contains $2^{m_1} + 2^{m_2}$ nodes and $2^{m_1+m_2}$ edges, where m_1 and m_2 are the numbers of reticulation nodes in N_1 and N_2, respectively) in the number of reticulation nodes, so that computing this measure may require exponential time to compute.

6.2.3 Networks and Splits

Splits are a tree-based concept—in the case of trees, splits and bipartitions are equivalent. We define the set of splits of a network N, denoted by $C(N)$, to be the union of the sets of splits of its induced trees:

$$C(N) = \bigcup_{T \in \mathcal{T}(N)} C(T)$$

Let N_1 and N_2 be two phylogenetic networks. We define the *false negative rate* (FN) and *false positive rate* (FP) between two networks N_1 and N_2 as follows.

- $FN(N_1, N_2) = \dfrac{|C(N_1) - C(N_2)|}{|C(N_1)|}$.
- $FP(N_1, N_2) = \dfrac{|C(N_2) - C(N_1)|}{|C(N_2)|}$.

Definition 6.2.4. The error rate between N_1 and N_2, denoted $m^{sp}(N_1, N_2)$, is $(FN(N_1, N_2) + FP(N_1, N_2))/2$.

While the three measures are equivalent on a tree, they generally differ on networks.

6.2.4 Illustrating the Three Measures

In this section, we illustrate the three measures of topological error on the two networks N_1 and N_2 in Figure 6.4. Table 6.1 lists the non-trivial tripartitions induced by the edges of the two networks. Based on their induced tripartitions, none of the edges of N_1 are compatible with the edges of N_2, and vice versa. Hence, $m^{tri}(N_1, N_2) = 1$, i.e., the tripartition-based error rate

Table 6.1: The tripartitions induced by the edges of networks N_1 and N_2 of Figure 6.4.

Network N_1		Network N_2	
Edge	Tripartition	Edge	Tripartition
1	$\langle\{A,B,C\},\ \emptyset,\ \{D,E\}\rangle$	1	$\langle\{A,B\},\ \{C(1)\},\ \{D,E\}\rangle$
2	$\langle\{D,E\},\ \emptyset,\ \{A,B,C\}\rangle$	2	$\langle\{D,E\},\ \{C(1)\},\ \{A,B\}\rangle$
3	$\langle\{A\},\ \{B(1)\},\ \{C,D,E\}\rangle$	3	$\langle\{B\},\ \{C(1)\},\ \{A,D,E\}\rangle$
4	$\langle\{C\},\ \{B(1)\},\ \{A,D,E\}\rangle$	4	$\langle\{D\},\ \{C(1)\},\ \{A,B,E\}\rangle$
5	$\langle\emptyset,\ \{B(1)\},\ \{A,C,D,E\}\rangle$	5	$\langle\emptyset,\ \{C(1)\},\ \{A,B,D,E\}\rangle$
6	$\langle\emptyset,\ \{B(1)\},\ \{A,C,D,E\}\rangle$	6	$\langle\emptyset,\ \{C(1)\},\ \{A,B,D,E\}\rangle$

between the two networks is 100%. Network N_1 induces the two trees T_1^1 and T_2^1, shown in Figure 6.5, and network N_2 induces the two trees T_1^2 and T_2^2 shown in Figure 6.7. We have the following error rates among the trees:

- $RF(T_1^1, T_1^2) = 1/3$.
- $RF(T_1^1, T_2^2) = 2/3$.
- $RF(T_2^1, T_1^2) = 0$.

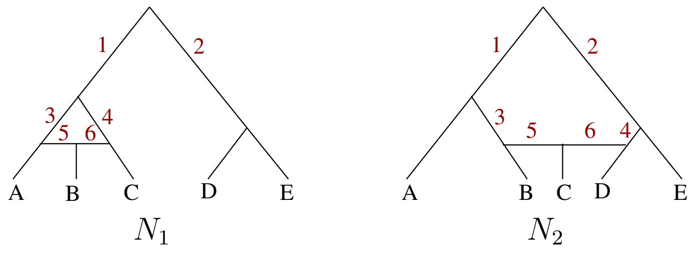


Figure 6.4: Two networks N_1 and N_2 used to illustrate the error rates computed by the three measures.

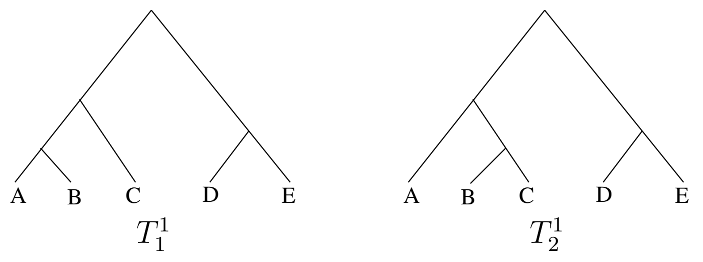


Figure 6.5: The two trees T_1^1 and T_2^1 induced by the network N_1 of Figure 6.4.

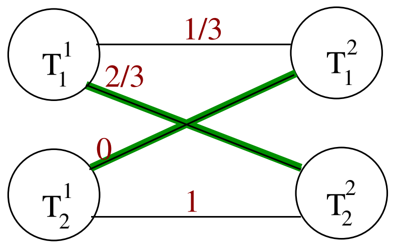


Figure 6.6: The weighted complete bipartite graph G^{N_1,N_2} that is constructed to compute $m^{tree}(N_1, N_2)$ for the two networks of Figure 6.4. The thick edges correspond to the minimum-weight edge-cover, and the sum of their weights is 2/3.

- $RF(T_2^1, T_2^2) = 1$.

Figure 6.6 shows the weighted complete bipartite graph G^{N_1,N_2}. The minimum-weight edge-cover of G^{N_1,N_2} equals 2/3, and hence $m^{tree}(N_1, N_2) = 2/3$. Table 6.2 lists all the non-trivial splits of the two networks N_1 and N_2 of Figure 6.4. Based on their induced splits, we have $FN(N_1, N_2) = 0$ and $FP(N_1, N_2) = 1/5$. Hence, $m^{sp}(N_1, N_2) = 1/10$. The two networks N_1 and N_2 in Figure 6.4 are different; indeed, the three error measures return non-zero values. Fur-

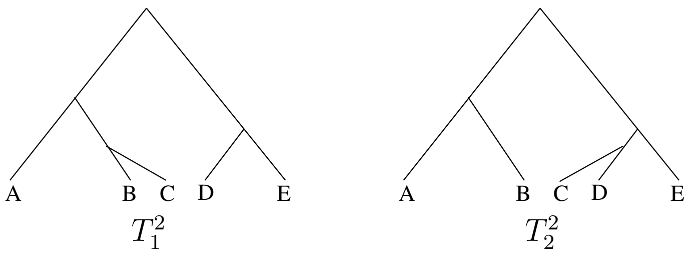


Figure 6.7: The two trees T_1^2 and T_2^2 induced by the network N_2 of Figure 6.4.

Table 6.2: The splits induced by the networks N_1 and N_2 of Figure 6.4.

Network N_1		Network N_2	
Split Number	Split	Split Number	Split
1	$\langle\{A,B,C\},\ \{D,E\}\rangle$	1	$\langle\{A,B\},\ \{C,D,E\}\rangle$
2	$\langle\{A,B\},\ \{C,D,E\}\rangle$	2	$\langle\{A,B,C\},\ \{D,E\}\rangle$
3	$\langle\{B,C\},\ \{A,D,E\}\rangle$	3	$\langle\{C,D\},\ \{A,B,E\}\rangle$
4	$\langle\{D,E\},\ \{A,B,C\}\rangle$	4	$\langle\{D,E\},\ \{A,B,C\}\rangle$
		5	$\langle\{B,C\},\ \{A,D,E\}\rangle$

thermore, by looking at the two networks, it is clear that no pair of (internal) edges from the two networks depict the same speciation or hybridization event. However, m^{tri} is the only measure that reflects that fact; the m^{sp} measure returns an error rate close to 0, i.e., that measure implies the two networks are very similar, which is misleading. The value computed by the m^{tree} measure falls in the middle. This observation is not surprising, given that m^{tri} is the only "network-intrinsic" measure among the three.

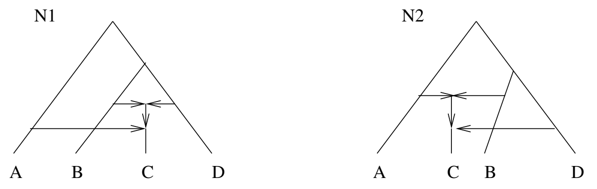


Figure 6.8: Two phylogenetic networks N_1 and N_2. We have $m^{tri}(N_1, N_2) \neq 0$, $m^{tree}(N_1, N_2) = 0$, and $m^{sp}(N_1, N_2) = 0$ although $N_1 \neq N_2$.

6.3 Tripartitions and Their Induced Metric

It is straightforward to establish that the three measures satisfy conditions C1 and C2 for any pair of phylogenetic networks. We now prove that the measure $m^{tri}(\cdot, \cdot)$ also satisfies condition C3 for all networks leaf-labelled by the same set of taxa. However, the two measures $m^{tree}(\cdot, \cdot)$ and $m^{sp}(\cdot, \cdot)$ may not satisfy C3 for general phylogenetic networks, as illustrated in Figure 6.8.

Definition 6.3.1. We say N is a **class-I** phylogenetic network if for every network node u in N has at least one sibling v that is a tree node. In other words, if u_1 and u_2 are the two parents of u, then there exists a tree node v such that either $(u_1, v) \in E(N)$ or $(u_2, v) \in E(N)$.

Theorem 2. *(From [102]) The pair $(\mathcal{N}, m^{tri})$, where $\mathcal{N}$ is the space of all class-I phylogenetic networks on n leaves $\{s_1, \ldots, s_n\}$, is a metric space.*

Before proving that theorem, however, we need a series of theorems, some of independent interest. Since the tripartition induced by an edge is defined by the head of that edge, we have the following observation:

Observation 1. *Let $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two edges of a phylogenetic network N. If $v_1 = v_2$ then $\theta(e_1) \equiv \theta(e_2)$.*

In addition to pairs of network edges that share the same network node, we can classify sets of distinct nodes that define compatible edges.

Definition 6.3.2. A subset $U \subseteq V$ in a network $N = (V, E)$ is **convergent** if and only if U satisfies the following three properties:

1. $|U| \geq 2$,
2. for every two nodes $u, v \in U$, we have neither $u \rightsquigarrow v$ nor $v \rightsquigarrow u$, and
3. for every leaf s reachable from some node in U, there exists a natural number k such that, for every node $u \in U$, there exists a directed path p from u to s containing exactly k network nodes.

Based on the definitions of tripartitions and convergent sets, we have the following lemma.

Lemma 2. *Let U be a convergent set of nodes in a network N. Then, for every pair $\langle e_1, e_2 \rangle$ of edges in the set $\mathcal{E} = \{(v, u) : u \in U\}$, we have $\theta(e_1) \equiv \theta(e_2)$.*

Proof. Let U be a convergent set such that $e_1 = (u, 1, v_1)$ and $e_2 = (u_2, v_2)$ are two edges with $v_1, v_2 \in U$. By condition (3) of Definition 6.3.2, we have that $A(e_1) \cup B(e_1) \equiv A(e_2) \cup B(e_2)$ and $C(e_1) \equiv C(e_2)$. We now show that $A(e_1) \equiv A(e_2) \equiv \emptyset$. Suppose $s \in A(e_1)$; then, $v_2 \not\rightsquigarrow s$ (otherwise, and since

v_1 and v_2 are mutually unreachable, we have $s \in B(e_1)$—a contradiction). This contradicts condition (3) of Definition 6.3.2. Hence, $A(e_1) \equiv \emptyset$, and similarly we establish that $A(e_2) \equiv \emptyset$. Therefore, we have $A(e_1) \equiv A(e_2)$ and $B(e_1) \equiv B(e_2)$. □

We now prove a theorem that classifies all equivalent edges in a network.

Theorem 3. *Let N be a phylogenetic network, and let $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two edges of N. We have $\theta(e_1) \equiv \theta(e_2)$ if and only if either $v_1 = v_2$ or $\{v_1, v_2\}$ is a convergent set.*

Proof. The "if" part follows immediately from Observation 1 and Lemma 2. For the "only if" part, assume $\theta(e_1) \equiv \theta(e_2)$. We have neither $v_1 \rightsquigarrow v_2$ nor $v_2 \rightsquigarrow v_1$ (otherwise, we have a node of indegree and outdegree 1 in the network—which is not allowed). Therefore, either $v_1 = v_2$ (in which case the result is established), or v_1 and v_2 are mutually unreachable. We now prove that in the latter case $\{v_1, v_2\}$ necessarily constitutes a convergent set.

Let s be a leaf of N that is reachable from v_1. Further, assume $s \in A(e_1)$. Since $\theta(e_1) \equiv \theta(e_2)$, it follows that $s \in A(e_2)$. However, this implies that s is reachable from the root via a path that passes through v_1 and via a path that passes through v_2 (which does not pass through v_1, since v_1 and v_2 are mutually unreachable). This contradicts the assumption that $s \in A(e_1)$. Hence, $A(e_1) = \emptyset$, and any leaf s reachable from v_1 is an element of $B(e_1)$.

Since $s \in B(e_1)$, and since $B(e_1) = B(e_2)$, it follows that there are two paths: p_1 with k ($k > 0$) network nodes from v_1 to s, and p_2 with k

($k > 0$) network nodes from v_2 to s. Hence, we have that $\{v_1, v_2\}$ satisfies the conditions of Definition 6.3.2. This completes the proof. □

Lemma 3. *Let $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two edges in a class-I network N. Then, $e_1 \equiv e_2$ if and only if $v_1 = v_2$.*

Proof. The "if" part follows directly from Observation 1. Assume $e_1 \equiv e_2$. From Lemma 2, it follows that either $v_1 = v_2$ or $\{v_1, v_2\}$ is a convergent set. In the former case, the result is established. Assume that $\{v_1, v_2\}$ is a convergent set. Assume that parent x_1 of v_1 has a tree node child y_1, and that parent x_2 of v_2 has a tree node child y_2 (the existence of those nodes is guaranteed, since N is a class-I network). Further, let e'_1 and e'_2 be the two edges incident into x_1 and x_2, respectively. Clearly, $A(e'_1) \cup B(e'_1) \neq A(e'_2) \cup B(e'_2)$ (otherwise, all paths from x_1 and x_2 to leaves of N would merge, violating the properties of class-I networks). Hence, $RS(v_1) \neq RS(v_2)$—a contradiction. Hence, $v_1 = v_2$. □

Using Theorem 3, we can now prove that $m^{tri}(N_1, N_2)$ satisfies condition C3 for class-I networks.

Theorem 4. *Let N_1 and N_2 be two class-I phylogenetic networks. If we have $m^{tri}(N_1, N_2) = 0$, then N_1 is isomorphic to N_2.*

Proof. We prove that $N_1 = N_2$ by construction. We show a bijection $f : V_1 \to V_2$ that satisfies both conditions in Definition 6.2.1. From the definition of

$m^{tri}(\cdot,\cdot)$, we have

$$\forall e_1 \in E(N_1),\ \exists e_2 \in E(N_2),\ e_1 \equiv e_2 \tag{6.1}$$

$$\forall e_2 \in E(N_2),\ \exists e_1 \in E(N_1),\ e_1 \equiv e_2. \tag{6.2}$$

Let v_1 be a node in N_1. We define f (top-down) as follows.

1. If v_1 is the root of N_1, then $f(v_1)$ is the root of N_2.
2. Otherwise, let $e_1 = (u_1, v_1) \in E_1$. We have $f(v_1) = v_2$ where $e_2 = (u_2, v_2) \in E_2$ and $e_1 \equiv e_2$.

We now prove that f is a bijection that satisfies both conditions of Definition 6.2.1.

I. *f is well-define.* Clearly, $f(v_1)$ is defined and unique when v_1 is the root. We now show that $f(v_1)$ is defined and unique when $indegree(v_1) > 0$. Assume that $v_2', v_2'' \in V_2$ such that $f(v_1) = v_2' = v_2''$. Then, by definition of f, we have three edges $e_1 = (u_1, v_1) \in E_1$, $e_2' = (u_2', v_2') \in E_2$ and $e_2'' = (u_2'', v_2'') \in E_2$ such that $e_1 \equiv e_2' \equiv e_2''$. By Lemma 2, and since $v_2' \neq v_2''$, we have that $\{v_2, v_2''\}$ is a convergent set.

II. *f is surjective.* Assume that there exists a node $u_2 \in V(N_2)$ that is not the image of any node of N_1 under f. Node u_2 cannot be the root or a leaf, since that would contradict the fact the networks are rooted and are labeled by the same set S of taxa. Thus there exists an edge $e = (w_2, u_2) \in E(N_2)$ to which no edge in N_1 is mapped. In other words, there does not exist an edge $e' \in E(N_1)$ with $e \equiv e'$, which contradicts our assumption of $m(N_1, N_2) = 0$.

III. *f is injective.* From Theorem 3, we know that two different edges $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ are equivalent if and only if we have $v_1 = v_2$. Hence, $f(v_1) = f(v_2)$ if and only if $v_1 = v_2$.

IV. *$(u, v) \in E(N_1)$ if and only if $(f(u), f(v)) \in E(N_2)$.* We first prove the "only if" part. Let $(u, v) \in E(N_1)$ and assume $(f(u), f(v)) \notin E(N_2)$. Assume $f(u) = u_1$ and $f(v) = v_1$. Either u_1 and v_1 are mutually unreachable, $u_1 \rightsquigarrow v_1$, or $v_1 \rightsquigarrow u_1$. Let e' be an edge incident into u and e'' be an edge incident into u_1. From Definition 6.3.1, Lemma 3, and Theorem 3, it follows that $e' \not\equiv e''$—a contradiction. Hence, $(f(u), f(v)) \in E(N_2)$. We prove the "if" part similarly, and this completes the proof. □

We are finally in a position to prove Theorem 2.

Proof. (Of Theorem 2) We prove the following:

1. $m^{tri}(N_1, N_2) \geq 0$ and $m^{tri}(N_1, N_2) = 0$ iff $N_1 = N_2$.

2. $m^{tri}(N_1, N_2) = m^{tri}(N_2, N_1)$.

3. $m^{tri}(N_1, N_3) \leq m^{tri}(N_1, N_2) + m^{tri}(N_2, N_3)$.

Property (1) follows immediately from the definition of m^{tri} and from Theorem 4. Property (2) follows from conditions C2 and C3, both of which are satisfied by m^{tri}, and Theorem 4. Measure $m^{tri}(\cdot, \cdot)$ is based on the symmetric difference of edge sets, suitably normalized; since the symmetric difference of sets satisfies the triangle inequality, so does m^{tri}, satisfying Property (3). □

6.4 k-gt-Networks and the Three Measures: Metric Spaces

We already know that m^{tri} is a metric on the space of all class-I phylogenetic networks leaf-labelled by a set S of taxa [102], and hence it is a metric on the space of all k-gt-networks leaf-labelled by the same set of taxa. It is straightforward to show that both m^{sp} and m^{tree} satisfy conditions C1 and C2. We now show that both measures satisfy condition C3 as well.

Lemma 4. *Let N_1 and N_2 be two gt-networks leaf-labelled by a set S of taxa. Then,*

$$m^{sp}(N_1, N_2) = 0 \Rightarrow N_1 = N_2.$$

Proof. Suppose $m^{sp}(N_1, N_2) = 0$. We show $N_1 = N_2$ by construction. We denote by $A^N(v)$ the set of taxa that are reachable from v in the gt-network N. For a node v in N_1, we define $f(v)$ as follows

- if v is not the reticulation node for any gall, then we have $f(v) = u$, with $A^{N_2}(u) = A^{N_1}(v)$, and
- if v is a reticulation node whose child is y, then we have $f(v) = u$, where u is the parent of $f(y)$.

The function f is well-defined for all non-reticulation nodes $v \in V(N_1)$: otherwise, there would exist an edge $e_1 = (u_1, v)$ in N_1 with $\pi_{e_1} \neq \pi_{e_2}$ for every edge $e_2 \in E(N_2)$, contradicting the assumption $C(N_1) = C(N_2)$. For a reticulation node v whose child is y, $f(y)$ cannot be the root of N_2, since that would imply $C(N_1) \neq C(N_2)$.

If we have $f(u) = f(v)$ for two distinct nodes u and v in a gt-network N_1, then u is a reticulation node whose child is v—otherwise, N_1 is not a gt-network. By definition of f, we can write $f(u) = y$, where y is the parent of $f(v)$; hence u and v are mapped to two distinct nodes in N_2. Similarly, every two distinct nodes y and z in N_2 must have two distinct source nodes in N_1. Hence f is injective. If there exists a node $y \in N_2$ with $f(u) \neq y$ for every node $u \in N_1$, then we have $C(N_1) \neq C(N_2)$. If y is the root of N_2, and since y is the only node in N_2 with $A(y) = L(N_2)$, it follows that there cannot exist a node r in N_1 with $A(r) = L(N_2)$; hence the two networks are defined over two different sets of taxa—a contradiction. If y is not the root of N_2, then y is the head of an edge in N_2 that induces a split π equal to $A^{N_2}(y)$, which implies there does not exist an edge in N_1 with the same split; this contradicts the assumption $C(N_1) = C(N_2)$. Hence f is surjective.

Finally, let $e_1 = (u_1, v_1)$ be an edge in N_1 with $f(u_1) = u_2$ and $f(v_1) = v_2$; further assume that (u_2, v_2) is not an edge in N_2. If u_1 is a reticulation node, then v_1 is the only child of u_1. By the definition of f, u_1 is mapped to the parent of $f(v_1)$ and hence $(f(u_1), f(v_1))$ is an edge in N_2. If u_1 is not a reticulation node, then, in addition to v_1, u_1 has another child, say y_1, and we have $A^{N_1}(u_1) = A^{N_1}(v_1) \cup A^{N_1}(y_1)$.

- If u_2 and v_2 are mutually unreachable, then neither $A^{N_2}(u_2)$ nor $A^{N_2}(v_2)$

is equal to either $A^{N_1}(u_1)$ or $A^{N_1}(v_1)$, which contradicts the definition of f.

- If v_2 is reachable from u_2 and we have $A^{N_2}(u_2) \cup A^{N_2}(v_2) = A^{N_1}(u_1) \cup A^{N_1}(v_1)$, then there exists a node z on the path from u_2 to v_2 with

$$A^{N_1}(v_1) \subseteq A^{N_2}(z) \subseteq A^{N_1}(u_1).$$

Then there exists a node z' in N_1 with

$$A^{N_1}(v_1) \subseteq A^{N_1}(z') \subseteq A^{N_1}(u_1),$$

which contradicts the assumption that (u_1, v_1) is an edge in N_1.

- If u_2 is reachable from v_2, then we have $A^{N_2}(u_2) \subseteq A^{N_2}(v_2)$, which in turn implies $A^{N_1}(u_1) \subseteq A^{N_1}(v_1)$. Since v_1 is reachable from u_1 in N_1, we can write $A^{N_1}(v_1) \subseteq A^{N_1}(u_1)$ and thus $A^{N_1}(u_1) = A^{N_1}(v_1)$, which implies that u_1 is a reticulation node whose child is v_1—contradicting the assumption that u_1 is not a reticulation node.

Hence, $(f(u_1), f(v_1))$ is an edge in N_2. We can prove the converse in the same manner. □

Lemma 5. *Let N_1 and N_2 be two gt-networks leaf-labelled by a set S of taxa. Then,*

$$m^{tree}(N_1, N_2) = 0 \Rightarrow N_1 = N_2.$$

Proof. Let N_1 and N_2 be two gt-network with $m^{tree}(N_1, N_2) = 0$. From Definition 6.2.3, we have $\mathcal{T}(N_1) = \mathcal{T}(N_2)$ and thus also $C(N_1) = C(N_2)$. From Lemma 4, we then also have $N_1 = N_2$. □

We now show the m^{sp} is a metric on the space of gt-networks leaf-labelled by a set S of taxa.

Lemma 6. *Let N_1, N_2, and N_3 be three gt-networks leaf-labelled by a set S of taxa. Then:*

Positivity: $m^{sp}(N_1, N_2) \geq 0$.

Symmetry: $m^{sp}(N_1, N_2) = m^{sp}(N_2, N_1)$.

Triangle Inequality: $m^{sp}(N_1, N_3) \leq m^{sp}(N_1, N_2) + m^{sp}(N_2, N_3)$.

Proof. The positivity and symmetry properties follow immediately from the definition of m^{sp}. The measure m^{sp} is based on the symmetric difference of edge sets, suitably normalized; since the symmetric difference of sets satisfies the triangle inequality, so does m^{sp}. □

With Lemmas 4 and 6, we have proved the following result.

Theorem 5. *The space $(\mathcal{N}, m^{sp})$, where $\mathcal{N}$ is the set of all gt-networks leaf-labelled by a set S of taxa, is a metric space.*

We now prove that m^{tree} is also a metric on the space of gt-networks; the proof follows the same lines as that for m^{sp}.

Lemma 7. *Let N_1, N_2, and N_3 be three gt-networks leaf-labelled by a set S of taxa. Then:*

Positivity: $m^{tree}(N_1, N_2) \geq 0$.

Symmetry: $m^{tree}(N_1, N_2) = m^{tree}(N_2, N_1)$.

Triangle Inequality: $m^{tree}(N_1, N_3) \leq m^{tree}(N_1, N_2) + m^{tree}(N_2, N_3)$.

Proof. The positivity and symmetry properties follow immediately from the definition of m^{tree}. Since the RF measure is a metric [161], it satisfies the triangle inequality, and the result follows. $\square$

Theorem 6. *The space $(\mathcal{N}, m^{tree})$, where $\mathcal{N}$ is the set of all gt-networks leaf-labelled by a set S of taxa, is a metric space.*

6.5 Computing the Splits of gt-Networks

In general, a phylogenetic network N with m reticulation nodes and n leaves has up to $2^m n$ splits. However a gt-network, like a tree, has a linear number of splits.

Lemma 8. *Let N be a gt-network on n leaves, with set $\mathcal{Q}$ of m galls; we have*

$$|C(N)| = |E(N)| + \textstyle\sum_{Q \in \mathcal{Q}} |E(Q)| - 6m.$$

Proof. Let Q be gall in a gt-network N, with $RE(Q) = \{e_1, e_2\}$. Further, assume tree T_1 is induced by N by removing edge e_1 from Q, and tree T_2 is induced by N by removing edge e_2. The edges of $RP^Q(T_1)$ induce a set X of bipartitions and the edges of $RP^Q(T_2)$ induce a set Y of bipartitions, with $X \cap Y = \emptyset$. Hence, the edges of each gall Q induce

$$|E(RP^Q(T_1))| + |E(RP^Q(T_2))| = 2|E(Q)| - 6$$

bipartitions. Each edge of N that is not on a gall induces exactly one bipartition; the number of edges of N that are not on any gall is $|E(N)| - \sum_{Q\in\mathcal{Q}} |E(Q)|$. Hence, the total number of bipartitions is $|E(N)| - \sum_{Q\in\mathcal{Q}} |E(Q)| + 2(\sum_{Q\in\mathcal{Q}} |E(Q)| - 3)$, which is equal to $|E(N)| + \sum_{Q\in\mathcal{Q}} |E(Q)| - 6m$. □

Algorithm **ALG2** in Figure 6.9 computes the splits of a gt-network in $O(n^2)$ time.

Theorem 7. *Given a gt-network N on n leaves, algorithm* **ALG2** *correctly computes $C(N)$.*

Proof. The proof follows from the observation that gall nodes that are neither the coalescent nor the reticulation node contribute two splits each: those are the two sets S_v^0 and S_v^1. Every other node contributes exactly one split to $C(N)$, so that we have $S_v^0 = S_v^1$ for such nodes. □

Algorithm **ALG2** takes $O(n^2)$ time for a gt-network with n leaves. For each node v, we use bit vectors of length n to represent the sets S_v^0 and S_v^1, so that the union and subtraction operations on two sets take $O(n)$ time. Since there are $\Theta(n)$ nodes in N, each requiring $O(n)$ time to compute the two sets, the algorithm takes $O(n^2)$ time.

6.6 The Relationship Between Splits and Tripartitions

Lemma 9. *Let N be a gt-network on n leaves that contains a set $\mathcal{Q}$ of m galls. We have*

$$|\Psi(N)| = |E(N)| - 2m.$$

Proof. Given a gall Q_x^w, the two edges incident upon x and the one edge issuing from x induce the same tripartition, which is different from every tripartition induced by any other edge. Therefore, there are $|E(N)| - 3m + m = |E(N)| - 2m$ distinct tripartitions in $\Psi(N)$. $\square$

Despite the different definitions of the tripartition and split measures, the two are closely related. For an edge e, whose induced tripartition is $\langle A(e), B(e), C(e)\rangle$, there are two splits: $\langle A(e), B(e) \cup C(e)\rangle$ or $\langle A(e) \cup B(e), C(e)\rangle$. Hence, the error rates computed using the two measures are comparable. We now formalize this result for gt-networks.

Lemma 10. *Let N be a gt-network and e be an edge in N; assume that e induces the tripartition $\langle A, B, C\rangle$. If $e \in E(Q) - RE(Q)$ for some gall Q, then e induces exactly two splits: $\langle A, B \cup C\rangle$ and $\langle A \cup B, C\rangle$; otherwise, e induces exactly one split: $\langle A, C\rangle$ (because we have $B = \emptyset$).*

Proof. Let $e = (u, v) \in E(Q) - RE(Q)$ for some gall Q_x^w; let U be the set of taxa under v, W the set of taxa under x, and L the set of all taxa. Let T be a tree induced by N. If W is reachable from v, then edge e induces the split

$\langle U, L-U\rangle$; otherwise, it induces the split $\langle U-W, L-U\cup W\rangle$. Since edge e induces the tripartition $\langle U-W, W, L-U\rangle$, the conclusion follows immediately.

When edge $e \notin E(Q) - RE(Q)$ for any gall Q, we have U under v and $L-U$ unreachable from v in all trees induced by N. Hence, e induces exactly one split, $\langle U, L-U\rangle$, and exactly one tripartition, $\langle U, \emptyset, L-U\rangle$, and the conclusion follows. □

Theorem 8. *Let N_1 and N_2 be two gt-networks. Then,*

$$|\Psi(N_1) \triangle \Psi(N_2)|/2 \;\leq\; |C(N_1) \triangle C(N_2)| \;\leq\; 2|\Psi(N_1) \triangle \Psi(N_2)|$$

where $\triangle$ denotes the symmetric difference[1] *of two sets.*

The proof follows immediately from Lemmas 8, 9, and 10. The theorem relates the (un-normalized) topological error between two gt-networks based on their splits and tripartitions.

[1]The symmetric difference of two sets A and B, denoted by $A \triangle B$, is the set $(A-B) \cup (B-A)$.

ALG2

Input: A gt-network N with n leaves.

Output: The set $C(N)$ of splits in N.

Algorithm
Proceed in a bottom-up fashion, computing the set splits at each node. For a node v, we maintain two sets S_v^0 and S_v^1.

- For a leaf v, maintain a set $S_v^0 = S_v^1 = \{l_v\}$, where l_v is the taxon labeling node v.
- For a node v, $S_v^0 = S_v^1 = S_y \cup S_z$, where y and z are the two children of v.
- For a reticulation node v, $S_v^0 = S_v^1 = S_y$, where y is the child of v.
- For a node v on a gall Q_x^w, maintain two sets
  - $S_v^0 = S_y^1 \cup S_z^1 - S_x$, and
  - $S_v^1 = S_y^1 \cup S_z^1$,

  where y and z are the two children of v.
- For a coalescent node v, $S_v^0 = S_v^1 = S_y^1 \cup S_z^1$, where y and z are the two children of v.

Upon termination, we set $C(N) = \cup_{v \in V(N)} \{S_v^0, S_v^1\}$.

Figure 6.9: Algorithm for computing the splits of a gt-network.

Chapter 7

Genes and Species: Trees Within Trees and Trees Within Networks

A gene tree is a model of how a gene evolves through duplication, loss, and nucleotide substitution. As a gene copy at a locus in the genome replicates and its copies are passed on to more than one offspring, branching points are generated in the gene tree. Because the gene copy has a single ancestral copy, barring recombination, the resulting history is a branching tree [106]. Sexual reproduction and meiotic recombination within populations break up the genomic history into many small pieces, each of which has a strictly treelike pattern of descent [70, 82, 105]. Thus, within a species, many tangled gene trees can be found, one for each nonrecombined locus in the genome.

A phylogenetic (species) tree might be defined as the pattern of branching of species lineages via the process of speciation. When reproductive communities are split by speciation, the gene copies within these communities likewise are split into separate bundles of descent. Within each bundle, the gene trees continue branching and descending through time. Thus, the gene trees are contained within the branches of the species phylogeny [106].

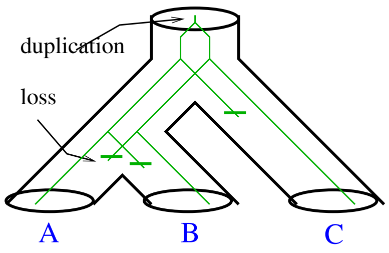
7.1 Gene Tree Incongruence

Gene trees can differ from one another as well as from the species tree. Such differences arise during the evolutionary process due to events such as duplication and loss, whereby each genome may end up with multiple copies of a given gene—but not necessarily the same copies that survive in another genome. Unless the genome is very well sampled, only a subset (sometimes only one copy, in fact) of the gene is used in phylogenetic analyses. As a result, the phylogeny for the gene may not agree with the species tree, nor with the phylogeny for another gene. In Figure 7.1 we illustrate two of the many evolutionary processes that can produce discordant gene trees, namely *lineage sorting* and *hybrid speciation.* One major difference between these two processes is that in the case of lineage sorting, the (species) phylogeny is still a tree; in the case of hybrid speciation, however, the (species) phylogeny cannot be modeled as a tree.

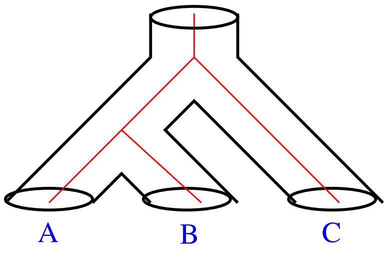
In Figures 7.1(a) and 7.1(b), we show an example of how lineage sorting operates. Figure 7.1(a) shows a gene evolves within the branches a species tree with one duplication event and three losses. Note that the species tree (denoted by the tubes) differs from the gene tree: based on this gene, B and C are sister taxa, and their most recent common ancestor and A are sister taxa.

Figure 7.1(b) shows a gene evolves within the branches of the same species tree; in this case, the topologies of the gene and species trees agree.

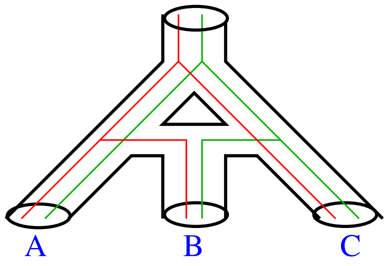
Since gene trees are used to estimate the underlying species tree, this



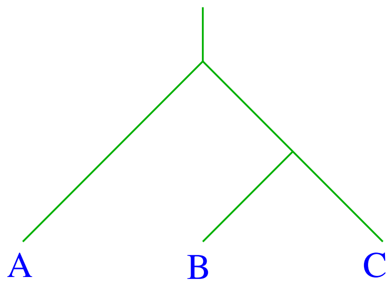
(a) Lineage sorting gene 1



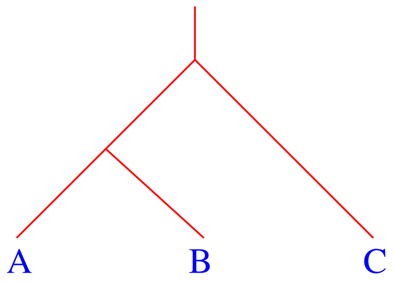
(b) Lineage sorting gene 2



(c) Phylogenetic network N



(d) Gene tree T_1



(e) Gene tree T_2

Figure 7.1: Gene trees, a species tree, and a species network for three species A, B, and C. The gene trees may be discordant due to lineage sorting or due to hybrid speciation. Further, the incongruence between the two gene trees may be similar in both cases. (a) Gene tree that disagrees with the (species) phylogeny due to lineage sorting. (b) Gene tree that agrees with (species) phylogeny. (c) A species phylogeny that contains one hybrid speciation event. (d) One of two possible gene trees inside the phylogeny in (c). (e) The second possible gene tree inside the phylogeny in (c).

can lead to problems in inferring species trees; however, these problems become evident only when two gene trees conflict each other. Note that these conflicts do not imply that the species have evolved under a network process. Therefore, in this case, inferring the species tree from a set of (potentially conflicting) gene trees is a problem of *reconciling* the gene trees, and explaining their differences through duplications and losses of genes.

Similarly, two gene trees may be discordant due to a hybrid speciation event. Figure 7.1(c) shows the evolutionary history of three species A, B, and C, where B is a hybrid. Shown inside the (species) phylogeny are two discordant gene trees (shown in Figures 7.1(d) and 7.1(e)). The topologies of the gene trees in Figures 7.1(a) and 7.1(b) agree with those in Figures 7.1(d) and 7.1(e), respectively. Hence, a major challenge in this case is to first determine whether the incongruence among gene trees is due to reticulation events (such as hybrid speciation and lateral gene transfer), or due to treelike events, such as gene duplication and loss.

7.2 Combined vs. Separate Analysis Approaches

Two basic approaches have been proposed for analyzing datasets containing data from different sources. In a *combined analysis*, one includes all the data in one combined dataset (via concatenation of the sequences) and infers a phylogeny on the basis of this combined dataset. For example, given several datasets of gene sequences, one could concatenate the sequences, thus resulting in species being represented by longer DNA sequences, which are then analyzed using standard methods. In a *separate analysis*, the method of choice is applied to each dataset separately, thus obtaining a tree (or set of trees) for each dataset. Then one attempts to combine, or reconcile, the trees. Both approaches have their proponents, although clearly when different data come from different trees (i.e., different tree topologies, not just different rates of evolution on the same underlying tree), a combined approach is a bad idea because a single tree does not describe the evolutionary process. For this reason, many people who prefer to do combined analysis nevertheless start with a separate analysis of each dataset and then analyze the results of these separate analyses for *congruence* (using techniques such as those described in [22, 84, 85]); only when the trees are sufficiently similar is a subsequent combined analysis then pursued.

Even when the underlying evolutionary trees of the different datasets are the same, there are challenges to doing a combined analysis. The statistical guarantees of good performance for both distance-based and ML methods depend very closely on the data fitting the model of evolution, and use the assumptions of the model in the analysis. Therefore, if the combined datasets violate the assumptions of the model, the method will not offer statistical performance guarantees. Furthermore, when the models are relaxed to accommodate the combined data, even ML methods are not guaranteed to be statistically consistent. For example, if we allow the positions within a DNA sequence to have arbitrarily defined rates, then ML methods, even given infinite data and run to optimality, can reconstruct the incorrect tree [176]!

7.3 The rSPR Operation

The rSPR operation transforms rooted trees into other rooted trees, and gene trees contained inside species networks are related to each other by rSPR operations; we now explain this relationship. Observe that the two rooted trees T_1 and T_2 in Figures 7.1(d) and 7.1(e) (induced by the network in Figure 7.1(c)) differ only in the location of the subtree that contains only taxon B. Tree T_2 can be obtained from T_1 by "pruning" the subtree that contains taxon B and "regrafting" it to another edge. In this case, we say that T_2 is obtained from T_1 by one *rooted subtree prune and regraft* (rSPR) operation.

Definition 7.3.1. (From [6]) A **rooted subtree prune and regraft** (rSPR) on a rooted binary tree T is defined as cutting any edge and thereby pruning a subtree, t, and then regrafting the subtree by the same cut edge to a new vertex obtained by subdividing a pre-existing edge in $T - t$. We also apply a forced contraction to maintain the binary property of the resulting tree.

The rSPR distance between two rooted binary trees T_1 and T_2, denoted by $d_{rSPR}(T_1, T_2)$, is the minimum number of rSPR operations needed to obtain T_2 from T_1. Computing the rSPR distance between trees plays a central role in the method that we propose for reconstructing networks. We formally define the (decision) rSPR distance problem as follows.

Definition 7.3.2. (The m-rSPR Distance Problem)

Input: Two binary trees, T_1 and T_2, leaf-labeled by a set S of n taxa, and a nonnegative integer m.

Question: Is $d_{rSPR}(T_1, T_2) = m$?

The m-rSPR Distance Problem is of unknown computational complexity [6]. However, for any constant m, we can solve the m-rSPR Distance Problem in polynomial-time, as we show in the following theorem.

Theorem 9. *Given two binary trees T_1 and T_2 leaf-labeled by a set S of n leaves, and a constant m, we can decide whether $d_{rSPR}(T_1, T_2) = m$ in $O(n^{2m})$ time.*

The proof follows directly from Definition 7.3.1 and Theorem 2.1 of [6], and is omitted.

7.4 Characterizing Trees Inside Networks

We begin by characterizing networks in general, using the model of [102], but with the added constraints that there is at least one regular speciation event between any two reticulation events, and that a species does not become extinct immediately after a reticulation event. Graph-theoretically, there is at least one tree node (whose two children are also tree nodes) on the directed path between any two reticulation nodes, and if one of the two children of a tree node is a reticulation node, then the other child is a tree node (see [102] for a discussion of the ramifications of missing taxa on reconstructing networks). In this case, we can obtain a nice characterization about the set of gene trees induced by a species network with m reticulations.

Theorem 10. *A species network N with m reticulation nodes induces 2^m distinct trees.*

Proof. We prove this by induction on m. It is easy to see that a network N with 0 reticulations is a tree, and hence induces one tree. Further, a network N with one reticulation induces two trees that differ in the location of the subtree rooted at the reticulation node (see Figure 7.1 for example). Assume that any network with m reticulations induces 2^m trees and consider a network N with $m+1$ reticulations. Let x be a reticulation node in N below which there are no other reticulation nodes. Let u_1 and u_2 be the two parents of x. The node u_1 is a tree node, and has another child v (a sibling of x), and u_2 is a tree node, and has another child w (different from v and a sibling of x). If we delete the edge (u_1, x), then the resulting network, N' has m reticulations, and by the induction hypothesis, N' induces 2^m distinct trees, where the subtree rooted at x is attached to node u_2 in all these trees. If we delete the edge (u_2, x), then the resulting network, N'' has m reticulations, and by the induction hypothesis, N'' induces 2^m distinct trees, where the subtree rooted at x is attached to node u_1 in all these trees. Hence, we have two sets of trees, each containing 2^m distinct trees, and clearly the two sets are different, due to the location of the subtree rooted at x. Therefore, we have 2^{m+1} distinct trees. □

As the example in Figure 7.1 illustrated, the gene trees induced by a species network are related through the rSPR operation. We extend this to the general case (with more than one reticulation) in the following theorem.

Theorem 11. *Let $\mathcal{T}$ be a set of 2^m distinct trees, $T_1, T_2, \ldots, T_{2^m}$. Then, $\mathcal{T}$ is the set of trees induced by a network N with m reticulations if and only if for every tree $T_i \in \mathcal{T}$,*

$$|\{T_j : d_{rSPR}(T_i, T_j) = k\}| = \binom{m}{k}, \tag{7.1}$$

where $0 \leq k \leq m$.

Proof. We first prove the direction "only if". Let $\mathcal{T}$ be a set of 2^m distinct trees induced by a network with m reticulations; we prove that (7.1) holds by induction on m.

For the base case of $m = 1$, let $\mathcal{T} = \{T_1, T_2\}$ be the set of two trees induced by a networks N with one reticulation. Let x be the reticulation node in network N, and let its two parents be u and v. Assume, without loss of generality, that x is a child of u in T_1, and x is a child of v in T_2. Hence, the two trees T_1 and T_2 differ only by the location of the subtree rooted at x. Obviously, tree T_2 can be obtained from T_1 by one rSPR move in which the subtree rooted at x is cut from its parent u, and connected to v. Hence, $d_{rSPR}(T_1, T_2) = 1$.

Assume that every network with m reticulations satisfies (7.1) and let N be a network with $m+1$ reticulations, and let x be a reticulation node below which there are no other reticulation nodes (i.e., “lowest” reticulation node). Further, assume u_1 and u_2 are the two parents of x. Let N' be a network with m reticulations obtained from N by removing the subtree T_x rooted at x, along with the two edges (u_1, x) and (u_2, x). Let T be a tree induced by N'. Then, in the set of trees induced by N', there are $\binom{m}{k}$ trees whose rSPR distance from T is k, and $\binom{m}{k-1}$ trees whose rSPR distance from T is $k-1$ (by the induction hypothesis). Each one of those trees will contribute two trees to the set $\mathcal{U}$ of trees induced by N, with the location of T_x being the only difference between the two copies of each tree. Half of those trees will maintain their rSPR distance from T and the other half will increase their rSPR distance (depends on where the subtree T_x is attached in each tree). Therefore, in the set of trees induced by N, the number of trees whose rSPR distance from T is k equals $\binom{m}{k} + \binom{m}{k-1}$, which equals $\binom{m+1}{k}$.

For the “if” part, let $\mathcal{T}$ be a set of 2^m trees that satisfies (7.1). Let T_1 and T_2 be two trees in $\mathcal{T}$ such that $d_{rSPR}(T_1, T_2) = m$. Then, T_2 can be obtained from T_1 by m rSPR operations, where in each operation an edge e_i is cut, thus pruning subtree t_i, which is then regrafted to edge e'_i, for $1 \leq i \leq m$; let $\mathcal{U} = \{t_i : 1 \leq i \leq m\}$. Every tree $T' \in \mathcal{T}$ such that $d_{rSPR}(T', T_1) = q$, $1 \leq q \leq m$, can be obtained from T_1 by q rSPR operations that involve only subtrees from the set $\mathcal{U}$; otherwise, there would exist a tree $T'' \in \mathcal{T}$ such that $d_{rSPR}(T'', T_1) > m$, which contradicts (7.1). The network N is obtained from T_1 by adding a set $\mathcal{E} = \{e^*_i\}$ of edges to tree T_1, where edge e^*_i connects edges e_i and e'_i (directed from e_i to e'_i). The resulting network N has m reticulations,

$u_1, u_2, \ldots, u_m$, and it induces the 2^m trees in $\mathcal{T}$. □

7.5 Ramifications of Missing Taxa on Network Reconstruction

7.5.1 Identifiability of Reticulation Events

Figure 7.2 illustrates a scenario in which identifying a reticulation event in a network may not be possible based solely on topological considerations. Figure 7.2(a) shows a network N on a set of five species, one of which (B) has become extinct. Based on *separate analyses* of the data [106, 128], the two trees in Figures 7.2(b) and 7.2(c) are the two possible gene trees that may be reconstructed. Those two gene trees may be *reconciled* into two different (phylogenetic) species networks as shown in Figures 7.2(d) and 7.2(e), each of which indicates a different reticulation event. In Figure 7.2(d), species C is the "product" of the reticulation event, whereas in Figure 7.2(e), species D is the product of the reticulation event. Thus, based on topological analysis alone, and in the presence of extinct or missing data, the identifiability of reticulation events may not be possible.

Nevertheless, this issue of identifiability, or the lack thereof, can be resolved if sufficient data are available so that the actual number of changes on the edges of the gene trees can be accurately estimated. In Figure 7.2(a), the variable associated with an edge in the true network denotes the actual amount of evolution on that edge. When sufficient data is available, those amounts of evolution can be estimated for the gene trees, as is shown in Figures 7.2(b)

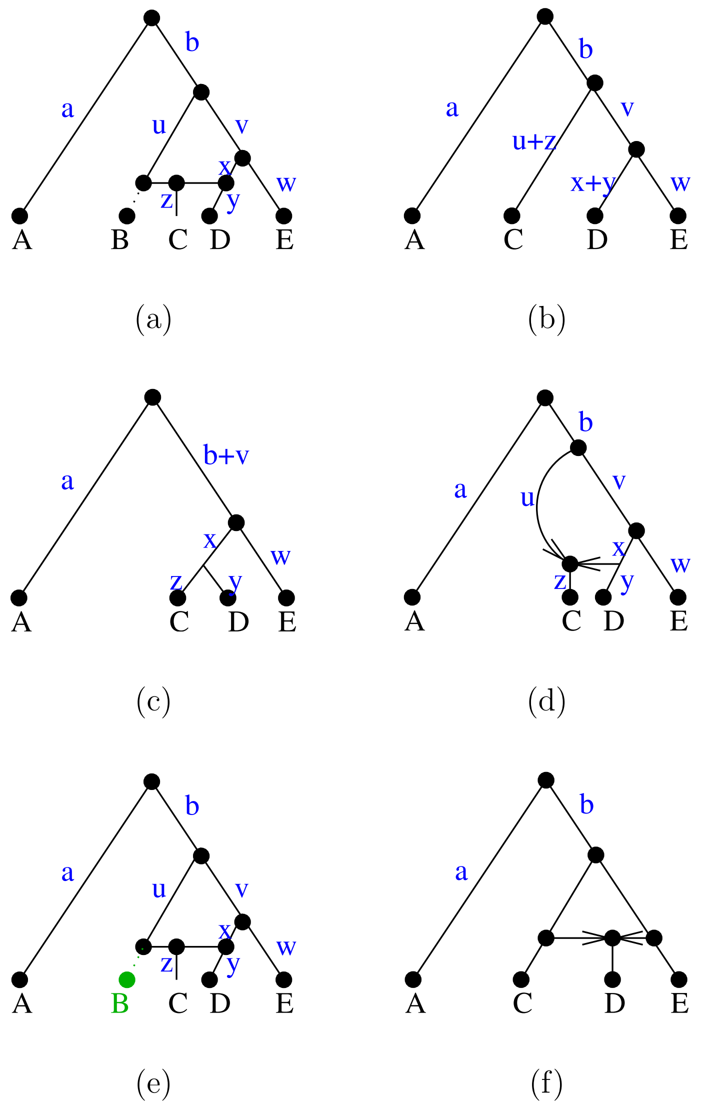


Figure 7.2: (a) A phylogenetic network N on 5 species, where species B had gone extinct. The network N induces the two trees shown in (b) and (c). The two networks N' and N'' in (d) and (e) represent different evolutionary histories (and, in particular, contains different reticulation events), yet each induces the two trees in (b) and (c). The network in (f) is an "augmentation" of (d) designed to satisfy the time coexistence property.

and 7.2(c). Once these amounts are estimated, they can be associated with the edges of the inferred network shown in Figure 7.2(d); yet, those amounts cannot be associated with the edges of the network in Figure 7.2(e), which rules out this network. Indeed, the network shown in Figure 7.2(d) captures the same evolutionary history of that captured by the network in Figure 7.2(a), when species B is eliminated and the internal node of indegree and outdegree 1 is eliminated, whereas the network in Figure 7.2(e) captures an evolutionary history different from that of the true phylogeny.

7.5.2 Reconstructing Missing Taxa

As stated in Section 4.2.1, a reticulation event takes place between two lineages that co-exist in time, requiring a phylogenetic network to satisfy the time coexistence property. However, as described in Section 4.2.2, missing taxa from a phylogenetic analysis may lead to a phylogenetic network that violates time coexistence. Figure 7.2(f) illustrates how to augment such a phylogenetic network to remedy that violation, by introducing nodes of indegree and outdegree 1. These nodes that are introduced "after the fact" may imply in most cases that a taxon (or a clade) is missing.

Figure 7.3(a) describes a phylogenetic network N on 5 species. With taxa B and D are missing from the phylogenetic analysis, the two (gene) trees in Figures 7.3(b) and 7.3(c) are reconstructed. Combining those two trees into a (species) network, we obtain the phylogenetic network in Figure 7.3(d), which clearly violates time coexistence. The three networks in Figures 7.3(e),

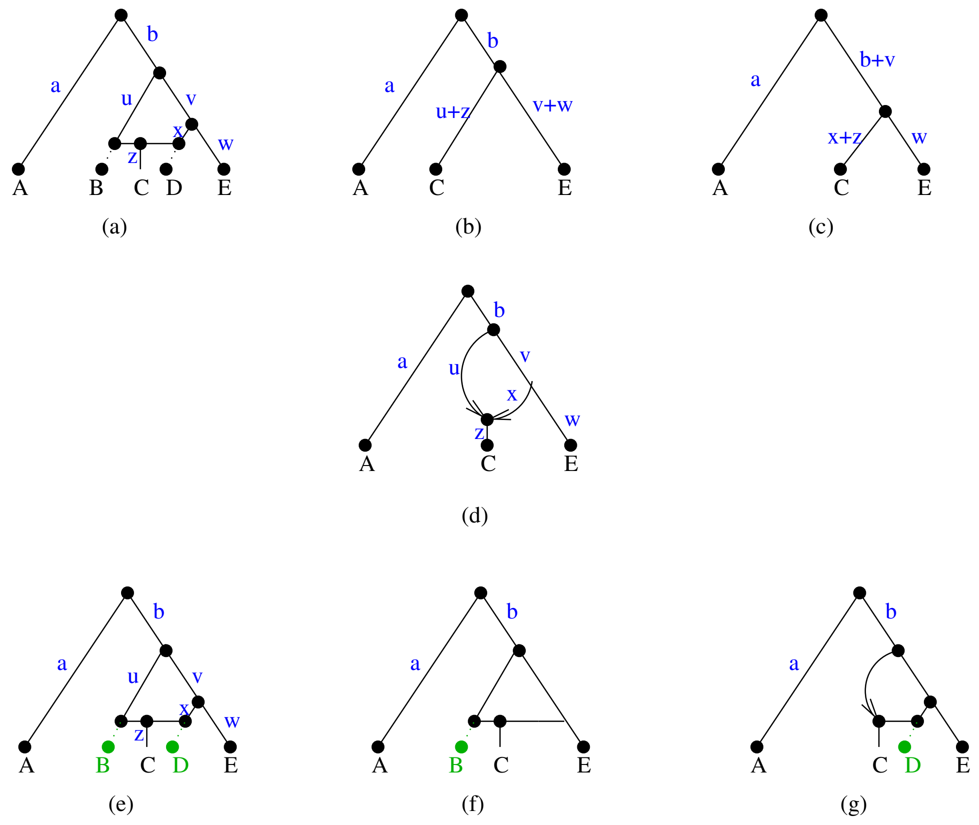


Figure 7.3: (a) A phylogenetic network N on 5 species, where species B and D are missing. The network N induces the two trees shown in (b) and (c). (d) A network N' that reconciles the two trees in (b) and (c). The networks in (e), (f) and (g) are augmentations of N' to satisfy time coexistence.

7.3(f), and 7.3(g) illustrate three possible ways of augmenting the network to satisfy time coexistence, each network giving rise to a different reticulation event and "detecting" one or two of the possible missing taxa.

Based on topology alone, the two trees in Figures 7.3(b) and 7.3(c) are identical. However, if the true amounts of change on the edges of the two trees could be estimated, the network in Figure 7.3(d) would be reconstructed for the species. Also, based on topology alone, the three networks in Figures 7.3(e), 7.3(f), and 7.3(g) are all valid augmentations of the network in Figure 7.3(d), but, if the true amounts of change on the edges were taken into account, the only valid network would be that of in Figure 7.3(e), which is also the true network.

Figure 7.4 illustrates a similar situation.

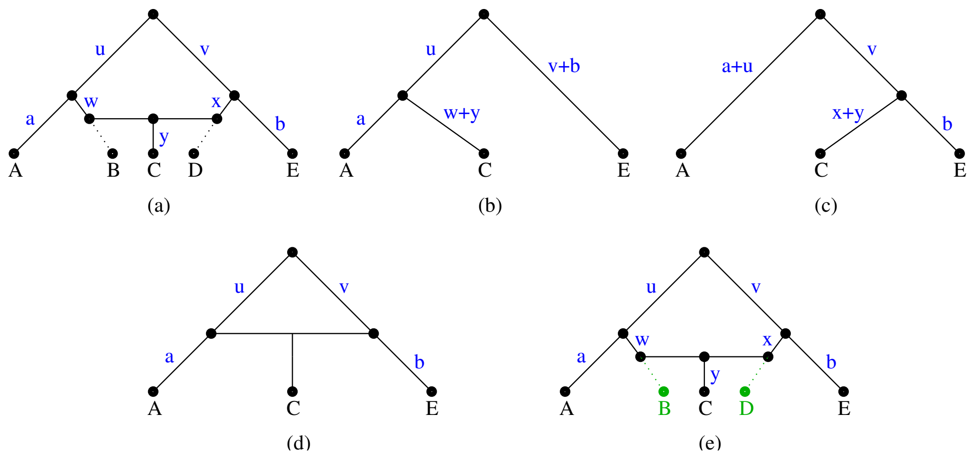


Figure 7.4: (a) A phylogenetic network N on 5 species, where species B and D are missing. The network N "induces" two trees shown in (b) and (c). (d) A network N' that induces the two trees in (b) and (c), but fails to capture the two missing species B and D. (e) A second network N'' that induces the two trees in (b) and (c) and captures the two missing species.

Chapter 8

Existing Methods for Detecting and Reconstructing Reticulate Evolution

As we saw in phylogenetic tree inference, the design and analysis of methods for the reconstruction of phylogenetic trees has several components:

- software for simulation studies: generating random model trees, simulating evolution down trees, and comparing inferred trees to model trees for accuracy
- algorithms and software for reconstructing phylogenetic trees

The phylogenetics community has produced much along these lines, and many of the best tools are quite good - provided the datasets are not too large. Similarly, we also need software for simulation studies for reticulate evolution, and algorithms and software for detecting reticulation and for reconstructing networks. In Chapters 5 and 6 we described our simulation tools and quality measures for studying network reconstruction methods. In this chapter, we describe the current state of methods for detecting and reconstructing reticulate evolution.

8.1 Detecting Reticulate Evolution

There are several methods which attempt to detect reticulation, and a few methods for reconstructing reasonable reticulate scenarios when reticulate evolution is suspected; however, much research needs to be done in order to develop better methods for both problems. The main challenge in detecting reticulation is that patterns in the data that are suggestive of reticulation may also be present due to other factors (such as lineage sorting, inadequate data, insufficient analyses, etc.), and it is difficult to distinguish between these different conditions; see [162] for a discussion of these issues. For this reason, among others, current methods for determining that reticulation has occurred have not been entirely successful. Reconstruction methods are also generally not yet sufficiently accurate for phylogeny reconstruction. In the following sections, we describe the existing approaches for detecting reticulate events at the population as well as species levels.

8.1.1 Detecting Recombination

Detecting recombination is a major topic of study in population genetics, with a commensurate number of publications. Studies of specific systems abound—any literature search using the keyword "recombination" will immediately bring up hundreds and any text on genetics will at least discuss the topic. Mostly, of course, such recombinations are meiotic in nature. In phylogenetic work, detecting recombinations (this time from a variety of sources) is at the heart of many approaches to the reconstruction of ancestral genomes or lines of descent [60, 70, 71, 79, 116, 118, 173, 180, 195]. Wiuf *et al.* [194] and Posada and Crandall [143] have studied the accuracy of methods for detecting recombination from a collection of DNA sequences; their papers contain and a wealth of references. Detecting the presence of recombination is but the first step; characterizing the recombinations that did take place is the goal. An intermediate goal along this path is to determine which recombination events might have taken place, as is done in many studies and implemented in programs such as SplitsTree [86] and NeighborNet [21]. A further step could determine the number of recombination events that took place [175], a goal that programs such as T-Rex [108] and the reconstruction algorithm of Hallett and Lagergren [4] attempt to reach. The last step is to produce a collection of recombination events that best explain the observed data, i.e., to produce one or more evolutionary networks that optimize some criterion (perhaps a generalization of a criterion used in tree reconstruction, such as minimum evolution, parsimony or maximum likelihood); few researchers have tackled that problem directly and none of the existing programs addresses it.

8.1.2 Detecting Horizontal Gene Transfer

Once again, the first order of business is to detect horizontal transfers. The goal of many biological studies has been to identify those genes that were acquired by the organism through horizontal transfers rather than inherited from its ancestors. In one of the first papers on the topic, Medigue [119] proposed the use of multivariate analysis of codon usage to identify such genes;

since then various authors have proposed other intrinsic methods, such as using GC content, particularly in the third position of codons (e.g., [96]). On the basis of such approaches, a database of genes acquired by horizontal transfer in prokaryotes has been established at `www.fut.es/∼debb/HGT/`. An advantage of intrinsic approaches is their ability to identify and eliminate genes that do not obey a tree-like process of evolution and thus could prevent classical phylogenetic methods from reconstructing a good tree. With the advent of whole-genome sequencing, more powerful intrinsic methods become possible, such as the location of suspect genes with each genome: such locations tend to be preserved through lineages, but a transfer event can place the new gene in a more or less random location. Thus a good look at the neighbors of a gene in the prokaryotic genome often enables biologists to identify horizontal transfers [97]. However, differential selection pressure, uneven evolutionary rates, and biased sampling can all give rise to false identification of HGT [40].

Non-intrinsic approaches bring us back to phylogenetic reconstruction. Here, the idea is to use phylogenetic reconstructions to identify discrepancies that could tag transfer events. An old question in phylogenetic reconstruction has been "to combine or not to combine?"—that is, given DNA sequences for several genes, are we better off concatenating the sequences or analyzing each set separately [22, 25, 27, 33, 83, 131, 192]? The common sense conclusion that many genes inherited through lineal descent would override the confusing signal generated by a few genes acquired through horizontal transfer appears wrong [18, 183]. Of course, one must first resolve the old problem of gene trees vs. species trees: discrepancies between the trees derived from different genes do not necessarily indicate reticulate evolution, but may simply testify to the incongruent evolution of two or more genes, all within a valid, tree-shaped evolution of the species. (For discussions of and algorithms for the gene tree/species tree reconciliation problem and some of its related problems, such as distinguishing orthologs from paralogs, see [8, 44, 103, 106, 135, 136, 138, 162, 178].) Distinguishing between the two is difficult in the absence of additional information.

With whole-genome sequencing, such information becomes available. Huynen and Bork [87] advocate two types of data: the fraction of shared orthologs and gene synteny. Synteny (the conservation of genes on the same chromosome) is of course not widely applicable with prokaryotes, but its logical extension, conservation of gene order, definitely is—and Huynen and Bork proposed to measure the fraction of conversed adjacencies, a notion that had been introduced earlier by Sankoff in a series of papers defining breakpoints (adjacencies that are not conserved) and their uses [15, 170]. Orthologs are a phylogenetic notion: two homologous genes are orthologs if they are the product of speciation from a common ancestor; in contrast, two homologous genes are paralogs if they are the product of duplication. Thus determining orthologs can be difficult. Daubin [28] combined orthology search and information from the DNA sequences themselves to improve the detection of horizontal transfers.

From a computing point of view, the problem can still be formulated as pure network reconstruction. Direct approaches include those of Hallett and Lagergren [4, 65] and Boc and Makarenkov [16].

8.1.3 Detecting Hybrid Speciation

Funk [54] and McDade [114] discuss the implications of hybridization for phylogenetic studies. Rieseberg and his colleagues have published numerous papers with case studies, discussions, and methodologies, all addressed at diploid hybrid speciation [42, 145–155]. Detecting hybrid speciation can be trivial: when the ploidy level (i.e., number of complete sets of chromosomes) changes, clearly hybrid speciation has occurred [147]. In other cases, the detection of hybrid speciation can be more difficult, especially when the parent species have similar sequences. As in the case of horizontal gene transfer, the true evolutionary history requires the network model. The two types of reticulate evolution—HGT and hybrid speciation—differ in significant ways. In HGT, the number of lineages does not change, but hybrid speciation directly creates a new lineage. In addition, under HGT the proportion of the donor genome to the recipient genome is relatively small, whereas in hybrid speciation is more nearly equal.

As discussed below, there are a small number of methods that attempt to detect and reconstruct hybrid speciation events [11, 21, 86, 171, 196], but none are entirely satisfactory, especially at reconstruction. In general, the methods produce an unacceptable number of false positives (edges that appear in the reconstruction, but are not part of the true network). The problems most likely arise because the methods tend to use combined data and because they lack sufficient biological rationale.

8.2 Reconstructing Reticulate Evolution

8.2.1 Combined Analysis Methods

Broadly speaking, four approaches to phylogenetic reconstruction in the presence of non-tree events have been developed. Since these have not been sufficiently studied in simulation studies, we do not yet know how well they perform, or whether these approaches are promising. Our discussion of the current network reconstruction methods is, therefore, quite brief.

1. With prokaryotic organisms, where the non-tree event is horizontal gene transfer (HGT), one common approach is to identify (by whatever means) the genes acquired through HGT, remove them, then reconstruct a tree based on the remaining genes. Most published bacterial phylogenies are trees and were either limited to one or a few genes for data, or use the whole genome, but only after eliminating genes identified as the product of HGT.

2. One can build a tree, then begin adding non-tree edges to turn it into a network, using a greedy approach to optimize some cost criterion. This is the approach used by Hallett and Lagergren [4, 65] (for horizontal transfers only), by Clement *et al.* [26], and by Makarenkov and his colleagues [108, 109].

3. Build many trees (perhaps using different subsets of the data) and attempt to reconcile them; where reconciliation fails, the conflict might be explained by a reticulation event. This is the basic idea behind median networks [11–13], as well as behind the molecular-variance parsimony approach of Excoffier *et al.* [45].

4. Characterize in advance any incompatibilities in the data (based on a distance matrix) and provide a collection of the possible resolutions through reticulation—leaving the biologist to choose which resolution is preferable. This approach is used in the splits-based methods [10, 21, 80, 86].

The first approach of course does not build a network; however, in groups of prokaryotes, where gene transfer is ubiquitous and there may not even exist a true "core" of genes, it has the advantage of representing the structured part of the information, with the understanding that nodes in the tree can be connected by a multitude of non-tree edges corresponding to HGTs. The last approach does not build or even propose a specific network, but presents all consistent choices—a potential problem when the number of choices is large.

8.2.2 Separate Analysis Methods

In 1997, Wayne Maddison [106] made an important observation which directly suggests a technique for reconstructing phylogenetic networks, via a "separate analysis" approach, which we now describe. Maddison observed that when there is one reticulation in the network, there are two trees within the network, and every gene evolves down one of these two gene trees (see Figure

7.1 in Chapter 7). Furthermore, the two trees are related to each other by a single rSPR move, and given the two trees, it is straightforward to construct the network that contained both trees. Maddison also suggested that when the network contains m reticulations, then any two trees contained in the network would be related to each other by a sequence of at most m rSPR moves (though the specific technique for producing the network from two of its constituent trees was not provided). Maddison's observations imply the following method for constructing phylogenetic networks:

- Step 1: For each gene dataset, infer a gene tree.
- Step 2: If the two trees are identical, return that tree. Else, find the minimum network that contains both trees.

In Chapter 9 we describe our new method, SpNet, for reconstructing species networks from gene trees. The method is based on Maddison's observation, but takes into account error in the gene tree estimates.

8.3 Software Related to Network Reconstruction

- **Pyramids**
  - `http://genome.genetique.uvsq.fr/Pyramids/`
  - Authors: J.C. Aude *et al.*
  - This program takes a distance matrix as an input and uses agglomerative algorithms to compute clusters. It allows for overlapping

  clusters, which can be used to represent reticulation events. Reticulation events may be placed among terminal nodes that are sister taxa.
  - Described in [34]
  - Platforms: Executables for Windows and Unix

- **Spectrum**
  - `http://taxonomy.zoology.gla.ac.uk/~mac/spectrum/spectrum.html`
  - Authors: Impkin Software (a division of Impkin Incorporated)
  - This program takes a set of sequences in the NEXUS format as an input and visualizes phylogenetic information without forcing it into a tree, thus avoiding the difficulty of choosing which is the "best" method for tree reconstruction and the whole issue of whether the data is tree-like.
  - Platforms: Executables for Mac and Windows are available

- **Arlequin**
  - `http://lgb.unige.ch/arlequin/`
  - Authors: Laurent Excoffier
  - This program supports population genetics analysis, such as estimation of gene frequencies, testing of linkage disequilibrium, and analysis of diversity between populations. It can also compute a Minimum Spanning Tree network.

  - Platforms: Mac, Windows, and Linux

- **PAL**
  - `http://www.stat.uni-muenchen.de/~strimmer/pal-project/`
  - Authors: Alexei Drummond and Korbinian Strimmer
  - A collection of Java classes for use in molecular phylogenetics. Among the tasks that have been implemented using this collection is computing the maximum likelihood of phylogenetic networks.
  - Platforms: JAVA source code

- **T-REX**
  - `http://www.fas.umontreal.ca/biol/casgrain/en/labo/t-rex/`
  - Authors: Vladimir Makarenkov
  - This program takes a distance matrix as an input, infers a tree, and then adds non-tree edges, thus creating a network, so as to minimize a certain least-squares loss function.
  - Described in [108]
  - Platforms: Windows, DOS, and Mac binaries, as well as C++ source code

- **PLATO**
  - `http://evolve.zoo.ox.ac.uk/Plato/main.html`
  - Authors: Nick Grassly

- This program takes sequential PHYLIP-style DNA sequences and their maximum likelihood phylogeny; using a likelihood approach with sliding window analysis and Monte Carlo simulation of the null distribution, it then detects anomalously evolving regions in the DNA sequences and assesses their significance. This may lead to the detection of, for example, recombination, gene conversion, or convergence; it may also reveal variable selective pressures along the gene sequences.
- Described in [59].
- Platforms: Mac (including PowerMacs) and source code for Unix systems

• **Bootscanning Package**

- by anonymous ftp from `http://www.ktl.fi` in directory `/hiv/mirrors/pub/programs`
- Authors: Mika Salminen and Wayne Cobb
- A series of shell scripts and programs that analyze DNA sequences for evidence of recombination. The code breaks the sequence into separate pieces that are analyzed for the bootstrap support of various groups; the code looks for evidence of significant conflict among trees for different parts of the sequence.
- Platforms: Sun executables

• **TOPAL**

  - `http://www.bioss.sari.ac.uk/ frank/Genetics`
  - Authors: Grainne McGuire
  - A collection of scripts and code that checks for evidence of past recombination events by looking for changes in the inferred phylogenetic tree TOPology between adjacent regions of a multiple sequence ALignment. The method detects recombinations by sliding a window along a sequence alignment and measuring the discrepancy between the trees suggested by the first and second halves of the window, using distance matrix methods.
  - Described in [115].
  - Platforms: Unix Bourne shell scripts and C code

- **reticulate**
  - `http://jcsmr.anu.edu.au/dmm/humgen/ingrid/reticulate.htm`
  - Authors: Ingrid Jakobsen and Simon Easteal
  - A compatibility matrix program for DNA sequences that includes tests for evidence of reticulate evolution (such as recombination). The program computes and displays a pairwise compatibility matrix for all pairs of sites and provides statistics on compatible pairs.
  - Described in [88]
  - Platforms: C source code for Unix and X Windows

- **RecPars**

  - `ftp.daimi.aau.dk` in directory `pub/empl/kfisker/programs/RecPars`
  - Authors: Kim Fisker
  - This program runs a parsimony analysis of DNA sequences and tries to find the best phylogenies for different regions of the sequences; different phylogenies lead it to postulate a recombination event between the respective segments.
  - Described in [71]
  - Platforms: C source code for Unix

- **Partimatrix**
  - `http://jcsmr.anu.edu.au/dmm/humgen/ingrid/partimatrix.htm`
  - Authors: Ingrid Jakobsen, Susan Wilson, and Simon Easteal
  - This program computes a "partition matrix" from aligned DNA sequence data. It finds partitions of the sequences into two groups and presents a matrix that describes conflicts and agreements among these partitions.
  - Described in [89]
  - Platforms: C source code for Unix systems with X Windows

- **Homoplasy test**
  - `http://www.biols.susx.ac.uk/Home/John_Maynard_Smith/`
  - Authors: John Maynard Smith and Noel Smith

  - This program implements the authors' homoplasy test for recombination in sequences.
  - Described in [173]
  - Platforms: QBASIC for DOS and must be run using QBASIC

- **LARD**
  - `http://evolve.zoo.ox.ac.uk/software/Lard/Lard.html`
  - Authors: Andrew Rambaut
  - This program detects the presence of recombination in a set of sequences. LARD looks at the set of sequences to discover which are the most plausible parents of a potentially recombinant sequence; at each possible breakpoint position, it runs a likelihood ratio test to check whether the three-species trees differ on the two sides of the breakpoint.
  - Described in [79]
  - Platforms: C source code and as a Macintosh executable

- **Network**
  - `http://www.fluxus-engineering.com/sharenet.htm`
  - Authors: Arne Röhl, Peter Forster, and Hans-Jürgen Bandelt
  - This program infers networks (which have more connections than trees) from non-recombining DNA, STR, amino acid, and RFLP

data. The networks are either reduced median networks nor median-joining networks.
  - Described in [11, 12]
  - Platforms: DOS executable

- **TCS**
  - `http://bioag.byu.edu/zoology/crandall_lab/tcs.htm`
  - Authors: Mark Clement and David Posada
  - This program estimates gene genealogies within a population, using a method that connects existing haplotypes in a minimum spanning tree (essentially a parsimony method).
  - Described in [26, 184]
  - Platforms: Java executables

- **SplitsTree**
  - `http://www-ab.informatik.uni-tuebingen.de/software/splits/`
  - Authors: D. Huson
  - This program outputs a network as a graphical representation of the incompatibilities in the dataset.
  - Described in [86]
  - Platforms: Unix and Windows

- **NeighborNet**

- `http://www.mcb.mcgill.ca/~bryant/NeighborNet/`
- Authors: D. Bryant and V. Moulton
- This program constructs a network that is a graphical representation of the compatibilities in the dataset.
- Described in [21]
- Platforms: Unix

• **Horizontal Transfer Program**

- `http://www.cs.mcgill.ca/~laddar/lattrans/`
- Authors: L. Addario Berry, M. Hallett, and J. Lagergren
- This program outputs a tree and a list of the horizontal gene transfer scenarios (i.e., a list of extra nontree edges that correspond to transfer events).
- Described in [4]
- Platforms: Java code

Chapter 9

Reconstructing Species Networks: Our New Method

As discussed in Chapter 8, Maddison's method uses a separate analysis where the (species) network can be reconstructed by first inferring individual gene trees from separate sequence datasets, and then reconciling the trees into a network. Maddison also showed the connection between phylogenetic network reconstruction and the calculation of rSPR distances (see Chapter 7 for the definitions). Maddison showed explicitly how to construct a network containing a single reticulation from its two trees (which are related by one rSPR move), and suggested that networks with additional reticulations could be inferred from individual gene trees, provided that the rSPR distance between two constituent gene trees could be calculated. However, Maddison did not show (or prove) how to use rSPR distance calculations in order to construct phylogenetic networks. Furthermore, the rSPR Distance problem is currently of unknown complexity [6] (a naïve algorithm would solve the problem in $O(n^{2m})$ time, where n is the number of leaves in each of the trees, and m is the rSPR distance between the trees).

In this chapter we consider the inference of "gt-networks" For this special case, we present polynomial time algorithms that provably reconstruct accurate phylogenetic networks, provided that accurate gene trees can be obtained. We also present polynomial time algorithms for reconstructing phylogenetic networks from inaccurate gene trees. In Chapter 10, we demonstrate the improvement in accuracy of these methods over two previous methods for phylogenetic network reconstruction in simulation.

9.1 Background

9.1.1 Graph-theoretic Definitions

Given a (directed) graph G, $E(G)$ denotes the set of (directed) edges of G and $V(G)$ denotes the set of nodes of G. We write (u,v) to denote a directed edge from node u to node v, in which case u is the *tail*, v the *head* of the edge, and u is a *parent* of v. The *indegree* of a node v is the number of edges whose head is v, while the *outdegree* of v is the number of edges whose tail is v. A directed path of length k from u to v in G is a sequence $u_0 u_1 \cdots u_k$ of nodes with $u = u_0$, $v = u_k$, and $\forall i$, $1 \leq i \leq k$, $(u_{i-1}, u_i) \in E(G)$; we say that u is the tail of p and v is the head of p.

Node v is *reachable* from u in G, denoted $u \rightsquigarrow v$, if there is a directed path in G from u to v; we then also say that u is an *ancestor* of v. Given a tree T and a subset L' of the leaves, we write $T|_{L'}$ to denote the subtree obtained by restricting T to leaves L', i.e., by removing all leaves not in L' and all incident edges. If X is a subtree of T, we denote by $T \setminus X$ the tree obtained by removing subtree X from T.

We denote by $L(T)$ the leaf-set of a tree T. An undirected path p of length k between u and v in a rooted tree T is a sequence $u_0u_1 \cdots u_k$ of nodes with $u = u_0$, $v = u_k$, and $\forall i$, $1 \leq i \leq k$, either (u_{i-1}, u_i) or (u_i, u_{i-1}) is an edge of T. If p is an undirected path in tree T, and whose two endpoints are u and v, we denote by $END(p) = (U, V)$ the two subtrees U and V attached to u and v, respectively, and that do not contain any edges from p. We use p to denote the path itself, as well as the edges of the path.

9.1.2 Strict Consensus and Compatibility Trees

Let T be a tree leaf-labeled by a set S of taxa. Each edge e in T induces a *bipartition* $\pi(e) = \{A(e)|B(e)\}$ on the set S, where $A(e)$ is the set of taxa "below" e, and $B(e)$ is the set containing the rest of the taxa. We denote by $C(T)$ the set of all bipartitions induced by tree T. We say that e_1 and e_2 (and their associated bipartitions) are *compatible*, denoted $e_1 \equiv e_2$, if there exists a tree T that induces both $\pi(e_1)$ and $\pi(e_2)$. This definition of compatibility naturally extends to sets of bipartitions, and hence also to trees.

If we contract an edge in T, thus identifying the endpoints of that edge, we obtain another tree T' on the same leaf set; T is then said to *refine* T', and T' is said to be a *contraction* of T. If T'' is the result of contracting a set of edges in T, then too T'' is a contraction of T, and T is a refinement of T''.

A set of trees is compatible if the trees have a common refinement; its minimal common refinement (called the *compatibility tree*) is unique. For any set of trees, the maximally resolved common contraction (called the *strict* *consensus tree*) is also unique. Both the compatibility and strict consensus trees can be found in $O(kn)$ time, where there are k trees on the same set of n leaves [30, 62, 187].

Given two trees T_1 and T_2, the set $U(T_1, T_2)$ contains all edges of T_1 that are not compatible with T_2; $U(T_2, T_1)$ is defined similarly. Note then that T_1 and T_2 are compatible if $U(T_1, T_2) = U(T_2, T_1) = \emptyset$.

If a tree T has a node v with indegree and outdegree one, we replace the two edges incident to v by a single edge; this operation on T is called *forced contraction.*

9.2 Reconstructing gt-Networks When Gene Tree Estimates are Accurate

There are two main limitations to Maddison's approach: (1) the construction of a network from two gene trees is only described explicitly when the network contains exactly one reticulation; (2) obtaining accurate binary trees in practice may not be possible in most cases. Our new method for reconstructing networks builds upon Maddison's observation. In this section we address the first limitation by showing how to accurately construct a gt-network, with any number of reticulations, from two of its constituent gene trees. However, since any two gene trees may not involve both parents in each reticulation, the best we can hope for is to reconstruct the minimal network that contains both trees; this is what we construct. We address the second limitation in the next section. The results in this chapter appeared in [128].

9.2.1 Efficient Reconstruction of gt-Networks from Gene Trees

Theorem 11 in Chapter 7 implies that given the "full" set of trees induced by a gt-network, we can reconstruct the network via a series of rSPR distance computations among the trees. However, given a pair of trees induced by a gt-network, we can only reconstruct a minimal (in terms of the number of reticulation nodes) gt-network that induces these two trees. In what follows, we show how to efficiently reconstruct such a minimal network from a pair of trees. Hereafter, we use n to denote the number of leaves in the trees as well as networks.

The intuition behind our algorithm is as follows. Given two trees T_1 and T_2, induced by a gt-network N, we first "mark" the edges of each tree that are incompatible with the other tree (symbolized by $U(T_1, T_2)$ and $U(T_2, T_1)$ in the proofs). If the two trees are m rSPR moves apart, the marked edges in tree T_1 would form m node-disjoint paths in T_1, and similarly for tree T_2. While necessary, this condition is not sufficient; an extra step is needed, in which, for each maximal path p_1 of marked edges in T_1, there must exist a unique maximal path p_2 of marked edges in T_2, where the endpoints of the two paths correspond to one rSPR move.

Lemma 11. *Let T_1 and T_2 be two trees induced by a gt-network N. Further, assume that gall Q_x^w in N was broken in the two different ways to obtain T_1 and T_2. Then, $RP^Q(T_1) \subseteq U(T_1, T_2)$, and $RP^Q(T_2) \subseteq U(T_2, T_1)$.*

Proof. Let RP^Q be formed of the two paths p_1 and p_2 whose tail is w. Further, assume p_1 is the path attached to edge e_1 and p_2 is the path attached to e_2, where $RE(Q) = \{e_1, e_2\}$. Let X be the subtree rooted at node x, T_1 be obtained by removing edge e_1 from Q, and T_2 be obtained by removing edge e_2 from Q. Then, in T_1, the leaves of X are under the edges of p_2 but not under the edges of p_1, whereas in T_2, the leaves of X are under the edges of p_1 but not under the edges of p_2. Hence, the edges on $RP^Q(T_1)$ are incompatible with the edges of $RP^Q(T_2)$, and vice versa. Therefore, we have $RP^Q(T_1) \subseteq U(T_1, T_2)$ and $RP^Q(T_2) \subseteq U(T_2, T_1)$. $\square$

Lemma 12. *Let T_1 and T_2 be two trees induced by a gt-network N. Further, assume that gall Q in N was broken in exactly the same way to obtain both T_1 and T_2. Then, $RP^Q(T_1) \cap U(T_1, T_2) = \emptyset$, and $RP^Q(T_2) \cap U(T_2, T_1) = \emptyset$.*

Proof. Since the gall Q is broken in exactly the same way to obtain the two trees T_1 and T_2, it follows that the edges on $RP^Q(T_1)$ induce the same bipartitions as those induced by the edges of $RP^Q(T_2)$. Hence, the edges of $RP^Q(T_1)$ and $RP^Q(T_2)$ are mutually compatible. Further, the edges of $RP^Q(T_1)$ are compatible with $E(T_2) \setminus RP^Q(T_2)$; otherwise, the network N would not be a gt-network (there would be two "overlapping" galls). Similarly, the edges of $RP^Q(T_2)$ are compatible with $E(T_1) \setminus RP^Q(T_1)$. Therefore, it follows that $RP^Q(T_1) \cap U(T_1, T_2) = \emptyset$ and $RP^Q(T_2) \cap U(T_2, T_1) = \emptyset$. $\square$

Let T be a tree induced by a gt-network N. We denote by $NG(T)$ the set of all edges e that are not on any gall in N. Formally, $NG(T) = \{e \in E(T) :$ for all galls Q in N, $e \notin RP^Q(T)\}$.

Lemma 13. *Let T_1 and T_2 be two trees induced by a gt-network N. Then, $NG(T_1) \cap U(T_1, T_2) = \emptyset$, and $NG(T_2) \cap U(T_2, T_1) = \emptyset$.*

Proof. Assume $e = (u, v)$ is an edge in $NG(T_1) \cap U(T_1, T_2)$. Let A be the subtree of T_1 rooted at v. Since $e \in U(T_1, T_2)$, then, for some edge $e' = (u', v')$ in T_2, $L(A) \cap L(B) \neq \emptyset$, where B is the subtree of T_2 rooted at v'. Let $X = L(A) \setminus (L(A) \cap L(B))$. Then, in tree T_1, X is under edge e, and in tree T_2, X is not under edge e'. Hence, edges e and e' are members of $E(Q)$ for some gall Q; a contradiction that $e \in NG(T_1)$. Therefore, $NG(T_1) \cap U(T_1, T_2) = \emptyset$; similarly, we prove that $NG(T_2) \cap U(T_2, T_1) = \emptyset$. □

Theorem 12. *Let N be a gt-network with q galls $\{Q_1, Q_2, \ldots, Q_q\}$, and T_1 and T_2 be two trees induced by N. Further, assume that exactly m of the q galls were broken in the two possible ways to obtain the two trees T_1 and T_2, and the other $q - m$ galls were each broken in a single way. Then, $U(T_1, T_2)$ forms m node-disjoint undirected paths in T_1, and $U(T_2, T_1)$ forms m node-disjoint undirected paths in T_2.*

The proof follows immediately from Lemma 11, Lemma 12, and Lemma 13, and is omitted. Stated differently, Theorem 12 implies that if T_1 and T_2 are two trees induced by a gt-network N such that $d_{rSPR}(T_1, T_2) = m$, then each of the two sets $U(T_1, T_2)$ and $U(T_2, T_1)$ forms m node-disjoint undirected paths in T_1 and T_2, respectively.

Let T_1 and T_2 be two trees induces by a gt-network N, such that $U(T_1, T_2)$ forms a set of node-disjoint undirected paths in T_1, and $U(T_2, T_1)$

forms a set of node-disjoint undirected paths in T_2. Let p_1 be one such path in $U(T_1, T_2)$, and p_2 be one such path in $U(T_2, T_1)$. Further, assume $END(p_1) = (U_1, V_1)$ and $END(p_2) = (U_2, V_2)$. We say that p_1 **yields** p_2 **in one rSPR move** (via subtree X), denoted $p_1 \models^X p_2$, if there exists a nonempty subtree X such that

1. X is a subtree of either U_1 or V_1,

2. X is a subtree of either U_2 or V_2, and

3. $p_1 \cap U(T_1', T_2') = \emptyset$ and $p_2 \cap U(T_2', T_1') = \emptyset$, where $T_1' = T_1 \setminus X$ and $T_2' = T_2 \setminus X$.

Theorem 13. *Let T_1 and T_2 be two trees induced by a gt-network N. Further, assume that $U(T_1, T_2)$ forms a set P_1 of m node-disjoint undirected paths $p_1^1, p_2^1, \ldots, p_m^1$ in T_1, and $U(T_2, T_1)$ forms a set P_2 of m node-disjoint undirected paths $p_1^2, p_2^2, \ldots, p_m^2$ in T_2. Then, $d_{rSPR}(T_1, T_2) = m$ if there is an injective function $f : P_1 \to P_2$ and m subtrees $X_1, X_2, \ldots, X_m$ such that $f(p_i^1) = p_j^2$ iff $p_i^1 \models^{X_i} p_j^2$, where $1 \le i, j \le m$.*

Proof. Let P_1 and P_2 be the two sets of paths in the lemma, and let f be the injective function. Let $p_i^1 \in P_1$ and $p_j^2 \in P_2$ be two paths such that $f(p_i^1) = f(p_j^2)$. Assume X_i is the subtree such that $p_i^1 \models^{X_i} p_j^2$. Then, X_i is the subtree whose pruning from T_1 and regrafting it to another edge (to obtain tree T_2) yielded paths p_i^1 and p_j^2 in the two trees, respectively. Since there are m such pairs of paths, there are m such subtrees X_i whose pruning and regrafting in T_1 would yield a tree T such that $U(T_1, T_2) = \emptyset$, which implies $T = T_1$. Hence, $d_{rSPR}(T_1, T_2) = m$. □

Theorem 14. *Let T_1 and T_2 be two binary trees induced by a gt-network N. We can decide whether $d_{rSPR}(T_1, T_2) = m$ in $O(mn)$ time.*

Proof. Preprocess the trees so that (1) For every two leaves s_i and s_j in either tree, the least common ancestor (LCA) of these two leaves can be found in constant time. This can be achieved in $O(n)$ time using the techniques from [30, 67], and (2) For any internal nodes, i, the number $\beta(i)$ of leaves below i can be found in constant time. Further, if S_i is the set of leaves under i, then $LCA(S_i)$ can be found in constant time. This can be achieved in $O(n)$ time using the techniques from [30]. After this preprocessing, computing $U(T_1, T_2)$ and $U(T_2, T_1)$ takes $O(n)$ time ($O(1)$ time for each edge, and there are $O(n)$ edges). This can be done by observing that an edge $e = (u, v)$ is in $U(T_1, T_2)$ if and only if $\beta(i) \neq \beta(LCA(S_i))$ (i is the number assigned to node v). It takes $O(n)$ time to check if $U(T_1, T_2)$ forms a simple path. Further, it takes $O(n)$ time to check if the conditions of Theorem 12 and Theorem 13 hold (we can find $f(p_i^1)$, if it exists, in $O(1)$ time, by using the "highest" node of path p_i^1 and finding its counterpart in T_2; to find that subtree X_i such that $p_i^1 \models^X p_j^2$, we need to compute the four pairwise intersections of a set in $END(p_i^1)$ and a set in $END(p_j^2)$, each of which takes $O(n)$ time, using bit vector representation of sets). Hence, we can decide whether $d_{rSPR}(T_1, T_2) = m$ in $O(mn)$ time. □

Now, it is straightforward to construct a gt-network N with m galls in $O(mn)$

time, given two induced trees T_1 and T_2 such that $d_{rSPR}(T_1, T_2) = m$.

Theorem 15. *Let T_1 and T_2 be two binary trees such that $d_{rSPR}(T_1, T_2) = m$. We can decide whether T_1 and T_2 are induced by a gt-network N with m reticulation events, and if so construct such N, in $O(mn)$ time.*

Proof. By Theorem 14, we can find the set of m subtrees $\{X_1, \ldots, X_m\}$, such that $p_i^1 \models^{X_i} p_i^2$, for $1 \leq i \leq m$, where p_i^1 is a path in $U(T_1, T_2)$, p_i^2 is a path in $U(T_2, T_1)$, and $f(p_i^1) = p_i^2$ (f is the injective function in the definition of $\models^X$). All this can be done in $O(mn)$ time. We form the gt-network N from T_1 as follows. For each path p_i^1 in $U(T_1, T_2)$ and its corresponding subtree X_i, X_i will be attached to one end of p_i^1. We add another edge from the other end of p_i^1 to the root of X_i, thus creating a network N with m galls. Since the paths are node-disjoint, N will be a gt-network. $\square$

9.3 Reconstructing gt-Networks When Gene Tree Estimates are Inaccurate

The main limiting factor in Maddison's approach is that methods, even if statistically consistent, can fail to recover the true tree; even on quite long sequences, some topological error is often present. This topological error can be tolerated in a phylogenetic analysis, but it makes the inference of phylogenetic networks from constituent gene trees difficult. To overcome these limits, we propose a method that allows for error in the estimates of the individual gene trees; consequently, our method performs much better in practice (as our simulation studies show).

Before we describe the method, we provide some insight into its design. When methods such as maximum parsimony or maximum likelihood are used to infer trees, typically a number of trees are returned, rather than a single best tree. For example, in maximum parsimony searches, especially with larger datasets, there are often many equally good trees (all having the same best score), and all can be returned (along with suboptimal trees, if desired). In maximum likelihood, although the best-scoring tree may be unique, the difference in quality between that tree and the next best tree(s) can be statistically insignificant, and so again, a number of trees can be returned [172]. A common output of a phylogenetic analysis is the strict consensus of these trees (that is, the most resolved common contraction of all the best trees found).

The interesting, and highly relevant, point here is the following observation, supported by both empirical studies on real datasets and simulations: *the strict consensus tree will often be a contraction of the true tree.* Thus, even when every tree in the set of best trees is a little bit wrong, the strict consensus tree (which contains only those edges common to all the best trees) is likely to be a contraction of the true tree. This observation suggests the following approach to inferring phylogenetic networks.

Proposed Approach

- **Step 1:** For each gene dataset, use a method (such as maximum parsimony or maximum likelihood) of choice, to construct a set of "best" trees, thus producing sets $\mathcal{T}_1$ and $\mathcal{T}_2$.

- **Step 2:** Compute the strict consensus tree t_i for $\mathcal{T}_i$, for $i = 1, 2$.
- **Step 3:** Find trees T_1 and T_2 refining t_1 and t_2 such that T_i refines t_i for each $i = 1, 2$, and T_1 and T_2 are induced trees within a gt-network with p reticulations, for some minimum p.

When $p = 0$, the two consensus trees are compatible, and we would return the compatibility tree; see Section 9.1. We now show how to handle the third step in this method when $p = 1$ (solving this for general p is currently an open problem). In this case, Step 3 involves solving the following problem.

Combining consensus trees into a network (the ConsTree-Network Problem)

- *Input:* Two trees, t_1 and t_2, on the same set of leaves (not assumed to be binary)
- *Output:* A network N inducing trees T_1 and T_2, such that N contains one reticulation, and T_i refines t_i, for $i = 1, 2$, if it exists; else *fail*.

We now provide a linear-time algorithm for this problem. There are two cases to consider: when the two consensus trees are compatible, and when the two trees are incompatible.

9.3.1 Compatible Consensus Trees

In most cases, if the consensus trees share a common refinement, we might believe the evolution to be tree-like (in which case we should combine the datasets, and analyze a tree directly). However, suppose we have reason to believe that a dataset has undergone reticulation, so that a tree is not an appropriate representation of the true tree. In this case, we can still seek reticulate evolutionary scenarios compatible with our observations. We begin with a simple lemma.

Observation 2. *Let t be a binary tree that refines an unresolved tree T, and let p be a path in tree t. Then, when restricted to the edges of T, p forms a path in T as well.*

Lemma 14. *Let t be an unresolved tree. Then, there exist two binary trees T_1 and T_2 that refine t and such that $d_{rSPR}(T_1, T_2) = 1$.*

Proof. Let x be a node with outdegree 3, and let v_1, v_2, and v_3 be the three children of node x. We obtain T_1 from t by removing the edges (x, v_1) and (x, v_2), adding a new node u with an edge (x, u), and then making v_1 and v_2 children of u. The tree T_2 can be obtained from t by removing the edges (x, v_2) and (x, v_3), adding a new node u with an edge (x, u), and then making v_2 and v_3 children of u. The rest of the nodes of T_1 and T_2 are resolved identically in both trees. It is obvious that T_2 can be obtained from T_1 by pruning v_2 from its parent and attaching it to edge (x, v_3) in T_1, and hence $d_{rSPR}(T_1, T_2) = 1$. □

Lemma 14 can be generalized to the case where t_1 and t_2 are two unresolved, yet compatible trees, as follows.

Lemma 15. *Let t_1 and t_2 be two compatible unresolved trees. Then, there exist two binary trees T_1 and T_2 that refine t_1 and t_2 respectively, and $d_{rSPR}(T_1, T_2) = 1$ if and only if t_1 and t_2 have a common refinement t that is not fully resolved. Furthermore, we can determine if these two trees exist, and construct them, in $O(n)$ time.*

Proof. The proof of the "if" part follows from Lemma 14. We prove the "only if" part. Let T_1 and T_2 be two binary trees that refine two unresolved (compatible) trees t_1 and t_2, such that $d_{rSPR}(T_1, T_2) = 1$. Since t_1 and t_2 are compatible, then they share a common refinement, t. The two binary trees T_1 and T_2 also refine t. Since T_1 and T_2 are different binary trees and refine the same tree t, it follows that t is not fully resolved. □

9.3.2 Incompatible Consensus Trees

We now address the last remaining case, where the consensus trees are incompatible. We begin with a simple lemma.

Lemma 16. *Let T_1 and T_2 be two binary trees that refine two unresolved trees t_1 and t_2. Then, $U(t_1, t_2) \subseteq U(T_1, T_2)$.*

Proof. Let $e \in U(t_1, t_2)$. Then, $e \in E(t_1)$, and consequently $e \in E(T_1)$. Further, e is incompatible with t_2, and hence is incompatible with T_2. It follows that $e \in U(T_1, T_2)$. Therefore, $U(t_1, t_2) \subseteq U(T_1, T_2)$. □

Lemma 17. *Let t_1 and t_2 be two unresolved incompatible trees. If there exist two binary trees T_1 and T_2 that refine t_1 and t_2, respectively, and such that*

$d_{rSPR}(T_1, T_2) = 1$, *then* $U(t_1, t_2)$ *and* $U(t_2, t_1)$ *are both simple paths in* t_1 *and* t_2*, respectively.*

Proof. Assume $U(t_1, t_2)$ is not a simple path in t_1. Then, by Theorem 12, it follows that $U(T_1, T_2)$ is not a simple path in T_1, and hence $d_{rSPR}(T_1, T_2) \neq 1$; a contradiction. Therefore, $U(t_1, t_2)$ forms a simple path in t_1. Similarly, we establish that $U(t_2, t_1)$ forms a simple path in t_2. □

Lemma 18. *Let* t_1 *and* t_2 *be two incompatible unresolved trees, such that* $U(t_1, t_2)$ *forms a path* p_1 *in* t_1*, and* $U(t_2, t_1)$ *forms a path* p_2 *in* t_2*. Further, let* $END(p_1) = (A_1, B_1)$ *and* $END(p_2) = (A_2, B_2)$*. Let* X_i*,* $1 \leq i \leq 4$*, be the following four sets:* $X_1 = (A_1 - A_2) \cap (B_2 - B_1)$*,* $X_2 = (A_1 - B_2) \cap (A_2 - B_1)$*,* $X_3 = (B_1 - A_2) \cap (B_2 - A_1)$*, and* $X_4 = (B_1 - B_2) \cap (A_2 - A_1)$*. Then, there exist two binary trees* T_1 *and* T_2 *that refine* t_1 *and* t_2*, respectively, and* $d_{rSPR}(T_1, T_2) = 1$*, if and only if there exists an* i*,* $1 \leq i \leq 4$*, such that*

*(**C1**)* $t_1|_{S \setminus X_i}$ *and* $t_2|_{S \setminus X_i}$ *are compatible,*

*(**C2**)* $t_1|_{X_i}$ *and* $t_2|_{X_i}$ *are compatible, and*

*(**C3**)* $t_1|_{S \setminus X_i}$ *contains all the edges in* $U(t_1, t_2)$*, and* $t_2|_{S \setminus X_i}$ *contains all the edges in* $U(t_2, t_1)$*.*

Proof. Assume that X_i, for some $1 \leq i \leq 4$, satisfies both conditions C1 and C2. Then, resolve $t_1|_{S \setminus X_i}$ and $t_2|_{S \setminus X_i}$ identically, resolve $t_1|_{X_i}$ and $t_2|_{X_i}$ identically, and finally attach the resolved subtrees $t_1|_{X_i}$ and $t_2|_{X_i}$ in their corresponding subtrees. The result is obviously two binary trees T_1 and T_2 that differ only in the location of the subtree leaf-labeled by X_i; i.e., $d_{rSPR}(T_1, T_2) = 1$. Let T_1 and T_2 be two binary trees that resolve the two incompatible unresolved trees t_1 and t_2, such that $d_{rSPR}(T_1, T_2) = 1$. By Lemma 16, $p_1 \subseteq U(T_1, T_2)$ and $p_2 \subseteq U(T_2, T_1)$. By Theorem 13, T_2 can be obtained from T_1 by pruning a subtree t' from one side of the path $U(T_1, T_2)$ and regrafting it on the other side of the path. It follows that $t_1|_{S \setminus L(t')}$ and $t_2|_{S \setminus L(t')}$ are compatible, and also that $t_1|_{L(t')}$ and $t_2|_{L(t')}$ are compatible (since they refine the same tree t'). It is straightforward to verify that $L(t')$ is equal to X_i, for some $1 \leq i \leq 4$. $\square$

We now state the major theorem of this section.

Theorem 16. *We can solve the ConsTree-Network Problem in $O(n)$ time. That is, given two unresolved trees t_1 and t_2, in $O(n)$ time we can find two binary trees T_1 and T_2 that refine t_1 and t_2, respectively, such that $d_{rSPR}(T_1, T_2) = 1$, when such a pair of trees exist. Further, once we have T_1 and T_2, we can compute a phylogenetic network with exactly one reticulation event inducing these trees in $O(n)$ additional time.*

Proof. We preprocess the trees so that: (1) For every two leaves s_i and s_j in either tree, the least common ancestor (LCA) of these two leaves can be found in constant time. This can be achieved in $O(n)$ time using the techniques from [30, 67], and (2) For any internal node, i, the number $\beta(i)$ of leaves below i can be found in constant time. Further, if S_i is the set of leaves under i, then $LCA(S_i)$ can be found in constant time. This can be achieved in $O(n)$ time using the techniques from [30]. After this preprocessing, computing $U(t_1, t_2)$ takes $O(n)$ time ($O(1)$ time for each edge, and there are $O(n)$ edges). It takes $O(n)$ time to check if $U(t_1, t_2)$ forms a simple path. By Lemmas 17 and 18, we first check whether $U(t_1, t_2)$ and $U(t_2, t_1)$ form simple paths. Then, we check whether conditions C1 and C2 of Lemma 18 hold; if so, then we can obtain two binary trees T_1 and T_2 that resolve t_1 and t_2, such that $d_{rSPR}(T_1, T_2) = 1$. Having preprocessed the trees, testing the conditions of these two lemmas can be achieved in $O(n)$ time. Using bit vectors to represent the sets of taxa, we can preprocess the trees in $O(n)$ time such that we store at each node the set of taxa under it; hence, it takes $O(n)$ time to compute the sets X_i, $1 \leq i \leq 4$. We construct N from T_1 and T_2 in $O(n)$ time using Theorem 15. Hence, the algorithm takes $O(n)$ time. □

9.4 SpNet: Our Technique for Inferring gt-Networks

SpNet, for Species Network, is a method we have designed for inferring networks (or trees, depending on the data) under realistic conditions. We base SpNet on the approach we outlined in the previous section, but we specifically use maximum likelihood for tree reconstruction, and we compute the strict consensus of the best two trees for each dataset. In order to facilitate a comparison to other methods, such as NeighborNet, we do not allow SpNet to return "fail", and so we apply Neighbor Joining (NJ) to all inputs on which we would otherwise return "fail."

SpNet

- **Step 1:** We find the best two trees on each dataset under maximum likelihood,
- **Step 2:** For each dataset, we compute the strict consensus of the two trees, thus producing the trees t_1 and t_2, and
- **Step 3:** If t_1 and t_2 are compatible, we combine datasets and analyze the combined (i.e., concatenated) dataset using NJ, thus returning a tree. Else, we apply our algorithm for ConsTree-Network to t_1 and t_2. If we can, we return a network N with one reticulation (if trees T_1 and T_2 exist refining t_1 and t_2, respectively, contained within the network N); if no such network exists, we apply NJ to the concatenated dataset, and return a tree. (Alternatively, we could simply return "fail".)

Chapter 10

Experimental Studies

In this chapter, we evaluate (using simulations) the performance of four methods: SpNet [128] (described in Chapter 9), Maddison's method [106], NeighborNet [21], and neighbor joining [165]. We used the tools of [127] (described in Chapter 5) to generate random networks, as well as simulate sequence evolution down those networks. We use the split-based measure, m^{sp}, of topological accuracy (described in Chapter 6) to quantify error.

As described in Chapter 9, SpNet constructs networks in two steps: first, using a "base" method of choice, gene trees are inferred on each of the gene datasets, and then, those trees are reconciled into a network. Using different base tree reconstruction methods, as well as different methods for removing "weakly supported" edges, many variants of SpNet can be defined. In this chapter, we study some of those variants. We evaluate the different variants of SpNet in terms of their "detection" quality, i.e., how well the methods estimate the number of reticulation events in the input. Choosing the best performing variant, we finally compare the performance of SpNet to NeighborNet as well as NJ in terms of their "reconstruction" quality.

The results in this chapter clearly exhibit that our method, SpNet, greatly outperforms Maddison's method, as well as NeighborNet.

10.1 Experimental Settings

10.1.1 Methods

We study the performance of three methods, namely, Neighbor Joining, NeighborNet, and SpNet. SpNet subsumes Maddison's approach, and we show how the two compare, in terms of detecting the right level of reticulation in a dataset.

Neighbor Joining (NJ) [165] Neighbor Joining is one of the most popular distance-based methods. NJ takes a distance matrix as input and outputs a tree. For every two taxa, it determines a score, based on the distance matrix. At each step, the algorithm joins the pair with the minimum score, making a subtree whose root replaces the two chosen taxa in the matrix. The distances are recalculated to this new node, and the "joining" is repeated until only three nodes remain. These are joined to form an unrooted binary tree. NJ takes $O(n^3)$ time, and we used PAUP* to run the method.

NeighborNet (NNet) [21] NeighborNet is an $O(n^3)$ method for constructing phylogenetic networks. Like NJ, it iteratively selects pairs of taxa to group together, but it does not join them immediately. Rather, at a later stage, it agglomerates pairs of pairs which share one node in common. NeighborNet generates a *circular split system* [10] rather than a hierarchy or a tree, which can subsequently be represented by a planar *split graph* [39]. In these graphs, bipartitions of *splits* of the taxa are represented by classes of parallel lines, and conflicting signals or incompatibilities appear as boxes. The authors of NeighborNet suggest this method can be used to detect complex evolutionary processes like recombination, lateral transfer and hybridization. We used the linux 1.2 version of the NeighborNet software package.

SpNet We implemented various versions of SpNet, depending on the reconstruction method we used to obtain the single trees and the method we used to obtain the consensus tree. For the former, we used Maximum Parsimony (MP) and Maximum Likelihood (ML), and for the latter we used strict consensus (SC). After SpNet obtained consensus trees on the datasets, it inspected the two trees (for the two sequence datasets). If the method detected that the two consensus trees were compatible, it concatenated the sequences and ran NJ on the concatenated dataset. If the two consensus trees were not compatible but contained two binary trees T_1 and T_2 such that $d_{rSPR}(T_1, T_2) = 1$, then it constructed a network with a single reticulation event (using the linear-time algorithm of [128], which is described in Chapter 9). In order to enable us to compare SpNet to other methods, which always return output, we modified SpNet as follows: in the event that SpNet would return "fail" rather than a tree or network, we instead return the neighbor joining tree of the concatenated sequences.

10.1.2 Tools and Measures

We used the tools of [127] (described in Chapter 5) to generate random networks, as well as to simulate the evolution of DNA sequences down those networks. To quantify the error rate of methods, and compare their topological accuracy, we used the split-based false positive and false negative rates (described in details in Chapter 6). The reason behind choosing split-based comparison (rather than tree-based or tripartition-based) is that the output of NeighborNet is a set of splits, rather than a phylogenetic network. Further, although split-based comparison of phylogenetic networks may not satisfy desired properties in all cases, it does satisfy all those properties under our experimental settings (0 and 1 reticulation events, and no taxon sampling).

10.1.3 Simulation Parameters

Diameters. We ran our studies on random networks with diameter 2; in order to obtain networks with other diameters, we scaled the edge lengths by scaling factors of 0.05, 0.1, and 0.25, producing networks with diameters of 0.1, 0.2, and 0.5, respectively.

Deviation factor. To deviate the networks from ultrametricity, we used a deviation factor 2: for each edge, we used a random variate x in the range $[-\ln 2, \ln 2]$ and multiplied the edge length by e^x.

Numbers of hybrids. NNet handles any number of reticulation events, whereas SpNet, at this stage, handles only the cases of 0 and 1 reticulations. Due to this limitation, we generated only networks with 0, 1, and 2 hybrids.

Numbers of taxa. We generated networks with 10 and 20 leaves, one network for each combination of diameter, number of taxa, and number of hybrids.

Rooting the networks. Our method, SpNet, assumes rooted trees and networks. To obtain rooted networks and trees, we used an outgroup in each network.

Sequences. We then evolved sequences on these networks using the GTR+Γ (with invariable sites) model of evolution. We used the parameter settings of [201], which are:

- Shape parameter for Γ: 0.8168
- Proportion of invariable sites: 0.1
- Base frequencies: A:0.1776 C:0.3336 G:0.2595 T:0.2293
- Rate matrix: A$\rightarrow$C:3.297 A$\rightarrow$G:12.55 A$\rightarrow$T:1.167 C$\rightarrow$G:2.060 C$\rightarrow$T:13.01 G$\rightarrow$T:1.00

We generates sequences of lengths 500, 1000, 2000, and 4000.

Numbers of datasets. For each combination of sequence length and network settings, we generated 25 sets of sequences.

Software. We used PAUP* [181] for NJ and ML, and the NeighborNet software package (the linux 1.2 version); we implemented our method, SpNet, using C++ and the LEDA library of data structures and algorithms [2].

Obtaining the ML trees in PAUP* To obtain the best 100 ML trees, we first ran a quick parsimony search to get 100 initial trees, and then, a search (using TBR branch swapping) was used to get to the best possible trees. The search was limited to five minutes for each dataset. The following is the PAUP block we used.

```
set monitor=yes maxtrees=100 increase=no criterion=parsimony;
hsearch start=stepwise addseq=random nreps=100 swap=tbr hold=1
nchuck=1 chuckscore=1;
filter best=yes;
set criterion=likelihood maxtrees=100 increase=no;
lset nst=6 rmatrix=(3.297 12.55 1.167 2.060 13.01)
basefreq=(0.1776 0.3336 0.2595) rates=gamma shape=0.8168
Pinvar=0.1 rescale=100;
hsearch start=current swap=tbr nchuck=100 nbest=100 chuckscore=no
timelimit=300;
```

10.2 Results and Analyses

In this section, we report on the empirical performance of our method, SpNet, as it compares to NNet. The section is organized into two parts. In the first part, we show the significant improvement SpNet achieves over Maddison's approach, in terms of detecting the number of reticulation events in the input dataset. Then, choosing the best performing version, we show that SpNet greatly outperforms NNet.

10.2.1 Detection Quality

In this section, we study the performance of separate analysis methods in terms of their *detection quality*. In other words, we want to see how well methods can estimate the number of reticulation events in the evolutionary history of a set of sequences.

We conduct the performance study in three steps:

Step 1: Compare the performance of SpNet on ML and MP trees. We observe that both types of trees give similar performance, under the conditions of our simulations.

Step 2: Compare the performance of Maddison's method on ML and MP trees.

Step 3: compare the best performing methods from steps 1 and 2.

As mentioned, the quality of methods is assessed in terms of their *rate* *of correct predictions*, which will be the values on the y-axes of the figures in this section. We now describe how this rate is computed for a method M.

Let N be a random model network that we generate in the study, and let i be the number of reticulation events in N.

- If $i = 0$, then N is a tree, and evolve two sequence datasets $\mathcal{S}_1$ and $\mathcal{S}_2$ down the same tree. Method M makes the correct prediction if it detects that both $\mathcal{S}_1$ and $\mathcal{S}_2$ have an identical evolutionary history.

- If $i = 1$, then N contains two trees T_1 and T_2. We evolve one sequence dataset $\mathcal{S}_1$ down T_1, and another sequence dataset $\mathcal{S}_2$ down T_2. Method M makes the correct prediction if it detects that the two sets have a phylogenetic network with one hybrid as their evolutionary history.

- If $i = 2$, then N contains two trees T_1 and T_2 whose SPR distance is 2. We evolve one sequence dataset $\mathcal{S}_1$ under T_1, and another sequence dataset $\mathcal{S}_2$ under T_2. Method M makes the correct prediction if it detects that the two sets have a phylogenetic network with at least two hybrids as their evolutionary history.

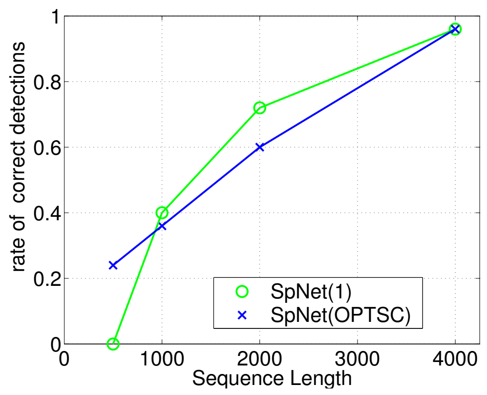
Since we generate 25 pairs of datasets in our simulations, we count how many times method M makes the correct prediction. The rate of correct predictions is that number, normalized by 25. This rate labels the y-axes in the figures of this section.

10.2.1.1 SpNet: ML vs. MP Trees

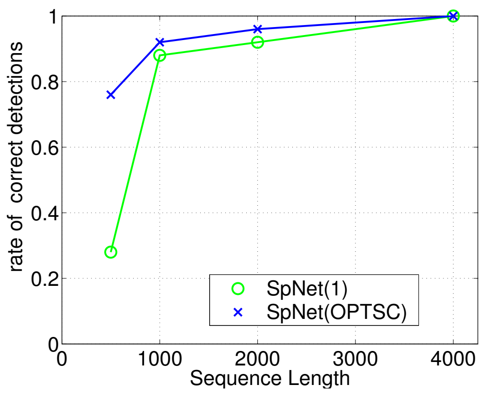
Figures 10.1(a) and 10.1(b) show the performance of SpNet using ML and MP trees, respectively. For the case of ML trees, the trees that were used in the strict consensus computation are the first best ones that led to one missing edge in the strict consensus tree. For the case of MP trees, all optimal trees found were taken into the strict consensus tree computation.

In the case where the model phylogeny is a tree, SpNet's performance is similar, whether MP or ML trees are used. In particular, the method's performance converges to the same rate of correct predictions at sequence length 4000. In the case of model networks with a single hybrid, SpNet's performance, especially for sequences longer than 1000 nucleotides, is the same using both MP and ML trees. For the case of model networks with two hybrids, SpNet, using ML trees, always detected that there were more than a single hybrid, whereas, using MP trees, the performance is slightly worse.

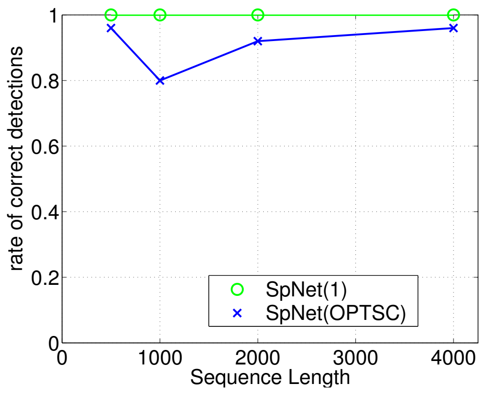
Note that, in order to obtain strict consensus of ML trees such that the strict consensus tree is missing one edge, exactly two ML trees are needed. However, in the case of MP trees, and since we use *all* optimal trees, the strict consensus tree may lack resolution more than that of the ML strict consensus tree. Hence, the strict consensus tree when using all optimal MP trees may be less resolved than the strict consensus tree of the two best ML trees. Hence, considering a pair of strict consensus trees of all optimal MP trees may underestimate the number of reticulation events.



(a) tree model phylogeny



(b) 1-hybrid model phylogeny



(c) 2-hybrid phylogeny

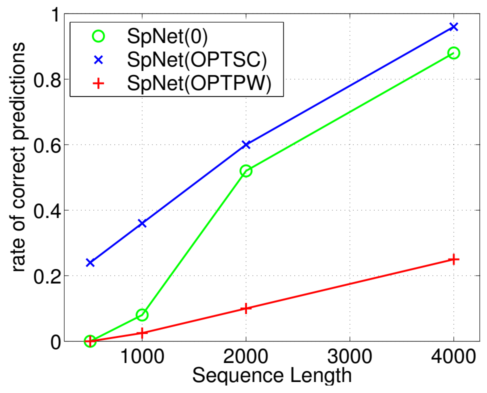
Figure 10.1: SpNet(1): SpNet run on strict consensus of ML trees, where the strict consensus trees are missing one edge each. SpNet(OPTSC): SpNet run on strict consensus of all optimal MP trees. Settings: 20-taxon networks, 0.1 scaling factor. The rate of correct predictions is the number of times the method predicted the correct number of reticulation events, normalized by 25—the total number of runs.

10.2.1.2 Maddison's Method: ML vs. MP Trees

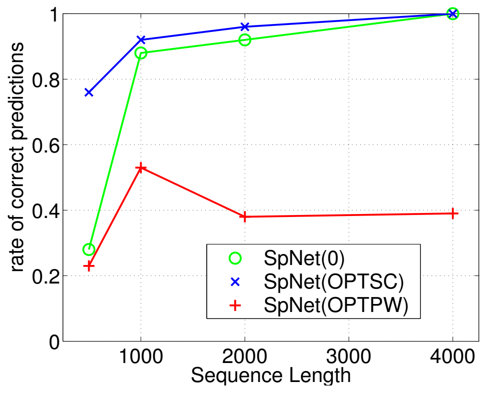
In this section we examine the performance of Maddison's method on ML and MP trees. In the former case, ML heuristics return a single optimal tree, which was used in Maddison's method; we call this method SpNet(0). In the latter case, and since MP optimizes a discrete criterion, more than one optimal tree may be found. In this case, Maddison's method is run on a pair of trees, each from the set of all optimal trees computed on the sequence dataset; we call this method SpNet(OPTPW). Since it is commonplace in phylogenetics to use the strict consensus of all optimal MP trees, we also compare the performance when using the strict consensus of *all* optimal MP trees found; we call this method SpNet(OPTSC). Figure 10.2 shows the results on 20-taxon networks.

The figure shows the Maddison's approach performs much better when using the best ML tree found than when considering pairwise comparison of optimal MP trees. Nonetheless, the figure shows that, when taking the strict consensus of all optimal MP trees, the performance improves drastically. One explanation for this is that the true tree is not any of the optimal MP trees, but rather it is a refinement of their strict consensus. Actually, this further supports the rationale behind our method, SpNet.

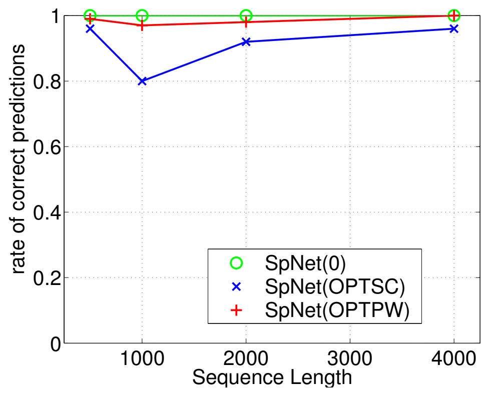
Further, using the strict consensus of all optimal ML trees yields better results than using the best ML trees in the cases of model trees and model networks with a single hybrid. In the case of model networks with two hybrids, using strict consensus of all optimal MP trees does not perform as well. An



(a) tree model phylogeny



(b) 1-hybrid model phylogeny



(c) 2-hybrid model phylogeny

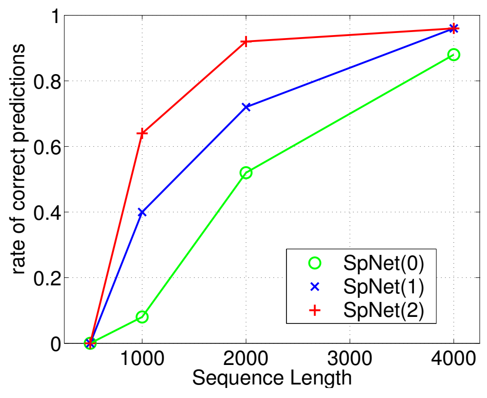
Figure 10.2: A comparison of Maddison's approach (SpNet(0)) on the single best ML trees, to an extension of Maddison's approach where we find all best MP trees and analyze them pairwise (SpNet(OPTPW)), and to SpNet run on the strict consensus of all the best MP trees (SpNet(OPTSC)). Settings: 20-taxon networks, 0.1 scaling factor. The rate of correct predictions is the number of times the method predicted the correct number of reticulation events, normalized by 25—the total number of runs.

explanation for this is that in the case of model networks with two hybrids, the strict consensus may lack resolution to the extent that the trees get closer (in terms of SPR distance), and hence the method starts under-estimating the number of hybrids in the model phylogeny.

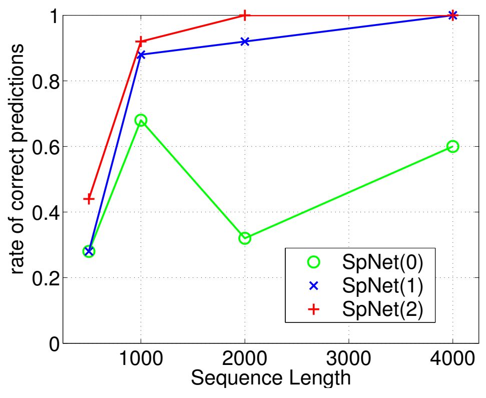
10.2.1.3 Maddison's Method vs. SpNet

In this step, we look at what happens when we change the resolution of the strict consensus trees, for the case where the strict consensus trees are based upon ML analyses. Maddison's method is obtained by using fully-resolved consensus trees, i.e., the best ML trees found. We study the effect of changing the resolution by using additional sub-optimal ML trees. Figure 10.3 shows the results on 20-taxon networks.

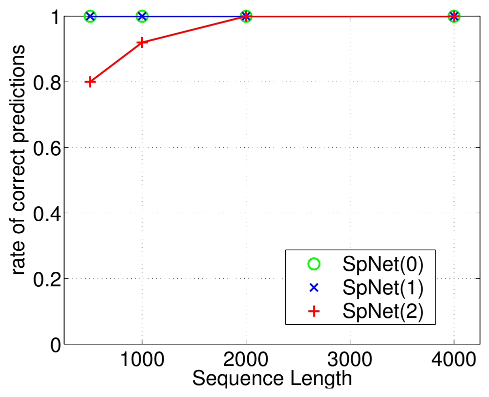
Figure 10.3 shows the improvement of our method, where we allow for one missing edge in the strict consensus tree (SpNet(1)) or two missing edges in the strict consensus tree (SpNet(2)), over Maddison's approach (SpNet(0)), where only the single best trees are used. However, our studies show that decreasing the resolution below a certain threshold starts leading to an under-estimation of the number of hybrids in the model phylogeny. The graphs show, as well, that SpNet rate of correct predictions converges to 1 as the sequence length grows; we did not observe the same for Maddison's method, especially when the model phylogeny contained a single hybrid.



(a) tree model phylogeny



(b) 1-hybrid model phylogeny



(c) 2-hybrid model phylogeny

Figure 10.3: SpNet run on the strict consensus of trees obtained in a maximum likelihood (ML) analysis. SpNet(i) indicates that i edges are missing from the strict consensus tree; hence SpNet(0) is Maddison's method. Settings: 20-taxon networks, 0.1 scaling factor. The rate of correct predictions is the number of times the method predicted the correct number of reticulation events, normalized by 25—the total number of runs.

10.2.2 Reconstruction Quality

Since SpNet (using strict consensus of ML trees) performs best in its class of methods based on separate analysis, we now compares its performance against NNet. In this section, we study the performance of methods in terms of their reconstruction quality, i.e., in terms of the topological accuracy of their output. As mentioned before, we study the topological accuracy based on the splits induced by networks.

In Figures 10.4 and 10.5, for SpNet, we used the strict consensus tree of the two best ML trees, so that the strict consensus tree was missing one edge in the former case, and missing two edges in the latter case. Both Figure 10.4 and Figure 10.5 show that all three methods (NeighborNet, SpNet, and Neighbor Joining) can be distinguished primarily on the basis of their false positive rates. That is, all three have excellent false negative rates (less than 5%) on trees and very good false negative rates (less than 10%) on networks with one reticulation, when given long enough sequences; however, NeighborNet has very poor false positive rates on both trees and networks with one reticulation, even at very long sequences (4000 nucleotides). In particular, our method, SpNet, has an excellent (i.e., very low) false positive rate, but NeighborNet has a very high false positive rate, even at long sequences. Furthermore, NeighborNet's false negative rate is quite comparable to that of SpNet, and so the huge difference in false positive rates is very important.

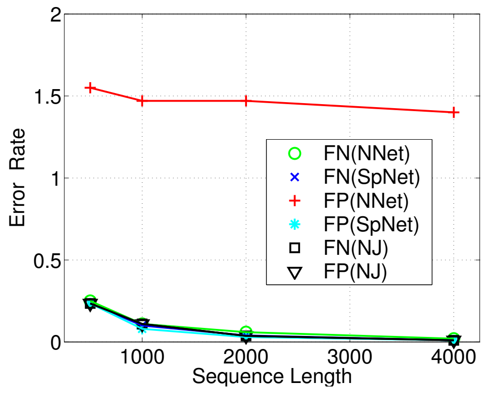
Since SpNet uses Neighbor Joining (NJ) to analyze datasets whenever it cannot infer a network with a single reticulation, a comparison between

SpNet and NJ is worth making. On trees they have essentially identical performance, as expected. On networks with a single reticulation, however, their performance is distinguishable: SpNet has an almost 0% false positive rate, which means that it produces a network with essentially no false edges, while NJ has (in these experiments) a false positive rate that is approximately 5%. The two methods have very close false negative rates. Thus, SpNet's performance on networks with one reticulation produces networks which are with some reliability *contractions* of the true network, while NJ's performance does not have the same reliability. The figures clearly show that NNet greatly overestimates the number of reticulation events, since it infers too many false positive splits. NNet does not seem to differentiate between a tree model phylogeny and a 1-hybrid-network model phylogeny, since the FP rates are similar in both cases. The graphs also show that the number of splits in the NNet output in our studies was about 150% of that of the true splits.

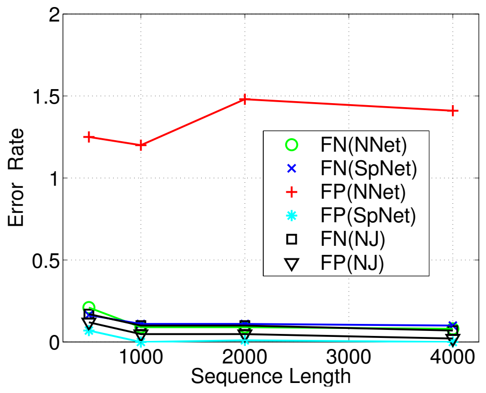
10.2.3 Major Challenge: Detection

Our experimental studies show that (1) our method clearly outperforms Maddison's method, and (2) a separate analysis approach significantly outperforms the NNet combined analysis approach (with the latter greatly overestimating the number of true reticulation events).

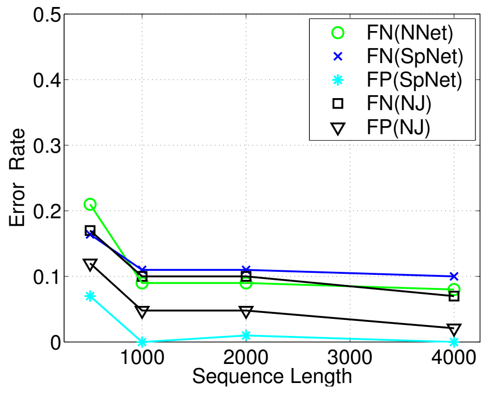
A major challenge that arises from this study is: how to improve the detection quality while maintaining good topological accuracy in the reconstructed networks?



(a) tree model phylogeny

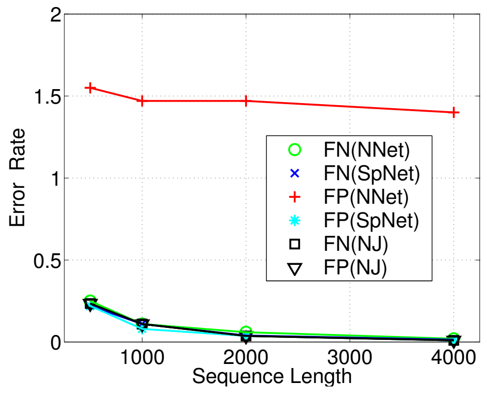


(b) 1-hybrid model phylogeny

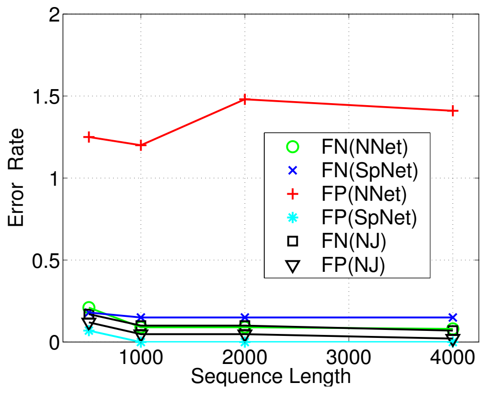


(c) 1-hybrid model phylogeny

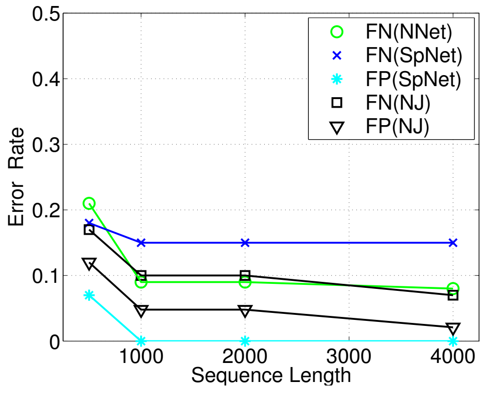
Figure 10.4: FN and FP error rates of NeighborNet (NNet) and SpNet on 20-taxon networks, with 0.1 scaling factor, with tree model phylogeny in (a), and 1-hybrid network in (b). The graph in (c) shows the results without the FP rate of NNet. SpNet was run on strict consensus trees of ML trees such that the strict consensus trees were missing one edge.



(a) tree model phylogeny



(b) 1-hybrid model phylogeny



(c) 1-hybrid model phylogeny

Figure 10.5: FN and FP error rates of NeighborNet (NNet) and SpNet on 20-taxon networks, with 0.1 scaling factor, with tree model phylogeny in (a), and 1-hybrid network in (b). The graph in (c) shows the results without the FP rate of NNet. SpNet was run on strict consensus trees of ML trees such that the strict consensus trees were missing two edges.

.

Figure 10.3 shows that the detection quality improves as the number of missing edges in the strict consensus tree increases, yet at some point the quality starts decreasing (when too many edges are missing). Further, we observed that the more edges are missing, the lower the topological accuracy becomes. Investigating different techniques of removing weakly supported edges is in order. Such techniques include non-parametric bootstrapping and Bayesian analysis techniques.

Part III

Phylogenetic Networks in Historical Linguistics

Chapter 11

Language Evolution

11.1 Introduction

Languages differentiate and divide into new languages via a process similar to how biological species divide into new species: communities separate (typically geographically), the language changes differently in each of the new communities, and in time people from separate communities can no longer understand each other. While this is not the only means by which languages change, it is this process which is referred to when we say, for example, "French and Italian are both descendants of Latin." The evolution of related languages is mathematically modeled as a rooted tree in which internal nodes represent the ancestral languages and the leaves represent the extant languages.

Reconstructing this process for various language families is a major endeavor within the historical linguistics community, but is also of interest to archaeologists, human geneticists, and physical anthropologists, for example, because an accurate reconstruction of how certain languages evolved can help answer questions about human migrations, the time that certain artifacts were developed, when ancient people began to use horses in agriculture, the identity of physically European mummies found in China, etc. (see in particular [107, 110, 160, 191]).

Careful scholarship over the last century has determined critical features and patterns that, combined with a statistical analysis, can be used to establish that languages share a common ancestor; examples of these features are shared idiosyncrasies in the grammars, shared idiosyncratic sound changes, and patterns of recurrent sound correspondences between words of the same meaning in different languages. Extending this fundamental statistical analysis, two techniques (the *comparative methods* [78] and *subgrouping through shared innovations*) have been developed which enable linguists to infer greater information about relatedness and properties of ancestral languages, and - to a limited extent - subgrouping as well (see [78, 120]). These techniques have established all known linguistic families and subfamilies, and are the basis of historical linguistic scholarship. Known families presently number close to 300, though ongoing comparative work on the languages of New Guinea and of South America - two of the linguistically most diverse and least described places on earth - may reduce this total to as low as 200.

Although these techniques provide firm evidence of relatedness between languages, they are significantly limited. The first limitation is that they have so far provided only limited information about subgrouping *within* sets of related languages. In particular, the problem of recovering detailed aspects of the branching pattern (i.e., the evolutionary tree for the family) has not always been possible using these techniques. Consequently, debates about the branching pattern for even the most well-established families (e.g., Indo-

European) have been significant, heated, and long-lasting. An even greater limitation of these methods is that they fail to indicate relationships between groups of languages when the most recent common ancestor lies more than approximately 6,000-8,000 years in the past.

Various researchers [35, 36, 58, 182] have noted that if communities are sufficiently separated after they diverge, then the inference of the phylogeny (i.e., evolutionary tree) for the languages can be inferred by comparing the characteristics of the languages (grammatical features, regular sound changes, and cognate classes for different basic meanings), and searching for "perfect phylogenies." However, the problem of determining if a perfect phylogeny exists, and then computing it, is NP-hard [17]. Consequently, efficient techniques for the inference of evolutionary trees for language families were not easily obtained. In the 1990's, various fixed-parameter approaches for the perfect phylogeny problem were developed (although inspired by the biological context rather than the linguistic one, see [47]). Subsequently, Ringe and Warnow worked together to fully develop the methodology (character encoding and algorithmic techniques) needed to apply these algorithms to the Indo-European language family. Their initial test of the methodology largely supported the claim that a perfect phylogeny should exist. Largely, but not entirely, because of the "Problem with Germanic". The Germanic family seemed to introduce non-treelike behavior, evidently acquiring some of its characteristics from its neighbors rather than (only) from its direct ancestors. Consequently, while the methodology seemed very clearly heading in the right direction, and even seemed to potentially answer many of the controversial problems in IE evolution (see [156, 159, 182, 188–190]), it became necessary to extend the model to address the problem of how characters evolve when the language communities remain in significant contact.

11.2 Historical Linguistics Data and Methods

11.2.1 Characters

Natural languages are classified based on data that is encoded in the form of characters. The various types of characters that are used in historical linguistics are [157]:

1. Phonological characters: These characters encode regular "sound changes". Changes in pronunciation in any given line of linguistic descent are overwhelmingly regular. It follows that if the contrast between two distinctive sounds ("phonemes") of a languages is lost, it can never be reestablished in its original distribution; in other words, *phonemic mergers are irreversible.* This is very useful since it imposes directionality on the evolutionary tree of the languages and helps determine where the root of the tree must lie. Unfortunately, though the set of sound changes in each line of descent is different, specific individual sound changes recur with some frequency (because all languages respond to the same phonetic pressures); this leads to the problem of *independent parallel development.* Historical linguists get around the problem by using groups of logically unrelated sound changes rather than single changes as characters. Since

a particular sound change (or a group of them) either has or has not occurred in the prior history of a language, phonological characters normally have only two states.

2. Lexical characters: These characters are up as meanings on a basis wordlist. For each languages the usual words are listed. Languages are then assigned the same state of a character if they exhibit “cognates” in that meaning. Cognates are words that have been inherited by two or more languages from their least common ancestor by direct linguistic descent alone (with no borrowing from other languages). Cognates are recognized by determining whether they can each be derived from the same word in the parent languages (protoform) by the sound changes peculiar to each line of descent.

3. Morphological characters: These characters are generally grammatical features of the languages. These are like lexical characters except that the items among which linguists search for cognates are inflectional affixes rather than vocabulary items.

11.2.2 The Comparative Method

One of the methods that have been used for a long time for interpreting historical linguistic character data is the *Comparative Method* [78]. This methods is used to verify relatedness between languages and to infer features of the ancestral languages of a group of related languages. The comparative method has two steps:

Step 1: Establishing sound correspondence in a set of related languages. In this step, words in the languages are compared for the same meanings, and patterns of sound correspondence between pairs of languages are observed.

Step 2: Establishing cognate classes. In this step, rules that explain all the sound correspondences are inferred and then used to establish the cognate classes for each lexical character. Let w_1 and w_2 be two different words from two different languages L_1 and L_2, respectively. w_1 and w_2 belong to the same cognate class if there is a common ancestor L of L_1 and L_2 and a word w in L such that both w_1 and w_2 can be derived from w in L through a series of sound change rules specific to the two languages. For example, using sound change rules, it can be shown that French *champ* and Italian *campo* are both descendants of Latin *campus*; thus the two words belong to the same cognate class [198].

Using the comparative method, raw data is transformed into mathematical encoding, where each character partitions the set of languages into cognate classes; those cognate classes correspond to character states.

11.2.3 The Warnow-Ringe Model for Trees

Ringe and Warnow [190] mathematically modeled the traditional techniques for comparing languages by imposing sets of constraints on the topology of the evolutionary tree (under the assumption that a tree is an appropriate graph-theoretic model of the evolutionary history of the dataset). The resulting model turns out to be very close to a classical model of biological evolution, called *Character Compatibility* or *Perfect Phylogeny*.

The character compatibility approach, although popular at earlier points in history for phylogenetic reconstruction in biology, is rarely used these days. However, it is a natural model for historical linguistics, as it follows naturally from linguistic assumptions which are stated or implied by the comparative method and which have been tested and found utterly reliable for more than a century.

Consider the result of examining all the words for the semantic slot (i.e., narrowly defined meaning) "foot" within some collection of Indo-European languages. English *foot*, German *fuss*, French *pied*, and Spanish *pie* are all cognate, but Russian *noga* is not. Cognate judgments clearly impose a partition of the groups of languages into disjoint cognate classes; cognate judgments also imply that if two leaves in the tree have cognates for some semantic slot, then all nodes on the path between them also have cognates for that semantic slot. Thus, for every cognate class of every given semantic slot, the nodes of the tree which contain elements of that cognate class should form a *subtree* of the true tree (i.e., the set of nodes with members of that cognate class are connected). Equivalently, it should be possible to remove a subset of the edges of the tree (but not their endpoints) so that each resultant subtree consists of exactly those nodes which contain cognates for that semantic slot. When a given semantic slot has this property on a tree, it is said to be "compatible" with the tree (recall Definition 2.3.3 in Chapter 2).

More generally, the classical methodology also provides ways of describing partitions of the languages for other kinds of linguistic properties, including sound changes, grammatical features, etc. Each such property can be organized as "qualitative character" which defines a partition of the languages into two or more "states". The classical methodology therefore implies that each qualitative character should be compatible with the true tree (when a tree is an appropriate model.

11.3 Character Compatibility and Perfect Phylogeny

Experience shows that it is easy to construct a comparative dataset using only qualitative characters that evolve without backmutation or parallel evolution; see the disucssion in [158]. What this means is that when the state of the qualitative character changes in the evolutionary history of the set of languages, it changes to a state which does not exist anywhere else on earth at that time, nor has it appeared earlier. We now formalize this concept mathematically.

Suppose T is a rooted binary tree describing the evolution of a set S of languages, and that a qualitative character α is defined for each of the languages in S. Thus, α is a function such that $\alpha : S \rightarrow Z$, where Z denotes the set of integers (i.e. each integer represents a possible state for α).

We say a qualitative character α is compatible (or "convex") on T if we can extend α to every internal node of the tree T, thus defining a qualitative character α', so that for every state, the nodes having that state form a connected subgraph of T. (In other words, $\forall z \in Z, \{v \in V(T) : \alpha'(v) = z\}$ is connected.)

It should be clear now why compatibility is unlikely in biology; qualitative characters in biomolecular sequences are the single positions within multiple alignments, and having only a fixed number of states (4 for DNA and RNA, 20 for amino-acids), compatibility of such qualitative characters on the true tree is highly unlikely. Even in morphology, features such as wings arose many times in evolutionary history.

However, Ringe and Warnow postulated that *all* properly encoded qualitative characters for the Indo-European data should be compatible on the true tree, if such a tree existed. They were right to a first approximation, but not exactly. Instead, their initial analysis revealed that almost no such tree existed; that is, no tree existed on which all the qualitative characters they considered were compatible. Such a tree, if it existed, would have been called a *perfect phylogeny*:

Definition 11.3.1. Let C be a set of qualitative characters defined on a set S of languages. A tree T is a **perfect phylogeny** for C and S if every qualitative character in C is compatible on T.

Determining if a perfect phylogeny exists is an NP-complete problem, yet solvable in polynomial time if any of the three parameters to the problem (the number of taxa, number of qualitative characters, or the maximum number of states per qualitative character) is bounded by a constant. See [5, 91, 117] for these results.

11.4 Non-treelike Evolution of the IE Languages and the Need for Phylogenetic Networks

The initial analysis of the Indo-European data done by Ringe, Warnow and Taylor in [189] demonstrated that the IE linguistic data is, nevertheless, "almost perfect": they found a tree on which the proportion of compatible characters to incompatible characters was enormous. (Even this was quite surprising; the existence of a tree on which a large proportion of characters is compatible is extremely unlikely in biological data analysis.) This suggested that the basic approach was a good one, but that the model had to be extended.

Largely the problem seemed to be the Germanic subfamily, which seemed to have remained in contact with other languages so that a tree was an inappropriate model of evolution. That is, the IE family must have evolved other than through clean speciation. When the group of languages contains some pairs of related dialects which have evolved in close contact with each other, the ability of the linguist to detect borrowing is greatly reduced. More precisely, whereas borrowing between clearly different speech forms is tightly constrained and clearly different from change in normal genetic descent, borrowing between closely related dialects is largely unconstrained and often indistinguishable from changes which could in principle be of very different types

[95, 158, 164]. In this case, a tree model is inappropriate, and the evolutionary process is better represented as a "network".

In Chapter 12 we show how we have extended the model of character evolution on trees to produce a model of how characters should evolve down networks. That is, we show how we can define "perfect phylogenetic networks" by extending the perfect phylogeny concept to the network case. Finally, in Chapter 13 we report on a new analysis of the IE dataset using our new methodology.

Chapter 12

Perfect Phylogenetic Networks

The evolution of a family of languages, when the languages evolve via clean speciation, is modeled as a rooted tree (typically bifurcating), so that internal nodes represent ancestral languages, and leaves represent the languages under study. In this case, it is reasonable to orient edges from the ancestral languages towards the descendent languages, so that all the edges in the tree are directed (from the root towards the leaves); these directions are consistent with the flow of time. However, when languages evolve in such a way as to be able to borrow from each other when they have not yet diverged very much, then additional edges are needed in order to show how characters evolve in the network. Since these edges represent exchanges between languages due to contact, we call them "contact edges". Furthermore, they are "bi-directional", so that characters can be borrowed in both directions. Such a graphical representation is called a "network" rather than a tree, to reflect the inclusion of these additional edges. Figure 12.3 gives an example of one such network.

We begin our discussion of how to extend the notion of character compatibility from trees to networks by observing that each character will evolve down one of the "trees contained in the network." Figure 12.3 shows a net-

	c_1	c_2
L_1	0	0
L_2	0	0
L_3	1	0
L_4	1	1
L_5	1	1

(a)

	c_1	c_2	c_3
L_1	0	0	0
L_2	0	0	1
L_3	1	0	1
L_4	1	1	0
L_5	1	1	0

(b)

Figure 12.1: (a) Five languages $L_1, \ldots, L_5$, with two characters c_1, and c_2. (b) The same five languages with a third character c_3. There exists a perfect phylogenetic tree for the languages and characters in (a); however, there does not exist a perfect phylogenetic tree for the data in (b).

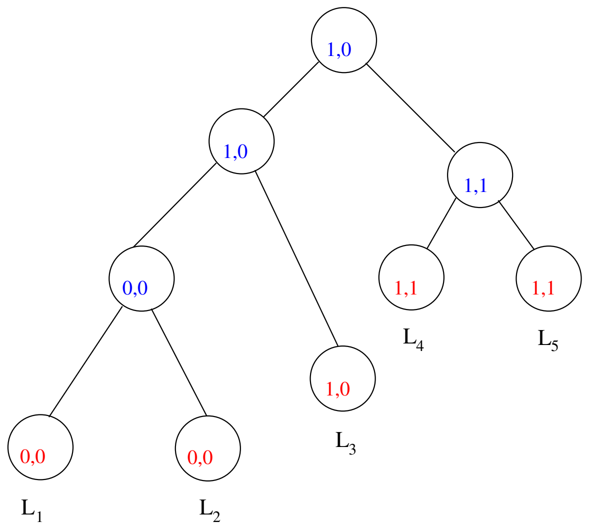


Figure 12.2: A perfect phylogeny T for the languages and character states of Figure 12.1(a).

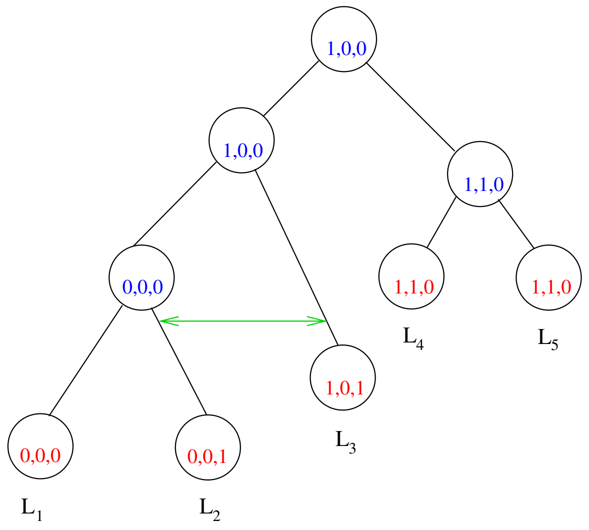


Figure 12.3: A perfect phylogenetic network N for the languages and character states of Figure 12.1(b).

work N that contains, in addition to the underlying tree, one contact edge between two reasonably closely related languages. The tree is characterized by a 3-tuple of characters c_1, c_2, c_3; each character has two states, 0 and 1. These are the languages and characters of Figure 12.1(b), for which there does not exist a perfect phylogenetic tree. The character states for each node are given; if this were a real example, the states at the terminal nodes ("leaves") would be coded from actual data, while those at the internal nodes would be inferred. Figure 12.4 shows the three possible trees within the network down each of which characters can evolve–that is, each of which potentially models the evolutionary history of one or more of the characters.

To motivate our model of character evolution down networks we begin

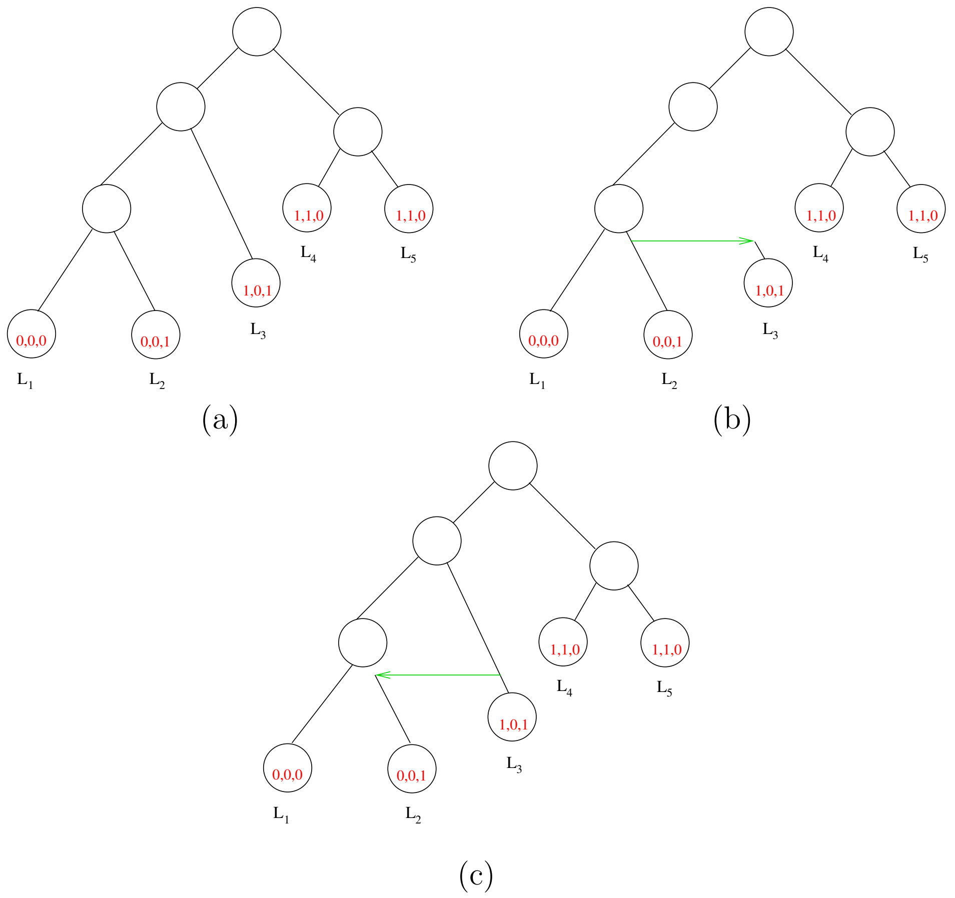


Figure 12.4: The three trees contained inside the network in Figure 12.3. While the network "reconciles" the evolutionary history of all three characters, each one of those characters actually evolved down exactly one of the three trees.

with the tree model. When trees are reasonable models of a language family's evolution we assume that qualitative characters are compatible with the tree: when a character changes state on an edge, it changes to a new state not yet in the tree (there being no backmutation, and parallel evolution having been excluded). But a network contains several evolutionary trees, and a character can evolve down any of them. We therefore say that a qualitative character c is "compatible" with a network N if c is compatible with at least one of the trees contained in the network.

The three characters of the network N shown in Figure 12.3 are compatible with N, since each of the characters is compatible with at least one of the three trees contained inside N: characters c_1 and c_2 are compatible with the tree in Figure 12.4(a), and character c_3 is compatible with both trees in Figures 12.4(b) and 12.4(c).

The assumptions inherent in the methodology of linguistic cladistics, when extended to the case where languages evolve on a phylogenetic network, imply that linguistic characters should be compatible on the true phylogenetic network. Just as in the case of trees, we will say that a network is a "perfect phylogenetic network" for a set of languages described by a set of qualitative characters if every character is compatible with the network (e.g., the network N in Figure 12.3 is a perfect phylogenetic network for the languages and characters in Figure 12.1(b)).

A perfect phylogenetic network can deviate from the tree model in a number of ways. The greater the number of characters that must evolve along lateral (i.e. contact) edges, the greater the deviation in terms of character compatibility; the greater the number of lateral edges, the greater the deviation in topological terms. Finally, the greater the number of borrowing events (i.e., character states transmitted on contact edges), the greater the deviation in terms of what might be called "loan parsimony"; this is not the same as either of the measures just mentioned, since a single item can be borrowed along more than one lateral edge.

We note that these different criteria measure different things, and hence different ways of deviating from a tree will be evaluated differently depending on the criterion chosen. For example, if the wave model is appropriate, we should not be able to find a clearly defined underlying genetic tree on which the vast majority of the characters are compatible; most characters may be involved in "borrowing", and so even if the number of contact edges is somewhat small (because constrained by geographical considerations, for example), the other two criteria should yield fairly large values. On the other hand, if the language family evolves in a mostly treelike fashion (most of the characters evolving down the underlying genetic tree), then the genetic tree should be discernible because of the compatibility pattern (i.e., there should not be two or more trees with a large percentage of the characters compatible on each). In such a case, there may be a fair number of contact edges and some characters which have a high propensity to be borrowed, and so those criteria (loan parsimony and the number of contact edges) might yield large values.

The debate about Indo-European's history has to some extent focused on these two extremes: the "wave model", in which there is no clear underlying genetic tree, and the "Stammbaum model", in which there is no significant borrowing. Our proposal explicitly takes account of intermediate possibilities, in which there is a clear genetic tree (with which a high proportion of characters is compatible) but some borrowing. Furthermore, our proposal allows for the deviation from a tree model to be measured along several partly independent parameters. (The number of contact edges and the number of incompatible characters are independent, but the third measure, loan parsimony, amounts to a combination of those two.) In this paper we both present this model, and attempt to discover where Indo-European (IE) fits within the model. Just how clearly can we identify an IE genetic tree? Is the evolution of IE largely treelike, or is a wave model really a better model? As we will see, our analysis shows dramatic support for the claim that Indo-European evolution is largely treelike: almost all (95%) of the characters evolve down our proposed genetic tree, and we only need three additional contact edges to explain all the data; thus all three criteria yield satisfyingly low scores. Finally, our proposed network is also largely consistent with known geographical and chronological constraints on IE linguistic history.

Since the simplest model is the most desirable, other things being equal ("Occam's Razor"), we will want to find a perfect phylogenetic network that optimizes all three mathematical criteria, with the smallest number of borrowing events, the smallest number of contact edges, and the highest percentage of characters compatible on the underlying genetic tree. Such a network would explain the evolution of all the characters (via genetic transmission and/or borrowing), and would not need to imply either parallel evolution or backmutation. It is worth noting that a perfect phylogenetic network always exists, because one can always construct a network in which all pairs of leaves are connected by contact edges; on such a network all characters are compatible. But since such a network fits all possible characters, it says nothing interesting about the evolutionary history of the dataset.

Finding the smallest number of borrowing events is obviously easier if one has first found (or estimated) the tree with which the largest number of characters is compatible and has added to it the minimum number of contact edges necessary to construct a perfect phylogenetic network. Accordingly our approach involves two steps:

- Given the set L of languages described by set C of qualitative characters, find or estimate the optimal "genetic tree" T for L. (If T is a perfect phylogeny, or deviates very little from a perfect phylogeny, it can be found; if it deviates too much from a perfect phylogeny, existing techniques may be insufficient to prove that the best tree discovered is in fact the optimal tree. See the discussion in [158].)
- If T is a perfect phylogeny, then return T. Otherwise, add a minimum number of contact edges to T to make it a perfect phylogenetic network.

(This is roughly similar to the approach of [7].) For example, the characters c_1 and c_2 in Figure 12.1(b) are compatible with the tree T in Figure 12.2, whereas character c_3 is not. By adding one contact edge to T, we obtain the network N of Figure 12.3, on which all three characters are compatible.

This is the approach that we used in this study in order to analyze the Indo-European dataset compiled by Ringe and Taylor. Because our data were close to tree-like, our analysis was able to complete in a reasonable amount of time (a few hours). In the next sections, we mathematically formalize the model and problems. We describe our analysis of the IE dataset in the next chapter.

12.1 The Mathematical Model

Our model of how languages evolve on networks references an underlying rooted tree (modeling "genetic descent") to which we then add bidirectional edges (modeling how linguistic characters can be transmitted through contact). Therefore, the underlying tree is rooted, and the edges of that tree can be naturally oriented from parent to child, whereas the additional edges are by design bidirectional, since contact between language communities can result in the flow of linguistic characters in both directions. We formalize this as follows:

Definition 12.1.1. A *phylogenetic network* on a set L of languages is a rooted directed graph $N = (V, E)$ with the following properties:

- $V = L \cup I$, where I denotes added nodes which represent ancestral languages, and

- E can be partitioned between the edges of a tree $T = (V, E_T)$, where $L \subset V$ are the leaves of T, and $E' = E - E_T$ are the "non-tree" edges. The edges in T are oriented from parent to child, and hence T is a directed, rooted tree. The edges in E' are bidirectional.
- N is "weakly acyclic", i.e., if N contains directed cycles, then those cycles contain only edges from E'.

We begin our discussion of how to extend the notion of character compatibility from trees to networks by observing that each character will evolve down one of the "trees contained in the network." See Figures 12.3 and 12.4 for a network with one contact edge and the three trees it contains. Note also the following:

Observation 3. *A network N with B non-tree edges induces at most 3^B trees.*

We now complete our extension of the character compatibility concept to networks.

Definition 12.1.2. Let $N = (V, E)$ be a phylogenetic network on L, $\mathcal{T}$ the set of trees induced by N, and let $c : L \rightarrow Z$ be a character. Then c is said to be *compatible* on N if c is compatible on at least one of the trees in $\mathcal{T}$.

For example, character c_3 in Table 12.1(b) is not compatible with the tree in Figure 12.2 (which is a network with no contact edges). However, all three characters in Table 12.1(b) are compatible with the network in Figure 12.3.

Theorem 17. *Let T be a phylogenetic tree on a set L of n languages, and assume that each language in L is assigned a state for the character α. We can test the compatibility of α on T in $O(n)$ time.*

Proof. Assume the states of α on the set L are $1, 2, \ldots, r$, for some integer r. We preprocess the input in order to compute the vector $c[1...r]$ defined by $c_i = |\{s \in L : \alpha(s) = i\}|$. Obviously we can compute this vector in $O(n)$ time.

Now, for each $i, 1 \leq i \leq r$, and each node v in the tree T, we define $B_i(v)$ to be

$$B_i(v) = \{x : \ x \text{ is a leaf of } T \text{ below } v \text{ and } \alpha(x) = i\}.$$

Note that if v is a node in T then $0 < |B_i(v)| < c[i]$ implies that in any labeling of the node v for which α is compatible, we must have $\alpha(v) = i$. Hence at each node v there is at most one state i satisfying this condition.

At each node v we will therefore compute the set $States(v)$ defined to be those state(s) i such that $0 < |B_i(v)| < c[i]$, as well as the value $|B_i(v)|$ for every $i \in States(v)$. If for any node v we have $|States(v)| > 1$, then we return "Incompatible", and exit; else, we return "compatible".

We now show how to compute this information. We do this from the bottom up, and it is trivial to compute these values for the leaves. So suppose v is a node in T and we have computed these values at its children, which are x and y. Note that the only candidates for elements of $States(v)$ must be drawn from $States(x) \cup States(y)$. For each $i \in States(x) \cup States(y)$, we set $|B_i(v)| = |B_i(x)| + |B_i(y)|$, and then compare this to $c[i]$ to see if we include i in $States(v)$. Since $|States(x) \cup States(y)| \leq 2$, we can therefore compute $States(v)$ in $O(1)$ time, and hence we can determine the compatibility of α on T in $O(n)$ time. $\square$

The assumptions inherent in the historical linguistic methodology, when extended to the case where languages evolve on a phylogenetic network, imply that linguistic characters should be compatible on the true phylogenetic network:

Definition 12.1.3. Let $N = (V, E)$ be a phylogenetic network on a set L, and C be a set of characters defined on L. N is called a Perfect Phylogenetic Network if all characters in C are compatible on N.

Now, suppose we are given a phylogenetic network. We need to determine if it satisfies the compatibility constraints:

Definition 12.1.4. Character Compatibility on Phylogenetic Networks (CCPN):

- **Input:** A phylogenetic network $N = (V, E)$ on a set L, and a set of characters C defined on L.
- **Question:** Is N a perfect phylogenetic network?

The CCPN problem is NP-hard in general, as we will now prove by a reduction from the Undirected Disjoint Connecting Paths problem.

Definition 12.1.5. UNDIRECTED DISJOINT CONNECTING PATHS

Input: graph $G = (V, E)$, collection of disjoint vertex pairs $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$.

Question: Does G contain k mutually vertex-disjoint paths, one connecting s_i and t_i for each i, $1 \leq i \leq k$?

The Undirected Disjoint Connecting Paths problem is NP-hard [57].

Theorem 18. *CCPN is NP-hard.*

Proof. Given a phylogenetic network N in which all nodes are labeled by the states of a character α, it can be checked in polynomial time whether α is compatible on N (by testing whether all nodes labeled by state i form a rooted connected component). Hence, CCPN is in NP. We now show that CCPN is NP-hard by a reduction from the undirected DISJOINT CONNECTING PATHS problem.

Given a graph $G = (V, E)$ and a collection of disjoint vertex pairs $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$, we build a phylogenetic network $N = (V', E')$ as follows:

- $V' = V \cup \{v' : v \in V\} \cup \{r, u, w\}$.

- $E' = E \cup E_0$,

where E_0 is the set of "tree edges":

$$E_0 = \{(r, v) : v \in V\} \cup \{(v, v') : v \in V\} \cup \{(r, u), (r, w)\}.$$

An edge (x, y) in E_0 is directed from x to y. The edges in E are the "non-tree" edges, and they are bidirectional. Node r is the root of the network. The set of leaves of the network is $V' \cup \{u, w\}$. In other words, the phylogenetic network N is comprised of an underlying tree $T = (V, E_0)$, and a set E of additional (bidirectional) contact edges (and set $V' - V$ of additional nodes that result from adding the contact edges). The resulting phylogenetic network N is weakly acyclic, since the underlying tree T is simply a star (whose center is node r).

The set C of characters has one character α. The states of α are set as follows:

- $\alpha(u) = \alpha(w) = 0$,
- $\alpha(s_i') = \alpha(t_i') = i$ for all $1 \leq i \leq k$, and
- $\alpha(v') = q_{v'}$, where $v' \neq s_i$ and $v' \neq t_i$ for all $1 \leq i \leq k$, and $q_{v'}$ is a natural number greater than k and which is unique to every such v'.

It is obvious that the resulting N is a phylogenetic network and satisfies the conditions of Definition 12.1.1. Figure 12.5(a) shows an example of a

"YES-instance" of the Disjoint Connecting Paths problem. The perfect phylogenetic network that results from the reduction on this graph is shown in Figure 12.6(a); Figure 12.6(b) gives the tree inside the network on which the character is compatible. Figure 12.5(b) shows an examples of a "NO-instance" of the Disjoint Connecting Paths problem. The phylogenetic network that results from the reduction on this graph is shown in Figure 12.7, and is not a perfect phylogenetic network.

We now prove that that G has k vertex-disjoint paths between every pair of vertices s_i and t_i (for all $1 \leq i \leq k$) if and only if the character α is compatible on network N.

For the "only if" part, let $p = (s_i, x_1, \ldots, x_l, t_i)$ be a path that connects s_i and t_i (p is vertex-disjoint with all other paths that connects the other pairs of nodes). In the resulting network N, $\alpha(s'_i) = \alpha(t'_i) = i$, and $\alpha(x'_j)$, for $1 \leq j \leq l$, is a unique number. The two leaves s'_i and t'_i are connected as follows: direct all edges on p away from s_i and towards t_i, and remove all tree edges incident into the nodes $x_1, \ldots, x_l, t_i$ (see Figure 12.6(b)). Finally, set $\alpha(s_i) = \alpha(t_i) = \alpha(x_j) = i$, for all $1 \leq j \leq l$. Since the path p is vertex-disjoint from the other paths connecting pairs of nodes, this labeling is well-defined, and clearly makes state i connected, for every $1 \leq i \leq k$. State 0 is connected via the root of the network.

For the "if" part, let N be a perfect phylogenetic network. Since there is a labeling of the nodes of N such that each state i, for $1 \leq i \leq l$, is connected, then there are k vertex-disjoint paths, each connecting a pair (s_i, t_i) of nodes

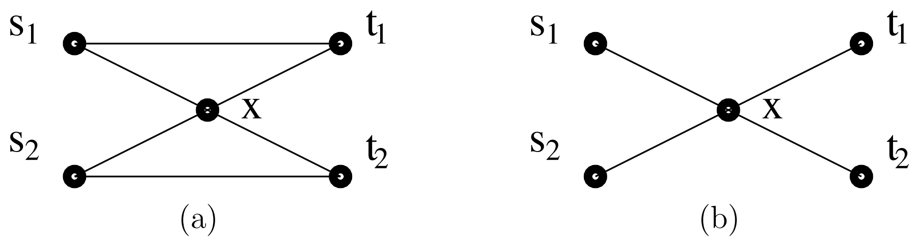


Figure 12.5: (a) A "YES-instance" of the Disjoint Connecting Paths problem: there are two vertex-disjoint paths, one connecting s_1 with t_1, and the other connecting s_2 with t_2. (b) A "NO-instance" of the Disjoint Connecting Paths problem: there do not exist two vertex-disjoint paths, such that one connects s_1 and t_1 and the other connects s_2 and t_2.

(the path that connects the pair (s_i, t_i) is formed of the nodes that are labeled i in N). This completes the proof, and hence CCPN is NP-complete. □

Nevertheless, for some special cases, the CCPN problem is solvable in polynomial time. For example, if N is a tree, and thus contains no non-tree edges, the problem is easily solved in polynomial time using standard techniques, and available in many phylogenetic software packages (e.g., PAUP* [181]). Furthermore, we may only wish to solve this problem for the case where the number of non-tree edges can be bounded by a small constant. In this case, we show that a polynomial time algorithm is achievable.

Theorem 19. *Let N be a phylogenetic network on a set L, and let C be a set of characters defined on L. We can determine if the set C is compatible on N in $O(kn3^B)$ time, where n is the number of leaves in N, $k = |C|$, and B is the number of non-tree edges in N.*

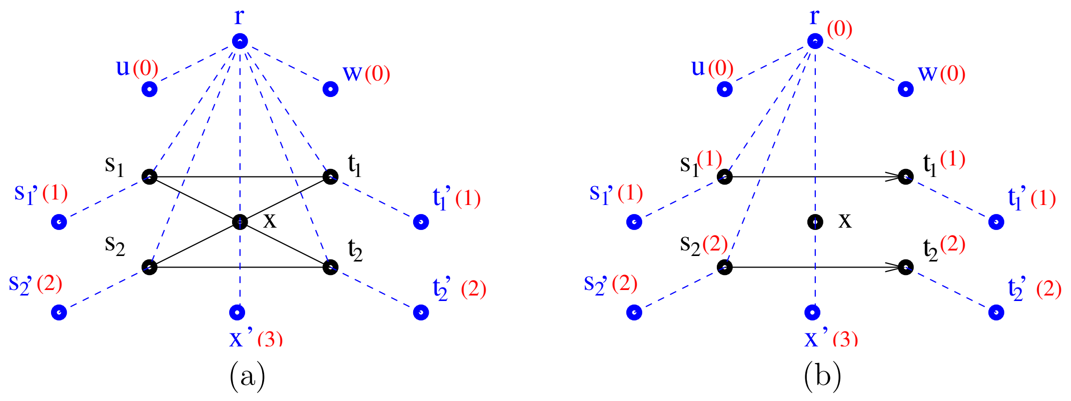


Figure 12.6: (a) The perfect phylogenetic network that is the output of the reduction in the proof of Theorem 18 for the graph in Figure 12.5(a). (b) A tree inside the network and on which the character is compatible. The internal nodes are labeled so as to make the character compatible. The dashed lines form the set E_0 of tree edges, while the solid lines denote the bidirectional contact edges.

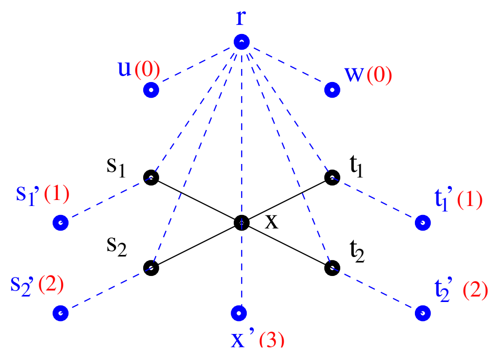


Figure 12.7: The phylogenetic network that is the output of the reduction in the proof of Theorem 18 for the graph in Figure 12.5(b). Clearly the character is not compatible on this network.

The proof of this theorem follows immediately from Definition 12.1.1, Observation 3, and Theorem 17. Hence, CCPN is *fixed-parameter tractable*, which means that running time of CCPN is bounded by a polynomial in the input whose degree does not depend upon the parameter B, although the leading coefficient does depend upon B [38]. This means that we can determine whether a phylogenetic network is perfect quite quickly, provided that the number of non-tree edges is small.

12.2 The PPN Reconstruction Problems

In general, we are not given the network directly, but need to find it. Two problems arise in this context: (i) given a set of characters, to find a network N on which each character is compatible, and (ii) given a tree T leaf-labeled by sequences of characters, to add a minimum number of edges to the tree, thus obtaining a network, so as to make all characters compatible. From an early "manual" analysis of the IE dataset, we found that adding at most four edges (between the Germanic subfamily and other subfamilies of IE) suffices to a obtain a phylogenetic network on which all characters were compatible. Hence, we expect the number B of non-tree edges to be reasonably small in practice. Therefore, the following is a natural optimization problem:

Definition 12.2.1. Minimum Size Perfect Phylogenetic Networks (MSPPN)

- **Input:** A set L of languages, set $\mathcal{C}$ of characters defined on L, and bound $B \in Z$,

- **Question:** Does there exist a phylogenetic network N on which each character in C is compatible, so that N contains at most B non-tree edges?

The MSPPN problem is also NP-hard, since the PP problem is a special case of it ($B = 0$). Consequently, bounding only the number of non-tree edges will not in itself lead to a polynomial-time solution. However, since PP is solvable in polynomial time for any fixed parameter, MSPPN may also be solvable in polynomial time if we can bound two or more of its parameters. Consider the following approach to solving MSPPN:

- **Step 1:** Solve maximum compatibility on the set C of characters, thus obtaining a set $\mathcal{T}$ of underlying phylogenetic trees on which a maximum cardinality subset of characters are compatible; let C_0 denote the characters that are incompatible on a tree $T \in \mathcal{T}$.
- **Step 2:** For each tree T in the set $\mathcal{T}$, add a minimum number of edges to T so as to make all characters in C_0 compatible on the resultant phylogenetic network N.

The second step involves solving an interesting problem, which we now formalize:

Definition 12.2.2. Minimum Increment to Perfect Phylogenetic Network (MIPPN)

- **Input:** Set C_0 of characters leaf-labeling a tree T

- **Output:** Network N formed by adding a minimum number B of edges to T so that C_0 is compatible on N.

For our purposes, we expect the parameters $|C_0|$ and B to be small, and hence we are interested in seeking algorithms whose running time is polynomial in n (the number of leaves in T), and which are fast for small values of B and/or small values of $k_0 = |C_0|$. For the case where B is small, the following theorem shows we can solve the MIPPN problem efficiently:

Theorem 20. *We can solve the MIPPN problem in $O(Bk_0 2^{4B} n^{2B+1})$ time, where n is the number of leaves in T, B is the minimum number of non-tree edges needed to obtain a perfect phylogenetic network, and k_0 is the number of characters.*

Proof. We will solve MIPPN by looking at all networks obtained by adding b bidirectional edges to T, where b ranges from 1 up to B, the first value for which one of the networks is a perfect phylogenetic network. The cost of this technique is clearly bounded by B times the cost of running the decision problem for B edges. A binary rooted tree T on n leaves has $2n-2$ edges. There are $\binom{\binom{2n-2}{2}}{B}$ possible ways of adding B edges to the tree T, which is $O(2^{2B}n^{2B})$. Each such possibility results in a network on n leaves, with B non-tree edges. From Theorem 2, it takes $O(k_0 n 3^B)$ to test the compatibility of the set of characters on each of those networks. Therefore, we can solve the problem in $O(Bk_0 2^{4B} n^{2B+1})$ time. □

As discussed above, for the IE dataset we study, most of the characters have evolved down the “genetic tree”, with few character states being borrowed along contact edges. Hence, an appropriate approach for analyzing the IE dataset is to first estimate the genetic tree, and then try to solve the MIPPN problem. We have implemented the algorithm given above, and analyzed the IE dataset; we report on our findings in the next chapter.

12.3 Related Work

12.3.1 The Answer-Set Programming Approach

Answer set programming [101, 111, 129] is a new form of declarative programming. It differs from traditional logic programming in that it represents solutions to a problem by “answer sets” rather than “answer substitutions”. Instead of Prolog, it uses *answer set solvers* – software systems capable of computing answer sets, such as CCALC [113], DLV [41], or SModels [130]. The idea of answer set programming is to represent a given computational problem as a logic program whose answer sets correspond to solutions, and to use an answer set solver to find an answer set for this program.

Various combinatorial search problems can be solved using answer set solvers. This has led to the applications of answer set programming to various fields, such as planning, graph theory, wire routing, product configuration, dynamic constraint satisfaction, model checking, reachability analysis, and logical cryptanalysis.

In [43], answer set programming was used to solve the MIPPN problem on the IE dataset, described in the next chapter. This approach took about a week of CPU time to find all solution, whereas our program (based on Theorem 20) took about eight hours to find the same set of solutions.

12.3.2 Perfect Phylogenetic Networks with Recombination

Perfect phylogenetic networks are relatively a new model, although Wang *et al.* [186] also study the problem of inferring *perfect phylogenetic networks with recombination.* Recombination operations originate from modeling mutations in DNA sequences. When recombination occurs, the evolutionary history of a set of species cannot be modeled by a rooted tree. Instead, it can be represented by a rooted *recombination topology* in which some nodes, called *recombination nodes*, have two parents. Similar to the perfect phylogeny problem, a recombination topology T has the property that its restriction on every equivalence class for a character state is connected. Such a recombination topology is called a *perfect phylogenetic network with recombination.*

We note that one difference between our definition of PPN and this definition is that the added non-tree edges to the original trees are bidirectional in our case, whereas in the case of Wang *et al.* those extra edges are unidirectional. However the problems are the same, otherwise.

In their paper, Wang *et al.* "proved" that the Minimum Size Perfect Phylogenetic Network problem (with respect to the binary case of PPN with recombination) is NP-hard, using a reduction from the *Subtree Prune and Regraft* (SPR) problem which they had proved to be NP-hard [72]. However, Allen *et al.* [6] showed that the NP-hardness proof of the SPR problem was flawed, and the SPR problem is still of unknown computational complexity. Therefore, the MSPPN problem is also still of unknown computational complexity.

The merged paths of a recombination node are the two paths from the node to the *least common ancestor* of the parents of the node. Wang *et al.* described a special case of the problem where networks satisfy the condition

- (C1) in a merged path of a recombination node, there is no node that is in the merged path of a different recombination node.

If a network satisfies (C1), the recombination nodes are "independent". Wang *et al.* described a polynomial-time algorithm that solves the MSPPN problem with respect to the binary case of PPN with recombination.

Theorem 21. *(Theorem 5 in [186]) For a given matrix M, a PPN with recombination satisfying condition (C1) exists if and only if either*

(a) *every O_i and O_j do not conflict; or,*

(b) *If O_i and O_j conflict,*

> **(b1)** *for any $k \notin \{i, j\}$, if O_k and O_i conflict then $O_k \cap O_i = O_i \cap O_j$, and if O_k and O_j conflict, then $O_k \cap O_j = O_i \cap O_j$,*
>
> **(b2)** *if two conflict pairs (O_i, O_j) and $(O_{k'}, O_{k''})$ having identical intersections, i.e., $O_i \cap O_j = O_{k'} \cap O_{k''}$, then either*

$\cdot$	c_1	c_2	c_3
a	1	1	1
b	0	1	0
c	1	0	0
d	0	0	1
e	0	0	0

Figure 12.8: A counterexample to Theorem 5 in [186].

(1) *$(O_i, O_{k'})$ and $(O_j, O_{k''})$ are conflict pairs and $(O_i, O_{k''})$ and $(O_j, O_{k'})$ are not conflict pairs, or*

(2) *$(O_i, O_{k''})$ and $(O_j, O_{k'})$ are conflict pairs and $(O_i, O_{k'})$ and $(O_j, O_{k''})$ are not conflict pairs,*

(b3) *for any conflict pair (O_k, O_l), if $O_k \cap O_l \neq O_i \cap O_j$, then either $O_k \cap O_l$ and $O_i \cap O_j$ are disjoint or one contains the other,*

(b3.1) *$O_k \cap O_l$ and $O_i \cap O_j$ are disjoint: then there exist distinct $O_{m_{ij}}$ and $O_{m_{kl}}$,*

(b3.2) *$O_i \cap O_j \subset O_k \cap O_l$: then there exists a character c_m satisfying (1) c_m is not conflict with any character, (2) $O_i \subseteq O_m$ and $O_j \subseteq O_m$, and (3) $O_m \subseteq O_k \cap O_l$.*

The matrix in Figure 12.8 is a counterexample to Theorem 5 in [186].

Chapter 13

Case Study: Analysis of an Indo-European Dataset

We have implemented a program for solving the MIPPN problem (defined in the previous chapter), and run it on a dataset of Indo-European (IE) languages for various candidate genetic trees. In this chapter, we report on our findings. While the results presented in this chapter represent the outcome of collaborative work with Don Ringe and Tandy Warnow, *all* liguistic analyses are exclusively due to Don Ringe (we include them here for the sake of completeness of the discussion).

13.1 The IE Dataset

The dataset we used was obtained from Don Ringe, and consists of 294 characters for 24 IE languages. We will first describe and explain our choice of languages and characters, then describe our coding of the characters. The languages are listed in Table 13.1. As can be seen, they represent all ten well-attested subgroups of the IE family (namely Anatolian, Tocharian, Celtic, Italic, Germanic, Albanian, Greek, Armenian, Balto-Slavic, and Indo-Iranian). To represent each subgroup we have chosen a language or languages

Table 13.1: The 24 IE languages analyzed.

Language	Abbreviation	Language	Abbreviation
Hittite	HI	Old English	OE
Luvian	LU	Old High German	OG
Lycian	LY	Classical Armenian	AR
Vedic	VE	Tocharian A	TA
Avestan	AV	Tocharian B	TB
Old Persian	PE	Old Irish	OI
Ancient Greek	GK	Welsh	WE
Latin	LA	Old Church Slavonic	OC
Oscan	OS	Old Prussian	PR
Umbrian	UM	Lithuanian	LI
Gothic	GO	Latvian	LT
Old Norse	ON	Albanian	AL

that are attested relatively fully at as early a date as possible. For instance, Indic is represented by early Vedic, since the Rigveda and other very early texts are extensive enough to provide us with data for most of our characters; but we have used "younger" Avestan rather than the earlier Gatha-Avestan to represent eastern Iranian, since the Gathas are too restricted for our purposes. Greek is represented by Classical Attic rather than Homeric, both because our attestation of Attic is far more extensive and because the Homeric language is known to be an artificial literary dialect. Similar decisions have been made in the other cases. We have used modern data for Welsh, Lithuanian, Latvian, and Albanian because earlier data are much less accessible and because we judged that it would make little difference in those cases. Because our method is character-based, not distance-based, the fact that the languages of our database are not contemporaneous has no negative effect on the results;

all that matters is whether the states of each character fit the branching structure of the tree. (In fact, it is advantageous to us to use the earliest attested languages, since these are more likely to have retained character states that are informative of the underlying evolutionary history. By contrast, distance-based methods, since they are required to work from contemporaneous languages, must use comparatively less phylogenetically informative data in some cases.)

In order to represent as many of the major subgroups as was practicable we were obliged to use some fragmentarily attested ancient languages for which only a minority of the lexical characters could be filled with actual data. The languages in question are Lycian (for which we have only about 15% of the wordlist), Oscan (ca. 20%), Umbrian (ca. 25%), Old Persian (ca. 30%), and Luvian (ca. 40%). At the other extreme we have complete or virtually complete ($\geq 99\%$) wordlists not only for the modern languages but also for Ancient Greek, Latin, Old Norse, Old English, and Old High German; we also have nearly complete ($\geq 95\%$) wordlists for Vedic, Classical Armenian, Old Irish, and Old Church Slavonic. Coverage of the remaining wordlists ranges from about 70% to about 85%. Gaps in the data are coded with unique states, which are compatible with any tree. Therefore, though they do not cause problems for our method, they do decrease the robustness of certain subgroups–which is, of course, realistic.

The inclusion of three Baltic languages and of four Germanic languages introduces parallel development in a considerable number of lexical characters, thus decreasing the amount of usable evidence. We have retained the full set of languages in the database because the internal subgrouping of Balto-Slavic and of Germanic are matters of ongoing debate in the specialist community. On the other hand, the inclusion of only two West Germanic languages–Old English and Old High German, the northernmost and southernmost respectively–potentially avoids much greater character incompatibilities, since the internal diversification of West Germanic is known to have been radically non-treelike (cf. [158]).

Our database includes 22 phonological characters encoding regular sound changes (or, more often, sets of sound changes) that have occurred in the prehistory of various languages, 13 morphological characters encoding details of inflection (or, in one case, word formation), and 259 lexical characters defined by meanings on a basic wordlist. The data were assembled by Don Ringe and Ann Taylor, who are specialists in IE historical linguistics, with the advice of other specialist colleagues. The characters were chosen as follows.

Ringe and Taylor attempted to find sound changes and sets of sound changes unlikely to be repeated that are shared by more than one major subgroup of the family; they were able to discover only three plausible candidates, which are our first three phonological characters. The remaining phonological characters define various uncontroversial subgroups of the family. It would have been possible to find many more such phonological characters and/or to use even larger sets of sound changes for some of them, but nothing would have been gained. For a detailed presentation of the phonological characters see [158].

In attempting to find viable morphological characters Ringe and Taylor made a startling discovery: the states of most morphological characters are either confined to a single major subgroup or are characteristic of the family as a whole, rendering them useless for the first-order subgrouping of the family. Several such morphological characters appear in our database, either because they are important inflectional markers or because they are useful for establishing one or more of the major subgroups. The database also includes all the morphological characters which might aid in determining the first-order subgrouping that Ringe and Taylor were able to discover. For details see [158].

Assembly of the lexical part of the database proceeded somewhat differently. Ringe and Taylor began with a version of the Swadesh 200-word list, since that is a standard comparative lexical set; to it they added about 120 meanings that appear to be culturally significant for older IE languages.

The database just described was the basis of the analysis reported in [158]. For the present project we have modified it in the following ways. We have excluded all polymorphic characters from the dataset (for the reasons outlined above); we even exclude those polymorphic characters which can be recoded as pairs of monomorphic characters (see [158]). We have also excluded all characters that clearly exhibit parallel development (whether or not they are compatible with any plausible tree). Those exclusions are the reason why we use fewer morphological characters, and many fewer lexical characters, than [158].

Ringe assigned character states to the languages in our dataset as follows. In the case of the phonological characters, a language either has or has not undergone a regular sound change (or set of regular sound changes) at some point in its prehistory; it is assigned one state if it has and another if it has not, so that phonological characters normally exhibit two states each. For all other characters, states are assigned on the basis of cognation classes. Words and inflectional affixes in two or more related languages are said to be cognate if the languages have inherited them from their most recent common ancestor by direct linguistic descent. For each character, all the members of each cognation class are assigned the same state; noncognate words and affixes are assigned unique states. We emphasize that all loanwords in a language are noncognates by definition, since they entered the language by a process other than direct, unbroken linguistic descent; thus they are assigned unique states. Readers should also be aware that cognation cannot be determined by inspection; a knowledge of the principles of language change and of the individual histories of all the languages is needed to make such a determination. More information about our coding of the data can be found in [158]; here we discuss only two points of interest not noted above.

First, of the 294 characters we used in our phylogenetic analysis, 256 of these are informative, which means that they can help distinguish between candidate phylogenies (by contrast, an uninformative character is compatible on every tree). Secondly, a considerable number of lexical characters can reasonably be coded in more than one way, because of partial cognations between the items; an example is given in [158]. (One morphological character is also double-coded.) Double codings (or, in a couple of cases, triple or even quadruple codings) increase the number of characters without augmenting the independent available evidence. In consequence, of the 294 characters of our database, 242 are independent. This is still a very substantial number for a comparative linguistic database. Finally we have "weighted" our characters in a maximally simple way. Every candidate tree is required to be compatible with all the phonological characters, or with all but two (P2 and P3, which define the "satem" subgroup and might reflect either shared descent in the strictest sense or have spread through a dialect continuum; see e.g. [77]). Every candidate tree is also required to be compatible with eight of our morphological characters (M3, M5-6, M8, and M12-15).

13.2 Phylogenetic Analysis

13.2.1 Overview

We analyzed five candidate genetic trees, three (Trees A, B, and C) that Ringe, Taylor, and Warnow have come to consider in recent years (shown in Figures 13.1–13.3), and two (Trees D and E) that were suggested to us by Craig Melchert (shown in Figures 13.4 and 13.5). We selected these trees because they are each compatible with the vast majority of the characters (the best of these trees is compatible with more than 95% of the characters, and the worst is compatible with 92% of the characters), but are also compatible with all the morphological characters, which are expected to be the most resistant to borrowing (cf. [120, 158]). (The first three trees differ from trees published in earlier work (such as [159]) because since about 2000 Ringe, Taylor, and Warnow have been using a larger, and corrected, set of characters.)

For each tree, we sought to add a minimum number of contact edges in order to produce a perfect phylogenetic network (PPN); three edges sufficed for all but one of these trees (which needed more than three). We then scrutinized each of the resultant networks to consider whether the proposed contact was feasible on the basis of known constraints (geographic and chronological) on the evolution of the IE family. This led us to reject all but 5 of the resultant perfect phylogenetic networks (three on Tree A, and two on Tree B). Of these, one (on Tree A) was the most believable. Interestingly, this most believable of the numerous PPNs we obtained also optimized each of our mathematical optimization criteria (number of incompatible characters, number of contact edges, and number of borrowing events). Thus, both mathematically and on the basis of known constraints, one solution is superior to all others, and suggests strongly that the IE family evolved largely in a tree-like fashion, sufficiently so that the underlying genetic tree is largely recoverable, and so that specific contact episodes between neighboring linguistic communities can also be detected.

In the remainder of this section we describe the candidate trees in detail, the differences between trees in terms of incompatibility patterns, and the PPNs based on these trees.

13.2.2 Our Candidate Trees

We analyzed five candidate genetic trees, three (Trees A, B, and C) that Ringe, Taylor, and Warnow have come to consider in recent years (shown in Figures 13.1–13.3), and two (Trees D and E) that were suggested to us by Craig Melchert (shown in Figures 13.4 and 13.5).

The main differences between these five trees is the placement of Germanic. The differing placement of Albanian in these trees is less important, since Albanian can attach anywhere within a fairly large region of each tree with equally good fit; its variable placement is a result of the fact that it has lost nearly all the diagnostic inflectional morphology and a large proportion of its inherited vocabulary. Thus each tree actually represents several trees which differ only with respect to exactly where Albanian attaches.[1] In contrast, the variable placement of Germanic appears to reflect a major idiosyncrasy of that subgroup's evolution, one that led Ringe, Taylor, and Warnow to conjecture in earlier papers that Germanic may not have evolved in a strictly tree-like fashion [158, 159]. Our detailed analysis of these different scenarios allows us to test each of the conjectured histories for Germanic.

The differing positions of Germanic in the five trees result in different character incompatibility patterns, as follows: for Tree A the 14 incompatible characters are all lexical; for Tree B the 19 incompatible characters include two phonological and 17 lexical characters; for Tree C the 17 incompatible

[1]In each of the five trees in Figures 13.1–13.5, Albanian can be shifted to any position within the region indicated by the thick lines (tree edges).

characters are all lexical; for tree D the 21 incompatible characters are all lexical, and for Tree E the 18 incompatible characters include two phonological and 16 lexical characters. Interestingly, the incompatible characters for Tree A are a proper subset of the incompatible characters for Tree C. Therefore Tree C will represent a preferred hypothesis for the IE genetic tree only if we can complete Tree C to a PPN which improves upon our best PPNs for Tree A either by the remaining mathematical criteria (the number of contact edges or the loan parsimony criterion), or by significantly greater conformity to known chronological or geographic constraints on IE linguistic history. The incompatible characters for trees A and B are incomparable: 12 lexical characters are incompatible on both trees, but two lexical characters are incompatible on Tree A and compatible on Tree B, and two phonological characters and three lexical characters are incompatible on Tree B but compatible on Tree A. Consequently both trees A and B remain roughly comparable candidates for the underlying genetic tree; a choice between them can reasonably be made only in light of a further analysis in which we extend them to PPNs.

Tree D (in Figure 13.4) differs from Tree A in not grouping Germanic and Albanian together and in not grouping Greek and Armenian together; Tree E (in Figure 13.5) differs from Tree A in its placement of Balto-Slavic. There were 21 incompatible characters for Tree D, all of which are lexical. The incompatible characters for Tree A are a proper subset of the incompatible characters from Tree D; as with our comparison between Trees A and C, we will only consider D a preferred hypothesis over Tree A if we can complete

Tree D to a PPN which improves upon our best PPN for Tree A in some way. There were 18 incompatible characters for Tree E, 2 phonological and 16 lexical; the set of incompatible characters is incomparable to the set of characters incompatible with Tree A.

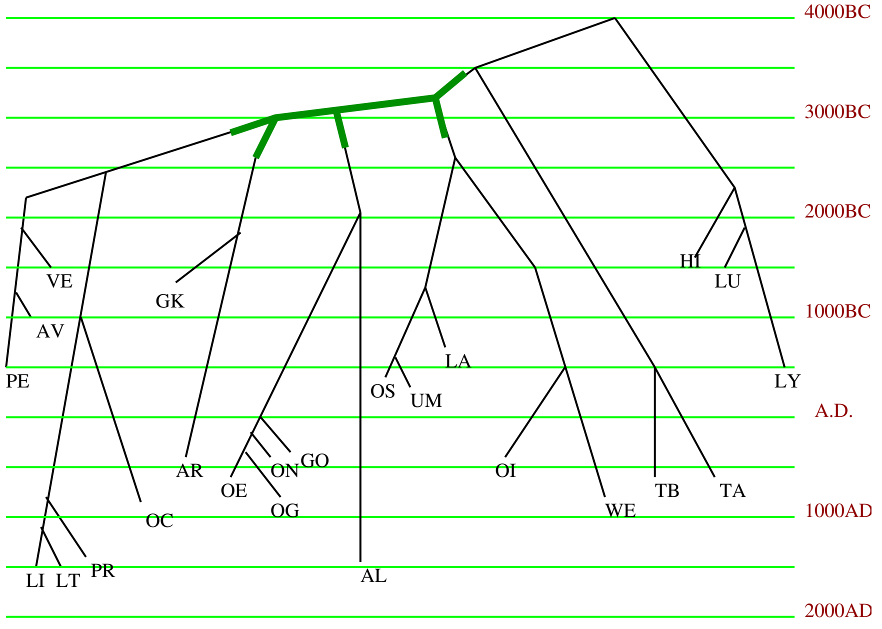


Figure 13.1: Tree A. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome.

13.2.3 Constructing the Perfect Phylogenetic Networks

The second phase of this analysis commenced after we had developed the algorithms and software to determine all the ways we could add a minimum number of edges to a tree in order to obtain perfect phylogenetic networks.

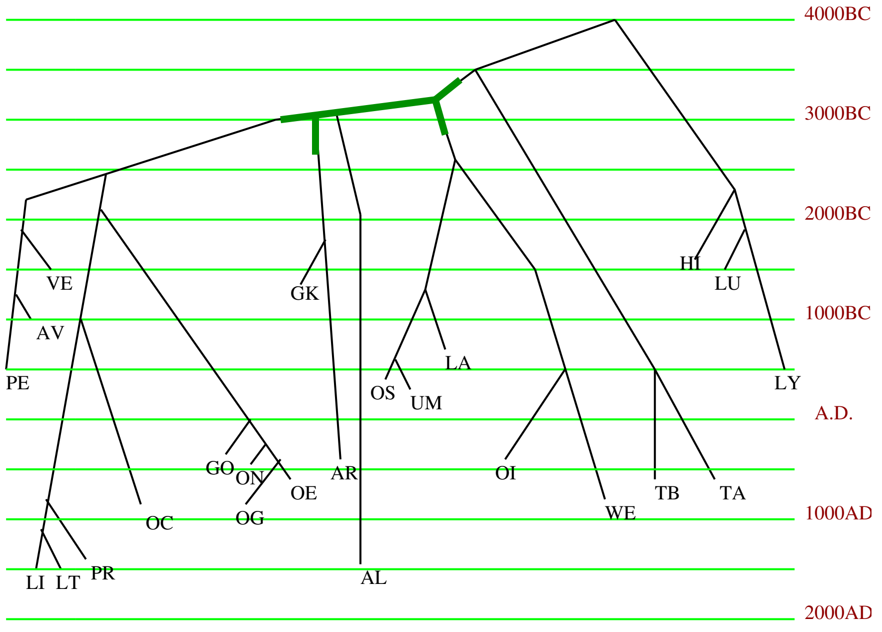


Figure 13.2: Tree B. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome.

The problem of adding a minimum number of edges to a tree to obtain a perfect phylogenetic network is computationally difficult; consequently we used a heuristic which allowed us to obtain solutions in a reasonable amount of time. Our technique for constructing a perfect phylogenetic network from a tree is as follows:

- **Pruning the candidate tree.** In this step, we modified our candidate tree by replacing certain rooted subtrees (i.e., clades) by single nodes, as long as two conditions held: first, all characters are compatible below the root of the subtree, and second, there is linguistic evidence that

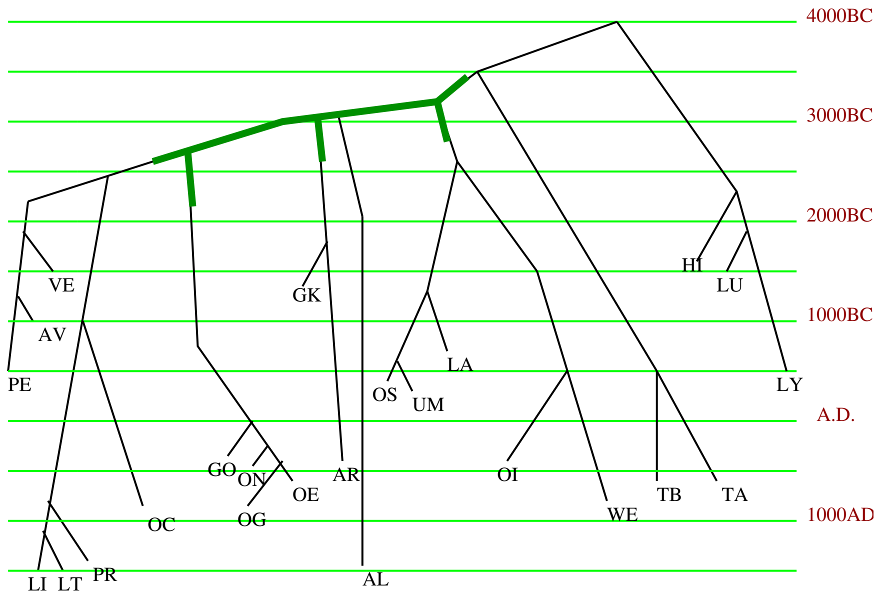


Figure 13.3: Tree C. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome.

suggests that undetected borrowing should not have occurred between a language in that clade and a language outside the clade. The subtrees of language groups that were replaced by their roots are: *Germanic*, *Celtic*, *Italic*, *Tocharian*, *Indo-Iranian*, *Anatolian*, *Greek-and-Armenian* (for those trees in which Greek and Armenian are sisters - i.e. for all our trees other than Tree D), and *Baltic*. Albanian and Old Church Slavonic remain as individual languages.

- **Adding the minimum number of contact edges.** After the pruned tree was obtained, the algorithm searched for all the ways we could add

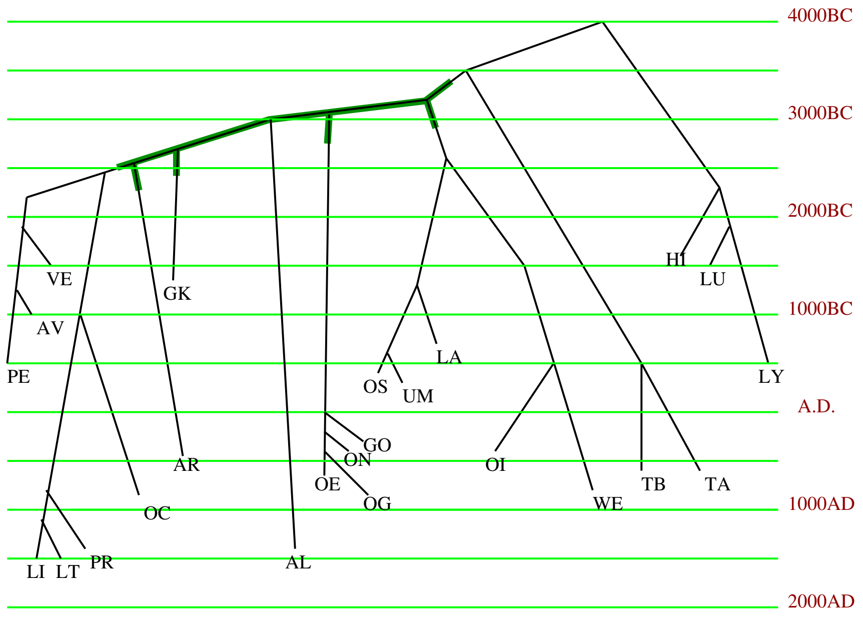


Figure 13.4: Tree D. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome.

a minimum number of contact edges and get a perfect phylogenetic network. This part of the analysis took eight hours on each of our candidate trees and found several networks with only three contact edges.

13.2.4 The Set of PPNs We Obtained

Table 13.2 summarizes the quantitative results of the analysis on each of the five trees. In the next section we analyze the solutions found on each of the five trees.

The result of our brute-force analysis produced 23 PPNs, each with

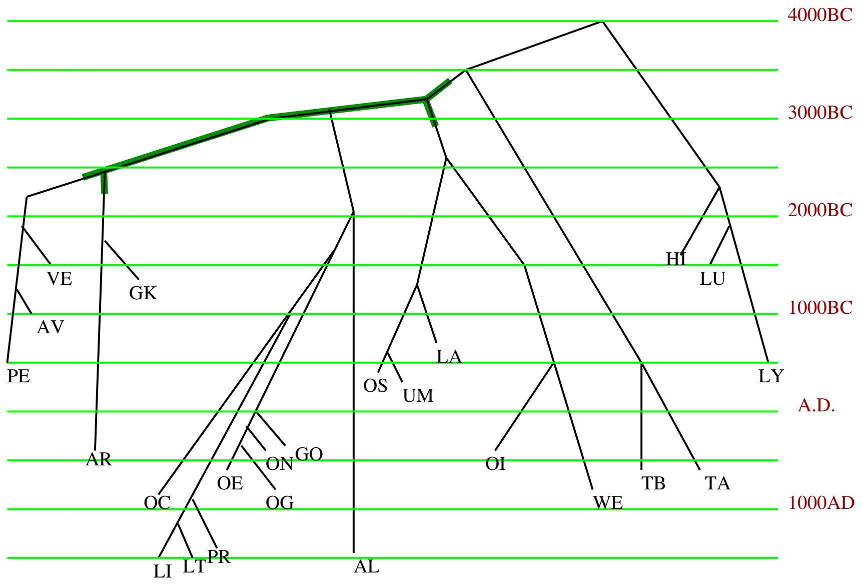


Figure 13.5: Tree E. The thick lines represent the region within which Albanian can attach without changing the quality of the outcome.

only three contact edges; 16 of these were based on Tree A, 4 were based on Tree B, one was based on Tree C, two on Tree D, and none on Tree E. We compared these 23 PPNs in two ways:

- with reference to linguistic and archaeological evidence, which can render certain proposed contacts unlikely or even impossible (cf. [110]), and
- according to the mathematical criteria we proposed earlier in the paper: (1) the number of characters compatible on the genetic tree, (2) the number of contact edges in the PPN, and (3) loan parsimony, i.e. the number of undetected borrowing events in the PPN.

Table 13.2: Summary of the results of the phylogenetic analysis of the five IE trees described in Figures13.1–13.5.

	Tree A	Tree B	Tree C	Tree D	Tree E
# incompatible characters	14	19	17	21	18
Min # contact edges needed	3	3	3	3	> 3
# 3-edge solutions found	16	4	1	2	0
# plausible solutions	3	2	0	0	0

Of the 23 PPNs with three contact edges that we found, five were consistent with known geographical and chronological constraints. Since three of these were based on Tree A (our preferred solution for the genetic IE tree, based upon the number of compatible characters), it seems reasonable to exclude PPNs based on any of our trees with more than three contact edges.

13.2.5 Comparison of the PPNs

In this section we compare the 23 PPNs we obtained. We present the comparison with respect to geographic and chronological constraints first (all of which is due to Don Ringe), and then the evaluation with respect to mathematical criteria. We observed several interesting things, the most striking of which is that the tree that optimized the mathematical criteria was also the most feasible with respect to the geographical and chronological constraints.

Finding the feasible solutions We begin our discussion with our favored candidate for the genetic tree, Tree A, shown in Figure 13.1. This was previously published in [158].

PPNs based on Tree A The dates assigned to the terminal nodes of Tree A (shown in Figure 13.1) are obtainable from historical data. Since our method is not distance-based, it is not necessary to use the dates of the substantial corpora which underlie our lexical lists. Instead we have here employed the dates of the earliest documentary material of each language which shows clearly that it was already different from its closest relatives. Dates assigned to the internal nodes are necessarily the result of informed guesswork, since proposed "glottochronological" methods for determining the dates of prehistoric languages have proved to be unreliable (see especially [14], none of whose objections have been effectively met by recent work, and Eska and Ringe, forthcoming). Dates for a few internal nodes can be fixed with reasonable certainty.[2] For instance, the complete archaeological continuity between the Yamna Horizon (up to ca. 2200 B.C., [110]), its eastern Andronovo offshoot, and cultures known to have spoken Indo-Iranian languages allows us to place Proto-Indo-Iranian in the temporal vicinity of 2000 B.C., give or take a couple of centuries (cf. the discussion of [110]). Since Indo-Iranian is one of the most deeply embedded subgroups in the tree, it follows that all the first-order branching must have occurred by that date. Most internal nodes, though, can be dated only within fairly wide ranges by a kind of linguistic "dead reckoning" and must therefore

[2]The reasoning in this and subsequent paragraphs, attempting to correlate linguistic events and historical and archaeological findings, has been commonplace in IE linguistics at least since Porzig 1954 [142]. The best summary of the relevant archaeological facts is [110].

be treated with caution.

The algorithm described above generated 16 solutions for Tree A, three of which were consistent with known constraints on the history of the IE family (cf. [110]). Those 16 solutions are described in Table 13.3 (the feasible solutions are highlighted in the table).

Table 13.3: The 16 3-edge solutions found on Tree A; the highlighted rows correspond to the three feasible solutions (1, 3, and 5). Each solution is described in terms of the three lateral edges added to Tree A to produce a PPN. The abbreviations are explained in Table 13.4. The rightmost column summarizes the minimum number of borrowing events needed on each of the 16 PPNs in order to make all characters compatible.

	Solutions			min # borrowing events
	Edge 1	Edge 2	Edge 3	
1	(PT,PS)	(PC,PBS)	(PC,PG)	19
2	(PIC,PB)	(PC,PG)	(PC,PGA)	20
3	(PT,PS)	(PI,PG)	(PI,PBS)	19
4	(PI,PG)	(PI,PGA)	(PIC,PB)	20
5	(PI,PG)	(PI,PGA)	(PG,PB)	17
6	(PT,PBSII)	(PI,PG)	(PI,PB)	19
7	(PT,PII)	(PI,PB)	(PI,PG)	18
8	(PT,PBS)	(PI,PB)	(PI,PG)	18
9	(PT,PS)	(PI,PB)	(PI,PG)	19
10	(PT,PB)	(PI,PB)	(PI,PG)	19
11	(PIC,PGA)	(PI,PB)	(PI,PG)	20
12	(PI,PB)	(PI,PGA)	(PI,PG)	20
13	(PC,PGA)	(PI,PB)	(PI,PG)	20
14	(PI,PG)	(PI,PB)	(PG,PGA)	20
15	(PI,PB)	(PAL,PGA)	(PI,PALG)	23
16	(PA,PBSII)	(PI,PG)	(PI,PB)	19

Three feasible PPNs based on Tree A were found; these are solutions

Table 13.4: Abbreviations for proto languages.

Abbreviations	
PT	Proto-Tocharian
PG	Proto-Germanic
PI	Proto-Italic
PC	Proto-Celtic
PIC	Proto-Italo-Celtic
PB	Proto-Baltic
PS	Proto-Slavic
PGA	Proto-Greco-Armenian
PBS	Proto-Balto-Slavic
PBSII	Proto-Balto-Slavic and Indo-Iranian
PII	Proto-Indo-Iranian
PAL	pre-Albanian
PALG	Proto-Albanian and Germanic
PA	Proto-Anatolian

1, 3, and 5 in Table 13.3. Of these, the first feasible PPN (solution 1, see also Figure 13.6) is clearly more plausible than the second feasible (solution 3, see also Figure 13.7).

Both posit a contact edge between Proto-Tocharian and Proto-Slavic. That is somewhat surprising, because it implies that an ancestor of Tocharian was still in eastern Europe in the last millennium B.C.E., and it seems clear that by the turn of the millennium they were within striking distance of Xinjiang (where the Tocharian languages are actually attested from about the 6th c. C.E.). However, we know very little about the prehistoric movements of speakers of Tocharian, and what we do know is that they were in contact with steppe-dwelling Iranian tribes; a long migration eastward in a relatively short period of time therefore does not seem out of the question. The PPN in Figure 13.6 also posits a contact edge between Proto-Celtic and Proto-Germanic, which is unobjectionable, and one between Proto-Celtic and Proto-Balto-Slavic, which is surprising; the PPN in Figure 13.7 also posits a contact edge between Proto-Italic and Proto-Germanic, which is likewise unobjectionable, and one between Proto-Italic and Proto-Balto-Slavic, which is likewise surprising. In other words, each of these PPNs posits that one of the "western" subgroups of the family was, at very early periods, in contact both with Germanic and with Balto-Slavic, and it is the connections with Balto-Slavic that are unexpected. But while it is clearly out of the question for Baltic and Slavic languages to have been in contact with Celtic or Italic languages during the historical period, the linguistic geography of eastern Europe can have been very different in, say, the 3rd millennium B.C.E. In particular, it is possible that speakers of Proto-Italo-Celtic occupied the Hungarian plain in about 3200 B.C.E. (David Anthony, personal communication), and that Italic and Celtic began to diverge in eastern Europe; and since it also seems possible that Germanic and Balto-Slavic evolved on the other side of the Carpathians, contact between both those groups and one of the "western" groups might have been possible for some centuries. Proto-Celtic, the more northerly of the two "western" protolanguages, is clearly a more plausible candidate, and so the first PPN is therefore probably preferable to the second. (Note that the relative chronological positions of the internal nodes of all our trees must be allowed to vary within certain limits. They cannot be fixed absolutely by archaeological data, and variation in the rate of linguistic change is still too poorly understood to render their calculation from internal evidence feasible.)

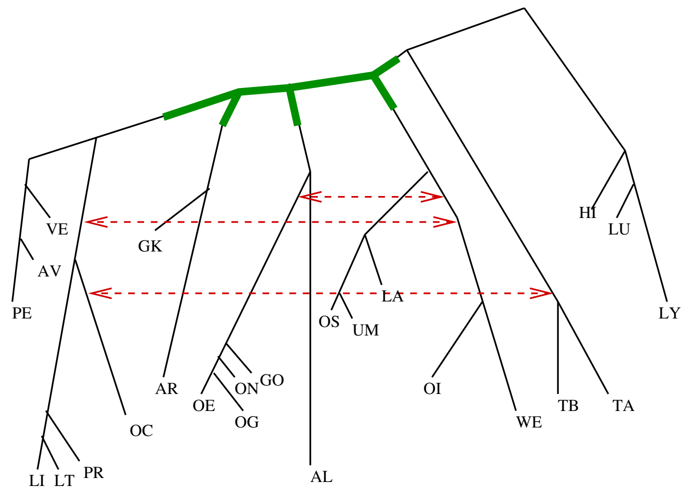


Figure 13.6: The first feasible perfect phylogenetic network based on Tree A (corresponding to Solution 1 in Table 13.3). The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional.

We now consider the third feasible PPN based on Tree A (solution 5, see Figure 13.8). This PPN posits a contact edge between Proto-Italic and Proto-Germanic, which is (again) unobjectionable; a second contact edge (at a later date) between Proto-Germanic and Proto-Baltic, which is highly likely; and a contact edge between Proto-Italic and Proto-Greco-Armenian, which is surprising but cannot be excluded (given how little we know about the prehistoric linguistic geography of Eastern Europe). Interestingly, however, of these

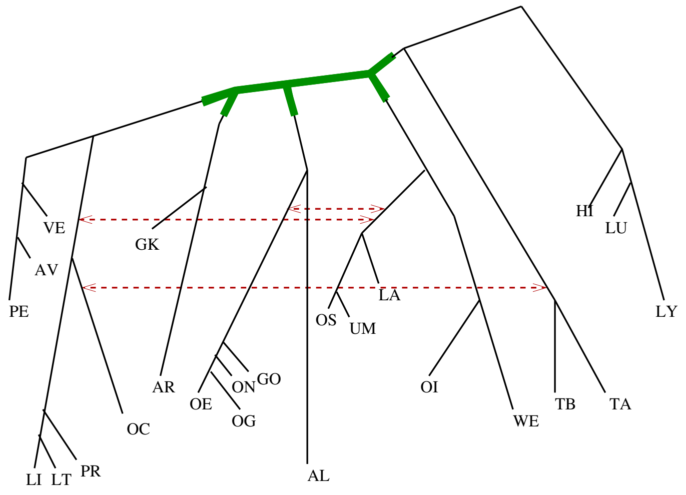


Figure 13.7: The second feasible perfect phylogenetic network based on Tree A (corresponding to Solution 3 in Table 13.3). The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional.

three contact edges, the questionable contact edge has the smallest support: only two of the characters ('free' and one alternative coding of 'young') must use that contact edge, compared to 6 on the first contact edge and 9 on the second. It is therefore possible that this questionable contact edge between Proto-Italic and Proto-Greco-Armenian is not realistic, and that some other explanation (such as undetected homoplasy?) should be found for the non-treelike evolution of 'free' and 'young'. The other two contact edges, which are very well supported, do seem realistic.

With the exception of the three aforementioned solutions, all other solutions are implausible or actually impossible for chronological and geographical

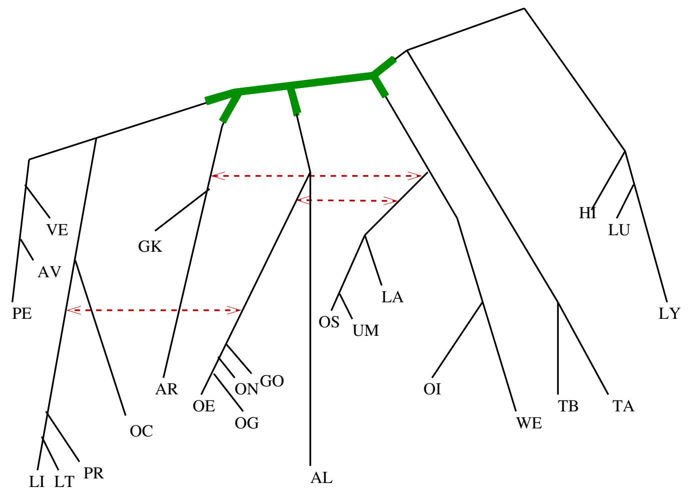


Figure 13.8: The third feasible perfect phylogenetic network based on Tree A (corresponding to Solution 5 in Table 13.3). The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional.

reasons, as follows. The temporal constraints which we applied to find candidate contact edges (see the previous section) are the ones which are easiest to define in formal terms, but they are not the only ones that exist; while it is clearly impossible for a living language to be in contact with its living ancestor, it is equally impossible for two living languages spoken at different periods in history to be in contact. Solutions 6 through 16 posit a contact edge between Proto-Italic (the ancestor of all the Italic languages), at some time after its separation from Proto-Celtic, and Proto-Baltic (the ancestor of all the Baltic languages), at some time after its separation from Proto-Slavic. Proto-Italic must have begun to diverge into the attested Italic languages well before 1000 B.C.E., because our earliest documents from the Osco-Umbrian subgroup, in about the 6th c. B.C.E., exhibit so many innovations not shared by Latin that it is clear that those two halves of Italic had been diverging for centuries. But Baltic is most unlikely to have begun diverging from Slavic by 1000 B.C.E., because Proto-Slavic seems still to have been more or less uniform in the 8th c. C.E., and Proto-Baltic and Proto-Slavic are so similar that they had probably been diverging for less than two millennia. In addition, Proto-Baltic was clearly spoken somewhere in northeastern Europe (not southwest of modern Poland); and while Proto-Italic may not have been spoken in Italy, it can hardly have been spoken anywhere to the northeast of modern Hungary. Solutions 2 and 4 posit a contact edge between Proto-Baltic and Proto-Italo-Celtic; since the latter is a direct ancestor of Proto-Italic, the chronological constraints exclude these two solutions a fortiori, though the geographical situation is considerably less clear.

PPNs based on Tree B The algorithm described above generated four 3-edge solutions for Tree B, two of which were consistent with known constraints on the history of the IE family. Those solutions are described in Table 13.5.

The first solution (solution 1 in Table 13.5) posits contact between pre-Italic and the ancestor of Germanic and Balto-Slavic (reasonably plausible), between Germanic and Celtic (highly plausible), and Tocharian and Slavic (as for two solutions on Tree A); the minimum number of borrowing events needed to make all characters compatible on this PPN is 21. This solution appears

Table 13.5: The 4 3-edge solutions found on Tree B; the two feasible solutions highlighted. Each solution is described in terms of the three lateral edges added to Tree B to produce a PPN. The abbreviations are explained in Table 13.4. The rightmost column summarizes the minimum number of borrowing events needed on each of the 4 PPNs in order to make all characters compatible.

	Solutions			min # borrowing events
	Edge 1	Edge 2	Edge 3	
1	(PI,PGBS)	(PG,PC)	(PT,PS)	21
2	(PG,PIC)	(PI,PBS)	(PT,PS)	23
3	(PI,PB)	(PC,PGA)	(PG,PIC)	25
4	(PI,PS)	(PC,PGA)	(PG,PIC)	24

possible, but it is not markedly better than the first two solutions on Tree A, and it is considerably less plausible than the third. The perfect phylogenetic network that corresponds to this solution is shown in Figure 13.9.

The next solution (solution 2 in Table 13.5) posits contact between pre-Germanic and Proto-Italo-Celtic (which might be possible if the Proto-Italo-Celtic node should actually be somewhat later than we have hypothesized), between Italic and Balto-Slavic (surprising, but not necessarily out of the question), and Tocharian and Slavic (as above); the minimum number of borrowing events needed to make all characters compatible on this PPN is 23. This solution, too, cannot be summarily excluded but is not as plausible as the third solution on Tree A. The PPN that corresponds to this solution is shown in Figure 13.10.

It is important to note that the two PPNs based on Tree B imply different chronological orderings of Proto-Italo-Celtic and Proto-Germano-Balto-

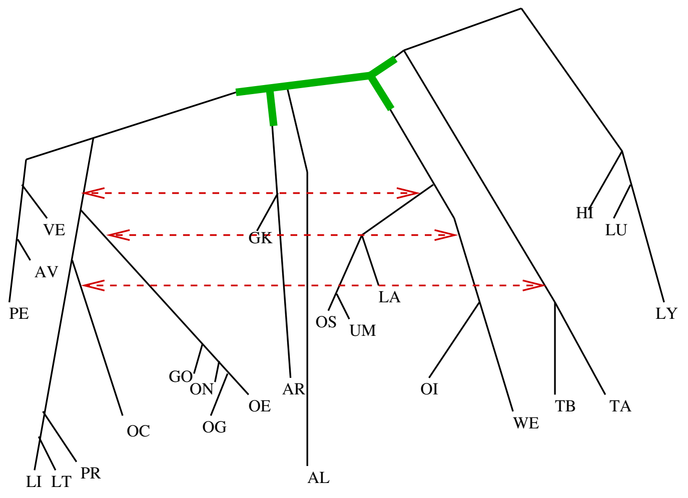


Figure 13.9: The first perfect phylogenetic network on Tree B. The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional.

Slavic. Both solutions need to be considered, since the times of internal nodes in the tree are somewhat indeterminate. Additional clarification about the dates of these internal splits would help clarify the relationships between these languages.

We now describe the two remaining solutions, and why we can eliminate these on the basis of known constraints. The first solution (solution 3 in Table 13.5) posits contact between Italic and Baltic but not Slavic, which seems impossible; the minimum number of borrowing events needed to make all characters compatible on this PPN is 25. The second solution (solution 4 in Table 13.5) posits contact between Italic and Slavic but not Baltic, which

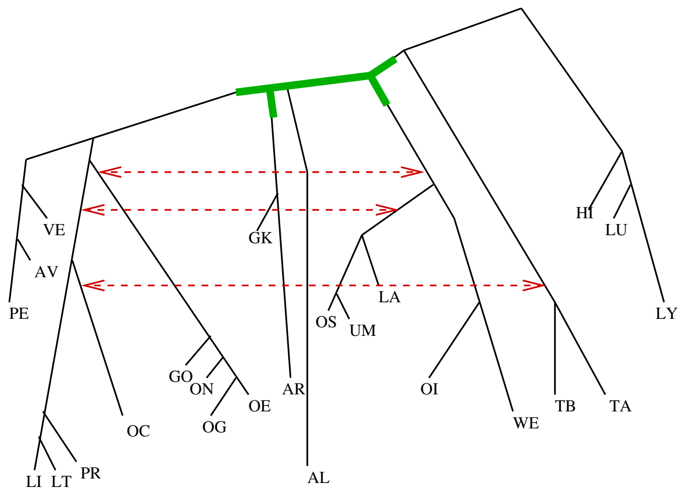


Figure 13.10: The second perfect phylogenetic network on Tree B. The solid black edges correspond to the underlying tree edges and indicate vertical transmission; those edges are directed. The dashed lines represent contact between language groups, and they are bi-directional.

likewise seems impossible; the minimum number of borrowing events needed to make all characters compatible on this PPN is 24. Both these solutions are thus infeasible.

PPNs based on Tree C The algorithm described above generated one 3-edge solution for Tree C. However, the solution posits contact between Italic and Baltic but not Slavic, which seems impossible both for chronological and for geographical reasons; the minimum number of borrowing events needed to make all characters compatible on this PPN is 24.

PPNs based on Tree D The algorithm described above generated two 3-edge solutions for Tree D. However, both solutions posit a contact between Italic and Baltic but not Slavic, which seems impossible both for chronological and for geographical reasons.

PPNs based on Tree E The algorithm described above did not find any 3-edge solutions for Tree E.

Additional Analyses

Since the Greco-Armenian subgroup is very weakly supported, and since the unity of Italic has been repeatedly impugned, we re-analyzed Tree A (Figure 13.1) without pruning the Italic and Greco-Armenian groups (recall that one of the steps in our phylogenetic analysis involves pruning maximally compatible subtrees). In this case, three new PPNs with three lateral edges were found (in addition to the 16 PPNs reported earlier). However, all three PPNs posit a lateral edge between Italic (at some time after its separation from Celtic) and Baltic (at some time after its separation from Slavic). Hence, the three solutions are implausible, as discussed earlier.

Summary of the Results Our five different plausible PPNs for the IE family exhibit interesting similarities and differences. In the first place, it appears that solutions with three lateral edges are possible only if Germanic is the outlier within the "core" (i.e. the residue of the family after Italo-Celtic has diverged), or if it is the nearest sister of Balto-Slavic. Four of the five solutions posit a contact episode between Slavic and Tocharian; that is an unforeseen, indeed a surprising, result. The remaining solution—the third on Tree A (solution 5)—is much less surprising; in fact, the contact between Germanic and Baltic which it posits is so highly plausible that this solution is probably preferable to the others on that ground alone (Ringe).

13.2.6 Comparison of All 23 PPNs With Respect to Mathematical Criteria

Recall the three mathematical criteria by which we evaluate PPNs: (1) the number of characters compatible on the genetic tree, (2) the number of contact edges in the PPN, and (3) loan parsimony, i.e. the number of undetected borrowing events in the PPN.

Of the 23 PPNs – feasible and infeasible together – the best that we can do with respect to criterion (1) is 14 (since all PPNs based on Tree A have only 14 characters incompatible on them, while PPNs based on the other candidate trees have more). All the PPNs we consider posit three contact edges, so there is no difference with respect to criterion (2). For the third criterion, the best we can do is 17, which is achieved by solution 5 on Tree A; all other solutions must posit at least 19 borrowing events (and usually more). Thus solution 5 on Tree A optimizes all three of the mathematical criteria, and is the unique optimal solution.

13.2.7 Discussion

Thus, our favored PPN – solution 5 based on Tree A – is optimal with respect to each of the three mathematical criteria, and also most believable with respect to the geographic and chronological constraints. Thus, it optimizes each of the four criteria. Our third PPN on Tree A is therefore the solution that we propose for the first-order evolution of the IE family. More research will of course be needed to confirm this. Because this PPN is so clearly better than the other scenarios, a closer look at it is justified. That is the focus of our next section.

13.3 Our Proposed Solution to IE Evolution

Our best solution for the historical diversification of the Indo-European language family posits Tree A (the tree found in [158]) as the underlying genetic tree and three contact edges; our proposed solution is thus the perfect phylogenetic network of Figure 13.8. We note that our network suggests *at most* three real episodes of contact between the relevant language groups. It makes sense to examine these three possible contact episodes to determine how much support our analysis suggests for each.

Two of the contact edges, both involving Germanic, do have a significant number of characters evolving down them; they are obviously necessary components of this PPN. The third edge, between Proto-Italic and Proto-Greco-Armenian, has only two characters transmitting states across it. Thus, while the contact edges that involve Germanic are well supported, the contact edge that does not involve Germanic seems debatable. It is possible that some other explanation of the evolution of the two involved characters ('free' and 'young') that will not involve borrowing can be found.

With respect to the question of whether the evolution of IE should be represented by a wave model or a tree model (for the most part), we believe we have shown that although a tree model does not fit the family's history perfectly, there is clear evidence that the underlying history is almost entirely treelike, and that it may be sensible to infer a genetic tree on which some borrowing (previously undetected) has occurred.

Bibliography

[1] http://evolution.genetics.washington.edu/phylip/software.html.

[2] http://www.algorithmic-solutions.com/enleda.htm.

[3] ftp://dimacs.rutgers.edu/pub/netflow/general-info/.

[4] L. Addario-Berry, M.T. Hallett, and J. Lagergren. Towards identifying lateral gene transfer events. In *Proc. 8th Pacific Symp. on Biocomputing (PSB03)*, pages 279–290, 2003.

[5] R. Agarwala and D. Fernandez-Baca. A polynomial-time algorithm for the perfect phylogeny problem when the number of character states is fixed. *SIAM Journal on Computing*, 23(26):1216–1224, 1994.

[6] B. Allen and M. Steel. Subtree transfer operations and their induced metrics on evolutionary trees. *Annals of Combinatorics*, 5:1–13, 2001.

[7] J. Alroy. Continuous track analysis: A new phylogenetic and biogeographic method. *Syst. Biol.*, 44(2):152–178, 1995.

[8] L. Arvestad, A.-C. Berglund, J. Lagergren, and B. Sennblad. Bayesian gene/species tree reconciliation and orthology analysis using MCMC. In *Proc. 11th Int'l Conf. on Intelligent Systems for Molecular Biology (ISMB03)*, volume 19 of *Bioinformatics*, pages i7–i15, 2003.

[9] D.A. Bader, B.M.E. Moret, and L. Vawter. Industrial applications of high-performance computing for phylogeny reconstruction. In H.J. Siegel, editor, *Proc. SPIE Commercial Applications for High-Performance Computing (SPIE01)*, volume 4528, pages 159–168, 2001.

[10] H.J. Bandelt and A.W.M. Dress. Split decomposition: a new and useful approach to phylogenetic analysis of distance data. *Mol. Phyl. Evol.*, 1:242–252, 1992.

[11] H.J. Bandelt, P. Forster, and A. Roehl. Median-joining networks for inferring intraspecific phylogenies. *Mol. Biol. Evol.*, 16(1):37–48, 1999.

[12] H.J. Bandelt, P. Forster, B.C. Sykes, and M.B. Richards. Mitochondrial portraits of human populations using median networks. *Genetics*, 141:743–753, 1995.

[13] H.J. Bandelt, V. Macaulay, and M. Richards. Median networks: speedy construction and greedy reduction, one simulation, and two case studies from human mtDNA. *Mol. Phyl. Evol.*, 16:8–28, 2000.

[14] K. Bergsland and H. Vogt. On the validity of glottochronology. *Current Anthropology*, 3:111–153, 1962.

[15] M. Blanchette, G. Bourque, and D. Sankoff. Breakpoint phylogenies. In S. Miyano and T. Takagi, editors, *Genome Informatics*, pages 25–34. Univ. Academy Press, Tokyo, 1997.

[16] A. Boc and V. Makarenkov. New efficient algorithm for detection of horizontal gene transfer events. In *Proc. 3rd Int'l Workshop Algorithms in Bioinformatics (WABI03)*, volume 2812, pages 190–201. Springer-Verlag, 2003.

[17] H. Bodlaender, M. Fellows, and T. Warnow. Two strikes against perfect phylogeny. In *Proceedings of the 19th International Colloquium on Automata, Languages, and Programming*, Lecture Notes in Computer Science, pages 273–283. Springer Verlag, 1992.

[18] J.R. Brown, C.J. Douady, M.J. Italia, W.E Marshall, and M.J. Stanhope. Universal trees based on large combined protein sequence data sets. *Nat. Genet.*, 28:281–285, 2001.

[19] J.R. Brown and P.V. Warren. Antibiotic discovery: Is it in the genes? *Drug Discovery Today*, 3:564–566, 1998.

[20] W.J. Bruno and A.L. Halpern. Topological bias and inconsistency of maximum likelihood using wrong models. *Mol. Biol. Evol.*, 16:564–566, 1999.

[21] D. Bryant and V. Moulton. NeighborNet: An agglomerative method for the construction of planar phylogenetic networks. In *Proc. 2nd Int'l Workshop Algorithms in Bioinformatics (WABI02)*, volume 2452 of *Lecture Notes in Computer Science*, pages 375–391. Springer-Verlag, 2002.

[22] J.J. Bull, J.P. Huelsenbeck, C.W. Cunningham, D. Swofford, and P. Waddell. Partitioning and combining data in phylogenetic analysis. *Syst. Biol.*, 42(3):384–397, 1993.

[23] S.B. Carroll, J.K. Grenier, and S.D. Weatherbee. *From DNA to Diversity*. Blackwell Science, 2001.

[24] J.K. Chambers and L.E. McDonald *et al.* A G protein-coupled receptor for UDP-glucose. *J. Biol. Chem.*, 275(15):10767–10771, 2000.

[25] P.T. Chippindale and J.J. Wiens. Weighting, partitioning, and combining characters in phylogenetic analysis. *Syst. Biol.*, 43(2), 1994.

[26] M. Clement, D. Posada, and K. Crandall. TCS: a computer program to estimate gene genealogies. *Mol. Ecol.*, 9:1657–1660, 2000.

[27] C.W. Cunningham. Can three incongruence tests predict when data should be combined? *Mol. Biol. Evol.*, 14:733–740, 1997.

[28] V. Daubin, M. Gouy, and G. Perriere. A phylogenomic approach to bacterial phylogeny: evidence of a core of genes sharing a common history. *Genome Res.*, 12:1080–1090, 2002.

[29] W.H.E. Day. Computationally difficult parsimony problems in phylogenetic systematics. *Journal of Theoretical Biology*, 103:429–438, 1983.

[30] W.H.E. Day. Optimal algorithms for comparing trees with labeled leaves. *Journal of Classification*, 2:7–28, 1985.

[31] W.H.E. Day and D. Sankoff. Computational complexity of inferring phylogenies by compatibility. *Syst. Zool.*, 35:224–229, 1986.

[32] F. de la Cruz and J. Davies. Horizontal gene transfer and the origin of species: lessons from bacteria. *Trends Microbiol.*, 8:128–133, 2000.

[33] A. de Queiroz, M.J. Donoghue, and J. Kim. Separate versus combined analysis of phylogenetic evidence. *Annu. Rev. Ecol. Syst.*, 25:657–681, 1995.

[34] E. Diday and P. Bertrand. An extension of hierarchical clustering: the pyramidal representation. In E.S. Gelsema and L.N. Kanal, editors, *Pattern Recognition in Practice II*, pages 411–424. Elsevier Science Publ., 1986.

[35] A.J. Dobson. Unrooted trees for numerical taxonomy. Unpublished manuscript.

[36] A.J. Dobson. Lexicostatistical grouping. *Anthropological Linguistics*, 11:216–221, 1969.

[37] W.F. Doolittle. Phylogenetic classification and the universal tree. *Science*, 284:2124–2129, 1999.

[38] R.G. Downey and M.R. Fellows. Fixed parameter tractability and completeness I: basic theory. *SIAM Journal of Computing*, 24:873–921, 1995.

[39] A. Dress and D. Huson. Computing phylogenetic networks from split systems, 1998. Manuscript.

[40] J.A. Eisen. Horizontal gene transfer among microbial genomes: New insights from complete genome analysis. *Curr Opin Genet Dev.*, 10(6):606–611, 2000.

[41] T. Eiter, N. Leone, C. Mateis amd G. Pfeifer, and F. Scarcello. A deductive system for non-monotonic reasoning. In *Proceedings of the 4th International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 363–374. Springer-Verlag, 1997.

[42] N.C. Ellstrand, R. Whitkus, and L.H. Rieseberg. Distribution of spontaneous plant hybrids. *Proc. Nat'l Acad. Sci., USA*, 93(10):5090–5093, 1996.

[43] E. Erdem, V. Lifschitz, L. Nakhleh, and D. Ringe. Reconstructing the evolutionary history of Indo-European languages using answer set programming. In *Proceedings of the Fifth International Symposium on Practical Aspects of Declarative Languages (PADL 2003)*, volume 2562 of *Lecture Notes in Computer Science*, pages 160–176, 2003.

[44] O. Eulenstein, B. Mirkin, and M. Vingron. Duplication-based measures of difference between gene and species trees. *J. Comput. Biol.*, 5:135–148, 1998.

[45] L. Excoffier, P.E. Smouse, and J.M. Quattro. Analysis of molecular variance inferred from metric distances among DNA haplotypes: application to human mitochondrial DNA restriction data. *Genetics*, 131:479–491, 1992.

[46] J. Felsenstein. Cases in which parsimony and compatibility methods will be positively misleading. *Syst. Zool.*, 17:401–410, 1978.

[47] J. Felsenstein. Numerical methods for inferring evolutionary trees. *Quarterly Review of Biology*, 57:379–404, 1982.

[48] J. Felsenstein. Phylogenies and the comparative method. *Amer. Nat.*, 125:1–15, 1985.

[49] J. Felsenstein. The troubled growth of statistical phylogenetics. *Syst. Biol.*, 50(4):465–467, 2001.

[50] J. Felsenstein. *Inferring Phylogenies.* Sinauer Associates, Inc., Sunderland, MA, 2003.

[51] J. Felsenstein and G. Churchill. A hidden Markov model approach to variation among sites in rate of evolution. *Molecular Biology and Evolution*, 13:93–104, 1996.

[52] W.M. Fitch. Toward defining the course of evolution: minimum change for a specified tree topology. *Syst. Zool.*, 20:406–416, 1971.

[53] L.R. Foulds and R.L. Graham. The Steiner problem in phylogeny is np-complete. *Advances in Applied Mathematics*, 3:43–49, 1982.

[54] V.A. Funk. Phylogenetic patterns and hybridization. *Ann. Mo. Bot. Gard.*, 72:681–715, 1985.

[55] D.J. Futuyma. *Evolutionary Biology.* Sinauer Assoc., Sunderland, MA, 1998.

[56] M.Y. Galperin and E.V. Koonin. Comparative genome analysis. *Methods Biochem. Anal.*, 43:359–392, 2001.

[57] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, New York, 1979.

[58] H.A. Gleason. Counting and calculating for historical reconstruction. *Anthropological Linguistics*, 1:22–32, 1959.

[59] N.C. Grassly and E.C. Holmes. A likelihood method for the detection of selection and recombination using nucleotide sequences. *Mol. Biol. Evol.*, 14:239–247, 1997.

[60] R.C. Griffiths and P. Marjoram. Ancestral inference from samples of DNA sequences with recombination. *J. Comput. Biol.*, 3:479–502, 1996.

[61] X. Gu. Maximum-likelihood approach for gene family evolution under functional divergence. *Mol. Biol. Evol.*, 18(4):453–464, 2001.

[62] D. Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21:19–28, 1991.

[63] D. Gusfield, S. Eddhu, and C. Langley. Efficient reconstruction of phylogenetic networks with constrained recombination. In *Proc. Computational Systems Bioinformatics (CSB03)*, 2003.

[64] P. Halbur, M.A. Lum, X. Meng, I. Morozov, and P.S. Paul. New porcine reproductive and respiratory syndrome virus DNA and proteins encoded by open reading frames of an Iowa strain of the virus are used in vaccines against PRRSV in pigs, 1994. Patent filing WO9606619-A1.

[65] M.T. Hallett and J. Lagergren. Efficient algorithms for lateral gene transfer problems. In *Proc. 5th Ann. Conf. on Research in Computational Biology RECOMB'01*, pages 149–156, 2001.

[66] E. F. Harding. The probabilities of rooted tree shapes generated by random bifurcation. *Adv. Appl. Prob.*, 3:44–77, 1971.

[67] D. Harel and R.E. Tarjan. Fast algorithms for finding nearest common ancestors. *SIAM Journal on Computing*, 13(2):338–355, 1984.

[68] J.A. Hartigan. Minimum mutation fits to a given tree. *Biometrics*, 29:53–65, 1973.

[69] M. Hasegawa, H. Kishino, and T. Yano. Dating the human-ape splitting by a molecular clock of mitochondrial DNA. *J. Mol. Evol.*, 22:160–174, 1985.

[70] J. Hein. Reconstructing evolution of sequences subject to recombination using parsimony. *Math. Biosciences*, 98:185–200, 1990.

[71] J. Hein. A heuristic method to reconstruct the history of sequences subject to recombination. *J. Mol. Evol.*, 36:396–405, 1993.

[72] J. Hein, T. Jiang, L. Wang, and K. Zhang. On the complexity of comparing evolutionary trees. *Discrete Applied Mathematics*, 71:153–169, 1996.

[73] D.M. Hillis. Primer: Phylogenetic analysis. *Current Biology*, 7:R129–R131, 1997.

[74] D.M. Hillis, J.J. Bull, M.E. White, M.R. Badgett, and I.J. Molineux. Experimental approaches to phylogenetic analysis. *Syst. Biol.*, 42:90–92, 1993.

[75] D.M. Hillis, C.Moritz, and B.K. Mable. *Molecular Systematics*. Sinauer Associates, Sunderland, MA, 1996.

[76] D.M. Hillis and J.P. Huelsenbeck. To tree the truth: Biological and numerical simulations of phylogeny. In D.M. Fambrough, editor, *Molecular Evolution of Physiological Processes*, pages 55–67. Rockefeller University Press, 1994.

[77] H.H. Hock. *Principles of Historical Linguistics*. Mouton de Gruyter, Berlin, 1986.

[78] H.M. Hoenigswald. *Language Change and Linguistic Reconstruction.* University of Chicago Press, Chicago, 1960.

[79] E.C. Holmes, M. Worobey, and A. Rambaut. Phylogenetic evidence for recombination in dengue virus. *Mol. Biol. Evol.*, 16:405–409, 1999.

[80] K.T. Huber, E.E. Watson, and M.D. Hendy. An algorithm for constructing local regions in a phylogenetic network. *Mol. Phyl. Evol.*, 19(1):1–8, 2001.

[81] R. Hudson and N. Kaplan. Statistical properties of the number of recombination events in the history of a sample of DNA sequences. *Genetics*, 111:147–164, 1985.

[82] R.R. Hudson. Properties of the neutral allele model with intergenic recombination. *Theor. Popul. Biol.*, 23:183–201, 1983.

[83] J.P. Huelsenbeck, J.J. Bull, and C.W. Cunningham. Combining data in phylogenetic analysis. *Trends in Ecol. and Evol.*, 11(4):151–157, 1996.

[84] J.P. Huelsenbeck, B. Rannala, and B. Larget. A Bayesian framework for the analysis of cospeciation. *Evol.*, 54(2):353–364, 2000.

[85] J.P. Huelsenbeck, B. Rannala, and Z. Yang. Statistical tests of host-parasite cospeciation. *Evol.*, 51(2):410–419, 1997.

[86] D.H. Huson. SplitsTree: a program for analyzing and visualizing evolutionary data. *Bioinformatics*, 14(1):68–73, 1998.

[87] M.A. Huynen and P. Bork. Measuring genome evolution. *Proc. Nat'l Acad. Sci., USA*, 95:5849–5856, 1998.

[88] I.B. Jakobsen and S. Eastel. A program for calculating and displaying compatibility matrices as an aid in determining reticulate evolution in molecular sequences. *Bioinformatics*, 12:291–295, 1996.

[89] I.B. Jakobsen, S.R. Wilson, and S. Eastel. The partition matrix: Exploring variable phylogenetic signals along nucleotide sequence alignments. *Mol. Biol. Evol.*, 14(5):474–484, 1997.

[90] T. Jukes and C. Cantor. Evolution of protein molecules. In H.N. Munro, editor, *Mammalian Protein Metabolism*, pages 21–132. Academic Press, NY, 1969.

[91] S. Kannan and T. Warnow. A fast algorithm for the computation and enumeration of perfect phylogenies when the number of character states is fixed. *SIAM J. Computing*, 26(6):1749–1763, 1997.

[92] Junhyong Kim and Tandy Warnow. Tutorial on phylogenetic tree estimation. Presented at ISMB (Intelligent Systems for Molecular Biology) 1999, Heidelberg, Germany. Available electronically at `http://kim.bio.upenn.edu/~jkim/media/ISMBtutorial.pdf`, 1999.

[93] M. Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *J. Mol. Evol.*, 16:111–120, 1980.

[94] C.G. Kurland, B. Canback, and O.G. Berg. Horizontal gene transfer: A critical view. *Proc. Nat'l Acad. Sci., USA*, 100(17):9658–9662, 2003.

[95] W. Labov. *Principles of language change, Vol. 1: internal factors.* Blackwell, Oxford, 1994.

[96] J.G. Lawrence and H. Ochman. Amelioration of bacterial genomes: rates of change and exchange. *J. Mol. Evol.*, 44:383–397, 1997.

[97] Emmanuelle Lerat, Vincent Daubin, and Nancy A. Moran. From gene trees to organismal phylogeny in prokaryotes:the case of the γ-proteobacteria. *PLoS Biology*, 1(1):e19, October 2003. DOI: 10.1371/journal.pbio.0000019.

[98] W-H. Li. *Molecular Evolution.* Sinauer Assoc., 1997.

[99] W-H. Li and D. Graur. *Fundamentals of Molecular Evolution, Molecular Phylogeny.* Sinauer Associates, Inc, 1991.

[100] D.A. Liberles, D.R. Schreiber, S. Govindarajan, S.G. Chamberlin, and S.A. Benner. The adaptive evolution database (taed). *Genome Biol.*, 2(8), 2001.

[101] V. Lifschitz. Answer set planning. In *Proc. ICLP-99*, pages 23–37, 1999.

[102] C.R. Linder, B.M.E. Moret, L. Nakhleh, A. Padolina, J. Sun, A. Tholse, R. Timme, and T. Warnow. An error metric for phylogenetic networks. Technical Report TR-CS-2003-26, University of New Mexico, 2003.

[103] B. Ma, M. Li, and L. Zhang. On reconstructing species trees from gene trees in terms of duplications and losses. In *Proc. 2nd Ann. Int'l Conf. Comput. Mol. Biol. (RECOMB98)*, pages 182–191, 1998.

[104] W. Maddison. A method for testing the correlated evolution of two binary characters: are gains or losses concentrated on certain branches of a phylogenetic tree? *Evol.*, 44:304–314, 1990.

[105] W.P. Maddison. Phylogenetic histories within and among species. *Experimental and molecular approaches to plant biosystematics. Monographs in systematics*, 53:273–287, 1995. Missouri Botanical Garden, St. Louis.

[106] W.P. Maddison. Gene trees in species trees. *Systematic Biology*, 46(3):523–536, 1997.

[107] V.H. Mair, editor. *The Bronze Age and Early Iron Age Peoples of Eastern Central Asia.* Institute for the Study of Man, Washington, 1998.

[108] V. Makarenkov. T-REX: Reconstructing and visualizing phylogenetic trees and reticulation networks. *Bioinformatics*, 17(7):664–668, 2001.

[109] V. Makarenkov and P. Legendre. From a phylogenetic tree to a reticulated network. *J. Comput. Biol.*, 2003. Accepted, to appear.

[110] J.P. Mallory. *In Search of the Indo-Europeans.* Thames and Hudson, London, 1989.

[111] V. Marek and M. Truszczynski. Stable models and an alternative logic programming paradigm. In *The logic programming paradigm: a 25-year perspective*, pages 375–398. Springer-Verlag, 1999.

[112] E.P. Martins. Phylogenies and comparative data, a microevolutionary perspective. *Phil. Trans. R. Soc. Lond. B*, 349:85–91, 1995.

[113] N. McCain and H. Turner. Satisfiability planning with causal theories. In A. Cohn, L. Schubert, and S. Shapiro, editors, *Proc. 6th Int'l Conf. on Principles of Knowledge Representation and Reasoning*, pages 212–223, 1998.

[114] L.A. McDade. Hybrids and phylogenetic systematics II: The impact of hybrids on cladistic analysis. *Evol.*, 46:1329–1346, 1992.

[115] G.F. McGuire, F. Wright, and M.J. Prentice. A graphical method for detecting recombination in phylogenetic data sets. *Mol. Biol. Evol.*, 14:1125–1131, 1997.

[116] G.F. McGuire, F. Wright, and M.J. Prentice. A Bayesian model for detecting past recombination events and DNA multiple alignments. *J. Comput. Biol.*, 7(1-2):159–170, 2000.

[117] F. McMorris, T. Warnow, and T. Wimer. Triangulating vertex colored graphs. *SIAM Journal on Discrete Mathematics*, 7(2):296–306, 1994.

[118] G. McVean, P. Awadalla, and P. Fearnhead. A coalescent-based method for detecting and estimating recombination from gene sequences. *Genetics*, 160:1231–1241, 2002.

[119] C. Medigue, T. Rouxel, P. Vigier, A. Henaut, and A. Danchin. Evidence for horizontal gene transfer in E. coli speciation. *J. Mol. Biol.*, 222:851–856, 1991.

[120] A. Meillet. *La Méthode Comparative en Linguistique Historique.* H. Aschehoug & Co., Oslo, 1925.

[121] T.J. Merritt and J.M. Quattro. Evidence for a period of directional selection following gene duplication in a neurally expressed locus of triosephosphate isomerase. *Genetics*, 159:689–697, 2001.

[122] B.M.E. Moret, U. Roshan, and T. Warnow. Sequence length requirements for phylogenetic methods. In *Proc. 2nd Int'l Workshop Algorithms in Bioinformatics (WABI02)*, volume 2452 of *Lecture Notes in Computer Science*, pages 343–356, 2002.

[123] S.R. Myers and R.C. Griffiths. Bounds on the minimum number of recombination events in a sample history. *Genetics*, 163:375–394, 2003.

[124] L. Nakhleh, B.M.E. Moret, U. Roshan, K. St. John, J. Sun, and T. Warnow. The accuracy of phylogenetic methods for large datasets. In *Proc. 7th Pacific Symp. on Biocomputing (PSB02)*, volume 7, pages 211–222, 2002.

[125] L. Nakhleh, U. Roshan, K. St. John, J. Sun, and T. Warnow. Designing fast converging phylogenetic methods. *Bioinformatics*, 17(90001):S190–S198, 2001. ISMB01 Conference.

[126] L. Nakhleh, U. Roshan, K. St. John, J. Sun, and T. Warnow. The performance of phylogenetic methods on trees of bounded diameter. In O. Gascuel and B.M.E. Moret, editors, *Proc. 1st Int'l Workshop Algorithms in Bioinformatics (WABI01)*, volume 2149 of *Lecture Notes in Computer Science*, pages 214–226, 2001.

[127] L. Nakhleh, J. Sun, T. Warnow, C.R. Linder, B.M.E. Moret, and A. Tholse. Towards the development of computational tools for evaluating phylogenetic network reconstruction methods. In *Proc. 8th Pacific Symposium on Biocomputing (PSB 03)*, pages 315–326, 2003.

[128] L. Nakhleh, T. Warnow, and C.R. Linder. Reconstructing reticulate evolution in species – theory and practice. In *Proc. 8th Ann. Int'l Conf. Comput. Mol. Biol. (RECOMB04)*, pages 337–346, 2004.

[129] I. Niemela. Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25:241–273, 1999.

[130] I. Niemela and P. Simons. Efficient implementation of the well-founded and stable model semantics. In *Proc. Joint Int'l Conf. and Symp. on Logic Programming*, pages 289–303, 1996.

[131] R.G. Olmstead and J.A. Sweere. Combining data in phylogenetic systematics: an empirical approach using three molecular data sets in the Solanaceae. *Syst. Biol.*, 43(4):467–481, 1994.

[132] G. Olsen. The "Newick's 8:45" tree format standard. http://evolution.genetics.washington.edu/phylip/newick_doc.html.

[133] S.P. Otto and J. Whitton. Polyploid incidence and evolution. *Annual Review of Genetics*, 24:401–437, 2000.

[134] R. Overbeek and N. Larsen *et al.* WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction. *Nucleic Acids Research*, 28(1):123–125, 2000.

[135] R. Page and M.A. Charleston. From gene to organismal phylogeny: Reconciled trees and the gene tree/species tree problem. *Mol. Phyl. Evol.*, 7:231–240, 1997.

[136] R. Page and M.A. Charleston. Reconciled trees and incongruent gene and species trees. In B. Mirkin, F.R. McMorris, F.S. Roberts, and A. Rzehtsky, editors, *Mathematical Hierarchies in Biology*, volume 37. American Math. Soc., 1997.

[137] R. Page and M.A. Charleston. Trees within trees: Phylogeny and historical associations. *Trends in Ecol. and Evol.*, 13:356–359, 1998.

[138] P. Pamilo and M. Nei. Relationship between gene trees and species trees. *Mol. Biol. Evol.*, 5:568–583, 1998.

[139] C.A. Phillips and T. Warnow. The asymmetric median tree: a new model for building consensus trees. *Discrete Applied Mathematics*, 71:311–336, 1996.

[140] A. Piccolboni and D. Gusfield. On the complexity of fundamental computational problems in pedigree analysis. Technical Report CSE-99-8, Computer Science Department, University of California, Davis, 1999.

[141] P.J. Planet. Reexamining microbial evolution through the lens of horizontal transfer. In R. DeSalle, G. Giribet, and W. Wheeler, editors, *Molecular Systematics and Evolution: Theory and Practice*, pages 247–270. Birkhauser Verlag, 2002.

[142] W. Porzig. *Die Gliederung des indogermanischen Sprachgebiets in neuer Sicht*. Winter, Heidelberg, 1954.

[143] D. Posada and K.A. Crandall. Evaluation of methods for detecting recombination from DNA sequences: Computer simulations. *Proc. Nat'l Acad. Sci., USA*, 98:13757–13762, 2001.

[144] A. Rambaut and N.C. Grassly. Seq-gen: An application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Comp. Applic. Biosci.*, 13:235–238, 1997.

[145] L.H. Rieseberg. The role of hybridization in evolution—old wine in new skins. *American Journal of Botany*, 82(7):944–953, 1995.

[146] L.H. Rieseberg. Distribution of spontaneous plant hybrids. *Proc. Nat'l Acad. Sci., USA*, 93:5090–5093, 1996.

[147] L.H. Rieseberg. Hybrid origins of plant species. *Annu. Rev. Ecol. Syst.*, 28:359–389, 1997.

[148] L.H. Rieseberg, S.J.E. Baird, and K.A. Gardner. Hybridization, introgression, and linkage evolution. *Plant Molecular Biology*, 42(1):205–224, 2000.

[149] L.H. Rieseberg and S.E. Carney. Plant hybridization. *New Phytologist*, 140(4):599–624, 1998.

[150] L.H. Rieseberg, R. Carter, and S. Zona. Molecular tests of the hypothesized hybrid origin of two diploid Helianthus species (Asteraceae). *Evol.*, 44:1498–1511, 1990.

[151] L.H. Rieseberg and N.C. Ellstrand. What can molecular and morphological markers tell us about plant hybridization? *Crit. Rev. Plant. Sci.*, 12(3):213–241, 1993.

[152] L.H. Rieseberg and C.R. Linder. Hybrid classification: Insights from genetic map-based studies of experimental hybrids. *Ecology*, 80:361–370, 1999.

[153] L.H. Rieseberg and D.A. Warner. Electrophoretic evidence for hybridization between Tragopogon mirus and T. miscellus (Compositae). *Syst. Biol.*, 12:281–285, 1987.

[154] L.H. Rieseberg and J.F. Wendel. Introgression and its consequences in plants. In R. Harrison, editor, *Hybrid Zones and the Evolutionary Process*, pages 70–109. Oxford Univ. Press, 1993.

[155] L.H. Rieseberg, J. Whitton, and C.R. Linder. Molecular marker incongruence in plant hybrid zones and phylogenetic trees. *Acta Botanica Neerlandica*, 45(3):243–262, 1996.

[156] D. Ringe. Some consequences of a new proposal for subgrouping the IE family. In B.K. Bergen, M.C. Plauche, and A. Bailey, editors, *24th Annual Meeting of the Berkeley Linguistics Society, Special Session on Indo-European Subgrouping and Internal Relations*, pages 32–46, 1998.

[157] D. Ringe. Personal communication, May 2002.

[158] D. Ringe, T. Warnow, and A. Taylor. Indo-European and computational cladistics. *Transactions of the Philological Society*, 100(1):59–129, 2002.

[159] D. Ringe, T. Warnow, A. Taylor, A. Michailov, and L. Levison. Computational cladistics and the position of Tocharian. In V. Mair, editor, *The Bronze Age and early Iron Age peoples of Eastern Central Asia*, pages 391–414. 1998.

[160] R.G. Roberts, R. Jones, and M.A. Smith. Thermoluminescence dating of a 50,000-year-old human occupation site in Northern Australia. *Science*, 345:153–156, 1990.

[161] D.R. Robinson and L.R. Foulds. Comparison of phylogenetic trees. *Math. Biosciences*, 53:131–147, 1981.

[162] A. Rokas, B.L. Williams, N. King, and S.B. Carroll. Genome-scale approaches to resolving incongruence in molecular phylogenies. *Nature*, 425:798–804, 2003.

[163] U. Roshan, B.M.E. Moret, T. Warnow, and T. Williams. Performance of supertree methods on various dataset decompositions. In O.R.P. Bininda-Emonds, editor, *Phylogenetic Supertrees.* Kluwer Publications, 2003.

[164] M. Ross. Social networks and kinds of speech-community events. In R. Blench and M. Spriggs, editors, *Archaeology and language I: theoretical and methodological orientations*, pages 209–261. Routledge, London, 1997.

[165] N. Saitou and M. Nei. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4:406–425, 1987.

[166] S.L. Salzberg and J.A. Eisen. Lateral gene transfer or viral colonization? *Science*, 293:1048, 2001.

[167] S.L. Salzberg, O. White, J. Peterson, and J.A. Eisen. Microbial genes in the human genome—lateral transfer or gene loss? *Science*, 292(5523):1903–1906, 2001.

[168] M. Sanderson. Analysis of rates (r8s) of evolution, 2002. http://ginger.ucdavis.edu/r8s/.

[169] M. Sanderson, B.G. Baldwin, G. Bharathan, C.S. Campbell, D. Ferguson, J. M. Porter, C. Von Dohlen, M.F. Wojciechowski, and M.J. Donoghue. The growth of phylogenetic information and the need for a phylogenetic database. *Syst. Biol.*, 42:562–568, 1993.

[170] D. Sankoff and M. Blanchette. Multiple genome rearrangement and breakpoint phylogeny. *J. Comput. Biol.*, 5:555–570, 1998.

[171] S. Sattath and A. Tversky. Additive similarity trees. *Psychometrika*, 42(3):319–345, 1977.

[172] H. Shimodaira and M. Hasegawa. Multiple comparisons of log-likelihoods with applications to phylogenetic inference. *Molecular Biology and Evolution*, 16:1114–1116, 1999.

[173] J. Maynard Smith and N.H. Smith. Detecting recombination from gene trees. *Mol. Biol. Evol.*, 15:590–599, 1998.

[174] Y.S. Song and J. Hein. Parsimonious reconstruction of sequence evolution and haplotype blocks: Finding the minimum number of recombination events. In *Proc. 3rd Int'l Workshop Algorithms in Bioinformatics (WABI03)*, volume 2812, pages 287–302. Springer-Verlag, 2003.

[175] Y.S. Song and J. Hein. Parsimonious reconstruction of sequence evolution and haplotype blocks: Finding the minimum number of recombina-

tion events. In *Proc. 3rd Int'l Workshop Algorithms in Bioinformatics (WABI03)*, volume 2812, pages 287–302. Springer-Verlag, 2003.

[176] M. A. Steel, L. A. Székely, and M. D. Hendy. Reconstructing trees when sequence sites evolve at variable rates. *J. Comput. Biol.*, 1(2):153–163, 1994.

[177] M.A. Steel. Recovering a tree from the leaf colourations it generates under a Markov model. *Appl. Math. Lett.*, 7:19–24, 1994.

[178] U. Stege. Gene trees and species trees: the gene-duplication problem is fixed-parameter tractable. In *Proc. 6th Workshop Algorithms and Data Structures (WADS99)*, volume 1663 of *Lecture Notes in Computer Science*. Springer-Verlag, 1999.

[179] K. Strimmer and V. Moulton. Likelihood analysis of phylogenetic networks using directed graphical models. *Mol. Biol. Evol.*, 17:875–881, 2000.

[180] K. Strimmer, C. Wiuf, and V. Moulton. Recombination analysis using directed graphical models. *Mol. Biol. Evol.*, 18(1):97–99, 2001.

[181] D. L. Swofford. PAUP*: Phylogenetic analysis using parsimony (and other methods), 1996. Sinauer Associates, Underland, Massachusetts, Version 4.0.

[182] A. Taylor, T. Warnow, and D. Ringe. Character-based reconstruction of a linguistic cladogram. In J.C. Smith and D. Bentley, editors, *His-*

torical Linguistics 1995, Volume I: General issues and non-Germanic languages, pages 393–408. Benjamins, Amsterdam, 2000.

[183] S.A. Teichmann and G. Mitchison. Is there a phylogenetic signal in prokaryote proteins? *J. Mol. Evol.*, 49:98–107, 1999.

[184] A.R. Templeton, K.A. Crandall, and C.F. Sing. A cladistic analysis of phenotypic associations with haplotypes inferred from restriction endonuclease mapping and DNA sequence data. III. Cladogram estimation. *Genetics*, 132:619–633, 1992.

[185] A. Tholse. Phylogenetic networks. Master's thesis, University of New Mexico, Albuquerque, New Mexico, August 2003.

[186] L. Wang, K. Zhang, and L. Zhang. Perfect phylogenetic networks with recombination. *J. Comput. Biol.*, 8(1):69–78, 2001.

[187] T. Warnow. Tree compatibility and inferring evolutionary history. *Journal of Algorithms*, 16:388–407, 1994.

[188] T. Warnow. Mathematical approaches to comparative linguistics. *Proc. Natl. Acad. Sci.*, 94:6585–6590, 1997.

[189] T. Warnow, D. Ringe, and A. Taylor. Reconstructing the evolutionary history of natural languages. Technical Report IRCS Report 95-16, Institute for Research in Cognitive Science, University of Pennsylvania, Philadelphia, 1995.

[190] T. Warnow, D. Ringe, and A. Taylor. Reconstructing the evolutionary history of natural languages. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 314–322, 1996.

[191] J.P. White and J.F. O'Connell. *A Prehistory of Australia, New Guinea, and Sahul.* Academic Press, New York, 1982.

[192] J.J. Wiens. Combining data sets with different phylogenetic histories. *Syst. Biol.*, 47:568–581, 1998.

[193] T. Williams and B.M.E. Moret. An investigation of phylogenetic likelihood methods. In *Proc. 3rd IEEE Symp. on Bioinformatics and Bioengineering BIBE'03*, pages 79–86. IEEE Press, 2003.

[194] C. Wiuf, T. Christensen, and J. Hein. A simulation study of the reliability of recombination detection methods. *Mol. Biol. Evol.*, 18(10):1929–1939, 2001.

[195] M. Worobey. A novel approach to detecting and measuring recombination: New insights into evolution in viruses, bacteria, and mitochondria. *Mol. Biol. Evol.*, 18(1):1425–1434, 2001.

[196] S.Z. Xu. Phylogenetic analysis under reticulate evolution. *Mol. Biol. Evol.*, 17(6):897–907, 2000.

[197] Z. Yang. Estimating the pattern of nucleotide substitution. *J. Mol. Evol.*, 39:105–111, 1994.

[198] S. Yooseph. *Phylogeny Construction and Consensus Methods.* PhD thesis, University of Pennsylvania, 1997.

[199] G. L. Yule. A mathematical theory of evolution based on the conclusions of dr. j.c. willis, frs. *Philosophical Transactions of the Royal Society*, 213:21–87, 1924.

[200] H. Zhu, J.F. Klemic, S. Chang, P. Bertone, A. Casamayor, K.G. Klemic, D. Smith, M. Gerstein, M.A. Reed, and M. Snyder. Analysis of yeast protein kinases using protein chips. *Nature Genetics*, 26(3):283–289, 2000.

[201] D. Zwickl and D. Hillis. Increased taxon sampling greatly reduces phylogenetic error. *Systematic Biology*, 51(4):588–598, 2002.

Vita

Luay Nakhleh was born in Israel on May 8, 1974. He received his Bachelor of Science degree in June 1996, from the Department of Computer Science at the Technion—Israel Institute of Technology. He received his Master of Computer Science degree in December 1998, from Texas A&M University. He started his PhD studies at the Department of Computer Sciences at The University of Texas at Austin in January 1999. He expects to receive his PhD degree in Computer Science in May 2004.

Mr. Nakhleh's research interests fall into the general areas of computational biology and bioinformatics. In particular, he works on computational phylogenetics and its applications in biology and historical linguistics. He has published numerous peer-reviewed articles in internationally renowned conferences and journals since the year 2001.

Permanent address: 12100 Metric Blvd. #1424
Austin, Texas 78758

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.