Phylogenetic Inference for Multidomain Proteins

Maureen Stolzer
mstolzer@andrew.cmu.edu

August 2, 2011

Department of Biological Sciences
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Dannie Durand, Chair
Veronica Hinman
Russell Schwartz
Craig Nelson (University of Connecticut)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.

Keywords: phylogenetics, molecular evolution, multidomain, reconciliation, membrane-associated guanylate kinase (MaGuK)

For Mom and Dad.
Without your unending support and encouragement,
this would not have been possible.

Abstract

In this thesis, I present a model of multidomain evolution with associated algorithms and software for phylogenetic analysis of multidomain families, as well as applications of this novel methodology to case-studies and the human genome.

Phylogenetic analysis is of central importance to understanding the origins and evolution of life on earth. In biomedical research, molecular phylogenetics has proved an essential tool for practical applications. Current molecular phylogenetic methods are not equipped, however, to model many of the unique characteristics of *multidomain* families. Genes that encode this large and important class of proteins are characterized by a mosaic of sequence fragments that encode structural or functional modules, called *domains*. Multidomain families evolve via *domain shuffling*, a process that includes insertion, internal duplication, and deletion of domains. This versatile evolutionary mechanism played a transformative role in major evolutionary transitions, including the emergence of multicellular animals and the vertebrate immune system.

Multidomain families are ill-suited to current methods for phylogeny reconstruction due to their mosaic composition. Different regions of the same protein may have different evolutionary histories. Moreover, a protein may contain domains that also occur in otherwise unrelated proteins. These attributes pose substantial obstacles for phylogenetic methods that require a multiple sequence alignment as input. In addition, current methods do not incorporate a model of domain shuffling and hence, cannot infer the events that occurred in the history of the family. I address this problem by treating a multidomain family as a set of co-evolving domains, each with its own history. If the family is evolving by vertical descent from a conserved set of ancestral domains, then all constituent domains will have the same phylogenetic history. Disagreement between domain tree topologies is evidence that the family evolved through processes other than speciation and gene duplication. My algorithms exploit this information to reconstruct the history of domain shuffling in the family, as well as the timing of these events and the ancestral domain composition. I have implemented these algorithms in software that outputs the most parsimonious history of events for each domain family. The software also reconstructs a composite family history, including duplications, insertions and losses of all constituent domains and ancestral domain composition.

My approach is capable of more detailed and accurate reconstructions than the widely used *domain architecture* model, which ignores sequence variation between domain instances. In contrast, my approach is based on an explicit model of events and captures sequence variation between domain instances. I demonstrate the utility of this method through case studies of notch-related proteins, protein tyrosine kinases, and membrane-associated guanylate kinases. I further present a large-scale analysis of domain shuffling processes through comparison of all pairs of domain families that co-occur in a protein in the human genome. These analyses suggest that *(1)* a remarkably greater amount of domain shuffling may have occurred than previously thought and *(2)* that it is not uncommon for the same domain architecture to arise more than once through independent events. This stands in contrast to earlier reports that convergent evolution of domain architecture is rare and suggests that incorporating sequence variation in evolutionary analyses of multidomain families is a crucial requirement for accurate inference.

Acknowledgments

I would first like to extend my greatest gratitude to my advisor, Dannie Durand, for her extraordinary support and encouragement. Dannie is one of the best research mentors in the world, and without her continued guidance, this work would not have been possible. I am sincerely grateful for the countless hours and late-night discussions on everything from my research to my academic career and class schedules, in addition to the incredible (and detailed) feedback on papers, presentations, posters, and thesis drafts. In my years at Carnegie Mellon University, Dannie has taught me how to be a great scientist, in addition to a productive member of the scientific community. Her confidence in me has brought a great, positive change in my confidence as a scientist — without her support and encouragement I would likely never have pursued my M.S. in Machine Learning. In addition, she has provided me with invaluable opportunities to improve not just my research skills, but also my skills in teaching, mentorship, critical reviews, and communication.

I would also like to thank all my committee members, Veronica Hinman, Russell Schwartz, and Craig Nelson, for their support, guidance, and insight. Over the years, they have provided valuable advice and suggestions on research and publication. The quality of this work would be diminished without their comments. I would also like to thank my M.S. advisor, Larry Wasserman, for his insight into "all of statistics."

This work would not have been possible without many other people. I would like to thank all members of the Durand Lab — Benjamin Vernot, Annette McLeod, Ravi Chinoy, Han Li, Jacob Joseph, Aiton Goldman, Deepa Sathaye, Minli Xu, Katie Siewert, Robbie Sedgewick, Nan Song, Narayanan Raghupathy, and Rose Hoberman — who have provided considerable scientific discussions and feedback on this work and who have created an amiable work environment. I would especially like to acknowledge Ben Vernot — this work would not have been possible without you. Ben was responsible for teaching me all I know about reconciliation and spent countless hours teaching me Notung code, instilling in me proper coding techniques, mentoring me as a research scientist, and providing constructive feedback on coding, writing, and presentations. He was also a great office mate, keeping me sane on our many all-night writing marathons. I would also like to thank Annette who is the one person that keeps the lab working like a well-oiled machine; Ravi for his hours of work implementing the composite history algorithm; Han for his help with the duplication, loss, transfer and incomplete lineage sorting algorithm and for continuing this work with an algorithm for rearrangement with transfers; Jacob Joseph and Aiton Goldman for their technical support, and especially Jacob for his database management; Deepa for her work implementing

rooting memoization; and Minli and Katie for continuing this project with the MaGuK family, as well as Minli continuing work on cycles.

Finally, so many people have contributed to making the past few years in graduate school rewarding and enjoyable, if not possible. First, I'd like to thank my entire family for their considerable support and understanding. Your pride and confidence in me has made everything worthwhile. I'd also like to thank Nina Senutovitch and Rebecca Bollinger — life-long friends I made through this program. They have kept me sane, in addition to providing a family away from home. I'd like to thank Becky for her guidance during the first years and being a wonderful friend and roommate with whom I could discuss anything. Nina has been a remarkable support system; our weekly dinners and "Lost" viewings provided something to look forward to every week (and, I'm sure, provided my parents with comfort, knowing I was eating a proper dinner at least once a week and not becoming too obsessed with work). Without the encouragement from so many of my friends and family, I don't think I would have made it these many years. Thank you, Mom, Dad, Julia, Lauren, Dave, Calleigh[1], Nina, Becky, Diana, Cara, Ericha, and the Johnson and Stolzer clans. I am overjoyed that my 95 year-old grandmother was able to see this completed and attend graduation.

[1]My niece — I believe how much joy you have brought into everyone's lives in such a short period of time

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Molecular phylogenetics is of central importance to understanding the origins and evolution of life on earth. Phylogenetic analysis has also become an essential technique in life science research in the 21st century, not only for answering evolutionary questions, but for a broad range of functional applications as well. In molecular evolution, phylogenetics is the foundation for investigating the patterns and processes of sequence evolution, identifying signatures of selection, and constructing substitution models. Molecular phylogenetics has proved invaluable for practical applications [1, 2], such as molecular epidemiology [3, 4], cancer progression [5, 6], bioremediation [7, 8], forensics [9–13], tracking rapidly evolving viruses [14, 15], and circumventing pesticide and drug resistance [16, 17].

Phylogeny reconstruction also has proven invaluable for the investigation of many important subjects such as function annotation, drug target selection, estimating species evolution, and correlating events with new functions. In model organism studies, evolutionary trees delineate the degree of functional conservation across species, serving as a guide in planning experiments. In drug design, phylogenetic analysis provides evidence of functional shifts or multiple functional roles, information that is crucial in assessing the suitability of drug targets [1]. Phylogenetic analysis is also an increasing source of inferential power in bioinformatic applications, such as homology based functional annotation [18–20] and predicting functionally active residues from correlated substitutions [21].

Substantial progress has been made in understanding the evolution and function of gene families as a result of 30 years of innovation in phylogenetic algorithms [24–26]. For single-domain families,

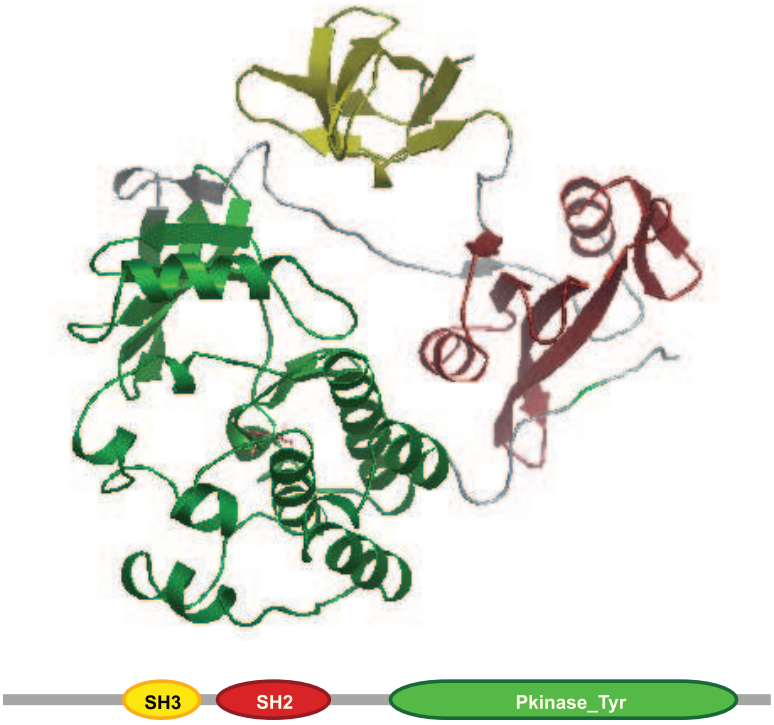


Figure 1.1: Three-dimensional structure of the multidomain tyrosine kinase protein, CSK, with the SH3 domain shown in yellow, SH2 in red, and tyrosine kinase domain in green. (Structure image generated with the Protein Workshop software [22] from PDB entry <1FMK > [23].) A schematic of the linear sequence of domains in the amino acid sequence encoding this structure is shown below with the same color scheme.

phylogeny reconstruction is well-studied, and good algorithms and tools are available. However, current molecular phylogenetic theory is not equipped to model many of the unique characteristics of *multidomain* gene families. Genes that encode this large and important class of proteins are characterized by a mosaic of sequence fragments that encode structural or functional modules, called *domains* (see Fig. 1.1). Multidomain families evolve via *domain shuffling* (Fig. 1.2), a process that includes insertion, internal duplication, and deletion of domains. Multidomain families can, and often do, contain proteins with different domain composition. Moreover, a protein may contain domains that also occur in otherwise unrelated proteins. As a result, different regions of the same protein may have different evolutionary histories [27–31]. Yet, standard phylogenetic methods typically rely on the implicit assumption that the entire sequence has the same evolutionary history. As a result, if a family contains varied architectures, it is not, in general, possible to construct a multiple sequence alignment (MSA) that captures all of the domains represented in family members. Also, domain shuffling events, inherently horizontal processes, are not included in these models. Thus, even if the family can be aligned, the reconstruction process will not infer

the shuffling events that occurred in the history of the family.

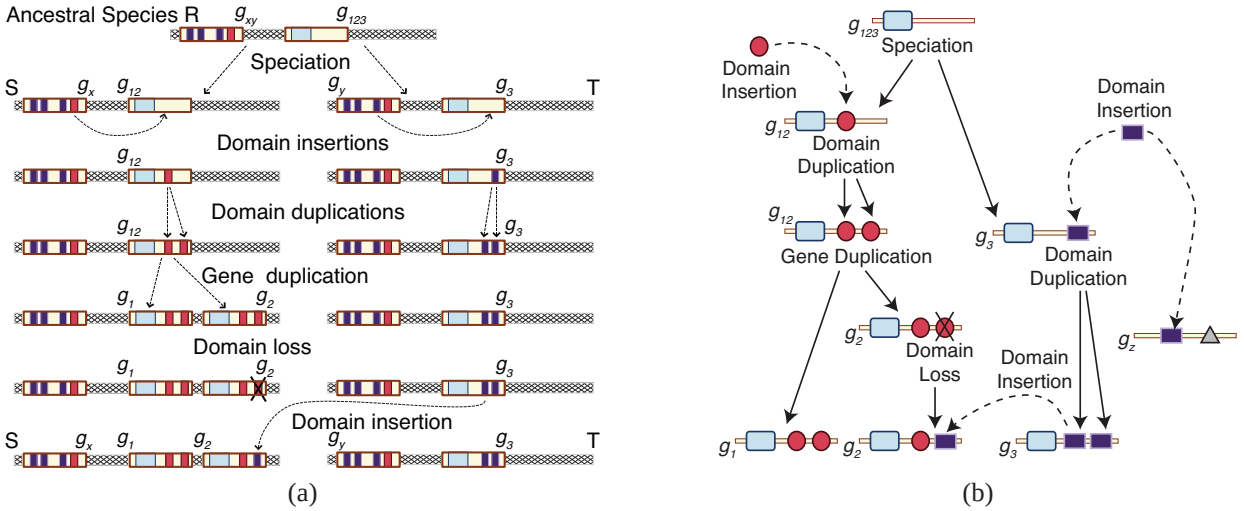


Figure 1.2: **(a)** Evolution of a hypothetical multidomain gene family in their genomic context. Hatched bars represent the chromosome; cream rectangles represent genes, and colored boxes represent domains. **(b)** Schematic of the multidomain family in (a) evolving by gene duplication and domain shuffling. Horizontal lines are genes; circles, boxes and triangles are domains. Solid arrows represent gene evolution; dotted arrows represent domain insertions. Note that inserted domains can originate inside or outside the gene family. Note that gene g_z does not appear on the chromosome in (a).

Phylogenetic methods for analyzing multidomain gene families are urgently needed because of their prevalence and functional importance. The percentage of eukaryotic proteins with two or more domains is 40% according to conservative estimates [32]. The expansion in size and complexity of multidomain families is closely linked to the evolution of cell signaling and cell adhesion. These families are also implicated in tissue repair, apoptosis, inflammation response, antigen recognition, and innate immunity [33–35]. Recent studies have also established strong links between multidomain proteins and mutations associated with cancer [36–39].

The **goal of my thesis** is to meet this need by developing phylogenetic methods to reconstruct the evolutionary history of a multidomain family. Here I present a novel methodology to infer the specific domain shuffling events in a multidomain family, as well as the domain architectures of ancestral proteins. This methodology builds on the locus model of family evolution where a gene family is defined as the descendants of an ancestral chromosomal locus. Consistent with this model, I define a multidomain reconciliation framework in which the domain superfamilies represented in a multidomain gene family are modeled as co-evolving entities. In this framework, a tree is constructed for each domain superfamily, and those trees are combined, using phylogenetic

reconciliation, to obtain a composite history of the gene family.

My methodology stands in contrast to prior, non-phylogenetic approaches investigating multidomain evolution. First, this novel reconciliation framework models multidomain evolution on multiple levels of biological organization: species, genes, domains, and sequence. Second, by taking trees for all domain superfamilies as input, my approach captures sequence variation within each domain family. Moreover, it is sufficiently informative to enable inference of the origin of an inserted domain. Third, the reconciliation framework includes a more detailed, explicit model of the events that mediate domain shuffling, rather than just domain gain and domain loss.

This methodology will allow evolutionary biologists to investigate processes of domain shuffling, bench biologists to identify functional orthologs for domains and multidomain genes, and cell biologists studying a specific multidomain family to complement experimental results with a phylogenetic analysis. The increased sensitivity associated with my method has the potential to yield very different conclusions about domain shuffling processes than those obtained using models that make more simplifying assumptions and take advantage of less information. Published articles that use the phylogeny of a single constituent domain to represent the evolution of an entire multidomain gene family are increasingly common. Since it is difficult to assess how often a tree based on a single domain accurately represents the history of other domains in the family, it is possible that many published accounts of multidomain family evolution are misleading. Using my methods, it is possible to assess the extent to which all domains in the family have the same history. Further, many analyses of multidomain evolution are based on the assumption that proteins with the same domain architecture evolved by vertical descent. Such methods cannot infer replacement of one instance with another, different instance from the same domain superfamily. By ignoring replacements that modify the protein but not the domain architecture, current methods may fail to recognize evidence of substantial changes in ligand specificity or protein sub-cellular localization. In contrast, because it uses information about the sequence of individual domain instances, my approach is capable of recognizing changes in a protein due to domain replacement.

Roadmap to this thesis. To set my methodology in the context of prior work, in Chapter 2, I will review current knowledge in the relevant areas of evolution and computational biology. In Chapter 3, I review the locus model of gene family evolution and prior work on reconciliation of co-evolving entities. I then present my novel approach to inferring the history of a multidomain gene family. In Chapter 4, I present four novel algorithms to realize this approach, all of which have been implemented in the NOTUNG software. These algorithms include: *(1)* Reconciliation

of a binary embedded tree with a non-binary reference tree to infer duplication, heuristic loss, and incomplete lineage sorting events; *(2)* Reconciliation of a binary embedded tree with a binary reference tree to infer horizontal transfer, duplication, and loss events; *(3)* Reconciliation (combining (1) and (2)) of a binary embedded tree with a non-binary reference tree to infer horizontal transfer, duplication, heuristic loss, and incomplete lineage sorting events; and *(4)* An algorithm to infer the composite history of events and ancestral states in a reference tree from a set of reconciled embedded trees. The next chapters present my application of these algorithms to various data sets in order to demonstrate the applicability and power of my approach and associated algorithms. In Chapter 5, I present my evolutionary analysis of three well-studied multidomain families from the literature: the protein tyrosine kinases [27], the Notch-related genes [28], and the membrane-associated guanylate kinases [30]. A discussion of the results from these analyses, and how they demonstrate the power of my approach over previous work, follows. Then, in Chapter 6, I present the results of a high-throughput analysis of domain shuffling in the human genome. This analysis demonstrates the types of genomic-scale information that can be inferred with my approach, as well as the applicability of my algorithms to large-scale studies. Finally, in Chapter 7, I discuss the strengths of my approach, including its power to infer domain shuffling events, and propose several promising directions for future research.

Chapter 2

Evolution of genes, domains and multidomain gene families

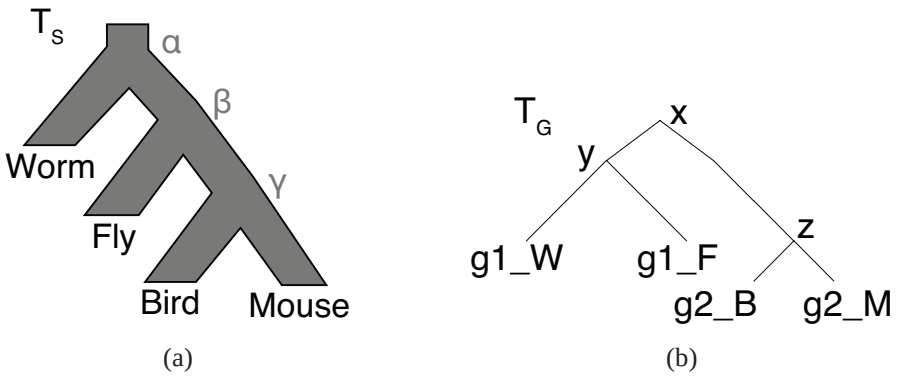
In this chapter, I review current methods for phylogeny reconstruction. I also summarize methods for modeling domains, as well as some of the current domain databases. This chapter also includes a discussion on the molecular mechanisms behind domain shuffling and a survey of previous work on the evolution of multidomain proteins and gene families.

2.1. Phylogeny reconstruction

Phylogenetic reconstruction is the process of reconstructing evolutionary histories of genes and species. *Phylogenetic (or evolutionary) trees* are commonly used to describe the evolutionary history of sites, genes, genomes, and species. In general discussions of phylogenetics, the generic term *taxon* is used to describe the entities on the nodes of an evolutionary tree (i.e., sequences in a gene tree or species in a species tree). A species phylogeny (e.g., Fig. 2.1a) is a tree representing a hypothesis concerning the evolutionary history of a group of species. Leaves represent modern day species. Internal nodes represent speciation events, and the species associated with these nodes are the common ancestral species. Species trees may be constructed from one or many gene sequences; from other types of molecular features, such as intron positions; or from morphological, behavioral, or physiological characters. Regardless of the type of data used, the goal is to reconstruct the history of populations of organisms, not the history of sequences. A key concern when using

sequence data to reconstruct species trees is to ensure that the sequence information used accurately reflects the history of the species. For example, unrecognized gene duplications or lateral gene transfers can result in incorrect species trees. The problem of determining species phylogenies continues to be an active area of research and debate [40]. In this thesis, I assume the species tree is known and focus on the evolution of multidomain families.

A gene tree (e.g., Fig. 2.1b) represents the evolution of a gene family. A *gene family* is a set of *homologous* genes, genes derived from a common ancestor by vertical descent [41]. New family members arise via gene duplication, lateral gene transfer, and speciation. A pair of genes that resulted from a speciation event are referred to as *orthologs*, while a pair of genes that result from a gene duplication are referred to as *paralogs*. Leaves in the gene tree represent contemporary sequences, which may be drawn from organisms in one or in more than one species. Bifurcations represent large-scale evolutionary events, such as speciation, gene duplication, lateral gene transfer, or incomplete lineage sorting. Internal nodes represent ancestral sequences.



(a) (b)

Figure 2.1: Definition of species and gene trees. **(a)** A binary species tree. **(b)** A hypothetical binary gene tree with genes sampled from species in (a).

The first step in reconstructing a gene family tree is to identify all gene family members in the species of interest and obtain a sequence for each family member. Second, these sequences are transformed into an abstract representation from which phylogenetic relationships can be constructed. This abstraction can be represented as either character data or pairwise distances between taxa. Parsimony and probabilistic methods operate on character data, and distance-based methods use pairwise distances. A character is an attribute that can take on two or more states. A *binary* character has two states, typically referring to the presence (1) or absence (0) of the character (e.g., "has wings" or presence of an intron). A *multistate* character has three or more states (e.g., A, C, G, T). These can also have numeric values (e.g., the number of mandibular hairs). Each taxon is described in terms of its particular character states, and mutational change is modeled as a change

Comparison of Phylogenetic Reconstruction Methods

Characteristic	MaxPar	Prob	Dist
Data Type	Character	Character	Distance
Topology	Yes	Yes	Yes
Branch Lengths	Number of Changes	Probability	Distance
Ancestral states	Yes	Most probable state	No
Dominant evolutionary force	Selection	Neutral	Neutral
Multiple substitutions	No	Yes	Yes

Table 2.1: Summary of differences and similarities among the three major methods for phylogeny reconstruction. MaxPar – maximum parsimony; Prob – probabilistic; and Dist – distance.

in character state. For sequence data, this transformation is typically achieved by constructing an MSA. Each column of the MSA is treated as a character. In this case, a change in state corresponds to a substitution of one nucleotide or amino acid for another. Sometimes sequence features, such as introns or small deletions, are treated as binary character states; e.g., the character associated with a particular intron position would be in state 1 if the sequence has the intron state and state 0 if it does not. Finally, the evolutionary tree is reconstructed from the MSA by searching the space of all possible trees, seeking the tree that best explains the MSA with respect to a given model of mutational change [24]. In the case of unrooted trees, for k contemporary taxa, there are $\frac{(2k5)!}{2^{k3}(k3)!}$ different trees to consider; for rooted trees, there are $\frac{(2k3)!}{2^{k2}(k2)!}$ topologies. Many heuristics have been developed to infer a tree without considering every topology; however, these methods are often not guaranteed to find the optimal tree. The resulting tree is a hypothesis for the evolutionary history of the gene family and will provide some or all of the following information, depending on the method used:

- Tree topology (or branching pattern),
- Branch lengths,
- Specific mutations that occurred on each branch, and
- Ancestral sequences.

Current models in molecular phylogenetics [24] fall into three categories: maximum parsimony, probabilistic, and distance-based methods. In general, the three methods differ in both the evolutionary model and the optimality criterion. I will briefly discuss each of these here (see Table 2.1 for a summary).

2.1.1. Parsimony

The primary assumption of parsimony methods is that mutations occur rarely and therefore the tree that requires the fewest mutations, or character state changes, to explain the data is the preferred evolutionary hypothesis [42]. In the small parsimony problem, the topology of the tree and the character states for leaf taxa are given. The goal is to infer ancestral character states of ancestral taxa (internal nodes in the tree) and the minimum (or minimum cost) set of events required for this topology. Calculating the minimum number of changes for a fixed topology is simple and can be determined using Fitch's Algorithm [43]. In the small parsimony problem, the topology of the tree and the character states for leaf taxa are given. The goal is to infer ancestral character states of ancestral taxa (internal nodes in the tree) and the minimum (or minimum cost) set of events required for this topology. Additional criteria can be imposed to model particular types of mutational processes. For example, Camin-Sokal parsimony models irreversible mutations, such as small DNA deletions, by imposing the additional restriction that each character may only change from state 1 to 0 once in the tree. Dollo parsimony, designed for features that are hard to gain, but easy to lose, such as introns, allows each character unlimited changes from 1 to 0, but only one change from 0 to 1.

The goal of the large parsimony problem, given the character states of a set of extant taxa, is to find the tree topology that minimizes the cost of the events required to explain the observed character states. The large parsimony problem is NP-complete: an exact solution requires enumeration and scoring of all possible tree topologies. If the set of taxa is sufficiently large, the problem can be tackled heuristically by sampling the space of tree topologies using a tree rearrangement strategy, such as nearest neighbor interchange or subtree pruning and regrafting. Note that the small parsimony problem is a subproblem of the large parsimony problem, since solution of the small parsimony problem is required to score each enumerated tree.

2.1.2. Probabilistic methods

Under the maximum likelihood framework, given character data D and a model of evolution M, the optimal T is defined to be the tree that maximizes the likelihood of observing the data given the model $P[D|T,M]$ [24, 44, 45]. Like maximum parsimony, phylogeny reconstruction by maximum likelihood estimation is also NP-complete and an exact solution requires enumeration of all tree topologies. In the maximum likelihood formulation, the likelihood, rather than the event cost, is

used to score each candidate tree.

Given the assumption that all positions (columns) in the MSA evolve independently, $P[D|T,M] = \prod_i P[D_i|T,M]$, where D_i is column i in the MSA. The Bayesian framework provides a related probabilistic approach and defines the optimal tree to be that which maximizes the posterior probability of the tree, given the data and the model, $P[T|D,M] = \frac{P[D,M|T]*P[T]}{\sum_T P[D,M|T]*P[T]}$. The probability distribution over all possible trees can be estimated using Markov chain Monte Carlo (MCMC) methods. These samples can then be used to estimate probabilities regarding the true tree.

Typically, the model of evolution is a parametrized Markov model describing the probability of all possible state changes for each character. Thus, it models state changes according to a model of sequence substitution. Depending on complexity, models are able to capture the following properties [24, 46, and works cited therein]: *(1)* the propensity of different types of character changes along a tree (the substitution rate); *(2)* a background base or residue distribution (the propensity of each character to appear), either for the entire sequence or at different sites; *(3)* different rates at different sites (e.g., different substitution rates at the N-terminus and the C-terminus, or, most descriptive, different substitution rates at each position in the sequence) and *(4)* branch lengths for a given tree topology. To calculate the likelihood of a given tree in which the topology and branch lengths are specified, all sites in the MSA and all combinations of internal nodes must be considered. However, branch lengths are usually not provided and must be inferred. For a given topology, there is no analytical method to calculate the optimal branch lengths. Rather, lengths are determined numerically by following the likelihood curve to a maximum.

More realistic models of substitution have more parameter values and are thus able to capture more information; less complex models capture less information. For example, a very simple model of sequence substitution for sites in a DNA sequence (Jukes and Cantor [47]) assumes that all nucleotide substitutions are equally likely, the background base distribution is 25% for each nucleotide, and that each position of the sequence has the same substitution rate. This model has a single parameter, the substitution rate α. Various models represent various compromises between complexity, accuracy, and the necessity to make additional limiting assumptions. As the number of parameters in the model increases, so does computational complexity. The amount of data needed to obtain an accurate estimate of the parameter values increases as well. In addition, different models of evolution may result in different reconstruction results.

In the cases where different models result in grossly different inferences, model selection is a crucial initial step in phylogeny reconstruction. Various statistical measures, such as likelihood

ratio tests and information criteria, have been applied to select the best model of evolution from a set of models [48, and works cited therein]. Under the likelihood ratio test, the relative merit of two models is assessed using the quantity $LRT = 2(l_1 - l_0)$, where l_i is the maximum log-likelihood under model i, and the first model is the more complex of the two. A large value of LRT implies that the more complex model significantly improves the inference and should thus be used; otherwise, the less complex model is favored, since an increase in complexity does not significantly improve results. Information criteria, on the other hand, provides a way to compare all models simultaneously. In this case, the log-likelihood l under each model is penalized by the number of its parameters, k, and, in some cases, the sample size n. The two widely used criteria are the Akaike Information Criterion ($AIC = -2l + 2k$) and Bayesian Information Criterion ($BIC = -2l + k\log n$), both of which represent the loss of information by using a given model. The smaller the value, the better the fit. These three approaches are all well-studied and available in software packages for model selection [48–50].

2.1.3. Distance methods

In addition to the use of character data, a tree can be inferred using *observed* distances between all pairs of taxa. In the context of molecular evolution, o is derived from the aligned sequences in the MSA (e.g., $o_{i,j}$ is defined to be the number of mutations that occurred between i and j). A tree with branch lengths implies a distance $d_{i,j}$ between leaves i and j. For each pair of input taxa i and j, $d_{i,j}$ is the sum of the branch lengths connecting i and j.

If there exists an unrooted tree that exactly fits the distances in o (i.e., $d_{i,j} = o_{i,j}\ \forall\ i, j$), then o is called an *additive* matrix. The four-point condition provides a simple test of additivity: for every four taxa i, j, k and l,

$$\begin{aligned} o_{i,j} + o_{k,l} &\leq \max(o_{i,k} + o_{j,l}, o_{i,l} + o_{j,k}), \\ o_{i,k} + o_{j,l} &\leq \max(o_{i,j} + o_{k,l}, o_{i,l} + o_{j,k}), \\ o_{i,l} + o_{j,k} &\leq \max(o_{i,j} + o_{k,l}, o_{i,k} + o_{j,l}). \end{aligned} \tag{2.1}$$

If there is no error, then observed distances should be additive and the unrooted tree that fits such distances can be found in polynomial time using the Neighbor Joining [51] algorithm. However, in practice, the observed distances are not additive, due to various sources of error, such as multiple

substitutions per site. While the total number of substitutions separating two sequences cannot be observed directly, except in very closely related sequences, it can be (partially) estimated from the number of observed mismatches using the evolutionary models cited above in the discussion of probabilistic methods.

In general, the problem of fitting pairwise distances to a tree is over determined and no exact solution exists. If the observed matrix is not additive, the optimal tree is defined to be the tree, with branch lengths that minimizes the discrepancy between the tree distances d and the observed distances o according to some metric. Common metrics include least squares ($\sum_{i=1}^{k}\sum_{j=i+1}^{k}(o_{i,j} - d_{i,j})^2$), the Fitch-Margoliash criterion ($\sum_{i=1}^{k}\sum_{j=i+1}^{k}\frac{(o_{i,j}-d_{i,j})^2}{o_{i,j}^2}$), and the minimum evolution criterion (the sum of all branch lengths, as determined by least-squares).

2.1.4. Rooting trees

The phylogeny reconstruction methods surveyed above will only infer an unrooted tree. If sequences obey a molecular clock, distances from the root to leaves will be the same for all leaves. A distance matrix with this property is called *ultrametric*. A test for ultrametricity is provided by the three-point condition, which states that for every three taxa i, j, and k,

$$\begin{aligned} o_{i,j} &\leq \max(o_{i,k}, o_{j,k}), \\ o_{i,k} &\leq \max(o_{i,j}, o_{j,k}), \\ o_{j,k} &\leq \max(o_{i,j}, o_{i,k}). \end{aligned} \tag{2.2}$$

In this case, the rooted tree can be inferred from the distance matrix in polynomial time using the Unweighted Pair Group Method with Arithmetic Mean (UPGMA) algorithm. However, for many datasets, the molecular clock hypothesis does not hold, and UPGMA may result in a tree with incorrect topology and/or branch lengths. A second approach in molecular phylogenetics is to use an outgroup species (e.g., an ursine sequence for a set of sequences from canids) or sequence from a more distantly related gene family member (e.g., a myoglobin sequence for hemoglobin data set). In this case, the root of the tree is placed on the branch leading to the outgroup. A third approach, described later in Sec. 3, attempts to place the root such that the number of events (duplication, loss, etc.) are minimized.

2.2. Domain superfamilies

The observation that genes are composed of distinct modules was first introduced in the 1970's [52, 53]. These modules are now commonly referred to as domains. A *domain* is a sequence fragment that acts as an independent module and is found in multiple sequence contexts. Domains have been defined variously as structural units, i.e., sequences that will fold into a well defined, three-dimensional shape independent of the surrounding amino acid sequence; functional units, i.e., sequence fragments that encode a particular function; or as "evolutionary" units, sequence fragments that are found in multiple sequence contexts. In many cases, all three definitions apply. I use the term "domain" as an abstraction of a particular structural fold or functional motif and the term *domain superfamily* to refer to the set of all amino acid sequences that encode that domain. A specific member of the domain superfamily is called a *domain instance*.

There are generally three steps to characterize a domain of interest: discovery, modeling, and recognition. Many of the first recognized domains, such as the Ig domain in 1973 [52, 54, 55] and the Rossmann domain in 1974 [56], were discovered through recognition of structures or sequences that occurred in otherwise unrelated proteins and which had shared similar functions. Today, systematic, *ab initio* approaches to domain discovery are employed [57–59]. Such methods for domain discovery on the sequence level are possible because the large amount of available sequence data makes it possible to recognize conserved patterns automatically. These methods use sequence information with inferred tertiary structure folding, integrating protein folding simulations and unsupervised machine learning techniques, and information about domain sequence properties to predict the likelihood of domain boundaries in sequence data.

Several domain databases have been constructed in the context of projects to obtain a comprehensive characterization of the protein domain universe [60–67] (summarized in Table 2.2). These databases store and organize models for each domain superfamily. Databases differ in their objectives, the definition of domain used, modeling methods, and whether the procedure is automated or curated. When modeling the sequence composition of a domain, the first step is to build a multiple sequence alignment (MSA) based on an all known instances of that domain. It is often valuable to express this information in a compact, yet informative, form. The simplest of these forms is the *consensus sequence* [68], where each residue of the model sequence is the most common residue at that position of the MSA. The *position-specific scoring matrix* (PSSM), or profile, represents the diversity in the MSA as a 20-row matrix that provides the likelihood of each amino acid at every position in the domain. While PSSMs provide more information than consensus sequences, they

do not easily capture gaps or insertions. The profile *hidden Markov model* (HMM) [69] does model this information through incorporation of transition probabilities between states – the conditional probability of the residue at a certain position depends on the previous residue.

Once a model is constructed, it can be used to search sequence databases to find other instances of that domain, which are then used to refine the model. The domain content of a full-length amino acid sequence can be determined by comparing the sequence to all models in one of these databases [60, 63–67] of probabilistic domain models. This typically results in a sequence annotated with starting position, ending position, and name or identifier of each domain identified.

2.3. Multidomain gene families

Multidomain gene families are gene families that encode proteins with two or more domains. These families can, and often do, contain proteins with varied domain architectures. Annotating whole genome sequences with domain databases has revealed the extent of genes that encode multidomain proteins: estimates of the percentage of proteins that have two or more domains range from 27% in prokaryotes and 40% in metazoans [32] to 40% – 60% in prokaryotes to 60% – 80% in eukaryotes [80].

Multidomain gene families are of particular evolutionary and functional importance. Multidomain families played a transformative role in key evolutionary transitions. They expanded preferentially coincident with the emergence of multicellularity in animals. A number of anecdotal studies [35, 81, etc.] have proposed that many metazoan families arose through a pattern of gene duplication followed by domain insertion, yielding the progenitors of major subfamilies in various families involved in cell-cell signaling and cellular adhesion. These subfamilies then expanded through further gene duplication. Additional expansions coincided with chordate and early vertebrate evolution [33, 35, 81].

Gene modularity is a powerful mechanism for the evolution of functional variation or interaction specificity within a gene family that performs a core molecular function. Many multidomain families are associated with fundamental molecular functions such as cell signaling and cell adhesion. As a consequence of their functional repertoire, multidomain families have important health implications, especially for apoptosis, tissue formation and repair, wound healing, immune response, blood-related functions, and the vertebrate nervous system [33–35]. Recent high-throughput screens have established strong links between multidomain proteins and mutations as-

Popular Domain Databases

Database	Type	Curated	Model	Notes
Pfam [61, 62]				
Pfam-A	sequence	yes	HMMs	Uses structure information, when available, to ensure correspondence to only one structural domain, improve the alignment and better define domain boundaries. Focuses on divergent domains.
Pfam-B	sequence	no	HMMs	All other domains which are not in Pfam-A, clustered automatically.
SMART [70, 71]	sequence	yes	HMMs PSSMs	Initial models are based on sequence clustering. Uses structure information when available. Only consists of information on domains and repeats with intra- or extracellular signaling functions.
CDD [72, 73]	sequence	yes	HMMs	Integration of Pfam and SMART domains with links to proteins in Entrez. Hierarchical classification is loosely tied to species evolution. Structure information is used for correction when available. Highly curated.
PRINTS [74]	sequence	yes	PSSMs	Creates an unweighted PSSM-based fingerprint (highly conserved motif) for every domain and uses this for classification.
SCOP/SUPERFAMILY [65]	structure	yes	HMMs	Widely used reference for hierarchical classification and structural information when curating sequence databases. Does not contain any domains without a solved structure.
CATH/GENE3D [75, 76]	structure	some	HMMs	Highly automated, but uses hand curation for determination of difficult domain boundaries and remote folds.
COGs [77, 78]	sequence	yes	MSA	Classification is based on proposed orthologs.
InterPro[a] [66, 79]	both	yes	Various	Integrates information from many domain databases including PROSITE, PRINTS, Pfam, ProDom, SMART, SCOP, CATH, TigrFams, PANTHER, and PIRSF.

Table 2.2: Summary of differences among some of the many databases available for domain identification and classification. Type refers to whether the database is structure or sequence based; Curated refers to whether or not the database is hand-curated.

[a]Note that the InterPro database is not a standard domain identification and classification database. Rather, it attempts to unify the information provided from numerous other databases.

sociated with cancer [36–38]. For example, more than half of the members of the multidomain Kinase family, the largest gene family, have known roles in one or more cancer processes [39]. In addition, the recent literature provides ample evidence of the need for phylogenetic methods for multidomain families, as illustrated by the recent September, 2010 issue of *Science Signaling* devoted to multidomain evolution.

2.4. Molecular mechanisms

Studies of domain and exon shuffling on the genomic level reveal that modular proteins arise through various genetic "accidents," in which aberrations in the cell's replication machinery result in duplications, deletions, and rearrangements of DNA. Our understanding of these processes is rapidly increasing due to two kinds of studies. A small, but growing number of studies describe cases where evidence of the particular domain shuffling mechanism that occurred is still discernible in the flanking DNA of genes that arose very recently [33, 82–108]. In addition, evidence is available from laboratory studies investigating microbial evolution in controlled environments such as chemostats or in cells engineered to have high mutation rates [109, 110].

These studies reveal the primary mechanisms that mediate gene duplication and domain shuffling: segmental duplication, non-allelic homologous recombination (NAHR), retrotransposition, non-homologous end joining (NHEJ), exonization of non-coding sequence, transposon-mediated insertion, and read-through errors [33, 82–108]. Many studies of young genes report novel architectures that arose through a combination of events including a gene duplication and a domain acquisition; one copy of the duplicated gene remains unchanged and can continue to perform the pre-duplication function, freeing the copy that acquired a new domain from purifying selective pressure. Unequal crossing over, typically mediated by NAHR, can increase or decrease the number of internal domain repeats. This process can also create new domain combinations when a fragment containing several genes is copied, if the breakpoints are in the middle of the gene. In this case, a new gene can arise that contains the 5' end of the last (interrupted) gene in the fragment and the 3' end of the first gene in the copied fragment. Transposition, on the other hand, is mediated by transposable elements (TEs) and allows for the integration of new elements into an existing gene. Some TEs facilitate exon shuffling by copying a fragment of DNA and integrating that fragment in another region of the genome. Other TEs, referred to as retrotransposons, can reverse transcribe and integrate themselves in new genomic regions. This can result in the integration of a whole gene, or only a gene fragment. Many gene duplications of this variety result in pseudogenes as the

inserted gene must recruit new regulatory sequence to be functional [95]; however, there are cases where retrogenes acquired a regulatory sequence and are expressed [85, 108]. The introduction of a start and/or stop codon within an exon can eliminate a domain by truncating the coding region. Read-through transcription errors, arising from either the mutation or deletion of a translation stop codon, can result in the fusion of two adjacent genes, creating a single, multidomain gene [83].

Early recognition of the existence of sequence fragments encoding the same modules in otherwise unrelated genes arose in the context of the discovery of introns [53, 111], leading to the hypothesis that the existence of introns increases mutational plasticity and facilitates domain insertions by increasing the probability that the new insertion will fall between domains and not disrupt existing structure [94, 112, 113]. This idea is supported by an observed correlation between exon and domain boundaries [94, 112].

The role of introns in domain shuffling was elaborated further with the concept of *intron phase*, the relative position of an intron between codons in the reading frame. An intron can interrupt a coding sequence between codons, i.e., between the third nucleotide of one codon and the first nucleotide of the next (phase 0); between the first and second nucleotide of a codon (phase 1); or between the second and third nucleotide (phase 2). Analogously, exon phases are defined by the flanking intron phase. A phase *i*-*j* exon is one with an upstream intron in phase *i* and a downstream intron in phase *j*. Symmetric exons are those in phase *i*-*i*. Insertion of a non-symmetric exon or sequence fragment disrupts intron phase. Such disruptions will result in frameshift mutations in downstream exons and are more likely to be deleterious [114, 115]. Insertions are more likely to lead to functional proteins if (a) the inserted fragment is symmetric and (b) the fragment phase is compatible with the phase of the intron where it is inserted (i.e., the insertion is *i*-*i* symmetric and is inserted into a phase *i* intron) [94, 113].

Domain shuffling as a process is more likely to be successful if inserted domains tend to have the same symmetric phase, and if introns in candidate multidomain proteins tend to have phases compatible with those domains. Genomic studies have found that phase 0 introns occur more often than expected, that most genes have a higher number of symmetric 0-0 exons than expected [83], and that there is a significant excess of symmetrical phase 0-0 domains (i.e., domains whose N-terminal exon phase starts with 0 and C-terminal exon phase ends with 0 so that insertion into a phase 0 intron is not disrupted) [94].

2.5. Previous research on multidomain evolution

Studies of genomic sequence can give us detailed understanding of mechanisms by which a particular multidomain protein evolved. However, such analyses are labor intensive, not suited to automation, and only possible in cases involving events so recent that the evidence is still visible.

As previously mentioned, standard phylogenetic methods cannot model multidomain gene families; thus, little work has been carried out on multidomain phylogenetics. However, the availability of domain databases, combined with comprehensive sets of protein sequences for large collections of genomes spanning the tree of life, have leveraged a wave of new investigations on patterns and processes of multidomain evolution using other computational approaches. These methods have attempted to extract evolutionary information without building a tree, or by using only partial phylogenetic information. Domain architecture statistics have been used to gain insight into *(1)* the nature and variety of domain combinations and co-occurrences [116–124]; *(2)* how the domain repertoire varies across genealogical lineages and functional groups [32, 123, 125, 126]; *(3)* plasticity in domain order [127, 128]; *(4)* the relative rates of different types of domain shuffling events [127, 129–132, 132–137]; *(5)* inference of ancestral architectures [96, 130, 134, 138–144]; *(6)* whether domain architecture formation is driven by neutral evolution or natural selection [116, 117, 145–151]; *(7)* the propensity for *convergent evolution* of domain architectures (i.e., the formation of the same architecture more than once through independent events) [139, 142, 152]; and *(8)* the extent of a domain's *promiscuity* (i.e., the propensity of a domain to co-occur with many other domains) [153–155].

2.5.1. Abstract models

First I review the abstract models that have been used in these studies, and then summarize their findings.

Domain architecture models

A commonly used abstract representation of domain content is the *domain architecture*, where each multidomain sequence is treated as a set or sequence of "tokens" (e.g. domain names or database IDs) representing the domain composition from the N- to C-terminus [118]. This ab-

straction has been used to achieve the computational efficiency necessary for genome-scale analyses [96, 154, 156, 157, and work cited therein]. In this framework, all instances of the same domain are indistinguishable. Sequence comparison is used only to determine domain content. Thereafter, sequence variation between domain instances is ignored. The abstract description of the states in the domain architecture model is very similar to the abstract description of genetic sequences — sequences are represented as a string of tokens from the nucleotide or amino acid alphabet; multidomain proteins are represented as a string of tokens from a much larger alphabet of domain superfamilies. The term "domain architectures" is also used to refer to the "bag of domains" model, in which the sequence is treated simply as the set of represented domains. In this model sequence information, domain order, and sometimes the number of copies of each domain, are ignored. A disadvantage of this approach is the possibility of errors due to misannotation (i.e., reporting no domain when a domain exists, or reporting an incorrect domain ID). However, work by Weiner et al. [127] reports that misannotations account for only a small fraction of all putative domains losses.

Event models

The set of molecular mechanisms by which protein architectures change over time are typically treated as a small number of abstract events that modify domain architectures. Different studies have used different event models and, unfortunately, it is not uncommon for two different studies to use the same words to describe different events. Event models can include any of the following: domain merging, fusion, fission, insertion, deletion/loss/death, duplication/gain/birth and innovation.

Domain architectures as character data

Many studies treat abstract domain architectures as a form of character data, as defined in the phylogenetic framework (pg. 8), although the connection between character data and the domain architecture model is frequently not acknowledged. In this model, the set of characters is equated with the set of domains in the protein universe. The domain content of a protein can be treated as binary character data, where 1 indicates presence and 0 indicates absence, or as multistate character data, where presence is represented by a positive number indicating the number of times the domain appears in the protein. A variant of this approach focuses on domain co-occurrence: for a given reference domain, the character state vector represents the set of domains that co-

occur with that reference domain. In a third variant, the taxa are species (or genomes) and domain architectures are treated as characters. For a given genome, the state of the character corresponding to a given domain architecture is non-zero if that architecture has been observed in that genome. Transforming multidomain proteins into character data allows the space of multidomain proteins to be examined by comparing domain architectures. The character state formulation has been applied to investigate various questions about multidomain evolution, often by adapting aspects of the phylogenetic framework, although full realization of multidomain phylogeny reconstruction based on character states has not been achieved.

Two variants of the small parsimony problem have been considered. The first uses the species tree as the fixed topology. Algorithms analogous to Sankoff's algorithm have been proposed to infer the set of domain architectures present in ancestral species and the events responsible for changes in the domain architecture complement over time. If the domain architectures are transformed into presence/absence vectors (i.e., a vector indicating whether each domain is present or absent in the protein), then the events included are either gain/insertion or loss/deletion. This approach has been used to investigate the propensity and spatial biases of various types of domain shuffling events (e.g., whether insertion is more common than duplication or the prevalence of insertions that occur at various locations in the architecture) [129, 140–142]. In investigations of the relative frequency of gene fusion and fission, syntenic information may be incorporated in the analysis as well. This method has also been used to study domain order [129] and convergent evolution of domain architectures [142]. A second variant focuses on changes in domain co-occurrence over time. In this case, the phylogeny is reconstructed from sequences of domain instances of a single superfamily using standard methods [139]. This selected reference domain is used as the fixed topology. Characters represent co-occurring domains. When using a domain tree, constructed from only one domain, it is important to realize that the tree describes only the evolutionary history of that single domain. Trees derived from other domains could have different branching patterns (e.g., the trees in Fig. 3.5) [27–31].

Algorithm 2.1

Input: T the phylogeny to be decorated with architectures, $setDA = \{DA\}$ a set of domain architectures with presence at leaves of T indicated.

```
decorateTree( T, setDA )
 1 r = root( T )
 2 pass1( r )
```

```
 3 pass2( r )

pass1( v )
 4 if isLeaf( v ) do
 5   for each da ∈ setDA
 6     if isPresent( v, da ) do
 7       label( v, da ) = ''present''
 8     else do
 9       label( v, da ) = ''absent''
10     return
11
12 else do
13  pass1( left(v) )
14  pass1( right(v) )
15  for each da ∈ setDA
16     if ( label(left(v), da) == ''present'' && label(right(v), da) == ''present'' ) do
17       label( v, da ) = ''present''
18     else if ( label(left(v), da) == ''absent'' &&
19                                     label(right(v), da) == ''absent'' ) do
20       label( v, da ) = ''absent''
21     else do
22       label( v, da ) = ''unknown''
23     return

pass2( v )
24 if isRoot( v ) do
25  for each da ∈ setDA
26     if ( label(v, da) == ''unknown'' ) do
27       label( v, da ) = ''present''
28
29 else do
30  for each da ∈ setDA
31     if ( label(v , da) == ''unknown'' ) do
32       label( v, da ) = label( parent(v), da )
33     if ( label(v, da) == ''present'' && label(parent(v), da) == ''absent'' ) do
34       event( v, da ) = ''gain''
35     else if ( label(v, da) == ''absent'' && label(parent(v), da) == ''present'' ) do
36       event( v, da ) = ''loss''
37     else do
38       event( v, da ) = ''none''
39
```

```
40  if !isLeaf( v ) do
41     pass1( left(v) )
42     pass1( right(v) )
43  return
```

Annotation of the fixed tree occurs in two passes (see Alg. 2.1). In the first pass, leaves and internal nodes are annotated with the presence/absence of domain architectures or domains. Leaves are assigned the leaf taxon's content. In the case of a species tree, this content is the set of domain architectures observed in that species; with a domain tree, this is the set of domains that co-occur with that domain instance. The ancestral content for internal node *v* is inferred by minimizing the number of gains and losses of content between *v* and its children. Calculating the minimum number of changes for a fixed tree topology is simple and can be determined using a process similar to Fitch's and Sankoff's algorithms [43]. Moving from the leaves to the root of the tree in postorder, content at internal node *v* in the species (domain) tree is inferred as follows: if an architecture (or domain) is observed in both children of *v*, label that architecture (domain) as "present" in *v*. If the architecture (domain) is in neither child, label that architecture (domain) as "absent" in *v*. Otherwise, label the architecture (domain) as "unknown" presence/absence in *v*.

A second pass from the root to the leaves, in preorder, removes the unknown labels by assigning content in *v* the same label as its parent. If the root has an architecture (domain) with an unknown label, the architecture (domain) is assigned as "present." Note that labeling unknown content at the root as present is arbitrary, and this content could also be absent in this node — there is just not enough information to determine. In addition, gains and losses in content are easily calculated in this pass. If an architecture (domain) is "present" in *v* but not in the parent of *v*, then a "gain" of that architecture (domain) is inferred. Otherwise, if the architecture (domain) is "absent" in *v* and "present" in the parent of *v*, then a "loss" is inferred. When using the species tree, an additional layer of event modeling may be included. In this case, for each gained architecture, the new architecture is inferred to be gained by a minimum number of combinations and/or rearrangements of existing architectures. This can be inferred with a simple dynamic programming algorithm (see [140]).

Examples of this algorithm for the family in Fig. 1.2 using both a species tree and a domain tree can be seen in Figs 2.2 and 2.3, respectively. In Fig. 2.2, the species tree is decorated with the observed domain architectures. Species *S* is labeled with the architectures of g_1 and g_2, while species *T* is labeled with g_3. In the first pass, the internal node, representing ancestral species *R*, has all three architectures labeled with "unknown" presence/absence. In the second pass, the architectures are

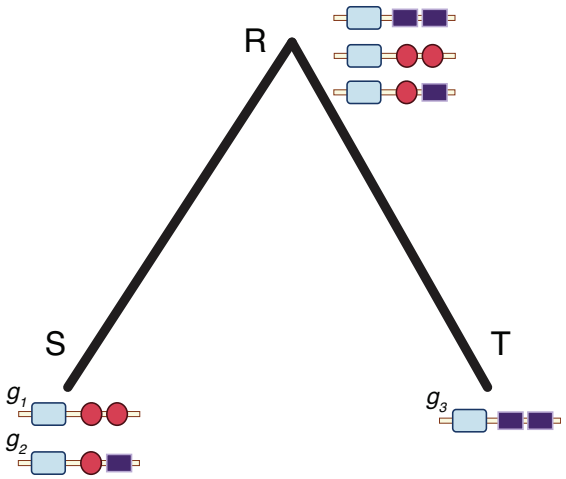


Figure 2.2: The history of the multidomain family in Fig. 1.2 inferred using domain gain-loss parsimony on the species tree.

all labeled as present since the internal node is the root. Thus, the two architectures containing red domains are lost along the edge to T and the architecture with two purple domains is lost along the edge to S. In Fig. 2.3a the tree would result from phylogeny reconstruction of the blue domain sequences. This tree was then annotated with ancestral architectures based on domain presence and absence. In the first pass, the leaves are labeled with the architectures from which the blue domain was sampled. For internal node g_{12} the blue domain and one red domain are labeled as present. The other red domain and the purple domain are labeled as unknown. At internal node g_{123}, the blue domain and one purple domain are labeled as present, and the one red domain (present in g_{12}) and the one purple domain are labeled as unknown, and the other red domain is absent. In the second pass, all domains labeled unknown on the root are assigned a present label, so g_{123} has the blue, one red, and two purple domains labeled present. Node g_{12} is then labeled with the blue and red domains originally labeled present in the first pass. The unknown red domain is labeled as absent and the purple domain as present to match the domains at the root. This annotation implies that a red domain was lost along the edge to g_3, a purple domain was lost on the edge to g_{12}, a red domain was gained and a purple domain lost along the edge to g_1, and no events occurred leading to g_2. Note that we could also assign unknown domains at the root as absent. In this case, we would infer the annotations seen in Fig. 2.3b, which implies a different set of events on the edges to g_3 and g_{12}: gain of a purple domain along the edge to g_3 and the gain of a red domain along the edge to g_{12}.

One attraction of the domain architecture model is that the abstract description of the multidomain

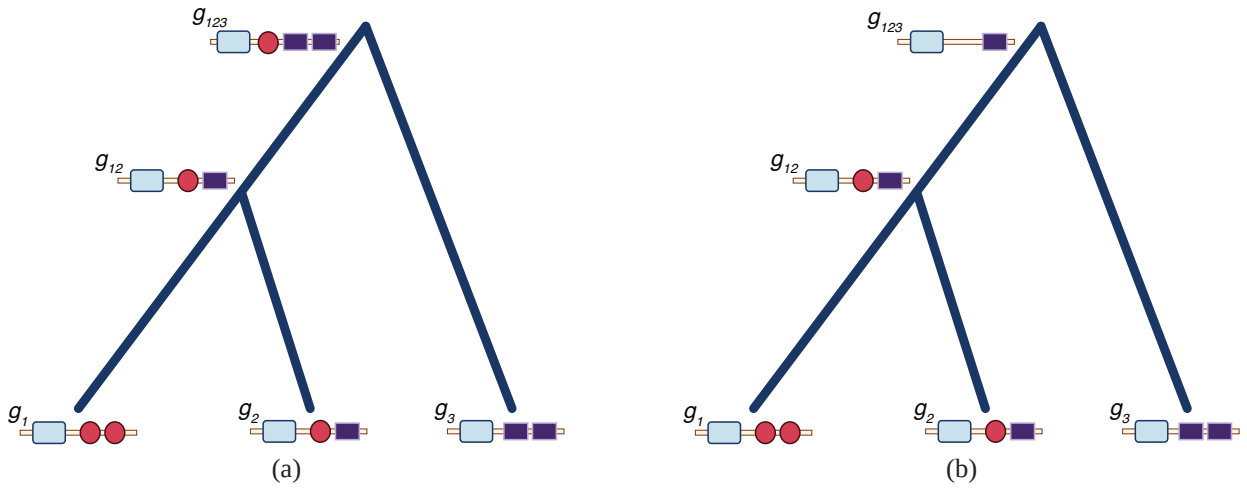


Figure 2.3: The history of the multidomain family in Fig. 1.2 inferred using domain gain-loss parsimony on the tree for the blue domain. **(a)** Ambiguously defined domains at the root are inferred to be present. **(b)** Ambiguously defined domains at the root are inferred to be absent.

protein is very similar to the abstract description of amino acid sequences. This analogy suggests that multidomain phylogeny reconstruction can be carried out by adapting the standard molecular phylogenetic framework to the multidomain realm. Transforming domain architectures into character data allows us to use phylogenetic methods based on character data. Thus, a multidomain tree can be constructed either by calculating the pairwise edit distance between architecture [134] or by employing a parsimony model [138, 143], as described below. Such trees can then be used to infer ancestral architectures (as above) and infer events.

The domain distance between two sequences is the number of domains that differ between the two architectures [134]. This measure is equivalent to the *edit distance*, which is the minimum number of events required to convert one architecture to another (e.g., gain, duplication, loss). Once all pairwise edit distances are calculated, the resulting distance matrix can be used to build trees (i.e., using Neighbor-Joining methods).

In the parsimony context, the domain architecture is represented by the presence/absence vector. In this model, state changes correspond to gaining ($0 \rightarrow 1$) or losing ($1 \rightarrow 0$) a domain. No distinction is made between various events that could result in gain (e.g., domain insertion versus gene fusion). Once architectures are transformed, standard parsimony methods, as discussed in Sec. 2.1 can be used to construct a tree.

A limitation of these methods is that after domains are identified, they do not take the sequence

information of domains into account, which may contradict such inferences. For example, no matter how great the sequence divergence between two genes, if their domain architectures are the same, they will have a distance of zero. Moreover, they do not provide information about the evolution of the multidomain family as a whole and provide only a coarse resolution of events.

Domain presence/absence networks

Another common approach to evolutionary analysis of domain architectures is to represent the protein universe as a domain network or graph, $G = (V, E)$ [117, 121–123, 132, 136, 137, 143, 146, 147, 149, 151, 158]. Each domain in the network is represented by a node $v \in V$ in this network. A pair of nodes, x and y, is connected by an undirected edge $e \in E$ if there is at least one protein in the genome that contains both an instance of domain x and an instance of domain y. The number of edges incident on a node v is called the degree of v. Note that the degree of node v in this network is equivalent to the propensity of domains to co-occur with other domains in the same protein. Variations on the domain graph include using directed edges to indicate domain order or providing edge weights, indicating the number of times two domains appear together in a domain architecture. Similarly, an architecture network has proteins represented by domain architecture as the nodes, and nodes are connected if the two architectures share a domain.

For a global approach, every domain instance or protein in the genome(s) of interest are used to construct the graph. These networks provide information on a global scale, such as the behavior and connectivity of a domain superfamily as a whole. Such approaches have been used to identify power law behavior (discussed later). In a local approach, only the domain instances and proteins in a specified multidomain family are used to construct the network. This approach provides information on domain and architecture behavior in a specific multidomain family. Behaviors of different multidomain families can then be compared. This approach has also been used to determine whether a Dollo parsimony tree can be constructed for the given multidomain family [143, 151].

Birth-death models

Birth-death (and innovation) models (BD(I)Ms) are probabilistic models used to understand the evolution of the size of a domain superfamily [117, 145–147, 159]. BD(I)Ms have been used for modeling a number of biological properties, including populations dynamics, genome evolution, distribution of paralogous gene family sizes [117, 126, 145], and protein-protein interaction net-

works [160,161]. In the context of multidomain evolution, the events considered are birth (gaining a new copy of a domain from an existing domain, through domain or gene duplication), death (loss of a previously existent domain), and, in some models, innovation (introduction of a new domain superfamily through the gain of a new domain fold).

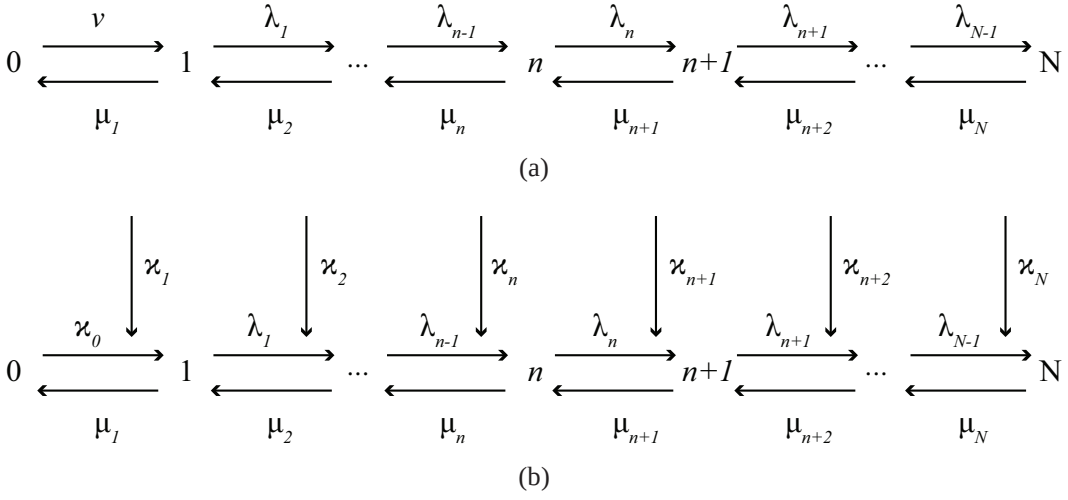


Figure 2.4: Schematics of **(a)** a typical birth-death model and **(b)** birth-death and gain model, with material transitioning between neighboring states. The probability of a transition to the right is a function of λ_n or $(\lambda_n + \kappa_n)$ in (a) and (b), respectively. For both (a) and (b), the probability of a transition to the left is a function of μ_n. The parameters λ_n, κ_n, and μ_n are rates of birth, gain, and death, respectively. (Adapted from Novozhilov et al. [148].)

The BD(I)M is a stochastic, Markov process (see Fig 2.4a) in which transitions are only allowed between neighboring states. We say that the system is in state n when the genome contains n instances of the domain of interest. An increase is a transition from state n to $n+1$ and is termed birth; similarly, a decrease is a transition from state n to $n-1$ and is termed a death. The size of the family at time t is described by random variable $X(t)$, such that

$$\Pr[X(t) = n | X(t_0) = m_0, X(t_1) = m_1, ..., X(t_k) = m_k] = \Pr[X(t) = n | X(t_0) = m_0] \; \forall \, t_i \leq t_0 < t. \quad (2.3)$$

The probability that the family increases from size n to $n+1$ during interval $(t, t+\Delta t)$ is a function of $\lambda_n \Delta t$, where λ_n is the birth rate. The corresponding probability that the family decreases from size n to $n-1$ during $(t, t+\Delta t)$ depends on $\mu_n \Delta t$, where μ_n is the death rate. When the *de novo* innovation event is included, the emergence of a new family with one member (i.e., a transition from state 0 to state 1) is also modeled. In the full BD(I)M, state probabilities, $p_n(t) = \Pr[X(t) = n]$

can then be described by the equations:

$$\frac{\mathrm{d}p_0(t)}{\mathrm{d}t} = -\lambda_0 p_0(t) + \mu_1 p_1(t) \tag{2.4}$$

$$\frac{\mathrm{d}p_n(t)}{\mathrm{d}t} = \lambda_{n-1} p_{n-1}(t) - (\lambda_n + \mu_n) p_n(t) + \mu_{n+1} p_{n+1}(t), 1 \leq n \leq N. \tag{2.5}$$

Note that this model does not take gene family organization into account and focuses only on the size of the domain superfamily.

A substantial number of studies [117,145–148] have used this approach to infer the parameters that best fit real and simulated data. These studies have typically concluded that domain evolution may be a largely stochastic process with natural selection having only a minimal influence on domain superfamilies in extant genomes [117, 145–148].

Unlike the previously described methods, this approach is one of the most challenging due to its computational complexity and the large amount of data required. However, it has the advantage of being able to simultaneously learn information about mutations and insertions/deletions and is the only model that is able to infer event rates.

2.5.2. Summary of findings from previous work

The abstract models discussed above have been used to analyze a range of genomes with multidomain proteins. These studies have provided a good basic understanding of the multidomain universe, including insights into domain architecture patterns across lineages and relative rates of common events. They have also been used to examine whether multidomain evolution is under selection, or whether domains evolve neutrally. Here I highlight some of the important observations and findings that have helped elucidate multidomain evolution.

Domain architecture patterns

The simplest information that these studies provided was a description of the multidomain universe through statistics about domain superfamilies and domain architectures.

Work on domain combinations primarily focused on the domain pair (or triplet), two (respectively three) domains that are found adjacent to one another in the same protein. The domain combi-

nations observed represent a very small fraction of possible domain combinations [112, 118, 125, 162, 163]. In addition, multidomain architectures consisting entirely of new domains are rare, while single-domain architectures are usually the result of the emergence of a new domain, not the fission of a multidomain architecture [130].

Many pairs contain domains from different families [32] – I will refer to these as *mosaic pairs*. Mosaic pairs specific to a particular lineage were assumed to be novel; that is, to have arisen in that lineage. Since the majority of mosaic pairs (ranging from 66% in Eukarya to 90% in Archaea) are comprised of superfamilies common to all lineages, novel pairs are likely the result of common superfamilies combining in novel ways rather than the combination of families specific to that branch [162]. Adjacent domains that appear in two or more distinct domain architectures are referred to as *supra-domains* [118, 119]. In an analysis of two and three domain combinations, approximately one fourth of all the observed combinations qualify as supra-domains [118, 119]. Statistically, over-represented supra-domains were identified and found to exhibit a bias toward eukaryotes.

When the N- to C-terminal orientation of domain was considered, more than 90% of mosaic pairs were seen in only one orientation [125, 128, 162, 164]. Examination of other domains that partner with supra-domains also showed that domain order is generally fixed [118, 119] (however, see Kummerfeld and Teichmann [136]). Promiscuous domains are observed more frequently at the C-terminus [131]. Circular permutations are sets of domain architectures with conserved domain content and order, when the N- and C-termini of the architecture are treated as though they were connected (i.e., the architecture is circular). For example, the architecture *BCA* is a circular permutation of the architecture *CAB*. Circular permutations are thought to evolve by either (a) independent insertions and deletions of domains, which is more common, or (b) duplication of the whole architecture, followed by domain loss at the termini [127]. When circular permutations in domain order are considered to be conserved, the frequency of domain conservation is even greater [127].

Comparative studies across species lineages [32, 122, 123, 125, 126, 130, 145, 147] have revealed patterns that are lineage-specific or common to all lineages. Multidomain evolution in each of the major species lineages (plants, animals, fungi, prokaryotes) has characteristic properties that distinguish it from processes of domain shuffling in other lineages. A core set of domains is found in all lineages, but many domains are lineage specific. The set of most promiscuous domains in prokaryotes, eukaryotes and archaea differs greatly. Multidomain families in multicellular organisms are larger (in the number of domains) and have more complex and varied archi-

tectures [35, 81, 130, 165–168]. This is particularly true for metazoans, and especially vertebrates, with plants running a close second. Multidomain complexity is substantially lower in prokaryotes and even lower in Archaea. Even when domain superfamilies are found across different lineages, they often participate in different architectures [123]. In addition, these lineage specific expansions are associated with specific families or biological processes, including neural and developmental functions [35, 81, 166–169]. A number of promiscuous domains in metazoa are involved in signalling [32, 121, 123, 125, 162].

In addition, these studies have revealed information about the relative propensity of events, including the following observations. Fusion events are more common than fission [129, 130, 132] by a factor of 4 [129] to 5.6 [130]. A fusion event results in two distinct genes joining into one gene, while a fission results in a single gene splitting into two distinct genes. Identification of a gene in one genome that is piece-wise homologous to more than one gene in a second genome suggests that the first gene may have arisen through gene fusion [132]. It has also been observed that indels (domain insertions and/or deletions) were the most frequent elementary event in bacteria genomes [131, 133]. Indels are more common than internal repetitions, tend to involve insertion or loss of a single domain at at time, and occur more frequently at the C and N-termini than in the middle of the architecture [130, 131, 133, 134, 165].

Tandem domain repeats may result from the duplication of more than one domain at a time. Only a small fraction of the proteome contains tandem repeats (i.e., multiple copies of the same domain adjacent in the architecture), and only a small percentage of domain superfamilies participate in repeats [125, 162]. The fraction of proteins with repeats is much higher in eukaryotes, especially vertebrates, than in prokaryotes [170]. Multicellular organisms are more likely to have longer repeats than unicellular organisms [125, 162]. The presence of repeated, adjacent domains is often thought to indicate domain duplication [32, 170]. Analyses of tandem repeats, based on sequence similarity suggest that repeats often expand through internal duplication of several domains at once [170, 171]. A possible explanation for this is offered by recent studies of folding stability and dynamics [80, and work cited therein] that show that tandem repeats of highly similar, homologous domains pose a greater risk of misfolding and aggregation. Wright et al. [172] further report that in titin, which contains many tandem copies of Ig and Fn3 domains, adjacent Fn3 domains are more dissimilar than more distant copies.

Relative importance of neutral evolution and selection in the formation of new architectures

Domains that have been observed to occur in many different protein contexts are referred to as *promiscuous* [153] or *versatile* [125]. Measures proposed for quantifying promiscuity include the number of other domains that co-occur with it in at least one protein (NCO) [125, 153]; the number of other domains found adjacent to it in at least one protein (NN) [120]; the number of different architectures in which it is found (NA) [32, 173]; the number of local architectures (domain triples) in which it is found (NTR) [32]; the weighted bigram frequency index (WBI) [155] and the domain versatility index (DVI) [174].

These are all static measures, based on domain architecture statistics. They cannot distinguish between proliferation by shuffling and proliferation by gene duplication, yet this distinction is very important in the phylogenetic context. A domain could be promiscuous either because it is *mobile* (i.e., frequently inserted into novel architectures) or because it is an *attractor* for insertions (i.e., insertions into the neighborhood of this domain tend to be selectively advantageous). For example, in Fig. 1.2, the blue domain is an attractor; the pink and purple domains are mobile. Yet promiscuity is frequently assumed to be synonymous with mobility. However, this hypothesis has not been tested. Note that the measures of promiscuity cited above can overestimate the mobility of a domain if the domain is, in fact, an attractor. These measures can also underestimate mobility: if domain *A* is observed next to domain *B*, it is not possible to know whether one insertion occurred, or repeated insertions and deletions occurred. To develop phylogenetic models of domain shuffling, it is necessary to be able to make the distinction between attractor domains and mobile domains. For example, according to simple measures, Kinase is a promiscuous domain. However, the evidence in the literature suggests that it is an attractor, not a mobile domain. It is substantially longer than typical promiscuous domains and does not have 0-0 phase [32]. In addition, phylogenomic analysis indicates that the Kinase family evolved by duplication of an ancestral, single-domain Kinase, followed by insertion of different domains into the resulting paralogs, and a second round of duplication [35, 81]. This suggests that the Kinase domain proliferated primarily by gene duplication. In contrast, SH3 and PDZ domains have similar promiscuity scores (except with WBI or DVI) as Kinase. Yet these domains *are* thought to be mobile: they have short, phase 0-0 sequences, do not appear in the single domain context, and are not thought to have evolved as progenitors of gene families [154, 174].

Apic and colleagues [125] argue that domains common to all lineages in the tree of life correspond to the largest proportion of domains and that their widespread phylogenetic distribution suggests

that these domains are ancient, which is consistent with a model of random duplication. However, the same argument does not explain the distribution of domain promiscuity, since the most promiscuous domains are clearly not oldest [121]. To the contrary, there is little overlap between the sets of most promiscuous domains in prokaryotes, eukaryotes and archaea.

A number of studies have observed that domain promiscuity [32, 121, 122, 125, 162, 175] (at least approximately [150]) and domain superfamily sizes [145, 175], can be described by a power law. A power law is a mathematical model in which the the probability of observing characteristic x follows the form $f(x) = c \cdot x^{-m}$, where c and m are constants. In double-log plots, $f(x)$ is a straight line with slope $-m$. This indicates that a small number of domain or gene families are very large or very promiscuous, while most families are small and static.

This observation has led to much speculation regarding multidomain evolution. Many scientists found this observation intriguing in light of a 1999 paper by Barabasi and Alberts [176] that proposed a simple procedure, called *preferential attachment*, that generates random graphs with a degree distribution that follows a power law. In terms of graph theory, preferential attachment theorizes that the probability of adding a new connection to a node is simply a function of node size. In other words, nodes have high connectivity simply because the node is bigger. In terms of domains, this implies that a domain has more, different partners, simply because there are more instances of that domain and a greater chance for another domain to be inserted next to it (or for it to insert next to a different domain). This observation was interpreted as evidence that evidence that domain evolution may, in fact, be neutral [116, 117, 145]

This influenced the implementation of Birth-Death models (see pg. 26) and other simulations of multidomain evolution context, which provided further evidence for neutral evolution [116, 117, 145–147, 149]. In these analyses, the evolution of the multidomain protein universe is simulated under one of two hypotheses: *(1)* domain shuffling events are random, or stochastic, processes (i.e., the growth of domain superfamilies and the formation of novel architectures is a neutral evolutionary process) and *(2)* domain shuffling events are under selection. The alternate hypotheses were evaluated by comparing global features (e.g., degree distribution, clustering coefficient) of the simulated network with the same features in real-world networks. The generative model that yields the best agreement between the network features is considered the best hypothesis for the evolutionary processes that drive multidomain evolution. Once a model has been selected, domain shuffling rates are inferred by selecting model parameters that maximize the similarity between the simulated and real-world networks. Studies implementing such models suggest that while domain shuffling has many characteristics of a preferential attachment process (i.e., neutral evolution) [117,

145–147], the promiscuity distribution generated with a model of selection is a better fit for the data [116]. This suggests that domain combinations have some stochastic properties, but are under selection.

Qian et al. [145] simulated an evolutionary model in which at each time step, a fold instance is selected uniformly at random and duplicated, or a novel fold is introduced with an innovation rate r. They showed that they were able to select parameters that resulted in equivalent size distributions for simulated and real fold families. Koonin and colleagues [117, 146, 147] extended the Birth-Death model to include deletion as well as duplication and innovation. In this case, the gain or loss of a domain is simply based on the size of the domain superfamily. They selected the model that best simulated real data in terms of the exponent the of the power law distribution and the time required to obtain duplication rates similar to those predicted from data by Lynch and Conery [177]. Based on these analyses, they drew the conclusion that multidomain evolution could simply be the result of neutral evolution and large domain superfamilies participating in more events, simply because of their size.

In a related approach, Vogel and colleagues [116] used parametrized generative models to simulate a domain network and test the hypothesis that particular domain combinations observed in nature are the outcome of selection. In the model of selection, a domain combination arises through a single fusion event and then proliferates via gene duplication; the simulator does not permit the same pair of domains to fuse twice. In the null model, all domain architectures are formed by domain insertion at a randomly selected location. Because the promiscuity distribution generated by the model of selection is a better fit for the data, Vogel et al. [116] conclude that domain combinations are under selection and, more specifically, that all instances of a given combination are descended from a single ancestral architecture. This result also suggests that gene duplication is a more common occurrence than domain insertion. Przytycka et al. [143] also simulated random, scale-free graphs using preferential attachment. Comparison of the domain graphs of multidomain superfamilies with simulated domain graphs of the same size and density, showed that these have very different the topological properties. Based on these results, they reject preferential attachment as a mechanism for multidomain protein evolution.

Further evidence that multidomain evolution is under selection has been provided by studies investigating the hypothesis that the convergent evolution of domain architectures is rare [116, 136, 139, 142, 143]. Two studies using quite different approaches, both based on simple domain architecture models, concluded that almost all instances of a given domain combination are descended from a single ancestral architecture [116, 142]. However, the results of more recent studies based on

more complex models, suggest that the propensity for convergent evolution of architectures may be more common than first supposed [139, 143], especially when the architecture contains highly promiscuous domains. Several recent studies of specific multidomain protein families have commented on instances where convergent evolution of domain architectures is the most compelling explanation for similar architectures in distantly related species (e.g. [178, 179]). The frequency of convergent evolution is an urgent question because many bioinformatics analyses are based on the implicit assumption that identical domain architectures must be related through vertical descent. This assumption, if false, could lead to incorrect conclusions in both evolutionary analyses and in practical applications, such as homology-based function prediction.

Tree parsimony methods (described on pg. 21), in particular, have been used to estimate the amount convergent evolution [140, 142]. Gough [142] superimposed domain architectures on a species tree and looked for cases where the same architecture appears in disjoint and distant subtrees, indicating that the same architecture arose in different species through independent events. Less than 4% of the architectures that Gough observed exhibited convergent evolution. Most of these cases correspond to independent instances of internal, tandem duplication. In a complementary approach, Forslund and colleagues [139] constructed trees for individual domain superfamilies, decorated each leaf with the domain architecture in which this domain instance appears, and then applied a standard parsimony analysis to infer domain architectures on internal nodes. If the majority of trees for domains in an architecture agreed on vertical descent or convergent evolution, the architecture was inferred to have arisen in such a manner. Otherwise, no conclusion was made for that architecture. They argue that convergent evolution is more common than previously thought, with approximately 12% of all architectures exhibiting convergent evolution.

Przytycka et al. [143] also tested this hypothesis by proposing two formal parsimony models of domain shuffling and relating these to local topological properties of the domain graph (described on pg. 26). These models are formal encodings of two hypotheses: the same domain pair forms only once in evolutionary history and domain architectures, once formed, persist through evolution. By inspecting local graph structures, the method determines whether it is possible to construct a parsimony tree that satisfies both conditions. Note that the fact that it is possible to construct a most parsimonious tree does not guarantee that the evolution of the superfamily was, in fact, parsimonious. However, if no tree exists that is consistent with the hypotheses *and* consistent with the domain graph for a particular domain superfamily, it can guarantee that convergent evolution played a role in the history of the superfamily. Thus, a negative outcome is more informative than a positive one. Przytycka et al. [143] applied this approach to a genome scale data set of domain

superfamilies and observed that the most promiscuous domain superfamilies did not satisfy the criteria, including Trypsin, EGF, IG, and SH3.

2.6. Summary and critique of previous methods

The benefits of these methods employing the domain architecture models include computational and conceptual tractability. While these approaches have provided a wealth of information on how the protein universe has formed, they are incomplete because they suffer from a number of limitations. Most of these methods are based on underlying assumptions that are mostly untested and often lead to "self-fulfilling prophesies."

First is the assumption that all instances of the same domain are indistinguishable and therefore sequence variation is ignored. However, in fact there is considerable sequence variation within domain superfamilies. A second assumption is that multiple observations of the same domain architecture are the result of vertical descent and formation of the same architecture through independent paths (i.e., convergent evolution) is extremely rare. Even when the architectures are identical, different domains in the architecture could have different histories [27–31] and would not be observed from architecture information alone. Thus, alternative, and possibly contradictory, information is not considered, which can lead to underestimation of events and inaccurate histories (see Figs. 2.2 and 2.3 for an example). More recent studies have used models that incorporate additional information (e.g., phylogenetic structure of individual families [139], domain order [136], local topology [143]) and reached different conclusions. Preliminary evidence suggests that a more detailed model will lead to more complete, if not fundamentally different, conclusions [151].

A third, problematic assumption is that graphs with power law distributed degree distributions arose by preferential attachment. This is not true for all such graphs. Hence, one should be wary about drawing evolutionary conclusions from degree distribution alone. In general, it is important to ensure that graph features used to compare data with a model are, in fact, suitable for distinguishing between alternate models [180]. Moreover, other distributions (Generalized Pareto, Yule Distribution) can masquerade as a power law, especially when the data available is limited [150]. In addition, most studies using generative models to test evolutionary hypotheses have not emphasized the importance of determining whether the features used to compare models actually have discriminative power. And while many of the parsimony-based methods include species and/or gene family classification in the model, the studies based on Birth-Death Models

and domain graphs have not incorporated phylogenomic considerations.

Finally, many studies recognize the change in the form of domain gain and loss, but do not include an explicit model of events. For sequences that evolve by substitution, changes in character state closely reflect mutation events. In contrast, although state changes are caused by duplication, insertion, and loss, it is not possible to determine from the state changes which events occurred because multiple combinations of events could result in the same changes in character state. Most of the methods surveyed here do not incorporate a formal model of events and often do not provide information about the particular insertions, deletions, and rearrangements of domains that gave rise to observed domain architectures. In addition, these studies provide a very coarse resolution of events, especially with the phylogenomic analyses.

Chapter 3

Models of multidomain evolution

Ideally, reconstruction of the evolutionary history of a gene family includes:

- the relationship between taxa, generally represented as a tree,
- ancestral states (i.e., ancestral sequences),
- the history of events (i.e., substitutions, gene duplications, etc.),
- a partial temporal ordering on those events.

Methods and technology to reconstruct the history of sequences that evolved by vertical descent are well-developed. Family reconstruction is of central importance to understanding the origins and evolution of life on earth. Tree reconstruction provides the most accurate way to identify orthologs. Phylogenetic context and ortholog identification provide essential information used in function annotation, where sequences of unknown function are annotated based on related sequences with experimentally determined function. The use of phylogenies and ortholog identification, rather than pairwise sequence comparison, is also essential for identifying potential drug targets. In addition, evolutionary trees provide a common mathematical framework for describing evolution at various levels of biological organization. Comparison of trees across multiple levels makes it possible to relate genetic innovations on the sequence level to physiological innovations on the cellular and organismal levels.

Current technology can infer changes in family copy number (i.e., gene duplication, loss, and transfer) through the comparison of a gene tree with a species tree. However, there is no formal methodology for modeling or inferring changes on an intermediate scale, such as insertion, duplication, and deletion of domains or sequence fragments.

The **goal of my thesis** is to develop, implement and test methods to support phylogenetic analysis of multidomain families (e.g., Fig. 3.6), based on the reconciliation of trees. Formally stated, given:

1. a multidomain gene family,
2. a reference tree,
3. a sequence phylogeny for each domain represented in the family, and
4. an abstract set of events (i.e., domain insertion, domain deletion/duplication, gene duplication, etc) and model of evolution,

my methods will infer

1. the events that occurred in the evolution of the multidomain family,
2. ancestral domain architectures, and
3. a reference tree annotated with domain shuffling events and ancestral architectures, representing the evolution of the multidomain family as a whole.

Before I describe my results, I first introduce the model of multidomain evolution upon which my approach is based and review the evidence that supports this model.

3.1. Locus model of multidomain gene family evolution

Under the *classical model of gene family evolution*, gene families evolve in a process of vertical descent from a common ancestor, through gene duplication, gene loss, lateral gene transfer (LGT), and co-divergence with host speciation [41, 43, 181]. Superimposed on these large scale processes, individual sequences evolve by point mutation. This process of evolution by vertical descent from a common ancestor is modeled by a tree. However, once domain organization is considered, sequences can arise where the constituent parts are derived from two different parent sequences.

I propose a phylogenetic framework based on the locus model, first proposed in [151, 173, 182], that defines evolutionary vertical descent for multidomain families in a way that is consistent with traditional models of homology. These families evolve through the events above *and* through *domain shuffling*, defined as domain insertion, loss, and internal duplication, where an insertion is defined as the acquisition of a new sequence fragment (the "mobile" domain) by an existing gene. This can occur through insertion of sequence fragments into the gene or by recruitment of adjacent exons. Formation of a new gene architecture by domain loss is also consistent with this model. In this *locus model of multidomain gene family evolution*, a gene family, whether it has one or more

domains, is defined as the set of genes descended from a common locus in the genome (see, for example, Fig. 1.2a). This model of multidomain gene families is well-defined for families that evolve through domain shuffling and is consistent with the classical single-domain model. Thus, it provides a natural framework for extending current phylogenetic methodology to multidomain sequences.

The restriction of domain acquisition by an existing gene guarantees that the ancestral locus is always well-defined, even when new domains are acquired by insertion. Therefore, we can define a family as a set of sequences that share a common ancestral locus. For example, Fig. 1.2a shows a hypothetical multidomain gene family evolving in a chromosomal context. The family originates as an ancestral gene g_{123} with a single domain. After a speciation event, the orthologous gene in each species evolves independently, with the copies undergoing domain insertion, domain duplication, domain loss, and gene duplication. The resulting family contains genes g_1 and g_2 in species S and g_3 in species T, all of which evolved from the common ancestral locus, g_{123}, in species R. The purple domains in g_3 share homology with g_y and the red domains in g_1 and g_2 share homology with g_x. However, g_y and g_x are not members of the gene family because they are not descended from locus g_{123}. The history of this family's evolution by vertical descent can be modeled as a tree, as seen in Fig. 1.2b. This gene family tree describes the evolution of the locus by vertical descent, augmented by "transfer edges" representing insertions. Fig. 3.6 shows the history of the locus as a tree in brown, augmented with horizontal insertion events (dashed arrows) and annotations regarding various other events.

Formally, in the context of the locus model, the history of a multidomain family is defined as a directed, acyclic graph, $T = (V, E_v, E_t)$, where V is the set of nodes (i.e, with extant loci as leaf nodes and ancestral loci as internal nodes), E_v is the set of tree edges indicating evolution by vertical descent, and E_t is the set of transfer edges. When only tree edges are considered, $T = (V, E_V)$ is a tree representing the history of the locus, in which every node (gene locus) in V (except the root) has exactly one parent node and every node (except the leaves) has exactly two children. Branch points indicate formation of a new locus by gene duplication or by speciation. An edge $e = (v, w)$ in E_t indicates that a sub-sequence of node v was duplicated and inserted into w. Annotating the multidomain family history with lateral events, E_t, reflecting domain insertions, results in a multidomain family tree with a reticulated (network-like) history (similar to a species trees with transfers), as shown in the augmented tree in Fig. 3.1a. Each node has zero or more incoming transfer edges. Note that genes can include sub-sequences with different histories descended from different "donor" sequences. However, in the context of the locus model, I distinguish be-

tween locus donors (represented by parental nodes and connected by tree edges) and other donors (connected by transfer edges). This is different from a recombination graph, in which a node may have two incoming tree edges both of which contribute equally. In this case, recombination is represented by a network as shown in Fig. 3.1b [183]

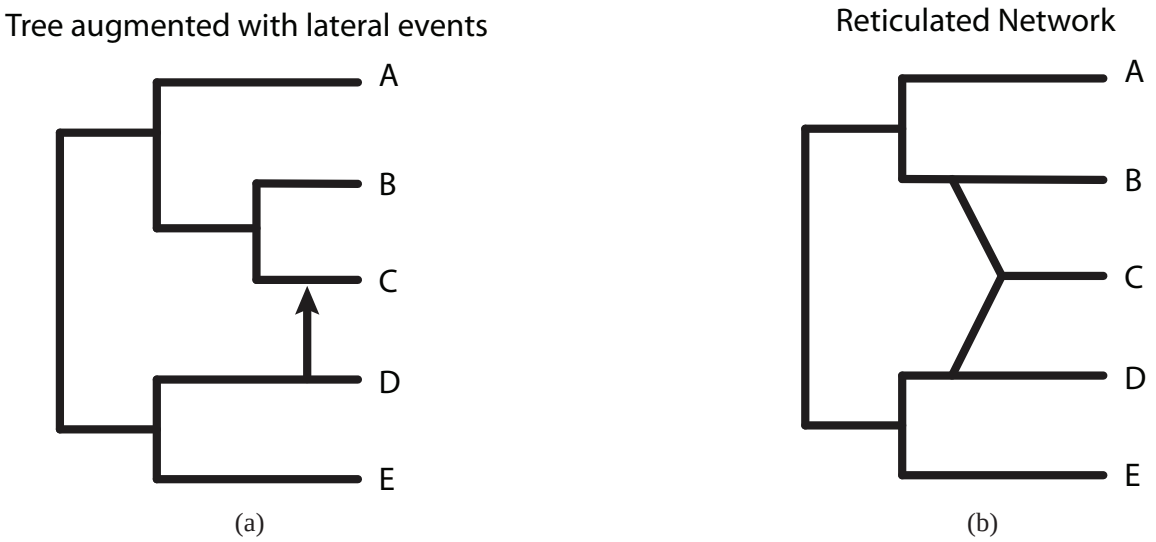


Figure 3.1: Differing views for taxa evolving with horizontal events. **(a)** A species tree shown with a horizontal transfer event from *D* to *C* is a tree augmented with transfer edges. **(b)** A species tree with a hybridization of *B* and *D* contributing to *C* is a network and is not tree-like. Figure adapted from Husen et al. [183].

The locus model is applicable to a broad range of multidomain families (see Song et al. [151] for a discussion of this evidence). While there are families that do not fit this model, in particular, those that families that originated from a *de novo* assembly of unrelated domains, such families are rare [96, 129, 134, 140].

I propose to model the evolution of multidomain families that have evolutionary histories that are consistent with locus model, as a set of co-evolving entities, where a multidomain protein is represented as a co-evolving set or sequence of *domain sequences*. Given a tree for each constituent domain, I propose algorithms to construct an augmented multidomain tree, inferring transfer edges, domain duplications and losses, and ancestral domain content. This framework uses a two phase approach to multidomain phylogeny reconstruction. In phase 1, the amino acid sequence of each domain instance that occurs in the multidomain family is extracted. For each domain family represented, a domain tree is constructed from the sequences using standard molecular phylogenetic techniques. In phase 2, domain shuffling events are inferred by comparing each domain tree with

trees representing higher levels of organization. For example, Fig. 3.6 shows the reconstructed history for the multidomain family in Fig. 1.2a inferred through the comparison (embedding) of domain trees with a gene family tree (Fig. 3.2a) and a species tree (Fig. 3.2b).

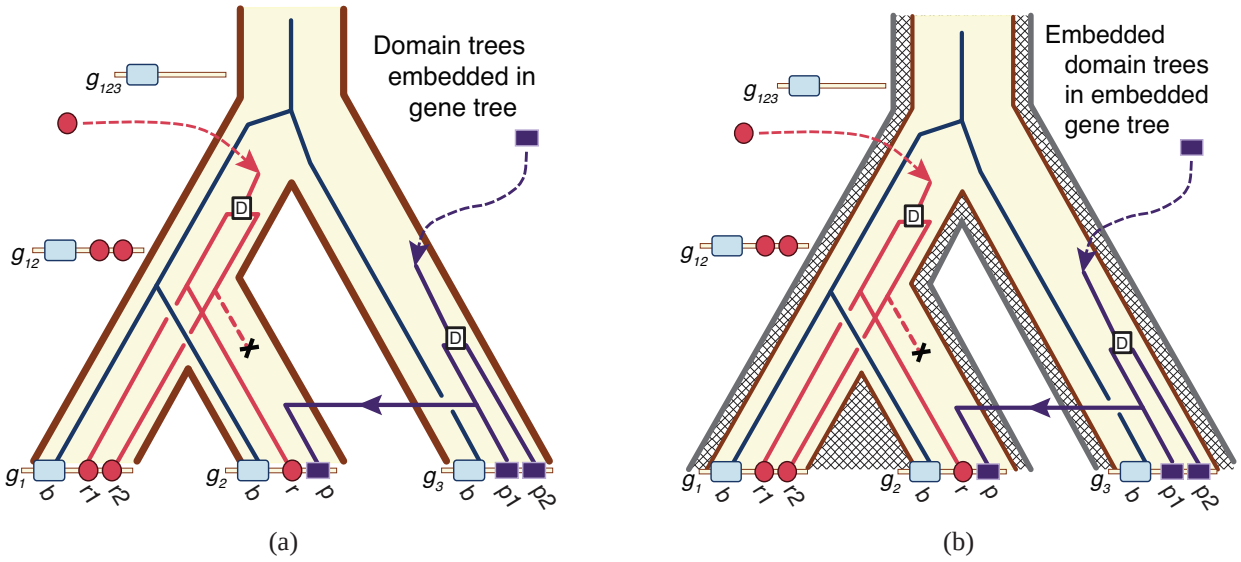


Figure 3.2: The evolutionary history of the hypothetical multidomain family in Fig. 1.2. This figure shows co-evolution on three different levels of organization: species evolution, gene family evolution and domain shuffling. Evolution on a fourth level of organization, nucleotide substitution, is implied but not shown explicitly. **(a)**. Domain trees embedded in the gene family tree, showing the evolution of the domains in the context of genes. Domain trees are represented by thin trees colored according to the domain they represent. The gene tree is the "fat" cream colored tree. **(b)** The evolutionary progression of the genes encoding the family, with embedded domain trees, is shown in the context of the phylogeny of the species that contains them. The species tree is the "fat" hatched filled tree.

This framework builds on a general model of historical associations between co-evolving entities that has also been applied to biogeography, symbiont-host relationships, and co-evolution between genes and species [184–186]. In each case, an entity at one level of biological organization (the *embedded taxon*) evolves in the context of an entity at a different level of biological organization (the *reference taxon*). Using this representation, multidomain phylogeny reconstruction can be carried out by adapting a different, well-established algorithmic framework, called *reconciliation*. In the next section, I describe the history of reconciliation and review the details of this framework in the gene tree/species tree context.

3.2. Gene tree-species tree reconciliation

Reconciliation [187] is the process of comparing trees representing two levels of biological organization to infer the co-evolutionary history of the trees. It relies on the observation that discordance between trees is evidence of evolutionary events other than co-divergence. Key features of this framework include *(1)* an explicit model of the events that determine associations between the embedded and reference taxa; *(2)* inference of historical associations between ancestral taxa from known associations between leaf taxa; *(3)* inference of historical events from comparison of the reference and embedded phylogenies; and *(4)* inference of a reference phylogeny via comparison with several embedded phylogenies. In the multidomain context, domains correspond to residents. The role of reference may be adopted by species, genes, or other domains.

Reconciliation approaches have been adapted in a number of different contexts, namely relationships between gene trees and species trees, between hosts and parasites, and between species and geographical areas [188, and work cited therein], demonstrating the generality of this framework. For each specific context, there is a different instantiation of *reference* taxon (e.g., the species or host) and *embedded* taxon (e.g., gene or parasite). Here I review reconciliation in terms of the gene and species trees context.

Discordance between a gene tree and a species tree is evidence that genes diverged through processes other than speciation. These events include gene duplication and loss, lateral gene transfer (LGT), and incomplete lineage sorting (ILS). Gene duplication, loss, and transfer events are all events that change the number of copies of a gene in a genome. Specifically, transfer events involve the duplication of a gene in one genome, followed by its displacement into another genome. Incomplete lineage sorting, on the other hand, is discordance between a gene and species tree due to allelic variation alone (see Fig. 3.3). Unlike duplication, loss and transfer events, it does not change the number of gene copies in the genome.

Reconciliation encompasses two related problems: *event inference* and *tree estimation*. In the event inference problem, both gene and species trees are known. Given a rooted gene, a rooted species trees, a mapping from leaves in the gene tree to the species from which each gene was sampled, and an evolutionary model, the goal of the *event inference* problem is to *(1)* infer the association between ancestral genes and ancestral species and *(2)* the set of events that best explains this association (illustrated in Fig. 3.4). Formally, the event inference problem is stated as follows: Formally stated, given:

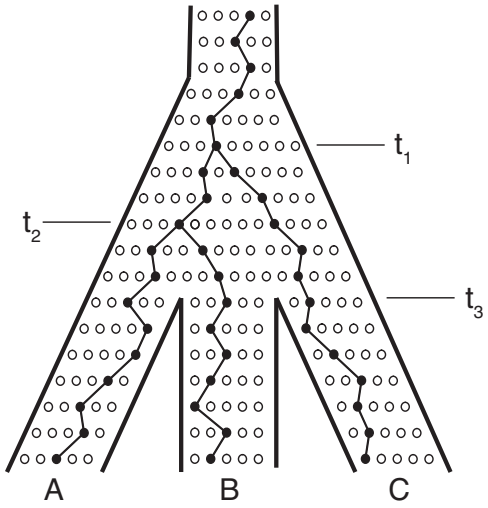


Figure 3.3: Evolution of a single genetic locus in the context of a population. Each row represents a generation of individuals in the population at a specific point in time. The the three possible binary resolutions are shown in Fig. 4.3

Reconciliation: Event Inference

Input:

1. an evolutionary model including
    a. a set of evolutionary events,
    b. an optimization criterion,
2. a binary, rooted species phylogeny,
3. a binary, rooted gene tree, and
4. a mapping of contemporary genes (leaves of the gene tree) to the species from which they are sampled (leaves of the species tree).

Output:

1. an extended mapping of ancestral genes to ancestral species, indicating the species in which each ancestral gene was present and
2. the gene tree augmented with the set of events that best explains the incongruence between the gene and species trees according to the optimization criterion.

If the species tree is not known, event inference can be used to solve the *estimation problem*, i.e., to infer the species tree. In this case, we are given a set of rooted gene trees and a mapping from each leaf in the gene tree to the species from which the gene was sampled. The goal is to infer the best species tree given these trees by searching the space of all possible trees with leaves equal to

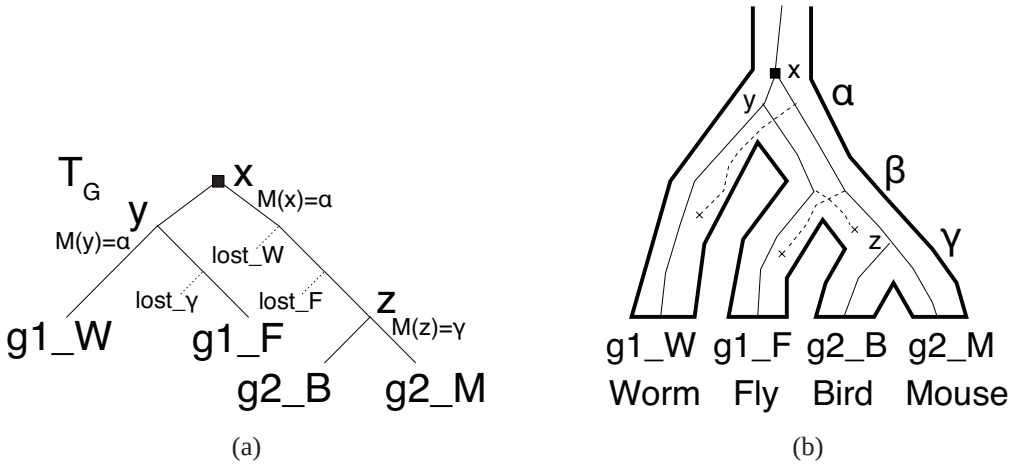


Figure 3.4: LCA reconciliation. **(a)** The gene tree from Fig. 2.1 reconciled with the species tree. **(b)** The gene tree embedded in the species tree. The black squares indicate duplications, and dotted lines indicate losses.

the mapping of contemporary genes. Formally, given

Reconciliation: Tree Estimation

Input:

1. an evolutionary model of events and an optimization criterion,
2. a set of binary, rooted gene trees, and
3. a mapping of contemporary genes (leaves of all the gene trees) to the species from which they are sampled (leaves of the species tree to be inferred).

Output:

1. the species tree that is the best, according to the optimization criterion.

This is achieved by scoring each candidate species tree based on the event inference reconciliation with each gene tree in $\{T_{G_i}\}$. Under event parsimony model, the best species tree is defined as the tree that results in the minimal cost for the sum of the events inferred through event inference. The tree estimation problem under the reduced duplication-loss parsimony model is NP-complete [189]. However, many efficient methods have been proposed in the literature [190–197]. For the remainder of this section, I focus on the event inference problem.

Event inference in reconciliation refers to a family of problems, with each specific instance defined by the event model and the optimization criterion used. Since first proposed by Goodman and colleagues [187], and formalized by Page [198], reconciliation of gene and species trees has been the target of a great deal of theoretical and algorithmic work [189, 198–205], and a number of

software packages for this problem are available [184, 206–213]. Prior theoretical work on the event inference problem, under the parsimony criterion, is summarized in Table 3.1.

Most reconciliation algorithms have focused on an event-parsimony criterion: given a set of events, with a cost associated with each event, the solution is the one that minimizes the total cost of the events required to explain the discordance between trees [187, 198, 214]. Speciation is assigned a zero cost, and duplication, loss, and/or transfer are assigned positive costs. Incomplete lineage sorting may be assigned either a zero cost or a positive cost. Costs are often assigned arbitrarily. A more principled approach is to select costs that maximize the statistical power of the method, where statistical significance is estimated by comparing the cost of the optimal solution with the distribution of costs obtained by reconciling randomized trees under an appropriate null model [188]. In addition to the parsimony approach, probabilistic approaches have been proposed [215–223]. In this case, the optimal reconciliation is the one that maximizes the probability of the inferred events, given the observed gene tree, leaf mapping, event model and event rates. These methods are appropriate for data sets in which duplication and loss are neutral, stochastic processes. Full Bayesian inference (including trees and rates) is computationally intensive and requires sufficient data to learn rates.

Most theoretical work on event inference has also focused on a subset of the event set. Some algorithms infer explicit event histories. That is, they determine whether a given node in the gene tree is a speciation, duplication, transfer, or incomplete lineage sorting and reconstruct the most parsimonious loss histories by inferring the species in which the loss occurred and placing these losses on edges of the gene tree. However, a number of algorithms only calculate the *number* of events, but not their location in the gene tree or species tree. Both approaches have their advantages. For example, simply counting events is a good approach for tree estimation, but for in-depth studies of individual gene families, more information is required and an explicit event history is preferred.

The most general model includes all four events: duplication, loss, transfer, and incomplete lineage sorting. Until now (see Alg. 4.3), no work has included such an event model. However, there has been a great deal of work with other reduced-event models. The most common of the reduced models is the two event (duplication and loss) model, with many groups ignore losses because of the false assumption that losses are unimportant [184, 187, 189, 199, 201, 203, 204, 206–208, 224–226]. One unusual approach by Chauve and colleagues [227, 228] has focused on an event model inferring only loss events.

More recently, event models that include lateral gene transfer (LGT) have been proposed. In the

past, transfer was ignored partly because incorporating transfers dramatically increases the computational complexity and partly because of the widespread belief that it does not occur in eukaryotes, despite considerable evidence to the contrary, at least in some lineages. In the simplest case, the event model includes only transfers [214, 229–233]. Tofigh et al. [234, 235] were the first group to consider an event model consisting of both duplication and transfer, together. This method did not minimize losses and only inferred counts, not the explicit event history. While Gorecki [236] has published a paper on inferring duplication, transfer, and loss, it has since been acknowledged that their approach includes an error and may not infer a most parsimonious reconciliation. Later in this thesis, I discuss my own work on event inference with a model including duplication, transfer, and loss that does infer an explicit event history.

Since the probability of incomplete lineage sorting decreases as time between speciation events increases [237–241], ignoring incomplete lineage sorting as a cause of discordance is justified if the branch lengths in the species tree are sufficiently long. In the above reduced event models, the assumption is made that the species tree is binary and the branches are significantly long enough that incomplete lineage sorting is improbable. When the species tree is non-binary, or branches are short, incomplete lineage sorting is a significant phenomenon that cannot be ignored [242]. Historically, a reduced event model with only incomplete lineage sorting has been well-studied [239, 243–249, for example]. Very recently, a number of papers have focused on a two event model, inferring incomplete lineage sorting and hybridization events [250, 251]. In prior work [252], we presented an algorithm for the reduced event model including gene duplication, loss, and incomplete lineage sorting.

When event models do not consider transfers, the event estimation problem has a unique solution and can be solved in polynomial time. In contrast, even the most reduced models containing transfer events are known to be NP-hard [231–234, 253, 254], and when including duplication and transfer events, the inference problem is NP-complete [234].

3.2.1. Notation and mathematical framework of binary reconciliation

The basic algorithmic structure and challenges of reconciliation are captured by reconciliation of gene and species trees under the duplication-loss parsimony model. To provide a foundation for the description of my results in subsequent chapters, I review that algorithm here. The trees shown in Fig. 3.4 will be used throughout to exemplify notation. In tree figures, the label g_s denotes a gene that is sampled from species s.

Comparison of Reconciliation Algorithms and Programs

Method/architect	Events			
	Duplication	Loss	LGT	ILS
[187]; [184, 198, 206]; [201]; [199]; [203]; [204]; [189]; [224]; and [225]	✓	✓		
[207, 208, 226]	✓			
[227, 228]	✓	✓		
[214]; [229]; [230, 231]; [232]; and [233, with genomic rearrangements]			✓	
[239, 243, 244]; [245]; [246, 247]; [248]; and [249]				✓
[234, 235]	✓		✓	
[236][1]	✓	✓	✓	
Stolzer, discussed here	✓	✓	✓	
[209][2]; [255]	✓	✓		✓
Vernot et al. [252]	✓	✓		✓
[250]; [251]			✓	✓
Lai and Stolzer, discussed here	✓	✓	✓	✓

Table 3.1: The various reconciliation approaches and packages defined by the event model employed.

Let $T_i = (V_i, E_i)$ be a rooted tree, where V_i is the set of nodes in T_i, and E_i is the set of edges. $L(T_i)$ is the leaf set of T_i and $L(v_i)$ refers to the leaf set of a subtree rooted at $v_i \in V_i$. The root node of T_i is denoted as $\rho_i = root(T_i)$. $C(v_i)$ and $p(v_i)$ refer to the children and parent of v_i, respectively. If v_i is binary, $r(v_i)$ and $l(v_i)$ denote the right and left children of v_i. For example, in Fig. 2.1b, $p(y) = x$, and $C(y) = \{g1\_A, g1\_B\}$, where $l(y) = g1\_A$ and $r(y) = g1\_B$. A non-binary node in a tree is referred to as a *polytomy*. A *monophyletic* group is a set of nodes consisting of a node and all of its descendants (i.e., a subtree of node v_i, denoted T_{v_i}; for example, in Fig. 2.1a, $\{\gamma, C, D\}$ forms a monophyletic group. The expression $u_i \geq_i v_i$ indicates that for $u_i \in V_i$, either u_i is v_i, or u_i lies on the path from v_i to $root(T_i)$. We say that u_i is the *ancestor* of v_i; analogously, v_i is the *descendant* of u_i. If $u_i \not\geq_i v_i$ and $u_i \not\leq_i v_i$, u_i and v_i are said to be *incomparable*. In Fig. 2.1b, $\rho_G = root(T_G) = x$ and $y \geq_G g1\_A$. I follow the computer science convention, in which the root is at the top of the tree, the leaves are at the bottom, and $p(v_i)$ is above v_i.

The basic tenet of reconciliation is that, if the gene tree co-diverges with the species tree, then each pair of parent and child nodes in the gene tree should map to a pair of parent and child nodes in the species tree. In the absence of duplication or loss, the immediate parent of a node, $p(v_G)$, will map to an immediate parent of $M(v_G)$ in the species host tree. Failure of this assumption is

evidence of incongruence, and indicates that events other than co-divergence have occurred. Reconciliation algorithms identifying nodes where this property is violated by comparing the parent and child nodes in the gene tree and the corresponding parent and child nodes in the species tree. If $M(p(v_G)) \neq p(M(v_G))$, the algorithm infers the minimum number of duplications and losses required to explain the discordance. The minimum cost set of events that explains these ancestral associations is also completely specified and easily calculated [207]. Thus, the reconciliation of the gene tree and species tree is completed with a two-step strategy: *(1)* infer the mapping $M(\cdot)$ between V_G an V_S and *(2)* use that mapping to test whether parent and child nodes disagree and to infer events if they do not. Formally, the event inference problem for duplication-loss parsimony is stated as follows:

Reconciliation with Duplication-Loss Parsimony

Input:

1. T_S: a binary, rooted species tree,
2. T_G: a binary, rooted gene tree, and
3. $\sigma = M : L(T_G) \rightarrow L(T_S)$: the mapping of contemporary genes, $L(T_G)$, to the species from which they are sampled, $L(T_S)$. $M(v_G) = v_S$ indicates that gene $v_G \in V_G$ was sampled from species $v_S \in V_S$.

Output:

1. $M : V_G \rightarrow V_S$: the mapping of ancestral genes, $V_G \setminus L(T_G)$ to ancestral species, V_S such that $M(v_G) = v_S$ indicates that ancestral gene v_G was present in ancestral species v_S, and
2. T_G augmented with duplication and speciation events assigned to V_G and loss events assigned to E_G.

LCA Mapping. Let T_G be a binary gene tree and T_S be a binary species tree such that the genes in $L(T_G)$ were sampled from the species in $L(T_S)$; we denote the species from which gene v_G was sampled, as σ. The mapping $M : V_G \rightarrow V_S$ is constructed from each node $v_G \in V_G$ to a target node $v_S \in V_S$. A mapping $M(v_G) = v_S$ implies that gene v_G was sampled from species v_S, and we say that v_S is the label of v_G. If v_G is a leaf node, $M(v_G) = \sigma(v_G)$ is the species from which sequence v_G was sampled. If v_G is an internal node, $M(v_G)$ is the least common ancestor (*LCA*) of the target nodes of its children:

$$M(v_G) = LCA(M(l(v_G)), M(r(v_G))). \tag{3.1}$$

For the three event model, $M(\cdot)$ is completely determined by the mapping between leaf taxa and is easily calculated with a greedy algorithm. In our example, $M(g1\_A) = A$, since it is a leaf; $M(x) = LCA(M(y), M(z)) = LCA(\alpha, \gamma) = \alpha$. From this mapping both gene duplications and gene losses can be inferred. We refer to this algorithm for calculating duplications and losses as *LCA reconciliation* in order to distinguish it from the new reconciliation algorithms proposed for multidomain evolution and for non-binary reference trees.

Gene Duplications. An inferred duplication at v_G implies that the duplication occurred between $p(M(v_G))$ and $M(v_G)$. The two resulting copies were present in species $M(v_G)$, and for at least one child c_G of $M(v_G)$ (if $M(v_G) \notin L(T_S)$), each copy persisted in at least one leaf (not necessarily the same leaf) of the subtree of T_S rooted at c_G. If $M(v_G) \in L(T_S)$, then both copies persisted in $M(v_G)$. Thus, a duplication is inferred at node v_G if and only if the children of v_G map to the same lineage in T_S; that is, there is some leaf $v_S \in L(T_S)$ such that both $l(M(v_G))$ and $r(M(v_G))$ are on the path from v_S to $root(T_S)$. This condition is true *iff*

$$M(v_G) = M(l(v_G)) \vee M(v_G) = M(r(v_G)). \tag{3.2}$$

By convention, duplications are assigned to nodes in V_G. Assigning a duplication to node $v_G \in V_G$ not only specifies its location in T_G, but also its location in T_S, via the mapping $M(\cdot)$. Every node in T_G that is not designated a duplication node is a speciation node. Fig. 3.4b shows a duplication at node $x \in T_G$, prior to the species divergence at α. A descendant of $l(x)$ persisted in species *B*, while a descendant of $r(x)$ persisted in species *C* and *D*; thus, both copies are represented in at least one leaf of the subtree rooted at β. The gene tree embedded in the species tree in Fig. 3.4b shows both copies of the gene on the edge (α, β). Although only one copy of the family survived in each species, discordance between the species tree in Fig. 2.1a and the gene tree in Fig. 2.1b provides sufficient evidence to infer a duplication at *x*. Because both *x* and one of its children (*y*) both map to α, Eq. 3.2 correctly identifies the duplication *x*.

Gene Losses. Losses can also be reconstructed from the mapping, $M(\cdot)$. In this case, the species, v_S, in which the loss occurred must be inferred explicitly. Assigning a loss in v_S to edge $(p(v_G), v_G) \in E_G$ indicates that v_G was present in both $M(p(v_G))$ and $M(v_G)$, and was lost on the path from v'_S to v_S, where v'_S is a species on the path from $M(v_G)$ to $M(p(v_G))$ (i.e., $M(v_G) <_S v'_S \leq_S M(p(v_G))$). For each $e = (p(v_G), v_G)$, the comparison of $M(p(v_G))$ and $M(v_G)$ determines

the losses assigned to e. If $p(v_G)$ is a speciation node, and no loss occurred, then $M(p(v_G))$ must be the parent of $M(v_G)$ in the species tree. Otherwise, we infer $depth(M(v_G)) - depth(M(p(v_G))) - 1$ losses on edge e. If $p(v_G)$ is a duplication node and no losses occurred, then $p(v_G)$ and v_G map to the same node in T_S. Otherwise, the number of inferred losses is $depth(M(v_G)) - depth(M(p(v_G)))$ — similar to speciation losses, but accounting for $p(v_G)$ and v_G mapping to the same node. The species associated with the losses inferred on $e = (p(v_G), v_G)$ are determined by walking up the species tree from $M(v_G)$ to $M(p(v_G))$. For each ancestral node $v_S \in V_S$ between $M(v_G)$ and $M(p(v_G))$, a loss is inferred in $l(v_S)$ or $r(v_S)$, whichever is not represented on the path from v_G to $p(v_G)$ in the gene tree. Both speciation and duplication loss situations arise in Fig. 3.4a. For example, consider the losses in Fig. 3.4a. Since x is a duplication node, $M(z) = \gamma \neq M(x) = \alpha$ indicates $depth(\gamma) - depth(\alpha) = 2$ losses occurred between x and z. These losses occurred in A and B. Also, since $p(g1\_B) = y$ is a speciation node, but $M(y) = \alpha$ is not the parent of $M(g1\_B) = B$ in T_S, losses are inferred in species C and D. Note that these two losses can be explained more parsimoniously by the loss of a single ancestral gene in the ancestral species, γ.

3.3. Multidomain reconciliation

I propose a methodology for inferring the evolutionary history of a multidomain family by integrating trees for each constituent domain, using a reconciliation framework analogous to the gene tree/species tree reconciliation, summarized above. This process exploits the fact that discordance between a domain tree and a reference tree is evidence that the domain diverged through processes other than co-divergence with the reference taxon. These processes include domain shuffling, gene duplication, gene loss, transfer, and incomplete lineage sorting.

Based on studies of molecular mechanisms (Sec. 2.4), I define a set of four *abstract domain shuffling events* that capture the various underlying mechanisms discussed in Sec. 2.4. These are represented in Fig. 1.2 *(1) Domain insertion* refers to any event that results in the acquisition of a new sequence fragment (the "mobile" domain) by *an existing gene*, whether that acquisition was mediated by NAHR, retrotransposition, duplicative transposition, or NHEJ. Note that in this model, a fusion between genes A and B is treated as an insertion of domains from B into gene A. *(2) Domain loss* can arise through unequal crossing over or disruption of splicing signals converting an exon into an intron. *(3) Internal domain duplication* often arise through unequal crossing over, but can also result from retro- or duplicative transposition. Evolution of the domain is also influenced by events on other levels of biological organization. *(4) Co-divergence* refers to events that are

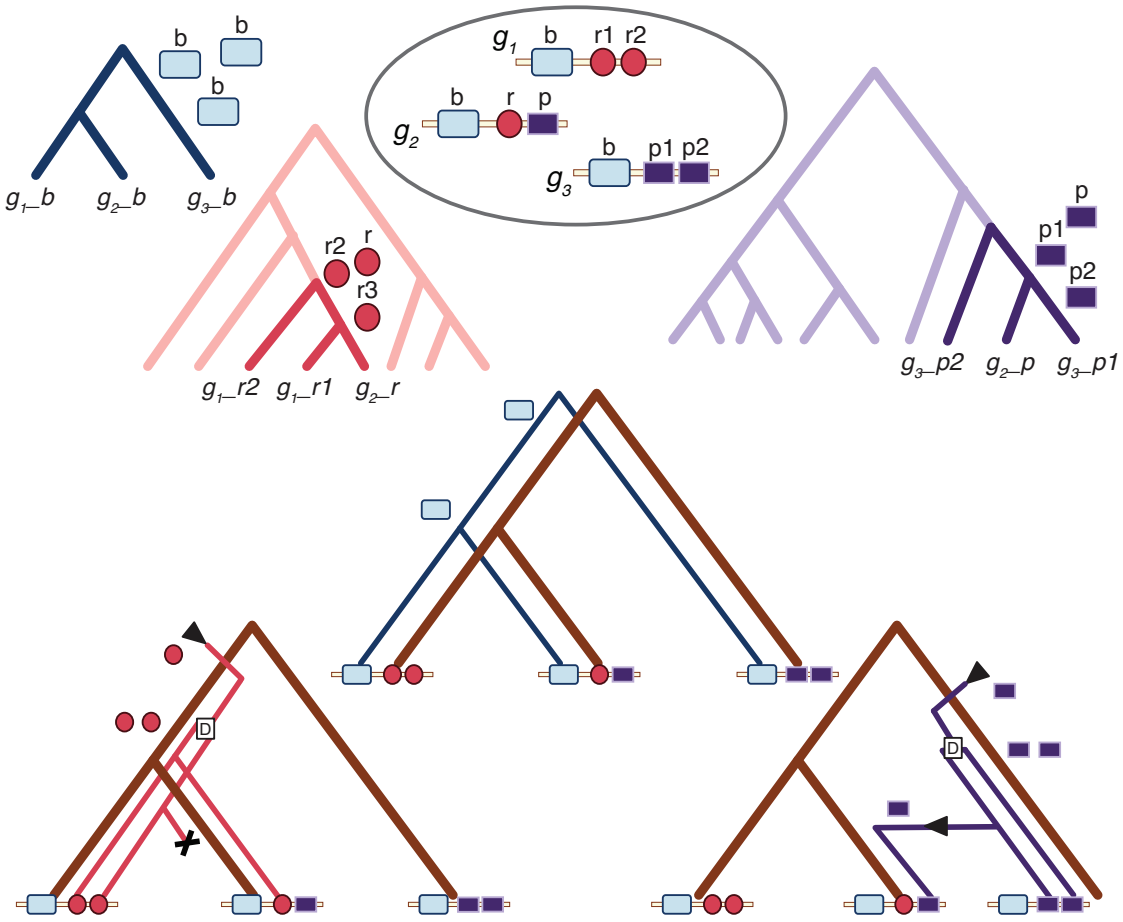


Figure 3.5: Domain trees for the multidomain family in Fig. 1.2 compared to the tree for the gene family locus to infer domain shuffling. Domain trees are colored according to the domain they represent. The red circle domain and purple rectangle domain trees are for all instances of the domain superfamily. Those domain instances present in the multidomain family are colored darker than the other instances. The tree for the gene family locus is brown. Contemporary domain architectures are shown in the bubble.

driven by the genome locus; i.e., a new instance of a gene that arose because of a speciation of gene duplication.

Any level of biological organization that co-evolves with the domain of interest can be used as the reference: another domain that co-occurs in the protein, species, or the multidomain gene family (as seen in Fig. 3.2b). The realization of my approach requires algorithms for domain tree reconciliation with event models appropriate for domain shuffling, algorithms to infer ancestral domain architectures, and the software implementation of these algorithms (see Fig. 3.7). Specifically:

– If the reference tree represents the history of a co-occurring domain or of the locus, then the events included in the model are: *(1)* co-divergence

due to speciation, gene duplication, transfers, or incomplete lineage sorting; *(2)* domain duplication; *(3)* domain insertion; and *(4)* domain loss events.

– If the reference tree is a species tree, the events included are: *(1)* co-divergence, which must be due to speciations; *(2)* duplication, which may be the result of gene or domain duplication or domain insertions *within* the species; *(3)* lateral events, which may be due to either transfers or cross-species domain insertions; *(4)* losses, which may be due to either gene or domain loss; and *(5)* incomplete lineage sorting.

Without considering all three levels of biological organization, information will be lost and, in some cases, incorrect histories could be inferred. When reconciling a domain tree with a species tree alone, it is not possible to distinguish between gene duplications, domain duplications and intraspecies transfers. Nor is it possible to distinguish between gene losses and domain deletions. Similarly, when the reference tree is a domain tree, it is not possible to determine whether a co-divergence was due to a speciation or gene duplication. While the reconciliation algorithm alone cannot distinguish between co-divergence events, if the reference tree represents the multidomain family, it can be reconciled with the corresponding species tree, in advance. This reconciled reference tree can then used to determine which type of event led to the co-divergence and also whether any gene losses have occurred. Solutions to these subproblems can still provide useful information.

In this framework, the event inference problem is as follows:

Multidomain Reconciliation

Input:

1. a binary, rooted reference tree T_R,
2. binary, rooted domain trees, T_{D_i} for every instance of domain superfamily D_i in the family, and
3. a mapping $M : L(T_{D_i}) \rightarrow L(T_R)\ \forall D_i$ of contemporary domains to contemporary reference taxa, where $M(v_D) = v_R$ if domain v_D is in reference taxon v_R.

Output:

1. a set of minimum-cost reconciled domain trees $\forall i$, where a reconciled tree is an augmented T_{D_i} in which:
    a. every ancestral domain node is labeled with the extended mapping $M : V_{D_i} \rightarrow V_R$, where $M(v_D) = v_R$ if the ancestral domain d was a constituent of reference taxon v_R;

   b. every node is determined to be a domain or gene duplication, lateral transfer event, domain insertion, or speciation; and
   c. gene losses and domain deletions are assigned to branches, and

2. a composite reference tree T_R, representing the history of the family and annotated with:
   a. inferred ancestral domain content on nodes;
   b. on branches, the set of (partially ordered) domain shuffling events that explains the incongruence between T_{D_i} and T_R, $\forall i$. Lateral events augment the tree by adding a directed edge between branches.

Domain trees are constructed from sequences of domain instances using standard molecular phylogenetics. An example of the output is shown in Fig. 5.9c. Ancestral domain architectures are not inferred by the reconciliation, but can be determined from the reconciled trees.

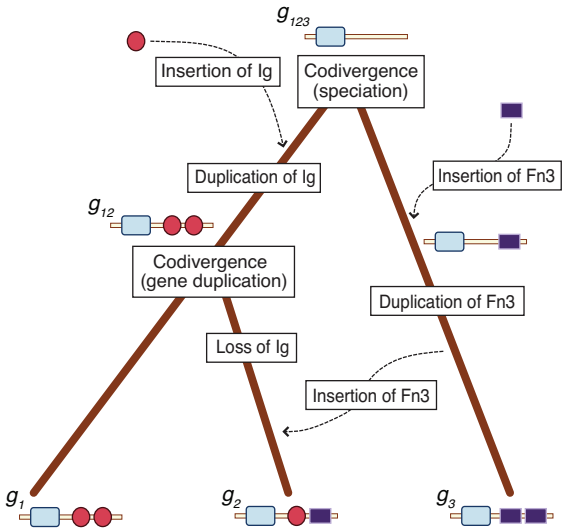


Figure 3.6: The history of the multidomain gene family in Fig. 1.2 annotated with the domain shuffling events that would be inferred using my multidomain reconciliation technique. Note that this hypothesis matches the true history of the family.

Depending on the reference tree employed, the application of this approach varies. When the species tree is the reference, we can establish the co-evolving associations between ancestral species and ancestral domain combinations. This provides information about domain content in ancestral species. Because we can directly reconcile a full domain superfamily tree representing *all*

instances of the domain in the species of interest (not just those instances in a given multidomain family) with a species tree, we can obtain information about the volatility of the domain family and how volatility changes across lineages.

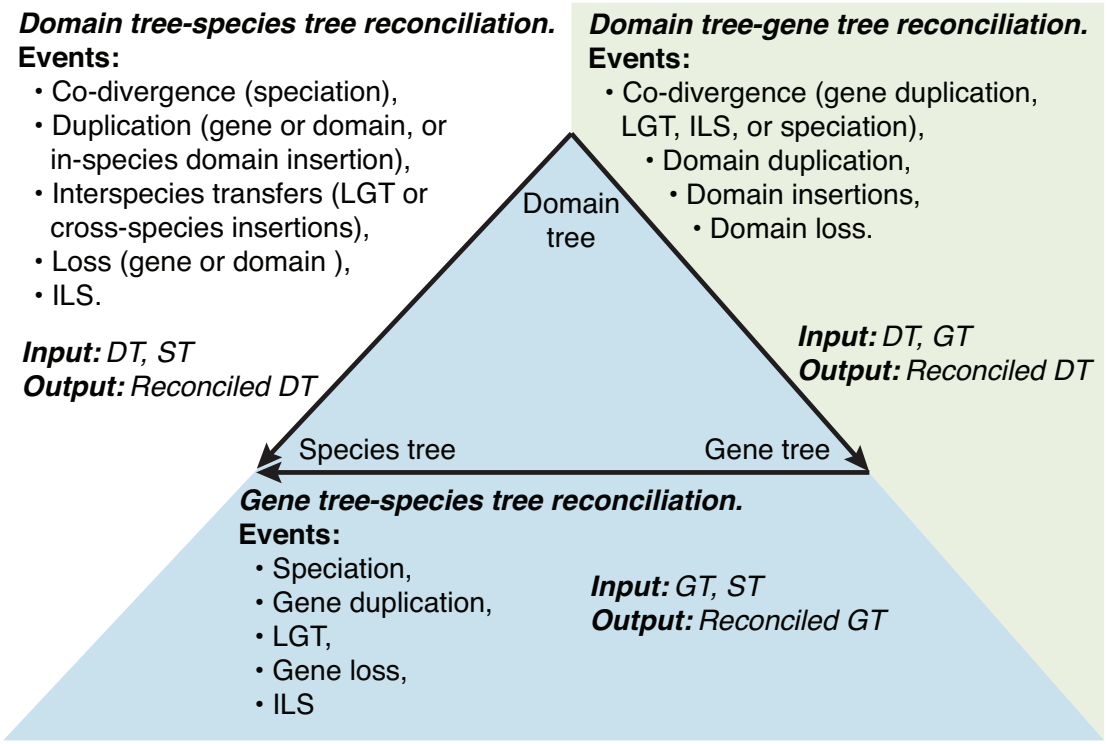


Figure 3.7: Diagram showing the different roles of the reference tree and the events that can be inferred with each instantiation. Top of the triangle: the domain tree; bottom right: the multidomain gene family tree; and bottom left: the species tree. Reconciling the domain tree with the gene tree is discussed in the green section. Reconciliation of a gene tree with a species tree is discussed in the blue section. Reconciliation of the domain tree with the species tree is discussed in the white section.

There are two applications of reconciliation using a domain tree as reference: reconciliation with any domain in the family and reconciliation with a primary domain. Using an arbitrary, co-occurring domain as the reference provides information about the promiscuity of one domain relative to another. This problem differs from the other problems considered in here in that we are considering the co-evolution of two entities at the same level of biological organization, in contrast to an embedded taxon evolving in a reference taxon. This situation is analogous to two parasites evolving within the same host, as opposed to a parasite co-evolving within a host. By comparing two entities co-evolving at the same level, we can infer the number of co-divergences, insertions, duplications, and deletions of domain domain D_1 *relative* to domain D_2. This type of

analysis allows analysis of all instances of two domains that co-occur, not just those that co-occur in the same multidomain family.

In some cases, we can take advantage of biological information to identify a *primary domain* of the family, and then use the tree from that domain as the reference. Many multidomain families arise from a progenitor gene [35,81,168], which undergoes gene duplication, followed by domain insertion, resulting in the progenitors of subfamilies with different domain architectures (e.g., Fig 1.2). These subfamilies then further expand through additional duplications. For families that follow this pattern, the evolutionary history of the *primary domain*[3] in the progenitor sequence is congruent with the history of the locus. For such families, the associated domain tree(s) can be treated as a proxy for the history of the locus. This is the most informative choice for the reference tree.

Anecdotal evidence suggests that the Kinase, Kinesin, Myosin, and ADAM families follow this pattern. For example, several lines of evidence suggest that the kinase domain is the "primary" domain for the protein tyrosine kinase family: All kinase domains share an origin [256,257]. Roughly 40% of kinases are single-domain proteins [258] and these occur primarily in more ancient kinase subfamilies. This is consistent with the hypothesis that single-domain kinases represent the ancestral state and that multidomain kinases arose through insertion of mobile domains into existing kinases. Moreover, kinase domains have characteristics that suggest they are not particularly mobile. Mobile domains tend to be small, and have 0-0 intron phase [32], yet neither is true of the kinase domain. I propose the following set of criteria, based on characteristics of known "primary" domains, for determining whether there exists in the multidomain family a domain that evolves only through vertical descent. These criteria include:

1. Evidence of vertical descent based on conserved synteny.
2. All instances of the domain are thought to share an origin.
3. Appearance of the domain only in one family (with some exceptions).
4. There is only one copy of the domain in each architecture of the family, which suggests that a single-domain protein was the ancestral state. I allow some exceptions here. For example, the kinase domain, a predicted primary domain, sometimes was locally duplicated. Often, one copy of this domain was later inactivated.
5. Conserved spatial features, such as a unique linker sequence, intron/exon structure, local architectures, etc.

[3]Note that we are not asserting that this domain is responsible for the primary function of the family, although this is often the case.

6. Appearance of the domain in single-domain families in more ancient lineages, suggesting that a single-domain protein was the ancestral state. In contrast, promiscuous domains, which are not "primary," rarely appear in single domain proteins.
7. Absence of traits characteristic of mobile domains, such as
   - short amino acid sequences or
   - 0-0 intron phase.

If there is more than one domain in the multidomain family that fits these criteria, an additional constraint, that the set of primary domains should have congruent trees, is imposed.

The prevalence of this pattern of evolution suggests that families with primary domains are not uncommon. However, there are cases where there is no "primary" domain. Either no single domain tracks the history of the locus or there may not be sufficient evidence to determine whether a domain evolved by vertical descent or not. In this case, the gene family tree can be estimated from the trees of all domains found in the family. Domain tree reconciliation can be treated as the estimation problem, in which the gene tree that minimizes the cost of domain shuffling events is sought. See Ma et al. [189, and works cited therein] for details on the tree estimation problem with reconciliation.

In this work, I focus only on multidomain families with a primary domain; gene family tree estimation from domain trees is an important problem for future work.

I have developed algorithms to satisfy the needs discussed here under the event-parsimony criterion, including: reconciliation with the events duplication, loss, incomplete lineage sorting; reconciliation for the three-event model with duplication, transfer, and loss; reconciliation with all four events (duplication, loss, transfer, and incomplete lineage sorting); and an algorithm to infer the composite history of the reference tree from all reconciled domain trees. In addition, these algorithms have been implemented in the reconciliation program, Notung [213, 252].

Chapter 4

Implementations of methods to infer domain shuffling events

In order to realize the co-evolutionary model of multidomain evolution that I proposed in Sec. 3.3, I have developed a number of different algorithms.

1. A reconciliation algorithm for duplication, heuristic loss, and incomplete lineage sorting, given a binary, *embedded* tree (the tree to be reconciled with the reference; i.e., the domain tree) and a non-binary reference tree [252]. The details of this algorithm have been published in the paper by Vernot, Stolzer, Goodman, and Durand [252]. This algorithm was implemented in NOTUNG by Ben Vernot.

2. A reconciliation algorithm for horizontal transfer, duplication, and loss, given binary embedded and reference trees. I have implemented this algorithm (for one optimal solution) into NOTUNG with a graphical user interface (GUI) update.

3. A reconciliation algorithm for duplication, heuristic loss, incomplete lineage sorting, and transfer given a binary embedded tree and a non-binary reference tree. This algorithm is based on algorithms (1) and (3) that I developed. It has been implemented in NOTUNG by Han Lai.

4. An algorithm to infer ancestral states of the reference tree and assign events to the reference tree given a set of reconciled embedded trees. This algorithm was implemented in NOTUNG by Ravi Chinoy, under my direction.

Notung. These algorithms have been implemented in NOTUNG. NOTUNG is a robust, general purpose software tool, developed in the Durand Lab, that provides a unified framework for incorporating information about duplication and loss into phylogenetic tasks [252, 259, 260]. NOTUNG provides a graphical interface for exploratory analysis and a command line interface for automated, high-throughput processing. It is widely used for a variety of analyses [261–273].

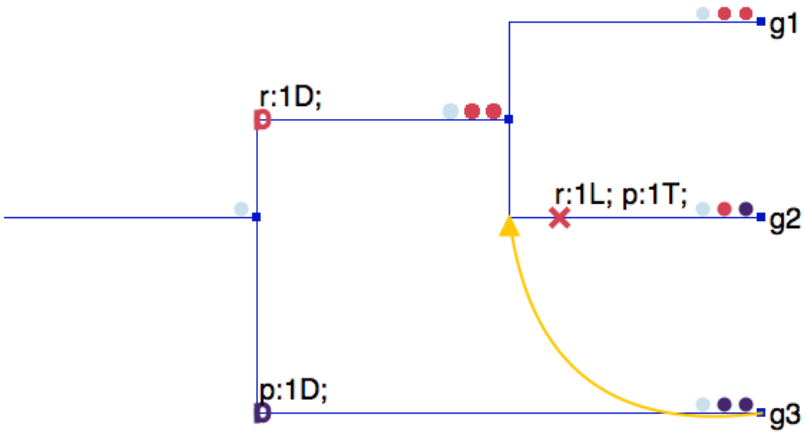
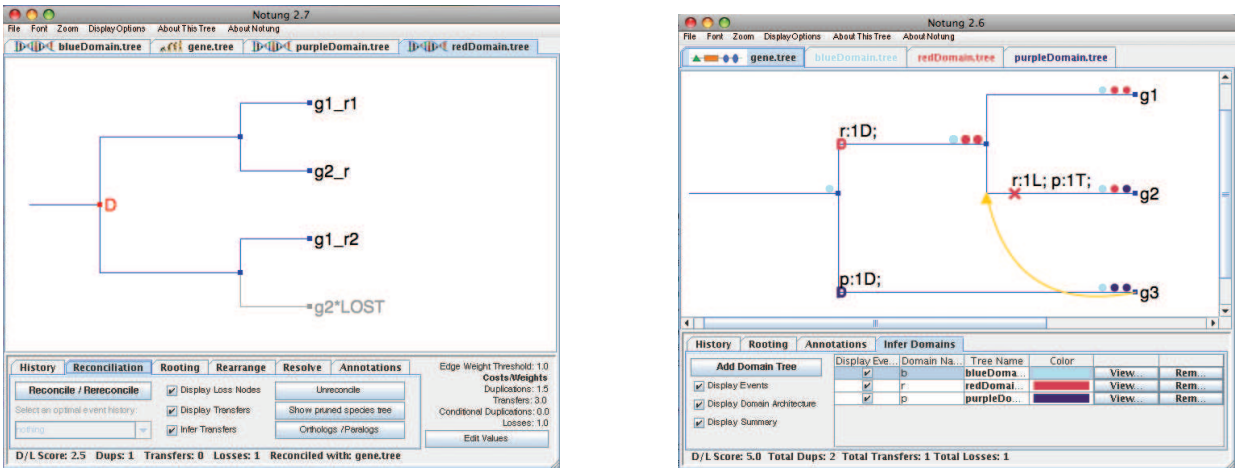


Figure 4.1: The history of the multidomain gene family in Fig. 1.2 inferred with multidomain reconciliation, as seen in Fig. 3.6, but inferred and drawn automatically in NOTUNG.

The implementation of these algorithms required major changes to data structures, the inference engine, and display. In addition, substantial changes were needed to accommodate reference trees representing composite multidomain histories. In NOTUNG, which performs gene tree-species tree reconciliation, the reference (species) tree is a passive entity which is not modified in the course of the analysis. Inferring a composite history for the reference family requires data structures that allow communication between reference and embedded trees, as well as bookkeeping for real-time updates between the individual domain trees and the reference tree. This also required an interaction panel with the reference tree view that allows the user to perform actions that invoke inference calculations and/or modify its appearance.

The GUI (shown in Fig. 4.2) has been modified to display horizontal events and domain content. It presents the composite tree with panels that allow the user to interact with the reference tree and to root both reference and domain trees by event parsimony; presents the reconciled domain trees with duplications, transfers and losses; presents composite reference trees with ancestral domain content and a summary of the events associated with each domain (e.g. r: 1L; p: 1T). The algorithm infers a partial ordering on the inferred domain shuffling events. The partial order of these events

is not currently presented visually, but is stored in the internal data structures and included in the output of the command line interface.



(a) (b)

Figure 4.2: Screen captures of NOTUNG showing my implementations of multidomain reconciliation algorithms. **(a)** The tree for the red domain in Fig. 1.2 reconciled with the tree for the gene family locus. **(b)** The composite history of the gene family showing domain shuffling events by reconciling the blue, red, and purple domains in Fig. 1.2 with the gene family tree.

With my developed algorithms, NOTUNG can now root a tree based on duplication, loss, and insertion parsimony. Under the assumption that gene duplication, loss, and insertion are rare events, the rooted, binary tree that requires the fewest events to explain the data is the best resolution of uncertainty. NOTUNG uses this parsimony principle to root an unrooted tree by minimizing the event cost. In order for this implementation to run efficiently, memoization of the reconciliation algorithm for rooting has been implemented. Incomplete lineage sorting (ILS) can be included in the cost, but this event type is often assigned zero cost.

4.1. Duplication-loss-incomplete lineage sorting algorithm

In order to reconcile the domain and/or the multidomain family tree with the species tree, we wish to be able to identify incongruence due to incomplete lineage sorting in addition to duplication and loss. The probability of incomplete lineage sorting increases as time between speciation events decreases [237–241]. Thus, when the species tree is non-binary or contains short branches, incomplete lineage sorting is a significant phenomenon that cannot be ignored [242]. I have developed

an algorithm, Alg. 4.1 that reconciles a rooted, binary embedded tree T_E with a rooted non-binary species tree T_S to infer the duplication, loss, and incomplete lineage sorting events in the history of the embedded family [252]. This is accomplished by mapping nodes in the embedded tree to *sets* of nodes in the reference tree. My novel set mapping approach allows us to test efficiently whether a discordance at a given node is a duplication or incomplete lineage sorting event. The maximum size of the set mapping to any node in T_E is $O(k_S)$, where k_S is the maximum outdegree in T_S (i.e., the size of the largest non-binary node). Using this mapping, incomplete lineage sorting, duplication, and heuristic loss events are inferred in $O(|V_E| \cdot (k_S + h_S))$, where h_S is the height of T_S.

All binary trees with k leaves are equally compatible with a non-binary node (polytomy) in the species tree with k children [237] (see Fig. 4.3). Therefore, we can treat a polytomy s as a set of hypotheses, or binary resolutions. For each polytomy $v_i \in V_S$, let $H(v_i)$ be the set of all possible binary trees, rooted at v_i, whose leaves are the children of v_i. Formally, given the k-tomy $v_i \in V_S$, let $H(v_i) = \{T_{ij} | L(T_{ij}) = C(v_i)\}$, where T_{ij} is a binary tree such that the leaves of T_{ij} are the children of v_i. In addition, let $H^*(T_S)$ be the set of all possible binary trees obtained by replacing each polytomy $v_i \in V_S$ with each tree $T_{ij} \in H(v_i)$. In other words, $H^*(T_S)$ is the set of all possible binary resolutions of T_S (see Fig. 4.3, for example). If T_S is binary, then $H^*(T_S) = \{T_S\}$. The number of trees in $H^*(T_S)$ is $\prod_{\forall v_i \in V_S} |H(v_i)|$, where $|H(v_i)| = \frac{(2k_i-3)!}{2^{k_i-2}(k_i-2)!}$ and $k_i = |C(v_i)|$, the polytomy size or number of children of v_i [274]. For example, if node v_i is the trichotomy α in Fig. 4.4a, then $H(\alpha) = \{(A,(B,\beta)),(B,(A,\beta)),(\beta,(A,B))\}$, and $H^*(T_S) = \{(A,(B,(C,D))),(B,(A,(C,D))),((C,D),(A,B))\}$. When reconciling the embedded tree T_E with every $T \in H^*(T_S)$, if $v_E \in V_E$ is a duplication in every reconciliation, then a duplication *must* have occurred at v_E. If at least one, but not all reconciliations indicate a duplication at v, then an incomplete lineage sorting event *may* have occurred. Under the parsimony principle, we will infer such a node to be an incomplete lineage sorting event. Notice that for the trees in Fig. 4.4, every $T' \in H^*$ would infer a duplication node at node y; however, this is not the case for node x. Therefore, node y is a duplication. On the other hand, node x is an incomplete lineage sorting event since there is a binary resolution in $H^*(T_S)$, namely $(A,(B,(C,D)))$, that does not infer a duplication. However, $H^*(T_S)$ grows superexponentially with the size of polytomies in T_S; therefore a more efficient method of identifying duplication and incomplete lineage sorting events is needed. The LCA mapping used in standard reconciliation (Eq. 3.1) is not sufficient because it cannot distinguish between incomplete lineage sorting and duplication (see for example, Fig. 4.4d, where LCA would incorrectly infer a duplication at x since $M(x) = M(r(x))$).

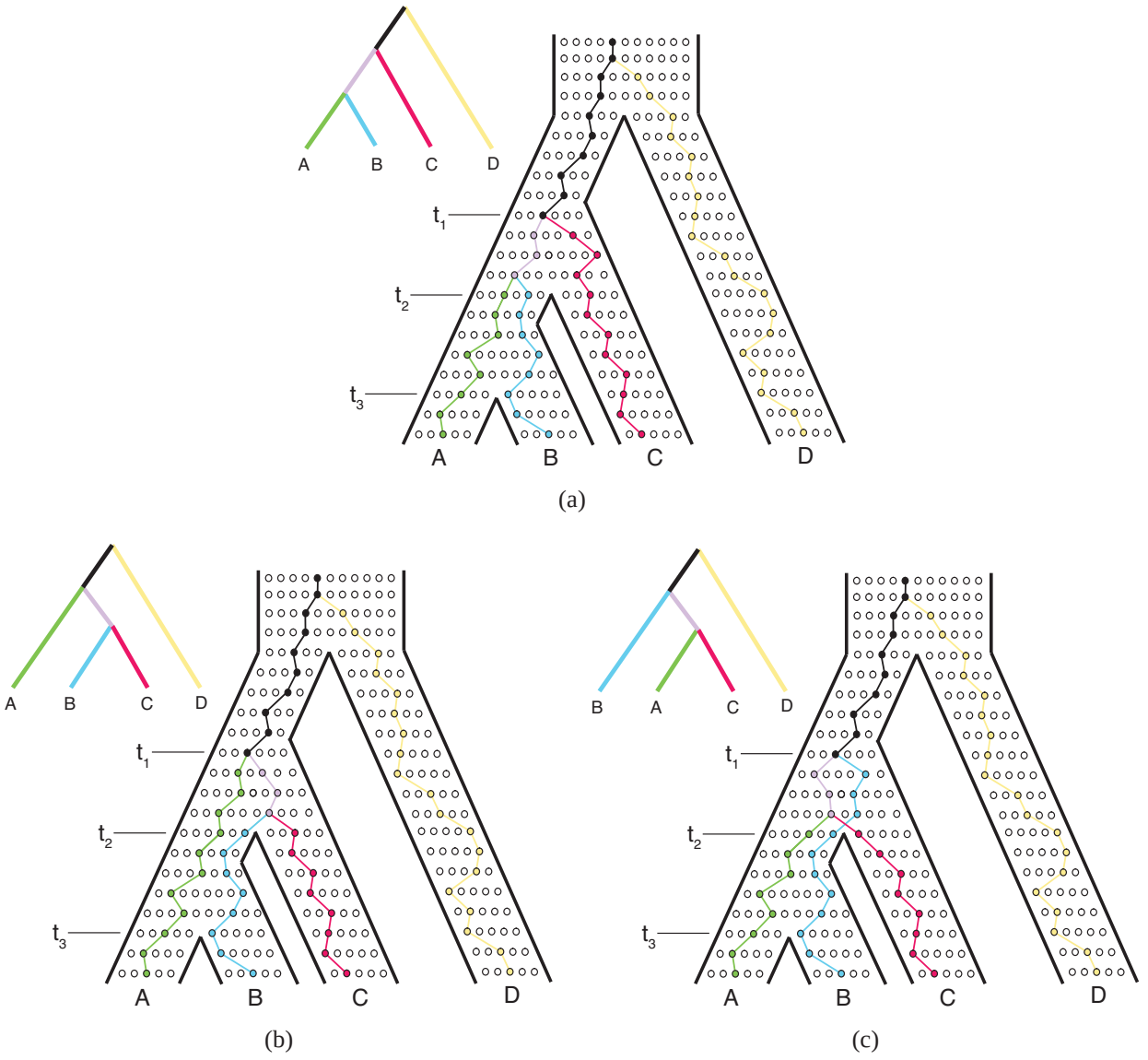


Figure 4.3: If branches in the species tree are sufficiently short (i.e., the branch between the ancestor of *A* and *B* and the ancestor of *A*, *B*, and *C*), incomplete lineage sorting is possible. These trees show the gene family evolving within the context of the species and populations. Gene trees are also shown as stick trees outside the species tree. Because the population has three different alleles (colored dots) present at time t_2, the alleles could sort randomly across *A*, *B*, and *C*, resulting in incomplete lineage sorting where the gene and species tree disagree, as in (b) and (c).

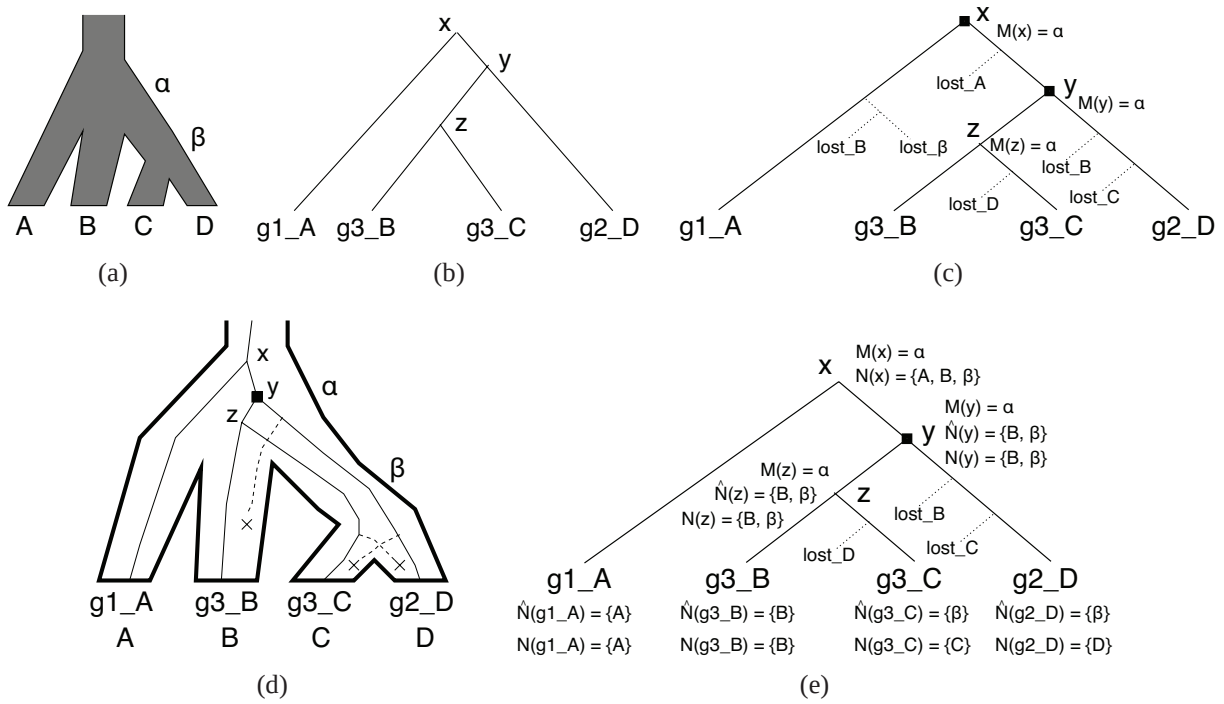


Figure 4.4: **(a)** A species tree with a polytomy at α. **(b)** A hypothetical gene tree sampled from the species in (a). **(c)** The gene tree from (b), which has been reconciled using the LCA algorithm. **(d)** The hypothetical gene tree embedded in species tree. **(e)** The gene tree labeled with the $\widehat{N}(\cdot)$ mapping and showing duplications and losses; node *x* is inferred to be ILS. Black squares indicate duplications. Losses are represented by dotted lines.

I have developed a novel mapping, $\widehat{N}$ (Eq. 4.1), that can be calculated efficiently and can correctly distinguish between duplication and incomplete lineage sorting events. A straightforward approach would be to map each node, v_E, in the embedded tree with *all* nodes (both leaves and internal nodes) in the species tree in which the embedded node was present. Using this mapping, a required duplication is inferred at v_E if the intersection of the sets of its children is non-empty. The size of the sets mapping the nodes in the embedded tree grows with the height of the tree and can contain as many as $O(|V_S|)$ elements. However, $\widehat{N}$ takes advantage of the observation that it is sufficient to store only those children of $M(p(v_E))$ in which descendants of v_E must have been present. In this case, $\widehat{N} : V_E \setminus root(T_E) \rightarrow V_S^+$ is defined to be

$$\widehat{N}(v_E) = \begin{cases} \{M(v_E)\}, & \text{if } M(\ p(v_E)\ ) \in L(\ T_S\ ), \\ \{h | h \in C(M(p(v_E))) \wedge \exists\, u_E \in L(v_E) \ni h \geq_S M(u_E), & \text{otherwise} \end{cases} \tag{4.1}$$

where V_S^+ is the powerset of V_S, excluding the empty set. With this mapping, node $v_E \in V_E$ is a

duplication *iff*

$$\widehat{N}(r(v_E)) \cap \widehat{N}(l(v_E)) \neq \emptyset. \tag{4.2}$$

The size of $\widehat{N}$ at any given node is bounded by the size of the largest polytomy in T_S, yet is sufficiently informative to identify and distinguish between incomplete lineage sorting and duplication[1]. Fig. 4.4e shows the mapping $\widehat{N}(\cdot)$. It correctly infers a duplication at y since $\widehat{N}(l(y)) \cap \widehat{N}(r(y)) = \{B, \beta\} \cap \{\beta\} = \{\beta\} \neq \emptyset$. It also correctly identifies incomplete lineage sorting at node x, since $\widehat{N}(l(x)) \cap \widehat{N}(r(x)) = \{A\} \cap \{B, \beta\} = \emptyset$.

For a given binary, embedded tree and binary species tree, there is exactly one most parsimonious loss history, with each loss unambiguously assigned to one edge in T_E, and associated with one node in T_S. In this case, it is possible to determine the set of losses assigned to an edge $e = (v_E, p(v_E))$ by comparing $M(v_E)$ and $M(p(v_E))$, without considering losses on any other edge of the gene tree. The total number of losses in the most parsimonious history can be determined by inferring losses on each edge independently and summing over all edges. In contrast, when T_S is non-binary, a reconciliation may have more than one equally parsimonious loss history because losses within a species polytomy are *ambiguous*. These losses may be assigned to one of several edges in the embedded tree. In addition, in embedded families in which two or more losses occurred, interactions between losses that can be assigned to the same edge of the embedded tree must be considered. Two factors contribute to the interactions between losses. First, losses that occurred in sibling species and that are assigned to the same edge in T_E may be replaced by a single loss in a common ancestor, decreasing the total loss count. Second, interaction of ambiguous losses with duplications in the embedded tree affects the total number of losses inferred. Pushing a loss below a duplication generally increases the number of losses; however, if these duplicated losses can be combined with other losses below the duplication, it may, in fact, decrease the total number of losses.

While reconciliation does not provide enough information to fully resolve the temporal order of these losses relative to other events, it is able to identify the set of *permissible edges* for a given loss. The particular edge within the permissible set to which a loss is assigned determines whether or not it can be combined with other losses and, hence, the total number of losses inferred. There are two criteria that dictate where losses can be combined. The first criterion follows from the standard binary case: losses forming a monophyletic clade (i.e., are a subtree T_{v_S} rooted at $v_S \in V_S$) are inferred as a loss in the ancestral root of the clade (i.e., in v_S). The second criterion focuses on polytomy "siblings": losses that correspond to leaves of subtrees whose roots are the children

[1] see Vernot et al. [252] for proofs

of a polytomy (i.e., losses are $\{L(T_{v_{S_i}})\}$ for $\{v_{S_i}\} \subset C(v_S)$, where $v_S \in V_S$ is a polytomy) can be combined into a single loss mapped to the set of children that were lost (i.e., $\{v_{S_i}\}$). This is possible because of the observation that there is some subtree T'_{v_S} in $T' \in H^*(T_S)$ such that the losses form a monophyletic clade, allowing us to infer a single loss in $root(T'_{v_S})$.

Rather than consider all possible loss placements and loss combinations, I have developed a heuristic to efficiently identify the fewest losses. An exact solution with complexity $O(|V_E|k_S 2^{2k_S})$ has been proposed by the Durand Lab and is discussed in Vernot et al. [252]. My heuristic has complexity $O(|V_E| \cdot (k_S + h_S))$, and although not guaranteed to return an optimal history, does very well in practice. In a dataset containing 1174 trees, the heuristic found the optimal solution in more than 99% of the cases studied [252]. The heuristic uses a greedy strategy that makes loss assignment decisions at each edge, without considering interactions with losses inferred on other edges. The strategy is to minimize duplicated losses by assigning each ambiguous loss to the permissible edge closest to the root. This guarantees that the loss will not be unnecessarily assigned below a duplication node, leading to the inference of two losses, instead of one. Losses assigned to an edge $e = (v_E, p(v_E))$ can occur in any of three sets of species:

- $C(M(v_E))$: The set of all children of $M(v_E)$.
- $\widehat{N}(v_E)$: The set of children of $M(p(v_E))$ that contain a descendant of v_E.
- $N(v_E)$: The set of children of $M(v_E)$ that contain a descendant of v_E.

$N(v_E)$ is a set variable I have developed to infer losses efficiently. $N : V_E \rightarrow V_S^+$ is defined to be

$$N(v_E) = \begin{cases} \{M(v_E)\}, & \text{if } M(\ v_E\ ) \in L(\ T_S\ ), \\ \{h | h \in C(M(v_E)) \wedge \exists\ u_E \in L(v_E) \ni h \geq_S M(u_E), & \text{otherwise.} \end{cases} \tag{4.3}$$

I define three tests, corresponding to the above sets, to determine losses along edge $e = (v_E, p(v_E))$, in E_E.

Test 1 – Skipped Species: If $M(v_E) \neq M(p(v_E))$ and $p(M(v_E)) \neq M(p(v_E))$, traverse the path from $M(v_E)$ to $M(p(v_E))$ in T_S, inferring a loss diverging from each intermediate species along this path (lines 29-31 in Alg. 4.1).

The procedure to infer these skipped losses is carried out in the *climb* procedure and is analogous to that used in LCA reconciliation (described on pg. 49). This test is applied to all edges in T_E, whether associated with a binary node or polytomy in T_S. If a node and its parent map to different nodes in T_S, we expect those nodes to correspond to child and parent

nodes in the species tree. Otherwise, genes in the intervening species must have been lost.

Test 2 – Duplication: If $p(v_E)$ is a required duplication, then losses are inferred in the species in $N(p(v_E)) \setminus \widehat{N}(v_E)$ at e (lines 18-21 in Alg. 4.1).

This test is applied to all edges where $p(v_E)$ is a duplication, whether associated with a binary node or polytomy in T_S. Note that if $M(p(v_E))$ is binary and $M(p(v_E)) = M(v_E)$, then $N(p(v_E)) = \widehat{N}(v_E)$ and no losses are inferred. Thus, this test reduces to that used in LCA reconciliation for binary nodes in T_S. When $M(p(v_E))$ is polytomy, losses may occur even when $M(p(v_E)) = M(v_E)$, in contrast to binary reconciliation.

Test 3 – Polytomy: If $M(v_E)$ is a polytomy and $M(p(v_E)) \neq M(v_E)$, then losses are inferred in the species in $C(M(v_E)) \setminus N(v_E)$ at e (lines 22-24 in Alg. 4.1).

This test is only applied when $p(v_E)$ is a speciation node and $M(v_E)$ is a polytomy. It verifies that each child of $M(v_E)$ contains a descendant of v_E. If not, one or more losses must be inferred.

After all losses are assigned, those losses that satisfy the appropriate criteria (described above) are combined.

The algorithm to construct the mapping and infer duplication, loss, and incomplete lineage sorting is shown in Alg 4.1. $\widehat{N}(\cdot)$, $N(\cdot)$, and $M(\cdot)$ are calculated by a postorder traversal of T_E. During this traversal, the algorithm performs each of the three loss tests described above to identify loss nodes. Because the tree is traversed in postorder, the permissible edge closest to the root is reached first, allowing the heuristic to assign the loss to the desired edge without explicitly determining the set of permissible edges for each loss. To ensure that $\widehat{N}(\cdot)$ is composed only of children of $M(p(v_E))$, a climbing step is executed to replace the set of nodes in $\widehat{N}(v_E)$ with the child of $M(p(v_E))$ which is ancestral to them. The `climb` procedure prevents $|\widehat{N}|$ from growing larger than k_S and assigns Polytomy and Skipped species losses. For any given path from $u_E \in L(T_E)$ to $\rho_E = root(T_E)$, we will climb in total from $M(u_E)$ to $M(\rho_E)$. Thus the *total* cost of calls to *climb* is $O(|V_E| \cdot h_S)$. For internal nodes, $N(v_E)$ is initialized to $\widehat{N}(l(v_E)) \cup \widehat{N}(r(v_E))$, each bounded in size by k_S. Using a suitable data structure, this step can be achieved in $O(\log(k_S))$ time per node. Using fast LCA queries, $M(\cdot)$ can be calculated in $O(|V_E|)$ time for the entire tree [275]. If the intersection of the sets $\widehat{N}(l(v_E))$ and $\widehat{N}(r(v_E))$ are non-empty, a duplication is inferred; incomplete lineage sorting is inferred if the node is not a duplication, but is a duplication under standard LCA reconciliation (Eq. 3.2). Testing for a duplication and assigning duplication losses takes $O(k_S)$ per node, while the test for incomplete lineage sorting is constant time per node. Thus, the total complexity for this

algorithm is $O(|V_E| \cdot (k_S + h_S))$.

Algorithm 4.1

```
Input:  T_E = (V_E, E_E); T_S = (V_S, E_S); σ, the M(·) of L(T_D) → L(T_R)

reconcileDLI( T_E, T_S, σ )
 1 for each v_E ∈ T_E in postorder do
 2   if ( v_E ∈ L(T_E) ) then
 3      // Leaf node case
 4      M( v_E ) ← σ( v_E )
 5      N( v_E ) ← { M(v_E) }
 6      N̂( v_E ) ← { M(v_E) }
 7   else do // Internal node case
 8   M( v_E ) ← LCA( M(l(v_E)), M( r(v_E)) )
 9   calculateDuplication( v_E )
 10  if ( v_E ≠ DUP ) then
 11     if ( M(v_E) == M(l(v_E)) || M(v_E) == M(r(v_E)) ) then
 12        event( v_E ) ← ILS
 13 return

calculateDuplication( v_E )
 14  // Update N̂( · ) for children by climbing
 15  N̂( l(v_E) ) ← climb( l(v_E), v_E )
 16  N̂( r(v_E) ) ← climb( r(v_E), v_E )
 17  N( v_E ) ← N̂( l(v_E) ) ∪ N̂( r(v_E) )
 18  if ( N̂(l(v_E)) ∩ N̂(r(v_E)) ≠ ∅ ) then
 19     event( v_E ) ← DUP
 20     // duplication losses for left child
 21     add N( v_E ) \ N̂( l(v_E) ) to losses( l(v_E) )
 22     // duplication losses for right child
 23     add N( v_E ) \ N̂( r(v_E) ) to losses( r(v_E) )

climb( u_E, v_E )
 24  // general polytomy losses
 25  if ( M(u_E) ∉ L(T_S) && M(u_E) ≠ M(v_E) ) then
 26     add C( M(u_E) ) \ N( u_E ) to losses( u_E )
 27  select x from N̂( u_E ) at random
 28  if ( x == M(v_E) || p(x) == M(v_E) ) then
 29     return N̂( u_E )
 30  while ( p(x) ≠ M(v_E) ) do
```

```
31     // skipped losses
32     if ( p(x) ≠ M(u_E) ) then
33        add siblings( x ) to losses( u_E )
34     // climb
35     x = p( x )
36  return { x }
```

4.1.1. Empirical testing

I tested this algorithm on several data sets, using the version implemented in NOTUNG 2.5.

First, I confirmed that this algorithm performs identically to LCA reconciliation when applied to a benchmark of 15 well-studied, binary trees [276–279], with no incomplete lineage sorting and verified that the results were the same as those generated by the binary version of NOTUNG, as well as those of the original authors.

Second, I compared the results of my reconciliation algorithms to those from LCA reconciliation for gene families in three species groups with known polytomies: *Anolis* [280], *Neoaves* [281] and Auklets [282]. Species trees were transcribed directly from the source articles. I constructed gene trees for two gene families in *Neoaves* (cytochrome-b and globin), three families in Auklets (cytochrome-b, cytochrome oxidase 1 and NADH-6), and one family in *Anolis* (NADH-2). Sequences were downloaded from NCBI [283], and multiple sequence alignments were constructed with T-Coffee [284]. Phylogeny reconstruction was performed using the PHYLIP package from Felsenstein (v. 3.6.1) and bootstrapped using the included SEQBOOT program. Branches with weak bootstrap support ($< 60\%$) in the globin tree from *Neoaves* were rearranged using models presented in [260].

Table 4.1 shows the number of leaves (l) in each tree, the size of the maximum polytomy (k_S) in each species tree, the number of duplications obtained by LCA reconciliation (B), the number of required duplications predicted by our algorithm (R), and the optimal number of losses. For each of these gene families, the exact and heuristic loss algorithms reported the same number of losses. Only one gene family, the globins, had more than one optimal loss assignment. As predicted, LCA reconciliation substantially overestimates required duplications.

In Vernot et al. [252], we also present an analysis on comparing heuristic and exact loss inference for all full trees in TreeFam 3.0 [285]. This analysis found that *(1)* the heuristic loss inference

Empirical results for testing the duplication, loss, incomplete lineage sorting algorithm

Gene family	Tree		Dupl.s		Losses
	l	k_S	B	R	
Neoaves [282]	12	10	-	-	-
cytochrome B	9	-	4	0	0
globin	17	-	7	4	7
Auklets [281]	5	4	-	-	-
cox1	5	-	1	0	0
cytochrome B	5	-	2	0	0
NADH-6	5	-	2	0	0
Anolis [280]	50	6	-	-	-
NADH-2	50	-	13	7	17

Table 4.1: Comparison of duplications inferred by NOTUNG 2.5 and LCA reconciliation.

inferred the minimal number of duplications ˜99% of the time; and *(2)* when the heuristic was not minimal, it was not very bad: the worst loss overcounting was of 4 losses in a tree with 249 losses. In addition, running time was measured for this dataset, showing that large-scale analyses are possible with this algorithm and software. Reconciling all 1174 trees in TreeFam took only 48 minutes (on a 3.2ghz OptiPlex GX620 computer) when using the heuristic losses.

4.2. Duplication-transfer-loss algorithm

In order to capture domain insertions, an algorithm that considers horizontal events, as well vertical events is required. Reconciliation with transfer (i.e., insertion or horizontal events), which I employ for multidomain reconstruction, is more difficult than other event models. First, for any model involving insertion, the event inference problem is NP-hard because insertions must obey temporal constraints [232, 233, 235]. Second, even when temporal constraints are not violated, in a model with horizontal events, taxa in the embedded tree can jump to distant locations in the reference tree. As a result, to determine the optimal mapping $M(v_D)$ for each ancestral node in $v_D \in V_D$, all possible reference node mappings must be considered. In addition, all possible child mappings must be examined. Third, each node branching in T_D could be the result of co-divergence with the reference taxa, or duplication or insertion of the domain. Thus, each of these three events must also be considered at each node. These properties invoke an additional level of complexity, as there may be multiple optimal mappings between V_D and V_R, multiple optimal sets of events, *and* multiple optimal child-pair mappings for each mapping. Any scenario that can be explained

by a transfer can also be explained by a combination of duplications and losses. At each node in V_D, the algorithm must iterate over all possible species nodes in V_R, all possible events, and all possible species nodes for each child to determine the value of $M(v_D)$ and the event history that minimizes the total event cost. None of the simpler models explore both the space of mappings and the space of events. Although heuristics for inference of transfer events alone have been proposed, few algorithms consider the complete duplication-loss-transfer event model.

My reconciliation algorithm, Alg. 4.2 explores the space of mappings and events to infer the most parsimonious history of domain duplication, loss, and insertion events. This algorithm:

- simultaneously calculates the minimal total cost of duplications, losses, and horizontal events required to explain the discordance between trees.

- infers the history, not just the number of events, indicating where in the tree each event occurred. (i.e., associating duplications, insertions, and losses with specific nodes and edges in the domain and reference trees).

This is achieved in three passes.

In the first pass, a dynamic program traverses the tree depth-first, visiting each node in the domain tree in post order. At each node, entries in minimal-cost event and traceback tables are calculated (`calculateTable`) for all possible mappings to nodes in the reference tree, all possible combinations of children node mappings, and for each node event-type: co-divergence, duplication, and insertion. Losses are calculated with each of these three events. The cost table, c, and the traceback table are two dimensional tables of size $n \cdot m$ ($n = |V_D|$ and $m = |V_R|$, with rows representing the nodes in T_D and columns representing nodes in T_R. Each entry, $c(v_D, v_R)$, stores the minimum cost of reconciling subtree T_{v_D} with T_R and mapping v_D to $v_R \in V_R$. Each entry in the traceback table is a triple $\{\texttt{EVENT},\ M(l(v_D)),\ M(r(v_D))\}$ representing the event and associated mappings of the left and right children of v_D, $l(v_D)$ and $r(v_D)$ respectively, that resulted in this optimal reconciliation.

Calculations for the cost table are based on a pair of recursion equations. If node v_D is a leaf node, $v_D \in L(T_D)$, then the only possible mapping is the reference taxon from which it was sampled, $\sigma(v_D)$, which is of no cost. Otherwise, the entry $c(v_D, v_R)$ is calculated as the minimum cost of the mapping of v_D to v_R and assigning a duplication, insertion, or co-divergence event to v_D. The recursion equations for the dynamic program are as follows [235]:

If $v_D \in L(T_D)$ and $v_R \in L(T_R)$, then

$$c(v_D, v_R) = \begin{cases} 0 & \text{if } v_R = \sigma(v_D), \\ \infty & \text{otherwise.} \end{cases} \tag{4.4}$$

Else, $v_D \in V_D \setminus L(T_D)$ and

$$c(v_D, v_R) = \min\{c_{\texttt{CODIV}}(v_D, v_R),\ c_{\texttt{DUP}}(v_D, v_R),\ c_{\texttt{INS}}(v_D, v_R)\}, \tag{4.5}$$

where `CODIV` is a co-divergence, `DUP` is a duplication, and `INS` is an insertion.

For each value of v_D and v_R, the cost of a duplication, an insertion, or a co-divergence at v_D is determined, and the minimum of these is assigned to $c(v_D, v_R)$. If two or more of these events incur a minimum cost, one is selected at random. In order to calculate $c_{\texttt{EVENT}}(v_D, v_R)$, we must consider all the possible mappings of the children of v_D that are consistent with the event at node v_D. To be consistent with a co-divergence event, the children of v_D must map to a node in $T_{l(v_R)}$, the subtree rooted at $l(v_R)$ and a node in $T_{r(v_R)}$, the subtree rooted at $r(v_R)$, respectively. If the children of v_D are mapped to the roots of these subtrees (i.e., to $l(v_R)$ and $r(v_R)$), then no losses are incurred; otherwise, losses must be inferred. In contrast, the mapping of the children of v_D is consistent with a duplication event if both children are mapped to the same node of one child mapping is comparable to the other. If both children map to v_R, then there is no evidence of loss. Losses are inferred if one or both children are mapped to nodes in $T_{r(v_R)}$ or in $T_{l(v_R)}$. The number of losses inferred follows from the equations for loss in standard reconciliation (on pg. 49) A mapping is consistent with an insertion event when one child of v_D is mapped to a node in T_{v_R} (possibly v_R, itself), and the other child is mapped to a node that is incomparable to v_R). Formally, these equations are:

$$\begin{aligned}
c_{\texttt{CODIV}}(v_D, v_R) &= \begin{cases} \min_{\forall(x,y)}\{c(l(v_D),x) + loss_{\texttt{CODIV}}(l(v_D),x) \\ \qquad + c(r(v_D),y) + loss_{\texttt{CODIV}}(r(v_D),y): \\ \qquad x \text{ incomparable to } y, \text{ and } lca(x,y) = v_R\ \} & \text{if } v_R \in V_R \setminus L(T_R), \\ \infty & \text{otherwise,} \end{cases} \\
c_{\texttt{DUP}}(v_D, v_R) &= \delta + \min_{\forall(x,y)}\{c(l(v_D),x) + loss_{\texttt{DUP}}(l(v_D),x) + c(r(v_D),y) \\
&\qquad + loss_{\texttt{DUP}}(r(v_D),y):\ x \leq_R v_R, y \leq_R v_R\}, \\
c_{\texttt{INS}}(v_D, v_R) &= \tau + \min_{\forall(x,y)}\{c(l(v_D),x) + c(r(v_D),y):\ x \leq_R v_R \text{ and } y \text{ incomparable to } v_R.\}
\end{aligned} \tag{4.6}$$

where δ is the cost of a duplication, λ is the cost of a loss, and τ is the cost of an insertion. To determine the optimal event and left and right child mappings to store in the traceback table, t, the arguments that provide the minimums for the cost table are stored.

To complete these tables, we must loop through all nodes in the domain tree, and for each domain node, we must loop through all nodes in the reference tree. In addition, for each domain-reference pair, we must also consider all possible reference mappings for the children of the domain. Thus, the time complexity to determine the optimal mapping of one domain-reference pair is $O(m^2)$ — for each mapping of the left child (of m possible mappings), consider every mapping of the right child (also of size m). Together, examining every domain-reference pair ($n \cdot m$ possible pairs), the time complexity to fill these tables is $O(n \cdot m^3)$. We can reduce the complexity by a factor of m, to $O(n \cdot m^2)$, if we take advantage of the observation that the optimal pair of children mappings can de determined by each child individually (see Alg. 4.2), rather then consider every possible pair of child mappings. The recursion equation in Eq. 4.6 becomes:

$$\begin{aligned}
c_{\texttt{CODIV}}(v_D, v_R) &= \min_{\forall x}\{c(l(v_D),x)\} + loss_{\texttt{CODIV}}(l(v_D),x) \\
&\quad + \min_{\forall y}\{c(r(v_D),y)\} + loss_{\texttt{CODIV}}(r(v_D),y): \\
&\quad x \text{ incomparable to } y,\ lca(x,y) = v_R,\ v_R \in V_R \setminus L(T_R), \\
c_{\texttt{DUP}}(v_D, v_R) &= \delta + \min_{\forall x}\{c(l(v_D),x) + loss_{\texttt{DUP}}(l(v_D),x): x \leq_R v_R\} \\
&\quad + \min_{\forall y}\{c(r(v_D),y) + loss_{\texttt{DUP}}(r(v_D),y): y \leq_R v_R\}, \\
c_{\texttt{INS}}(v_D, v_R) &= \tau + \min_{\forall x}\{c(l(v_D),x): x \leq_R v_R\} + \min_{\forall y}\{c(r(v_D),y): \ y \text{ incomparable to } v_R\}.
\end{aligned} \tag{4.7}$$

Once the entire tree has been traversed, the minimal cost solution is selected.

The second pass traverses the domain tree to generate a candidate optimal reconciliation from these cost and traceback tables. First, the optimal reconciliation at the root, $\rho_D = root(T_D)$ is selected:

$$M(\rho_D) = \arg\min_{v_R \in V_R}\{c(\rho_D, v_R)\}. \tag{4.8}$$

The event at this node and the children mappings can then be recalled from the traceback table,

$$\{event(\rho_D),\ M(l(\rho_D)),\ M(r(\rho_D))\} = \ t(\rho_D, M(\rho_D)). \tag{4.9}$$

This process is continued, until all nodes in T_D are mapped to nodes of T_R and are assigned a co-divergence, duplication, and insertion event. Then, losses can be determined. This process is

similar to the `climb` function in Alg. 4.1. One loss is assigned to the edge above node $v_D \in V_D$ for each "skipped" node from $M(v_D)$ to $M(p(v_D))$, and for $M(p(v_D))$ if v_D is a duplication and $M(v_D) \neq M(p(v_D))$.

If we only care about a single optimal reconciliation (i.e., a random reconciliation from the set of all possible, optimal reconciliations), we simply have to find the best mapping at the root of T_D, and traverse the domain tree, pulling data from the traceback and cost tables. In this case, the traceback complexity is just $O(n+m)$ — finding the minimum cost mapping at the root is $O(m)$ and traversing the domain tree is $O(n)$. However, if we would like to review *all* optimal reconciliations (as in Alg. 4.2), there is an increase in complexity. In this case, we must keep track of all solution paths in the traceback table (i.e., by having entries in the cost and traceback tables be a *list* of solutions). Each optimal scenario (the mapping and associated events) are stored in the tables; if a scenario generates a better cost, all previous stored information is cleared and the new scenario is stored. This allows us to keep track and generate *all* optimal scenarios. The traceback is still initiated with finding the optimal mapping of the root, which is still $O(m)$. When traversing T_D, we must consider every optimal mapping and event; however, it is possible that many mappings produce an optimal solution. Also, a mapping may be optimal with multiple children mapping pairs. Thus, the time complexity for the traceback is $O(n \cdot m^3)$ — n to traverse the tree and at each node consider all optimal mappings $O(m)$ and all optimal children pair mappings $O(m^2)$.

The third pass checks the candidate reconciliation for violations of temporal constraints that can arise if insertions create a cycle. In particular, an event history has a cycle if it includes a transfer from taxon x to an ancestor of taxon y *and* a transfer from taxon y to an ancestor of taxon x. This is not permitted as it would require that both that x existed before y and that y existed before x, which is impossible. If a cycle is detected, it is repaired. Cycle checking runs in polynomial time: $O(n+m^2)$. However, the problem of repairing a cycle, if one is detected, is NP-complete [232,233, 235]. My work on domain shuffling in the human genome (see Sec. 6) performed 3400 domain reconciliations, of which only 6 contained a cycle, suggests that cycles are rare.

Algorithm 4.2

Input: $T_D = (V_D, E_D)$; $T_R = (V_R, E_R)$; σ, the $M(\cdot)$ of $L(T_D) \rightarrow L(T_R)$

```
reconcileDTL( T_D, T_R, σ )
 1 initializeTables()        //Initializes the cost and traceback tables.
 2 for each v_D ∈ V_D in postorder do
```

```
 3   calculateTable( v_D )
 4 bestRoot = argmin_{v_R ∈ V_R}{ c(ρ_D, v_R) }
 5 traceback( ρ_D, bestRoot )
 6 return

initializeTables()
 7   // Each table is 2-D with V_D as rows and V_R as columns.
 8   for each v_D ∈ V_D do
 9      for each v_R ∈ V_R do
10         c( v_D, v_R ) ← ∞  // total event cost for subtree rooted at v_D and labeled v_R.
11
12         // This table is not required; however, if this information is stored,
13         // the traceback function does not need to recompute it.
14         // Each entry is a set:  {best event, best left child map, best right child map}
15         // Default is:  { co-divergence, empty, empty }.
16         t( v_D, v_R ) ← { CODIV, ∅, ∅ }
17   return

calculateTable( v_D )
18   if ( v_D ∈ L(T_D) ) then // Leaf case
19      // Can only map to reference taxon from which it was sampled or an ancestor,
20      // indicating a co-divergence with losses.
21      v_R ← σ( v_D )
22      c( v_D, v_R ) ← 0
23      for each v'_R ∈ ancestor( v_R ) do
24         totalLoss ← λ * ( height(v_R) − height(v_R) − 1 )
25         c( v_D, v'_R ) ← totalLoss
26   else do
27      for each v_R ∈ V_R in postorder do
28         calcCost( v_R, CODIV ) // Co-divergence
29         calcCost( v_R, DUP )  // Duplication
30         calcCost( v_R, INS )  // Insertion
31   return

calcCost( v_R, event )
32   if ( event == CODIV ) then
33      // In a co-divergence, one child is labeled as a right descendant, and the other
34      // child is labeled as a left descendant.
35      // Cannot be a leaf label because it has no children.
36      if ( v_R ∈ L(T_R) ) then
37         return
```

```
38    else do
39       childLabelLeft = descendants( l(v_R) )
40       childLabelRight = descendants( r(v_R) )
41 else if ( event == DUP ) then
42    // Either child can be labeled with the node or any of its descendants.
43    childLabelLeft = descendants( v_R )
44    childLabelRight = childLabelLeft
45 else do    // Insertion case
46    // With an insertion, one child must be labeled with the node and the other child
47    // labeled something incomparable (neither ancestor or descendant).
48    // Label can't be root because it has no incomparable nodes.
49    if ( v_R == ρ_R ) then
50       return
51    else do
52       childLabelLeft = incomparable( v_R )
53       childLabelRight = descendants( v_R )
54
55 for each vL_R ∈ childLabelLeft do
56    for each vR_R ∈ childLabelRight do
57       minAll = c( v_D, v_R )
58       //Initialize costs for this node.
59       totalLoss = 0; sum1 = ∞; sum2 = ∞
60       if ( event == CODIV ) then
61          // Only costly events are losses
62          totalLoss = λ * ( 2 * height(v_R) − height(vL_R) − height(vR_R) − 2 )
63          // totalLoss = λ * ( height(v_R) − height(vL_R) − 1 +
64          //       height(v_R) − height(vR_R) − 1 )
65          // Check both children for both labels
66          sum1 = c( l(v_D), vL_R ) + c( r(v_D), vR_R ) + totalLoss
67          sum2 = c( r(v_D), vL_R ) + c( l(v_D), vR_R ) + totalLoss
68       else if ( event == DUP ) then
69          // Event cost is from loss and duplication.
70          totalLoss = λ * ( 2 * height(v_R) − height(vL_R) − height(vR_R) )
71          // Don't need to swap, since vL_R and vR_R are pulled from identical sets.
72          sum1 = c( l(v_D), vL_R ) + c( r(v_D), vR_R ) + totalLoss + δ
73       else do
74          // Only cost is from insertion.  Most parsimonious to infer direct insertion
75          // from donor, vL_R to recipient, vR_R, with no losses.
76          sum1 = c( l(v_D), vL_R ) + c( r(v_D), vR_R ) + τ
77          sum2 = c( r(v_D), vL_R ) + c( l(v_D), vR_R ) + τ
78
```

```
79        if ( sum1 ≤ minAll )
80           minAll = sum1
81           c( v_D, v_R ) ← sum1
82           t( v_D, v_R ) ← { event, vL_R, vR_R }
83        if ( sum2 ≤ minAll )
84           minAll = sum2
85           c( v_D, v_R ) ← sum2
86           t( v_D, v_R ) ← { event, vR_R, vL_R }
87
88    return

traceback( v_D, v_R )
89    { event, bestLeftLabel, bestRightLabel } = t( v_D, v_R )
90    if ( v_D ∉ L(T_D) ) then
91       traceback( l(v_D), bestLeftLabel )
92       traceback( r(v_D), bestRightLabel )
93    M( v_D ) ← v_R
94    event( v_D ) ← event
95
96    if ( v_D == ρ_D ) then
97       return reconciled T_D
98    else do
99       return
```

Algorithm 4.2 (a)

```
calcCost( v_R, event )
    ⋮
54    for each vL_R ∈ childLabelLeft do
55       minLeftLeft = ∞; minRightLeft = ∞
56       //Initialize costs for this node.
57       totalLoss = 0; costLeftLeft = ∞; costRightLeft = ∞
58       if ( event == CODIV ) then
59          // Only costly events are losses
60          totalLoss = λ * ( height(v_R) − height(vL_R) − 1 )
61          costLeftLeft = c( l(v_D), vL_R ) + totalLoss
62          costRightLeft = c( r(v_D), vL_R ) + totalLoss
63       else if ( event == DUP ) then
64          // Event cost is from loss and duplication.
65          totalLoss = λ * ( height(v_R) − height(vL_R) )
```

```
66        // Don't need to swap, since vL_R and vR_R are pulled from identical sets.
67        costLeftLeft = c( l(v_D), vL_R ) + totalLoss + δ
68     else do
69        // Only cost is from insertion.  Most parsimonious to infer direct insertion
70        // from donor, vL_R, to recipient, vR_R, with no losses.
71        costLeftLeft = c( l(v_D), vL_R ) + τ
72        costRightLeft = c( r(v_D), vL_R ) + τ
73     if ( costLeftLeft ≤ minLeftLeft ) then
74        minLeftLeft = costLeftLeft
75        leftLabelLeft = vL_R
76     if ( costRightLeft ≤ minRightLeft ) then
77        minRightLeft = costRightLeft
78        rightLabelLeft = vL_R
79
80  for each vR_R ∈ childLabelRight do
81     minLeftRight = ∞; minRightRight = ∞
82     //Initialize costs for this node.
83     totalLoss = 0; costLeftRight = ∞; costRightRight = ∞
84     if ( event == CODIV ) then
85        totalLoss = λ * ( height(v_R) − height(vR_R) − 1 )
86        costRightRight = c( r(v_D), vR_R ) + totalLoss
87        costLeftRight = c( l(v_D), vR_R ) + totalLoss
88     else if ( event == DUP ) then
89        totalLoss = λ * ( height(v_R) − height(vR_R) )
90        // Don't add dup cost, because added to node with vL_R label
91        costRightRight = c( r(v_D), vR_R ) + totalLoss
92     else do
93        // Don't add ins cost, because added to node with vL_R label
94        costRightRight = c( r(v_D), vR_R )
95        costLeftRight = c( l(v_D), vR_R )
96     if ( costRightRight ≤ minRightRight ) then
97        minRightRight = costRightRight
98        rightLabelRight = vR_R
99     if ( costLeftRight ≤ minLeftRight ) then
100       minLeftRight = costLeftRight
101       leftLabelRight = vR_R
102
103 sum1 = minLeftLeft + minRightRight
104 sum2 = minRightLeft + minLeftRight
105 if ( sum1 < sum2 && sum1 < c(v_D,v_R) ) then
106    c( v_D, v_R ) ← sum1
```

```
107    t( v_D, v_R ) ← { event, leftLabelLeft, rightLabelRight }
108 else if ( sum2 < c(v_D,v_R) ) do
109    c( v_D, v_R ) ← sum2
110    t( v_D, v_R ) ← { event, leftLabelRight, rightLabelLeft }
111
112 return
```

Algorithm 4.2 (b)

```
reconcileDTL( T_D, T_R, σ )
 1 initializeTables()        //Initializes the cost and traceback tables.
    ⋮
 4 { bestRoot } = argmin_{v_R∈V_R}{ c( ρ_D, v_R ) }
 5 for each v_R ∈ { bestRoot } do
 6   traceback( ρ_D, v_R )
 7 return
    ⋮
calcCost( v_R, event )
    ⋮
78       if ( sum1 < minAll ) then
79          minAll = sum1
80          c( v_D, v_R ) ← sum1
81          clear t( v_D, v_R )
82       if ( sum1 == minAll ) then
83          add { event, vL_R, vR_R } to t( v_D, v_R )
84       if ( sum2 < minAll ) then
85          minAll = sum2
86          c( v_D, v_R ) ← sum2
87          clear t( v_D, v_R )
88       if ( sum2 == minAll )
89          add { event, vR_R, vL_R } to t( v_D, v_R )
90
91  return

traceback( v_D, v_R )
92  for each { event, bestLeftLabel, bestRightLabel } ∈ t( v_D, v_R ) do
93    if ( v_D ∉ L(T_D) ) then
94       traceback( l(v_D), bestLeftLabel )
95       traceback( r(v_D), bestRightLabel )
96    M( v_D ) ← v_R
```

```
97     event( v_D ) ← event
98
99     if ( v_D == ρ_D ) then
100       return reconciled T_D
101    else do
102       return
```

4.3. Duplication-transfer-loss-incomplete lineage sorting algorithm

We now have a reconciliation algorithm to compare a binary embedded tree with a non-binary reference tree in order to infer duplication, heuristic loss, and incomplete lineage sorting and a reconciliation algorithm to compare a binary embedded tree with a binary reference tree in order to infer horizontal transfer, duplication, and loss. But what if we would like to compare a binary embedded tree with a *non-binary* reference tree to infer all four types of events: duplication, horizontal transfer, loss, and incomplete lineage sorting? There are no solutions to this problem in the literature, but such a reconciliation algorithm is needed when reconciling gene families or domain superfamilies in species where gene transfer events and incomplete lineage sorting are not uncommon. To satisfy this need, a third reconciliation algorithm was developed in a collaboration with Han Lai. This algorithm builds on the two algorithms above. In particular, novel set mappings N and $\widehat{N}$ in Alg. 4.1 and the dynamic programming approach and search of the reconciliation space in Alg. 4.2 were combined to create the dynamic programming algorithm, Alg. 4.3 that infers the optimal set of duplication, transfer, heuristic loss, and incomplete lineage sorting events given a binary embedded tree and non-binary reference tree.

The basic structure of this algorithm is similar to that in Alg. 4.2, with some exceptions. We now consider co-divergence, duplication, and incomplete lineage sorting events together because of their intrinsic dependence upon one another. When considering these events (referred to as `NOTINS` in the algorithm), the set mappings $\widehat{N}$ and N, defined in Eqs. 4.1 and 4.3, are used to distinguish between co-divergence, incomplete lineage sorting, and duplication. It is important to note that $\widehat{N}$ and N for node v_E rely on the $\widehat{N}$ and N of v_E's children and on the node v_R to which v_E is mapped. Because the dynamic program considers all possible mappings of a node and its children, we must keep track of the $\widehat{N}$ and N sets for each of these mappings as well. This is

accomplished by defining two new tables, $t\widehat{N}$ and tN, such that $t\widehat{N}(v_E, v_R)$ and $tN(v_E, v_R)$ contain, respectively, $\widehat{N}(v_E)$ and $N(v_E)$ for the optimal reconciliation of T_{v_E} with v_E mapped to v_R.

In this new algorithm, the $\widehat{N}$'s of the node's children are calculated using a `climb` function, similar to the one in Alg. 4.1. This function climbs from the mapping of the child node (e.g., vL_R) to the mapping of the parent node, v_R, in order to infer the number of polytomy and skipped species losses. It also defines the children of v_R in which descendants of the embedded child node (e.g., $l(v_E)$) must have been present. The N of v_E is then calculated as the union of the children $\widehat{N}$'s, similarly to Alg. 4.1. As defined in Eq. 4.2, the event at node v_E was a duplication if $\widehat{N}(l(v_E)) \cap \widehat{N}(r(v_E)) \neq \emptyset$ for the current nodes and mappings under consideration. Else, if v_R equals vL_R or vR_R, the event at v_E is incomplete lineage sorting (congruent to $M(v_E)$ equal to $M(l(v_E))$ or $M(r(v_E))$ in Alg. 4.1). Otherwise, the event at v_E was a co-divergence. If the reconciliation of T_{v_E} with v_E mapped to v_R is optimal, $\widehat{N}(l(v_E))$, $\widehat{N}(r(v_E))$, and $N(v_E)$ are stored in $t\widehat{N}(l(v_E), vL_R)$, $t\widehat{N}(r(v_E), vR_R)$, and $tN(v_E, v_R)$, respectively. During traceback, $N(v_E)$ and $\widehat{N}(v_D)$ are retrieved from the tN and $t\widehat{N}$ tables just as $M(v_E)$ and the event(v_E) are retrieved from the traceback table t.

Algorithm 4.3

Input: $T_E = (V_E, E_E)$; $T_R = (V_R, E_R)$; σ, the $M(\cdot)$ of $L(T_E) \rightarrow L(T_R)$

```
reconcileDTLI( T_E, T_R, σ )
 1 initializeTables( )        //Initializes the cost and traceback tables.
 2 for each v_E ∈ V_E in postorder do
 3   calculateTable( v_E )
 4 bestRoot = argmin_{v_R ∈ V_R}{ c(ρ_E, v_R) }
 5 traceback( ρ_E, bestRoot )
 6 return

initializeTables( )
 7   // Each table is 2-D with V_E as rows and V_R as columns.
 8   for each v_E ∈ V_E do
 9     for each v_R ∈ V_R do
10       c( v_E, v_R ) ← ∞  // total event cost for subtree rooted at v_E and labeled v_R.
11
12       // Tables for keeping track of the children that were visited.  Essential for
13       // distinguishing between duplications and incomplete lineage sorting events
14       // Default is the empty set.
15       tN̂( v_E, v_R ) ← ∅
```

```
16        tN( v_E, v_R ) ← ∅
17
18        // This table is not required; however, if this information is stored,
19        // the traceback function does not need to recompute it.
20        // Each entry is a set:  {best event, best left child map, best right child map}
21        // Default is:  { co-divergence, empty, empty }.
22        t( v_E, v_R ) ← { CODIV, ∅, ∅ }
23  return

calculateTable( v_E )
24  if ( v_E ∈ L(T_E) ) then  // Leaf case
25     // Can only map to reference taxon from which it was sampled or an ancestor,
26     // indicating a co-divergence with losses.
27     v_R ← σ( v_E )
28     c( v_E, v_R ) ← 0
29     t̂N( v_E, v_R ) ← { v_R }
30     tN( v_E, v_R ) ← { v_R }
31     for each v'_R ∈ ancestor( v_R ) do
32        totalLoss = λ * ( height(v_R) − height(v_R) − 1 )
33        c( v_E, v'_R ) ← totalLoss
34        t̂N( v_E, v_R ) ← C( v'_R )
35        tN( v_E, v_R ) ← C( v'_R )
36  else do
37     for each v_R ∈ V_R in postorder do
38        calcCost( v_R, NOTINS )  // Co-divergence, Incomplete lineage sorting or
39                                 // Duplication
40        calcCost( v_R, INS )  // Insertion
41  return

calcCost( v_R, type )
42  if ( type == NOTINS ) then
43     // For co-divergence, ILS, or duplication, children of v_E are mapped to children
44     // of v_R or v_R itself.
45     // If the children mappings are on the same path, then there is a duplication.
46     // Else, if if v_R is a polytomy and at least one descendent of v_E was already mapped
47     // to another child of v_R, there was ILS
48     // Otherwise, there was a co-divergence.
49     childLabelLeft = descendants( v_R )
50     childLabelRight = childLabelLeft
51  else do    // Insertion case
52     // With an insertion, one child must be labeled with the node and the other child
```

```
53    // labeled something incomparable (neither ancestor or descendant).
54    // Label can't be root because it has no incomparable nodes.
55    if ( v_R == ρ_R ) then
56       return
57    else do
58       childLabelLeft = incomparable( v_R )
59       childLabelRight = descendants( v_R )
60
61 for each vL_R ∈ childLabelLeft do
62    for each vR_R ∈ childLabelRight do
63       minAll = c( v_E, v_R )
64       //Initialize costs for this node.
65       totalLoss = 0; sum1 = ∞; sum2 = ∞
66       if ( event == NOTINS ) then
67          { N̂_L1, lossL1 } = climb( l(v_E), vL_R, v_R )
68          { N̂_R1, lossR1 } = climb( r(v_E), vR_R, v_R )
69          N1 = N̂_L1 ∪ N̂_R1
70          totalLoss1 = λ * ( lossL1 + lossR1 )
71          if ( N̂_L1 ∩ N̂_R1 ≠ ∅ ) then
72             event1 = DUP
73             // Duplication losses
74             if ( N1 ≠ N̂_L1 ) then
75                totalLoss1 += λ
76             if ( N1 ≠ N̂_R1 ) then
77                totalLoss1 += λ
78             sum1 = δ + totalLoss1
79          else do
80             sum1 = totalLoss1
81             if ( v_R == vL_R || v_R == vR_R ) then
82                event1 = ILS
83             else do
84                event1 = CODIV
85
86          { N̂_R2, lossR2 } = climb( r(v_E), vL_R, v_R )
87          { N̂_L2, lossL2 } = climb( l(v_E), vR_R, v_R )
88          N2 = N̂_L2 ∪ N̂_R2
89          totalLoss2 = λ * ( lossL2 + lossR2 )
90          if ( N̂_L2 ∩ N̂_R2 ≠ ∅ ) then
91             event2 = DUP
92             // Duplication losses
93             if ( N2 ≠ N̂_L2 ) then
```

```
94              totalLoss2 += λ
95            if ( N2 ≠ N̂_R2 ) then
96              totalLoss2 += λ
97            sum2 = δ + totalLoss2
98          else do
99            sum2 = totalLoss1
100           if ( v_R == vL_R || v_R == vR_R ) then
101             event2 = ILS
102           else do
103             event2 = CODIV
104
105         sum1 += c( l(v_E), vL_R ) + c( r(v_E), vR_R )
106         sum2 += c( r(v_E), vL_R ) + c( l(v_E), vR_R )
107
108       else do
109         event1 = INS; event2 = INS
110         // Inferring insertion from donor, vL_R to recipient, vR_R.
111         { N̂_L1, lossL1 } = climb( l(v_E), vL_R, v_R )
112         { N̂_R1, lossR1 } = climb( r(v_E), vR_R, v_R )
113         sum1 = c( l(v_E), vL_R ) + c( r(v_E), vR_R ) + τ + λ * ( lossL1 + lossR1 )
114         N1 = N̂_L1 ∪ N̂_R1
115
116         { N̂_L2, lossL2 } = climb( l(v_E), vR_R, v_R )
117         { N̂_R2, lossR2 } = climb( r(v_E), vL_R, v_R )
118         sum2 = c( r(v_E), vL_R ) + c( l(v_E), vR_R ) + τ + λ * ( lossL2 + lossR2 )
119         N2 = N̂_L2 ∪ N̂_R2
120
121       if ( sum1 ≤ minAll )
122         minAll = sum1
123         c( v_E, v_R ) ← sum1
124         t( v_E, v_R ) ← { event1, vL_R, vR_R }
125         tN̂( l(v_E), vL_R ) ← N̂_L1
126         tN̂( r(v_E), vR_R ) ← N̂_R1
127         tN( v_E, v_R ) ← N1
128       if ( sum2 ≤ minAll )
129         minAll = sum2
130         c( v_E, v_R ) ← sum2
131         t( v_E, v_R ) ← { event2, vR_R, vL_R }
132         tN̂( l(v_E), vR_R ) ← N̂_L2
133         tN̂( r(v_E), vL_R ) ← N̂_R2
134         tN( v_E, v_R ) ← N2
```

```
135
136 return

traceback( v_E, v_R )
137 { event, bestLeftLabel, bestRightLabel } = t( v_E, v_R )
138 if ( v_E ∉ L(T_E) ) then
139    traceback( l(v_E), bestLeftLabel )
140    traceback( r(v_E), bestRightLabel )
141 M( v_E ) ← v_R
142 N( v_E ) ← tN( v_E, v_R )
143 N̂( v_E ) ← tN̂( v_E, v_R )
144 event( v_E ) ← event
145
146 if ( v_E == ρ_E ) then
147    return reconciled T_E
148 else do
149    return

climb( v_E, v_R, u_R )
150 losses = 0
151 // general polytomy losses
152 if ( (v_R ∉ L(T_R) && v_R ≠ u_R) && (C(v_R) ≠ tN(v_E,v_R)) ) then
153    losses++
154 select x from tN̂( v_E, v_R ) at random
155 if ( x == u_R || p(x) == u_R ) then
156    return tN̂( v_E, v_R )
157 while ( p(x) ≠ u_R ) do
158    // skipped losses
159    if ( p(x) ≠ v_R ) then
160       losses++
161    // climb
162    x = p( x )
163 return { x, losses }
```

4.4. Algorithm to infer composite history of reference tree

Given a multidomain family tree and a set of reconciled domain trees, I developed an algorithm (Alg. 4.4) to construct a composite reference tree history with complete ancestral architectures and

inferred events from all domains. There is no prior work addressing this problem, because it does not arise in other instantiations of the co-evolutionary framework, such as gene tree-species tree reconciliation. In those problems, only the history of the *embedded* tree is of interest.

My algorithm constructs this history by transferring information from each reconciled domain tree to the appropriate branch in the reference tree. Determining the ancestral architectures is simply a matter of copying ancestral states from domain to host tree. However, there are challenging questions regarding the transfer of events and how to present such information to the user. The transfer of events is more complex because multiple events may be associated with a single branch and the order of events is partially constrained by the structure of the domain trees. Only events affecting instances of the same lineage in the domain tree effect relative timing. Instances that are unrelated on that branch in the reference tree have no impact on relative timing. For example, in Fig. 3.2, the events in the $g3$ lineage include a duplication and transfer to $g2$ of domain p. In the $g3$ lineage, there is the insertion from $g3$ of p and the loss of an r domain. We know that in $g3$ the duplication of p must have occurred before the insertion, because it involved one of the resulting paralogs. However, in $g2$ it is not possible to determine whether the loss of r occurred before or after the insertion. Therefore, transferring events from domain to host tree requires determining the (partial) temporal order of the events associated with each host tree branch.

Algorithm 4.4

Input: $T_R = (V_R,\ E_R)$; $setDT = \{\ T_{D_i} = (V_{D_i},\ E_{D_i})\ \}$, the set of all domain trees (for each domain D_i) reconciled with T_R.
Losses have been added; for such nodes, the event is LOSS.

```
annotateReference( T_R, setDT )
 1 for each T_Di ∈ setDT do
 2   for each v_Di ∈ V_Di do
 3     v_R = M( v_Di )  // Get the mapping of the node
 4     // Get the event at this node and add to list of events at mapped reference node
 5     add event(v_Di) to events( v_R, D_i )
 6     if ( event(v_Di) == LOSS ) then
 7       add v_Di to losses( v_R, D_i )
 8     else do
 9       if ( v_Di ∈ L(T_Di) || event(v_Di) == CODIV ) then
10         // Domain D_i was present in v_R and not a node effecting copy number.
11         // Counting such nodes will overcount domains.
12         add v_Di to architecture( v_R, D_i )
```

```
13      if ( event(v_Di) == DUP ) then
14         add { v_Di, C(v_Di) } to duplications( v_R, D_i )
15      else if ( event(v_Di) == INS ) then
16         donor = v_R
17         if ( M( l(v_Di) ) ≠ v_R ) ) then
18            // For an insertion, one child of v_Di must have the same mapping as v_Di
19            // (descendant of the donor).  The other child is the result of the
20            // insertion (the mapping of that child is the recipient of the insertion).
21            recipient = M( l(v_Di) )
22            add { v_Di, l(v_Di), recipient } to insertions( v_R, D_i )
23         else do
24            recipient = M( r(v_Di) )
25            add { v_Di, r(v_Di), recipient } to insertions( v_R, D_i )
26         e = (donor, recipient)
27         if ( e ∉ E_R) then
28            add e to E_R
```

Chapter 5

Case-studies: Histories of multidomain families from the literature inferred using the co-evolutionary framework

Several studies in the literature [27–30, 139] have used comparison of domain trees to gain insight into the history of domain shuffling using informal approaches based on visual inspection of the trees. However, inference by visual inspection is error prone and too difficult for large data sets — even small trees can lead to very complicated scenarios (see, for example, the MaGuK family in Fig. 5.7). In contrast, using my algorithms and software, this type of analysis can be carried out consistently and on a much larger scale. In order to demonstrate the utility and effectiveness of the work presented in Chapter 4, I selected and analyzed a set of three multidomain families with diverse domain architectures discussed in the literature [27, 28, 30]: the protein tyrosine kinases, the Notch-related genes, and the membrane-associated guanylate kinases (MAGuKs). The original studies compared domain trees, through visual inspection, to gain insight into the history of domain shuffling events. In this section, I present these families, the results from my analyses, and discuss the impact of using my novel co-evolutionary approach.

Tree construction. Trees in these studies were either transcribed from the literature or reconstructed from sequences. When necessary, I reconstructed trees using the following procedure:

1. Initial multiple alignments were constructed using the T-Coffee program [286].

2. I further edited these multiple alignments manually using GeneDoc [287], considering information from the literature such as conserved and functional positions.

3. Domain trees were constructed using the Neighbor Joining [51] program in Phylip [288].

4. Phylip's Seqboot program was used to generate 100 bootstrap replicates for each tree.

5. Reference trees were generally rooted using an outgroup sequence or information from the literature, as described below. Domain trees were rooted using the rooting parsimony approach in NOTUNG, described on pg. 59.

5.1. Notch-related proteins

The Notch family consists of transmembrane receptor proteins that mediate cell-cell interactions and signalling that are important during development in metazoa [28, 289, and work cited therein]. Family members are characterized by a series of EGFL repeats, a set of three LNG domains, a transmembrane domain, and a series of six cdc10/Ankyrin (Ank) domains. The co-evolution of domains in this multidomain family was first investigated in phylogenetic context by Maine et al. [28], who constructed trees for the extracellular sub-sequence of the family and the LNG and Ank domains in eight metazoan species. Their phylogenies were constructed by first aligning sequences using ClustalV [290] and then building trees with Phylip [288]. They then compared the trees for the repeated LNG and Ank domains with the tree for the extracellular sequence. Because the trees were fairly consistent, they concluded that the ancestral family contained the same domain architecture with three LNG domains and six Ank domains. For my analysis, I transcribed these trees and analyzed them with my methods in NOTUNG (Fig. 5.1). Non-binary nodes and weakly supported edges were resolved using event parsimony in NOTUNG. My results suggest a similar conclusion to those made by Maine et al. However, while the trees are largely consistent with one another, Maine fails to comment on some of the observed tree disagreement. My analysis not only locates this disagreement, but infers the possible domain shuffling events: *(1)* the first Ank domain in rat Notch1, xenopus Xotch, human TAN1, and mouse Notch1 are the result of an ancestral domain duplication and reciprocal losses; *(2)* the second LNG domain in mouse MotchA is the result of a domain insertion from the second LNG domain in mouse MotchB; and *(3)* the first LNG in mouse Motch1 is the result of a domain insertion from the first LNG in mouse MotchA. The extracellular sequences, which were treated as the reference in the paper and my analysis, were

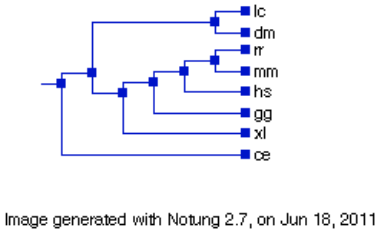
also reconciled with the species tree, indicating a number of ancestral and lineage-specific gene duplications that were also noted in the paper.

5.2. Protein tyrosine kinases

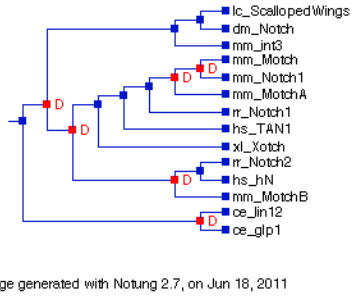
Protein tyrosine kinases (PTKs) phosphorylate proteins by transferring a phosphate group from ATP to a tyrosine residue in the protein. They are involved in many important cellular processes, including cell signaling, cell growth, cell differentiation, metabolism, cell-cell adhesion, cell motility, and cell death. PTKs are generally divided, both functionally and evolutionary, into two classes: transmembrane receptor-linked PTKs and cytoplasmic non-receptor PTKs [291]. Fig. 5.2 shows the family tree, as represented by the kinase domain, with both the cytoplasmic and receptor classes. I built this tree from the sequences of all kinase domains in humans. The kinase domain is thought to be a primary domain (as defined on pg. 55), and is thus used as the reference tree to represent the evolution of the multidomain PTK family.

As seen in Fig. 5.2, many members of the cytoplasmic class contain the SH3 and SH2 domains, often as a pair. The phyletic distribution of these domains suggests several alternate hypotheses for the origins of these domain architectures. Was this the result of a single insertion of the domain pair, followed by losses of the pair in some descendants and losses of the SH3 domain in others and by a duplication of SH2 in others? Or were there an insertion of the pair in the Tec/Abl/Src/Frk/Csk/Fes ancestor with loss of SH3 in Fes, an insertion of SH3 in Ack, and an insertion with duplication of SH2 in Syk? According to the domain gain-loss approach (see pg 21), shown in Fig. 5.3, based only on domain architectures, the SH3-SH2 pair were gained at the ancestor of the Fes, Csk, Frk, Src, Tec, and Abl subfamilies and persisted in all domain architectures, with the exception of the loss of SH3 in the Fes subfamily. In addition, two SH2 domains were gained in the Syk subfamily and one SH3 domain was gained in the Ack subfamily. However, this approach does not consider the sequence evolution of the SH3 and SH2 domains, which may contradict this inference.

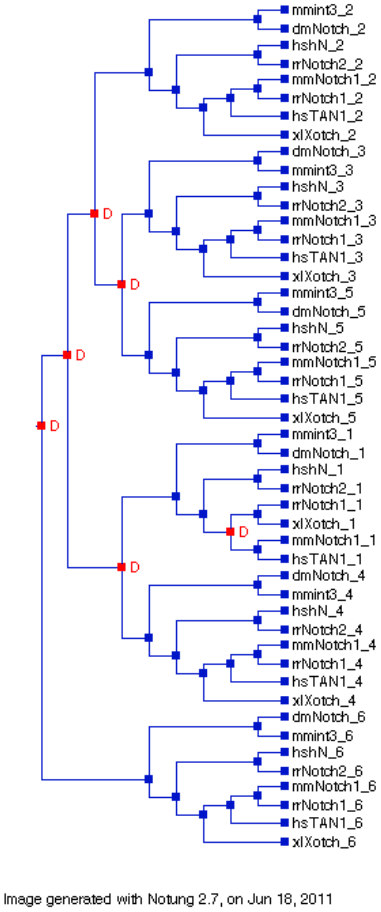
Nars et al. [27] originally analyzed this family, seeking the answers to these questions. They built trees for the SH3, SH2, and kinase domains were aligned using the Pileup program in GCG [292], and domain trees were constructed using maximum parsimony in the PAUP* program [293]. Trees were rooted using outgroup sequences. After a visual inspection to compare the trees, the authors concluded that since the domain trees were consistent, these domains co-evolved. I transcribed the



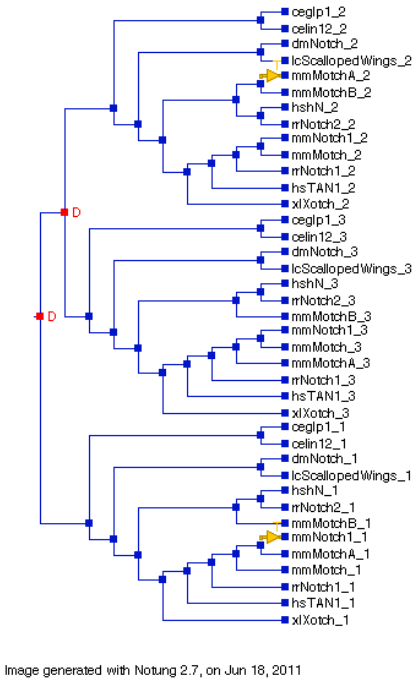
(a)



(b)



(c)



(d)

Figure 5.1: Trees from Maine et al. [28] **(a)** Species tree for the eight species. Abbreviations: ce - *Caenorhabditis elegans*; xl - *Xenopus laevis*; gg - *Gallus gallus*; hs - human (*Homo sapiens*); mm - *Mus musculus*; rr - *Rattus rattus*; dm - *Drosophila melanogaster*; and lc - *Lucilia cuprina* (blowfly). **(b)** Tree for the Notch extracellular sequences, reconciled with the species tree in (a). **(c)** Tree for the six repeated Ank domains in the sampled Notch genes reconciled with the extracellular sequence tree in (b). **(d)** Tree for the three LNG domains in the Notch gene family reconciled with the extracellular sequence tree in (b).

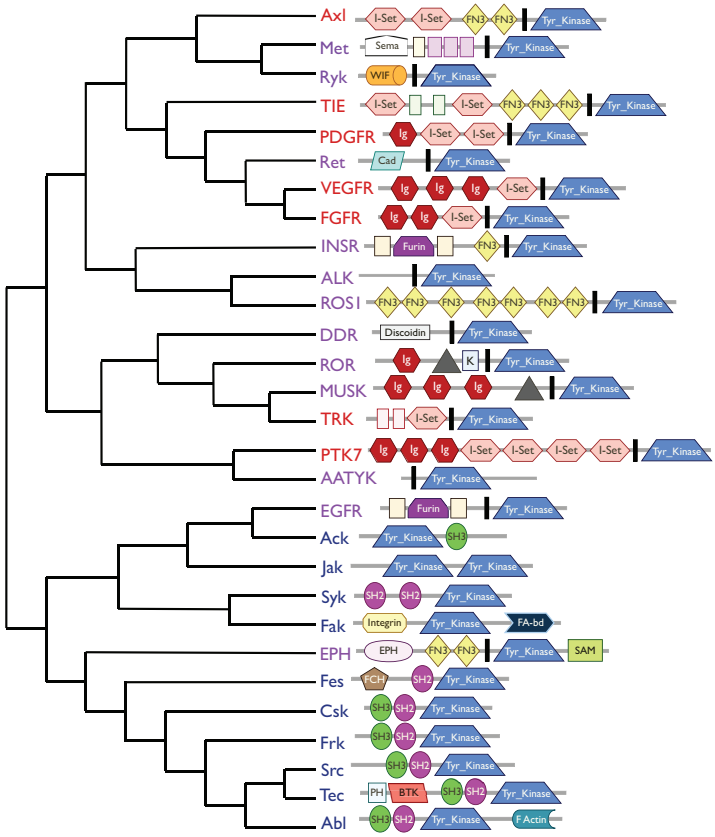


Figure 5.2: Tree for the PTK gene family, built from the sequence of the kinase domains. Subfamilies are collapsed to leaf nodes, and representative domain architectures for each subfamily are shown. Transmembrane receptor PTKs have a transmembrane domain, denoted as a black rectangle in the architectures. Transmembrane PTKs with an I-set domain are denoted with red colored names; those with an I-set domain have a purple colored name. Cytoplasmic PTKs are denoted with blue colored names. (Domain architectures were adapted from Robinson et al. [291]).

trees from this paper and used my new algorithms in NOTUNG to analyze these domains systematically. Because the trees had a number of non-binary nodes, I used NOTUNG to resolve these nodes using event parsimony. My analysis (seen in Fig. 5.4) generally agrees with the assessment of Nars et al. However, my analysis proposes three domain shuffling events that were not presented in the paper. In particular, *(1)* that the domains co-evolved from the insertion of SH3-SH2 in the Tec/Abl/Src/Frk/Csk/Fes ancestor, not the ancestor of all; *(2)* that the SH2 pair in the Syk subfamily (KSyk and Zap70) was the result of a domain insertion followed by a domain duplication; and *(3)* that the SH3 in the Ack subfamily (PTK6 and SrmS) was the result of a domain insertion. Nars et al. failed to comment on how the two SH2 domains arose in Syk or the observation that the Syk and Ack families are separate from the other SH3-SH2 containing families, which would require a number of losses to explain an insertion in the ancestor of all SH3-SH2 containing families.

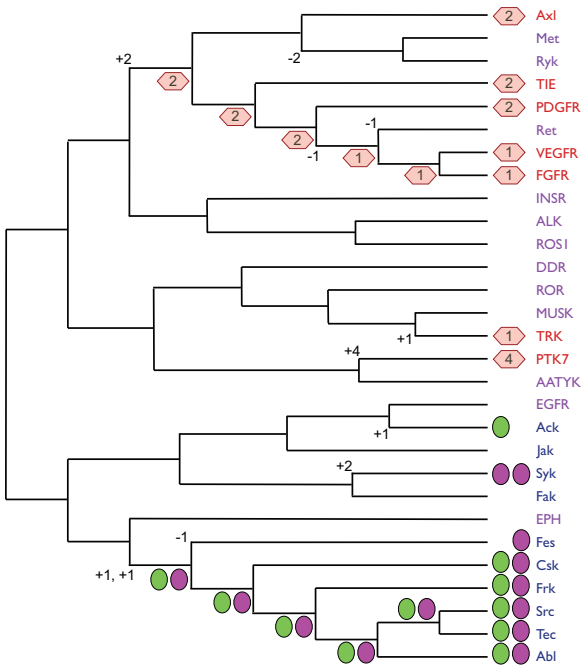
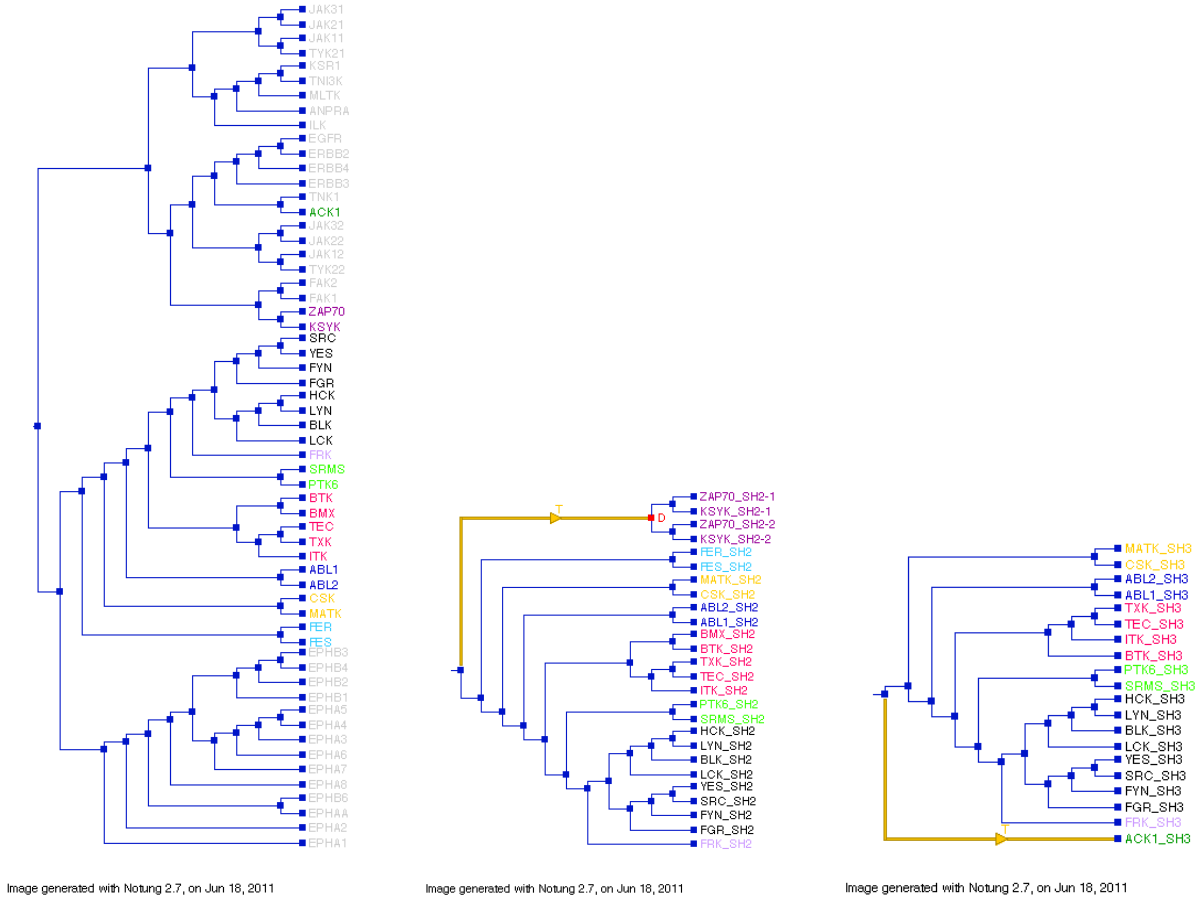


Figure 5.3: History of the gain and losses of SH2, SH3, and I-set domains in the PTKs using the domain gain-loss approach with the kinase tree, representing the history of the gene family.

I further analyzed this family by examining the evolutionary history of the immunoglobulin I-set domains. As can be noticed from Fig. 5.2, many members of the receptor sub-class contain repeated I-set domains. I sought to determine whether these domains, like SH2 and SH3, were the result of a single ancestral insertion, or whether additional domain shuffling occurred. The domain gain-loss approach (Fig. 5.3) infers an ancestral gain of two I-set domains in the Axl/Tie/PDGFR/VEGFR/FGFR ancestor followed by two losses in the Met/Ryk subfamily ancestor and one loss in each of the Ret/VEGFR/FGFR ancestral subfamily and the Ret subfamily. In addition, there were independent gains of four I-set domains in PTK7 and one I-set domain in NTRK3 (Trk).

Using my method in NOTUNG to compare the Ig I-set domain tree to the kinase reference tree revealed some interesting insights (see Figs. 5.5 and 5.6). First, the I-set domains in PTK7 appear to be the result of an ancestral insertion followed by local duplications, rather than inheritance of an ancestral PTK with multiple I-set domains. In this reconciliation, one I-set domain was present in the ancestor of all growth factor receptors (TIE1, PDGFRs, VGFRs and FGFRs) and the Axl (MERTK, UFO, and TYRO3), Met (MET and RON), and Ryk subfamilies. This domain was subsequently lost in the Met/Ryk ancestor and duplicated independently in the growth factor receptors and the Axl subfamily. Both copies of the domain in the Axl subfamily were lost in UFO.

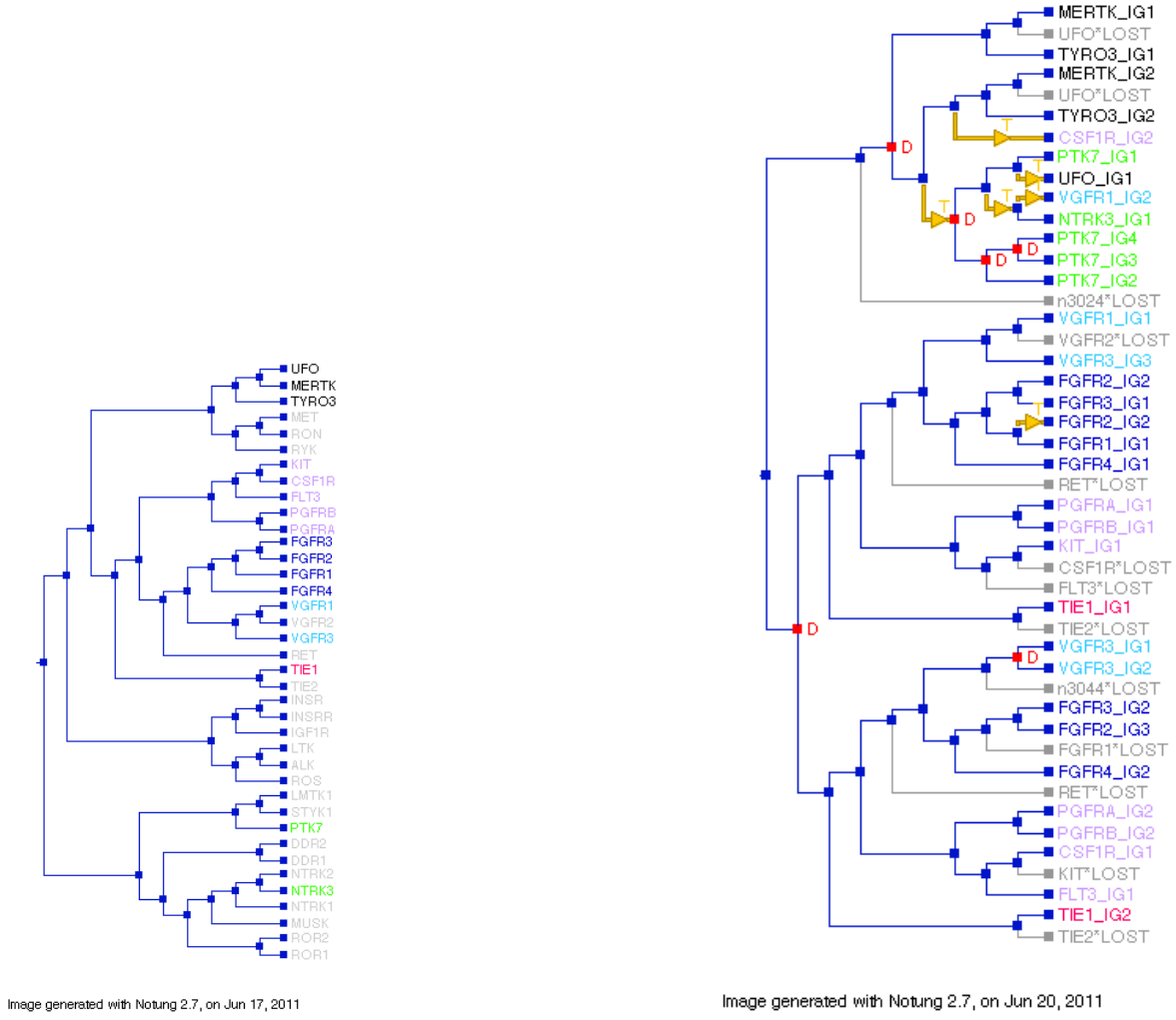


(a) (b) (c)

Figure 5.4: Trees constructed by Nars et al. [27]. All trees were rooted by using an outgroup sequence. **(a)** The kinase domain tree. Gray leaf nodes represent those genes without an SH2 or SH3 domain. Other leaves are colored according to subfamily membership. **(b)** Tree for the SH2 domains in all PTKs, reconciled with the kinase tree in (a). **(c)** Tree for the SH3 domains in all PTKs, reconciled with the kinase tree in (a).

Also, one copy of the duplicated domain in the growth factor receptors was lost in CSF1R, Kit, Flt3, VGFR1, RET, TIE2, and the ancestor of VGFR2 and VGFR1. The first I-set domain in VGFR3 was further duplicated. There were also a number of more contemporary domain insertions. The second I-set in CSF1R was the result of a domain insertion from an ancestral I-set in the Axl subfamily. The second I-set in VGFR1 resulted from a domain insertion from the I-set domain in NTRK3, which, in turn, was the result of a domain insertion from the first I-set in PTK7. The second I-set in FGFR2 was the result of a domain insertion from the I-set in FGFR1. Finally, the I-set domain in UFO was the result of an insertion from the first I-set in PTK7.

Fig. 5.6 shows a schematic of this reconstruction of the history of the I-set and kinase domains



(a) (b)

Figure 5.5: Trees I constructed for the PTKs. Gray leaf nodes represent those genes without an I-set domain. Other leaves are colored according to subfamily membership. **(a)** Tree for the kinase domain, treated as reference. This tree was rooted using information from the literature. **(b)** The tree for all I-set domains in PTKs, reconciled with the kinase domain in (a). This tree was rooted using event parsimony

in the context of the gene family history. This reconstruction shows that analyzing both domain architecture and sequence information (as represented by the trees) may imply much more domain shuffling activity than inferred from architecture alone. In the latter case, a number of the more recent insertions in my reconstruction would not be inferred. In addition, the prediction that the second I-set domain in the Axl subfamily is not orthologous to the second I-set domain in the other growth factors, would not be possible. For this case with the Ig I-set domain, by adding sequence information and my new method, the history of this family is more understandable.

5.3. Membrane-associated Guanylate Kinases (MaGuKs)

As observed in the history of I-set domains in the PTKs, not all domain repeats are the result of a single ancestral domain duplication. Rather, similar patterns of tandem arrays of I-set domains arose independently several times during evolution. This presented the question of whether this observation is unique to the Ig domain or to the PTK family. As seen in Fig. 5.7, many members of the membrane-associated guanylate kinase (MaGuKs) family have the same number of PDZs in a tandem array (e.g., the three PDZs in all members of the DLG (DLG 1-4) and Zo subfamilies). It was, thus, also of interest to investigate whether the PDZ repeats were inherited from an ancestral gene, or whether there was more domain shuffling activity in this family.

Until recently, the MaGuKs were thought to be uniquely metazoan [30, 294], but have since been shown to exist in the premetazoan lineages of the protist *Capsaspora owczarzaki* [295] and the choanoflagellate *Monosiga brevicollis* [296]. The members of the MaGuK family act as scaffolding proteins in various types of intercellular junctions, exhibiting a broad range of specific functions within this general category. They are involved in many cell-cell communication and signal transduction functions (see, for example [297, and work cited therein]). The MaGuKs originally evolved from catalytically active guanylate kinases, which are responsible for transferring a phosphate group from ATP to a guanosine monophosphate (GMP) residue. Since this divergence, however, they have lost the GMP binding and phosphorylation capabilities [297]. All MaGuKs contain a single guanylate kinase (GuK) domain, one or more PDZ domains, and either an SH3 domain or WW motifs, both domains associated with protein-protein interactions. MaGuKs exhibit considerable diversity in domain content, including both variation in the number of PDZ copies and the presence of additional, auxiliary domains.

MaGuK architectures are characterized by a common pattern with variations typical of many multi-

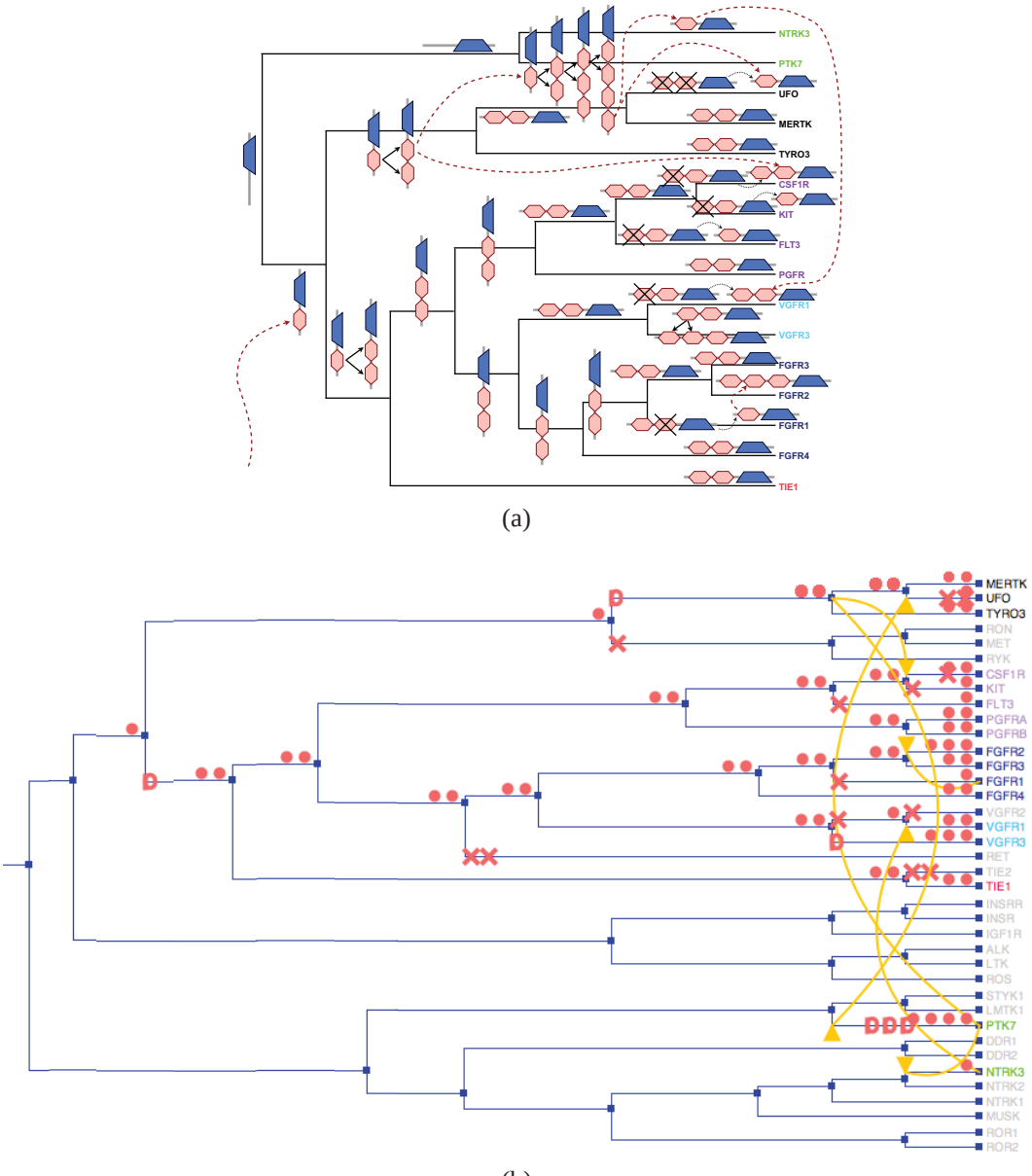


Figure 5.6: Schematic history of the gene family showing domain shuffling events in the history of the I-set domain. **(a)** The schematic developed by inspection of the reconciled I-set domain tree. **(b)** The schematic as inferred and drawn automatically in NOTUNG.

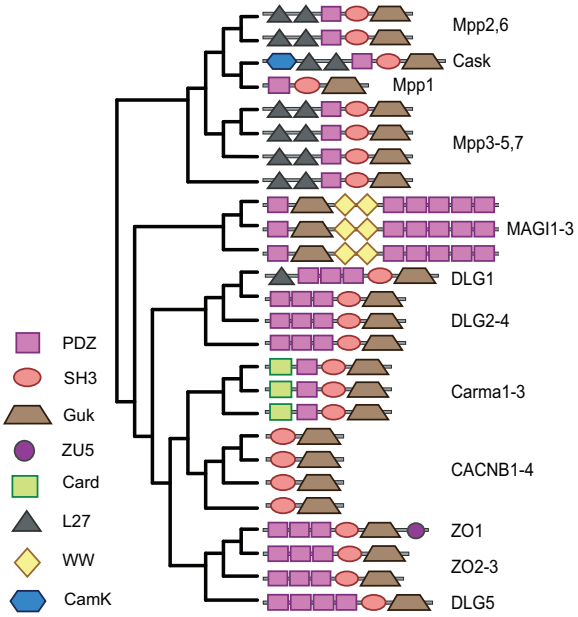


Figure 5.7: The MaGuK gene family as represented by the tree for the GuK domain. Leaves are decorated with domain architectures.

domain families. What forces drive this interplay between commonality and variation? Were these architectures the result of an ancestral domain architecture that persisted, like the SH3-SH2-PTK architectures in protein tyrosine kinases, or was more domain shuffling involved in the formation of this family, as with the I-set domains in PTKs?

The co-evolutionary history of these domains was first investigated by te Velthuis, et al. [30], who built trees for individual domains and compared them through inspection. Their trees were constructed by aligning sequences in ClustalX [298] and then building Bayesian trees with MrBayes [44] and maximum likelihood trees with PhyML [299]. The authors concluded that the PDZ-SH3-GuK architecture evolved only once; their conclusions contained no inferences of any domain insertions beyond the ancestral ones. They did infer that the proliferation of PDZs in the MAGI subfamily was due to domain duplication in that subfamily and that the proliferation of PDZs in the DLG and ZO subfamilies was the result of domain duplications in their common ancestor. However, the trees they constructed and evaluated included many non-binary nodes. Thus, much information was lost, as these nodes provide no information on history, and their analysis was basically the domain gain-loss approach based on the GuK tree. In order to further investigate this family and its evolutionary history, I built trees for the GuK, SH3, and PDZ domains in human and compared them using NOTUNG to infer the domain shuffling events. Sequences for all GuK, SH3, and PDZ domain instances in human MaGuKs were obtained from [30]. The GuK

tree was rooted using the functional GuK domain as an outgroup [30]. The roots of the SH3 and PDZ trees were inferred using event parsimony with NOTUNG. For this family, the GuK domain was treated as the reference. Several lines of evidence support the use of the GuK domain tree as a proxy for the locus history. First, it is thought that the ancestral MaGuK protein had a GuK domain (and possibly other domains as well), which was inherited by vertical descent by all contemporary MaGuKs. The varied MaGuK architectures subsequently arose through insertion, deletion, and duplication in the descendants of this progenitor. This assumption is supported by the observation that the GuK domain only appears in a restricted set of architectures, compared with the other, more promiscuous, domains in the family. Further, the phylogenetic distribution of functional GuK domains is broader than the catalytically inactive GuK, suggesting that the progenitor GuK resulted from an ancestral GuK domain through a single, ancient loss-of-function event [30, 297]. Additional evidence derives from near perfect phylogenetic congruence between the SH3 and GuK domains as discussed below.

Applying the domain gain-loss approach (see Fig. 5.8) suggests that the ancestral MaGuK had a single SH3, which was lost in the MAGI subfamily, and a single PDZ. In addition, it would infer that PDZs expanded by two domains in the the common ancestor of the MAGI/DLG/ZO/Carma/CACNB subfamilies and again by three domains in the ancestor of the MAGI subfamily and by one domain in DLG5. These PDZs would have then evolved through vertical descent, without further shuffling, with the exception of the Carma/CACNB ancestor, which lost two PDZs and the CACNB subfamily, which lost the third PDZ. Note, however, that this inference process ignores sequence variation in the SH3 and PDZ domains.

Using my method to compare the SH3 and GuK trees revealed that, for the most part, they have similar histories, as shown in Fig. 5.9. My analysis suggests that it is likely that the ancestor of all MaGuKs had a single SH3 domain, which was lost in the MAGI clade (MAGI 1-3). In addition, in contrast to the analyses of te Velthuis et al., my analysis also suggests that the contemporary SH3 domain in the Carma subfamily (Carma 1-3) may be the result of a domain insertion from the ancestor of the ZO subfamily (ZO 1-3 and DLG5) and loss of the ancestral SH3 domain, rather than a result of vertical descent.

The comparison of the GuK tree with the tree for PDZ, Fig. 5.9, also reveals that more domain shuffling may have occurred in the history of this family than previously assumed. Reconciliation reveals repeated internal domain duplications, losses, and insertions between genes, suggesting recent domain swapping between the ZO, DLG, and Carma subfamilies. My analysis suggests that the ancestor of all MaGuKs had a single PDZ domain; this domain evolved through vertical

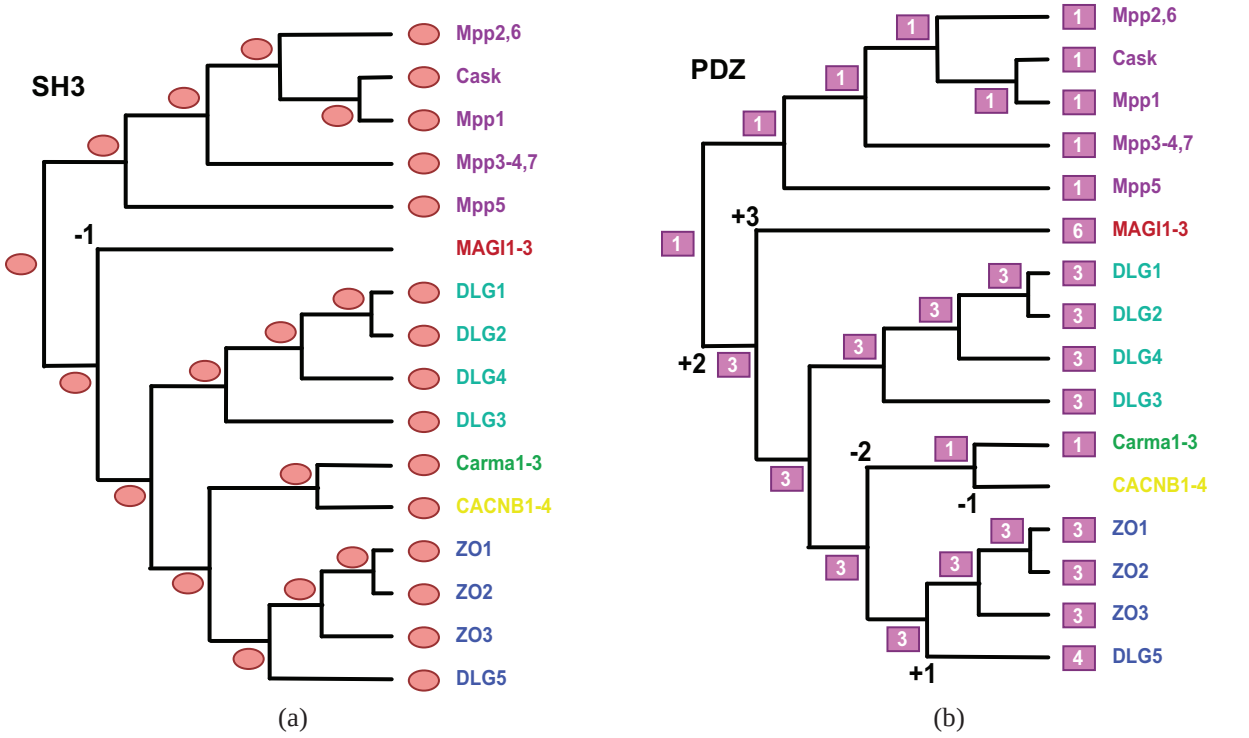
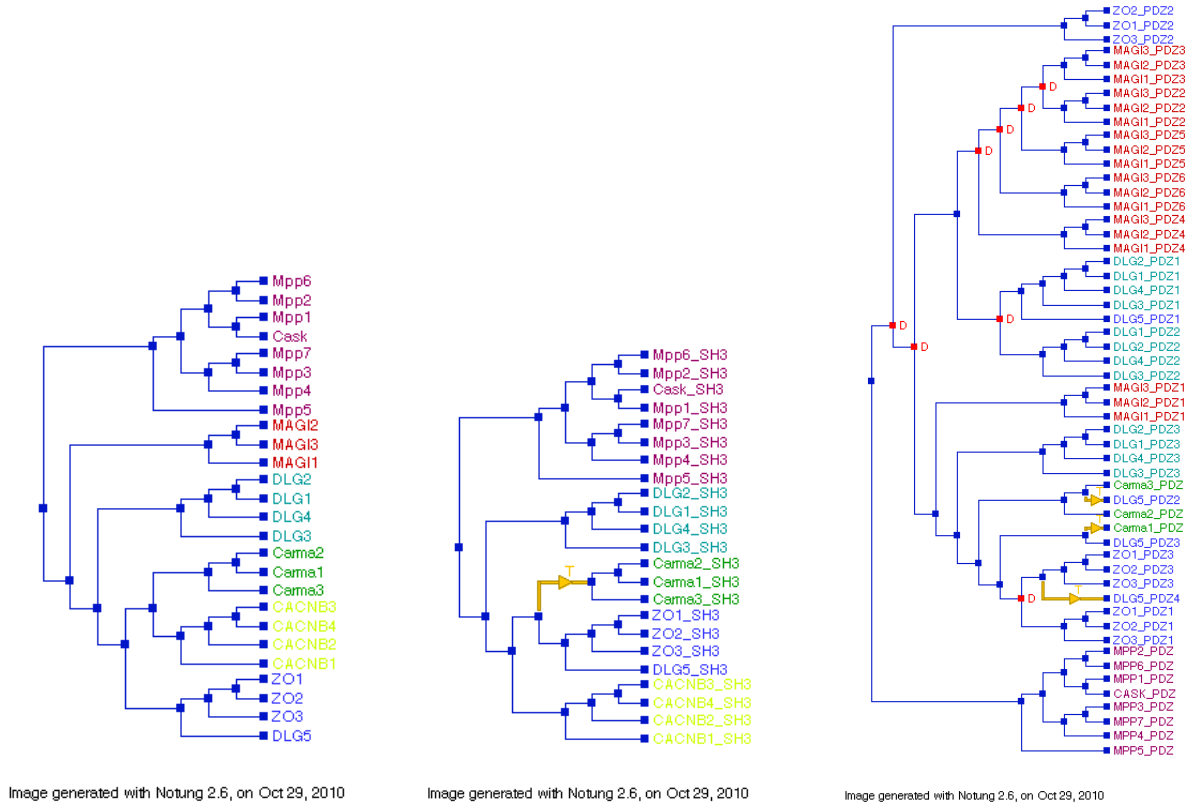


Figure 5.8: History of the **(a)** SH3 and **(b)** PDZ domains in MaGuK as inferred using the domain gain-loss approach. Clades that have the same domain architectures on the leaves have been collapsed to a single node (i.e., MAGI 1-3).

descent to form the present copy of PDZ in all members of the Mpp clade (Mpp 1-7 and Cask). This ancestral domain was duplicated twice more in the ancestor of all MaGuKs except the Mpp's. In the MAGI subfamily, one of these copies was lost, while another was duplicated another four times, resulting in the multiple repeats seen in the MAGI. A different domain duplication occurred in the ancestor of DLGs, ZOs, Carma's, and CACNBs (CACNB1-4), followed by differential losses. As with the analysis of the I-set domains in PTKs, contemporary domain insertions were also observed: the second PDZ in DLG5 is the result of a domain insertion from Carma3; the fourth PDZ in DLG5 is the result of a domain insertion from the third PDZ in the ZO subfamily; and the PDZ in Carma1 is the result of a domain insertion from the third PDZ in DLG5.

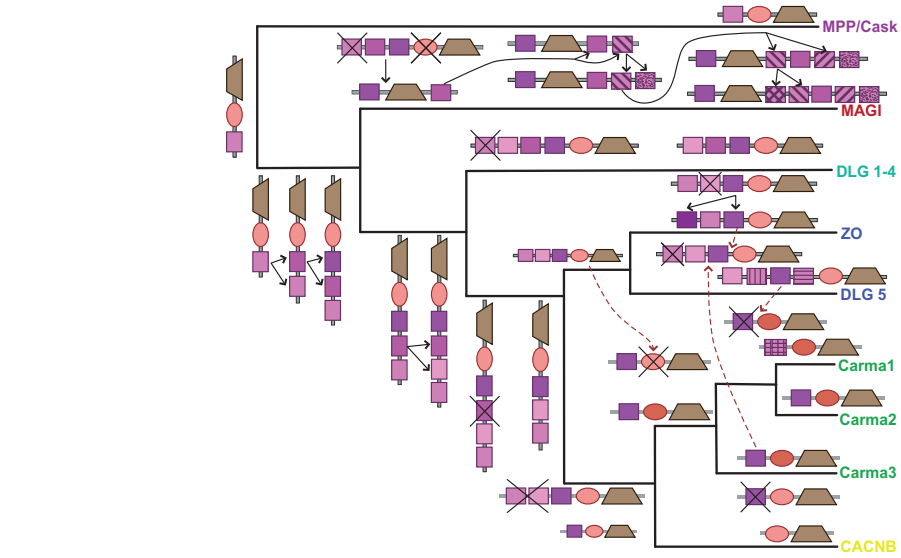
Fig. 5.10 shows a schematic, based on the NOTUNG output, conveying the combined histories of the PDZ, SH3, and GuK domains. This figure reveals that many of the inferences based on the domain gain-loss approach are not supported by the sequence data: *(1)* the common ancestor of the DLG, ZO, Carma, and CACNB subfamilies had four, not three, PDZ domains; *(2)* a cassette of three PDZs was not transmitted to the DLGs and ZOs by vertical descent - rather, a series of



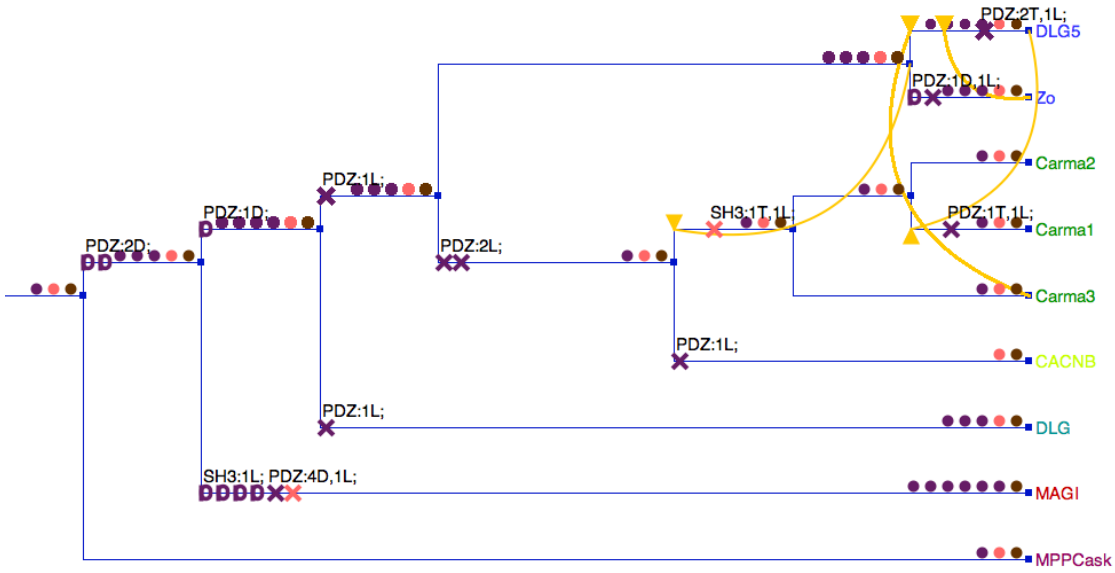
(a) (b) (c)

Figure 5.9: Trees I constructed for the MaGuK gene family. Leaves are colored according to subfamily membership. **(a)** Tree for the GuK domain, treated as reference. This tree was rooted using active guanylate kinase domains from outside the family. **(b)** The tree for all SH3 domains in MaGuKs, reconciled with the GuK domain in (a). **(c)** The tree for all PDZ domains in MaGuKs, reconciled with the GuK domain in (a). Trees in (b) and (c) were rooted using event parsimony

duplications, losses and insertions gave rise to the same domain architecture in all the subfamily members; and *(3)* PDZs in the same position in the architecture of different genes are not more closely related to each other than to other PDZs. This suggests that analyses based on domain architectures alone may underestimate the extent of domain shuffling and convergent evolution of domain architectures that is occurring and underscores the value of methods that include sequence comparison. It further suggests that estimates of domain shuffling rates using such methods may give very different results than from previous studies and that new genomic analyses using these approaches are imperative.



(a)



(b)

Figure 5.10: Schematic history of the gene family showing domain shuffling events in the history of the GuK, SH3, and PDZ domains. Monophyletic clades with no events were collapsed to a single leaf node. **(a)** The schematic developed by inspection of the reconciled domain trees. **(b)** The schematic as inferred and drawn automatically in NOTUNG.

5.4. Event cost parameter selection

In the methods described here, a linear combination of the number of duplications, insertions, and losses is used to score candidate event histories by their total, weighted event cost, referred to as the *DTL-score* (i.e., DTL-Score $= \delta * \text{num}_{\text{dup}} + \tau * \text{num}_{\text{ins}} + \lambda * \text{num}_{\text{loss}}$). Different event cost parameters could result in different optimal event histories. For example, when the combined cost of one duplication and one loss is less than the cost of an insertion ($\delta + \lambda < \tau$), the optimal history for the family in Fig. 1.2a is a duplication of the red circle domain followed by a loss (as in Fig. 3.6). This solution has a DTL-score of $\delta + \lambda$. However, if $\tau > \delta + \lambda$, the optimal event history for the same trees is an insertion of domain $r1$ from g_2 into g_1, resulting in a DTL-score of τ.

How should cost parameters be selected in order to obtain the most accurate inference? For case studies, evidence such as synteny, LTR repeats, transposon integration sites, and intron loss can be used to guide selection among several candidate event histories. For large scale analyses, maximum likelihood estimation can be used to estimate parameters for a parsimony model if the probability of an event on a given branch is small [24]. For example, rates can be estimated by modeling domain shuffling as a birth death process, as in my Master's thesis work [300] or similar to Karev et al. [147, 159]. Rates can then be converted into event cost parameters by taking the negative natural log of the rate (i.e., $-\ln r_i$ for rate r_i of event i) [24].

In all the above analyses, parameter values were selected such that the duplication cost was more than the loss cost, and the insertion cost (τ) was more than a loss plus a duplication (i.e, $\lambda < \delta$ and $\delta + \lambda < \tau$). Here, I demonstrate that for in-depth studies, the space of all reconciliations can be explored empirically, and that optimal solutions for different parameter functions correspond to contiguous, distinct regions of the cost-parameter space. For this study, I reconciled the SH3 and PDZ domain trees with the GuK domain tree using various duplication, insertion, and loss parameter values between 1 and 100. Specifically, I calculated the DTL-score for all parameter value triples $(\delta, \tau, \lambda) \in F^3$, where $F = (1, 5, 10, 15 \ldots 100)$.

For the SH3 reconciliation, there were only three different optimal solutions, summarized in Table 5.1. In the space of all parameter values, shown in Fig. 5.11, these three different reconciliations occupy distinct regions that overlap only at the boundaries. Note that in order to display the parameter space in two dimensions, the index x and y variables are defined as the ratios τ/λ and δ/λ, respectively. Based on the number of events associated with the different reconciliations, the following inequalities define the borders of the regions in the parameter space, corresponding to

All reconciliations of SH3 in MaGuK

Reconciliation	Duplications	Insertions	Losses	Region equation
(1)	0	1	2	$\lambda \leq \tau \leq \delta + 2\lambda$
(2)	1	0	4	$\delta + 2\lambda \leq \tau$
(3)	0	3	0	$\tau \leq \lambda$

Table 5.1: Summary of the three different reconciliations possible depending on the parameter values used. Each region, seen in Fig. 5.11 is defined by the number of different events. The final column provides the boundary equations, in terms of duplication cost δ, insertion cost, τ, and loss cost λ, defining which solution will be reached depending on parameter values.

the distribution of reconciliations.

$$\begin{aligned} \text{Reconciliation (1)} \qquad & \tau + 2\lambda \leq 3\tau \qquad \tau + 2\lambda \leq 4\lambda + \delta \\ & \lambda \leq \tau \qquad \tau \leq 2\lambda + \delta \\ & \therefore\ \lambda \leq \tau \leq 2\lambda + \delta \end{aligned} \tag{5.1}$$

$$\begin{aligned} \text{Reconciliation (2)} \qquad & \delta + 4\lambda \leq \tau + 2\lambda \qquad \delta + 4\lambda \leq 3\tau \\ & \delta + 2\lambda \leq \tau \qquad \frac{1}{3}\delta + \frac{4}{3}\lambda \leq \tau \\ & \therefore\ \delta + 2\lambda \leq \tau \qquad \text{since } \frac{1}{3}\delta < \delta \text{ and } \frac{4}{3}\lambda < 2\lambda \end{aligned} \tag{5.2}$$

$$\begin{aligned} \text{Reconciliation (3)} \qquad & 3\tau \leq 2\lambda + \tau \qquad 3\tau \leq 4\lambda + \delta \\ & \tau \leq \lambda \qquad \tau \leq \frac{4}{3}\lambda + \frac{1}{3}\delta \\ & \therefore\ \tau \leq \lambda \qquad \text{since } \lambda < \frac{4}{3}\lambda + \frac{1}{3}\delta \end{aligned} \tag{5.3}$$

With the reconciliation of the PDZ and GuK trees, there were many more (16) different optimal solutions, as described in Table 5.2. As with SH3, the different optimal reconciliations all occupied distinct regions only overlapping at the boundaries of these regions. Equations for these boundaries can similarly be defined in terms of parameters. The partition of the parameter space and the associated reconciliations can be pre-computed. Then, the reconciliation for any choice of parameter values can be retrieved from this pre-computed set, by determining which region is associated with those parameter values.

The number of optimal event histories depends on the size of the trees and the extent of topological incongruence As these results show, for small problem instances there are only a few optimal

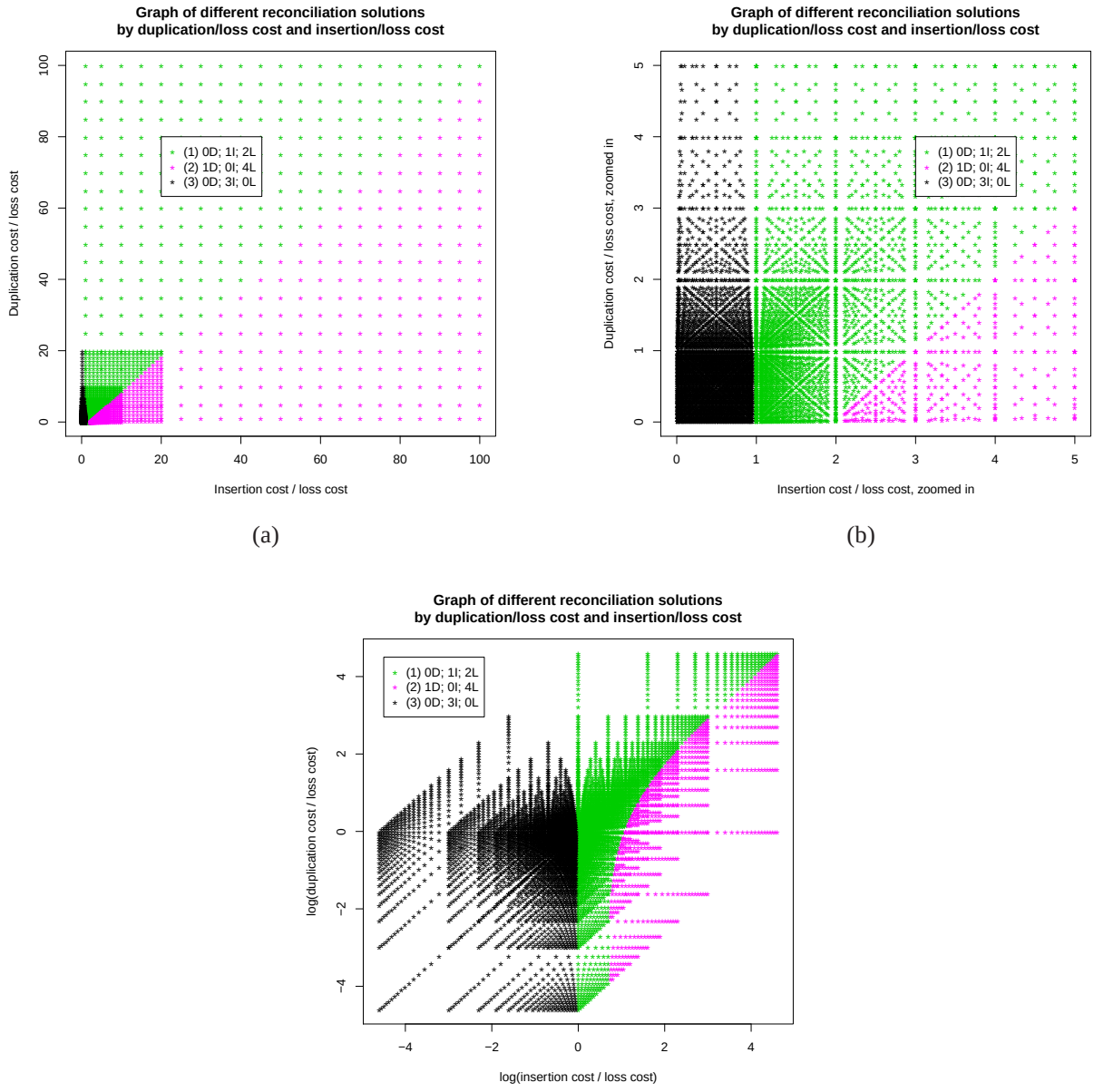


Figure 5.11: Parameter analysis of the SH3 domain tree reconciled with the GuK domain tree, as shown in the parameter space of δ/λ and τ/λ. Points represent a single reconciliation under one specified parameter triplet $\{\delta, \tau, \lambda\}$. Colors represent the inferred number of events as shown in the legend. **(a)** Reconciliations for all examined parameter triplets. **(b)** The plot in (a) zoomed in to show detail. **(c)** The data from (a) plotted according to a log on both axes.

All reconciliations of PDZ in MaGuK

Reconciliation	Duplications	Insertions	Losses	Region
(1)	0	20	0	$2\tau \leq \delta$ $\tau \leq \lambda$
(2)	0	18	2	$2\lambda \leq 2\tau \leq 5\lambda$ $2\tau \leq \delta$
(3)	0	16	7	$5\lambda \leq 2\tau \leq \delta$
(4)	4	8	7	$3\delta \leq 6\tau \leq 4\delta + 2\lambda$ $5\lambda \leq 2\tau$
(5)	6	5	8	$2\delta + \lambda \leq 3\tau \leq 3\delta$ $2\delta + 6\lambda \leq 5\tau$
(6)	11	0	20	$2\delta + 9\lambda \leq 2\tau$
(7)	9	2	11	$2\delta + 4\lambda \leq 2\tau \leq 2\delta + 9\lambda$
(8)	8	3	9	$\delta + \lambda \leq \tau \leq \delta + 2\lambda$ $2\lambda \leq \tau$
(9)	7	4	8	$5\lambda \leq 3\tau$ $2\delta + 6\lambda \leq 5\tau$ $\lambda \leq \delta \leq \tau \leq \delta + \lambda$
(10)	4	10	2	$\lambda \leq \tau \leq \delta \leq 2\tau$ $5\tau \leq 2\delta + 6\lambda$ $\lambda \leq \delta \leq \tau \leq \delta + \lambda$
(11)	4	12	0	$\delta \leq 2\tau \leq 4\delta$ $3\tau \leq \delta + 2\lambda$ $\tau \leq \lambda$
(12)	5	9	2	$\delta \leq \tau \leq 2\delta$ $2\delta + 4\lambda \leq 6\tau \leq 6\delta + 3\lambda$ $5\tau \leq 2\delta + 6\lambda$
(13)	7	7	3	$3\lambda \leq 3\tau \leq 5\lambda$ $6\delta + 3\lambda \leq 6\tau \leq 2\delta + 8\lambda$
(14)	8	4	7	$\delta + 4\lambda \leq 3\tau \leq 6\lambda$ $\delta \leq \lambda$
(15)	7	8	2	$2\delta \leq \tau \leq \lambda$ $\delta + 2\lambda \leq 3\tau$
(16)	6	11	0	$6\delta \leq 3\tau \leq \delta + 2\lambda$

Table 5.2: Summary of the 16 different reconciliations possible depending on the parameter values used. Each region, seen in Fig. 5.12 is defined by the number of different events. The final column provides the boundary equations, in terms of duplication cost δ, insertion cost, τ, and loss cost λ, defining which solution will be reached depending on the parameters.

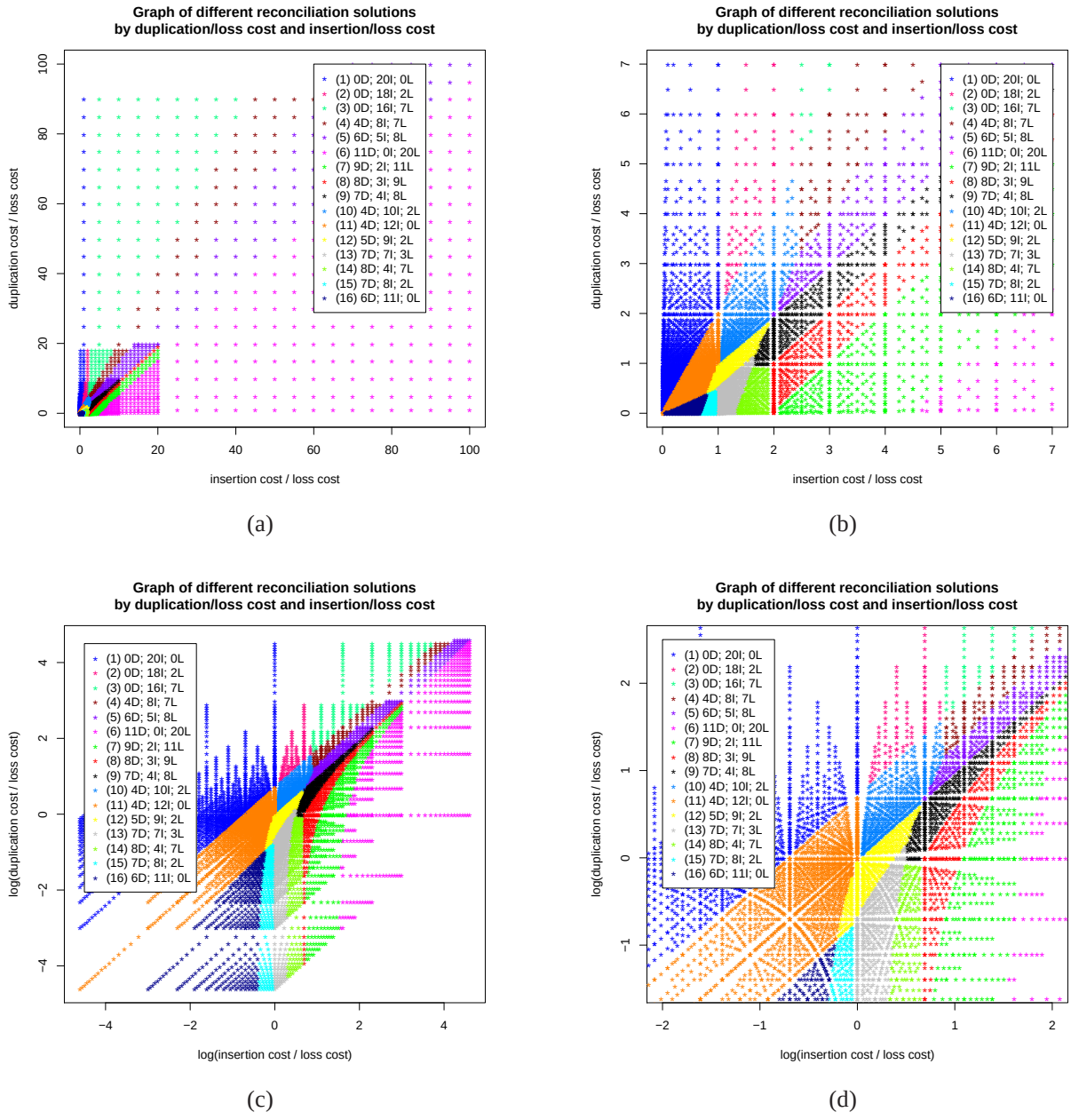


Figure 5.12: Parameter analysis of the PDZ domain tree reconciled with the GuK domain tree, as shown in the parameter space of δ/λ and τ/λ. Points represent a single reconciliation under one specified parameter triplet $\{\delta, \tau, \lambda\}$. Colors represent the inferred number of events as shown in the legend. **(a)** Reconciliations for all examined parameter triplets. **(b)** The plot in (a) zoomed in to show detail. **(c)** The data from (a) plotted according to a log on both axes. **(c)** The plot in (c) zoomed in to show detail.

histories. Further, each history corresponds to a contiguous region of the parameter space that overlaps with other regions only at well-defined, limited boundaries. Thus, for in-depth analyses of multi-domain families, all optimal solutions for different parameter-space regions can be explored.

5.5. Significance of results

The studies and results provided here have demonstrated how my methods and associated software can be used to infer the history of domain shuffling events of a multidomain family. In addition, I have demonstrated that the analysis of such families using a tree from only a single domain and inferring ancestral states based on the domain gain-loss approach alone may underestimate the amount of shuffling. In particular, a number of recent studies have investigated patterns of domain shuffling using such approaches based on domain architecture alone. The sequences of individual instances of a particular domain were not compared. The authors of these studies [134, 140, 142] concluded that a specific domain architecture usually arose only once in an ancient ancestor was inherited by vertical descent by all members of the family. However, my analyses that take sequence information to account using tree reconstruction and reconciliation, shown in Figs. 5.6 and 5.10, suggest that, for the studied families, the conclusions of these architecture-based studies do not hold.

Major insights can already be observed from a thorough study of the families presented here. First, it appears that cassettes containing the same number of domain copies may have evolved independently in different family members. For example, the three PDZ cassettes in the Zo1-3 clade were the result of ancestral domain duplication, domain loss, and local domain duplication. Similarly, while the SH3 domain in most MaGuK subfamilies was inherited from an ancestral instance by vertical descent, The single SH3 in the Carma subfamily results from replacement of the ancestral SH3 domain by a lateral insertion from the ancestral ZO. If sequence information were not taken into account, the above inferences would be missed because the conserved domain content would be thought to result from a single set of ancestral events, under the gain-loss parsimony assumption. My work, based on tree comparison, indicates that much more duplication, loss, and insertion occurred than a parsimony analysis of domain architectures alone would suggest. Of particular note, including sequence evolution can suggest domain insertions not visible to domain architecture methods. In addition, when considering only architectures, domains in the same position in different genes would appear to be recent homologs; however, my work shows this may not be the case always. For example, the second I-set domain in FGFR3 and FGFR4 in the PTKs

are not closely related to the second I-set domain in FGFR2; rather, they are recent homologs to the *third* I-set domain in FGFR2. An analysis through inspection, however, would suggest that all domain copies in position 1 are recent homologs, and so on.

Note that we often see a domain insertion with domain loss resulting in the replacement of the domain in one architecture with a domain from another architecture. One hypothesis for this observation, alternative to domain shuffling, is the occurrence of gene conversion. Domains that have undergone gene conversion will appear to be more closely related than they actually are. A number of studies have investigated the propensity for gene conversion among duplicated *genes* as a function of properties such as sequence similarity and spatial proximity [301–310]. However, I am unaware of a systematic evaluation of gene conversion among domains, although specific instances have been reported [311, for example]. My methods could be used to identify possible candidates of gene conversion, which could then be further analyzed using more detailed information about the gene (i.e., with other methods of gene conversion detection, such as [312–314] or [311]).

All these results are exciting because my approach has the potential to overturn prior results based on the domain gain-loss approach, such as inferred relative rates of domain shuffling and the currently held assumption that convergent evolution of the same architectures is rare [139, 142]. If the gain-loss approach underestimated the number of events, then previously inferred relative rates of events are likely inaccurate. In addition, it appears that the same architectures, especially when including repeated domains, may have evolved more than once. This suggests that my method has the potential to yield much greater insight into the process of domain shuffling than current methods by explicitly inferring domain shuffling events through phylogenetic methods, rather than considering static properties.

Chapter 6

Co-evolution of co-occurring domains in the human genome

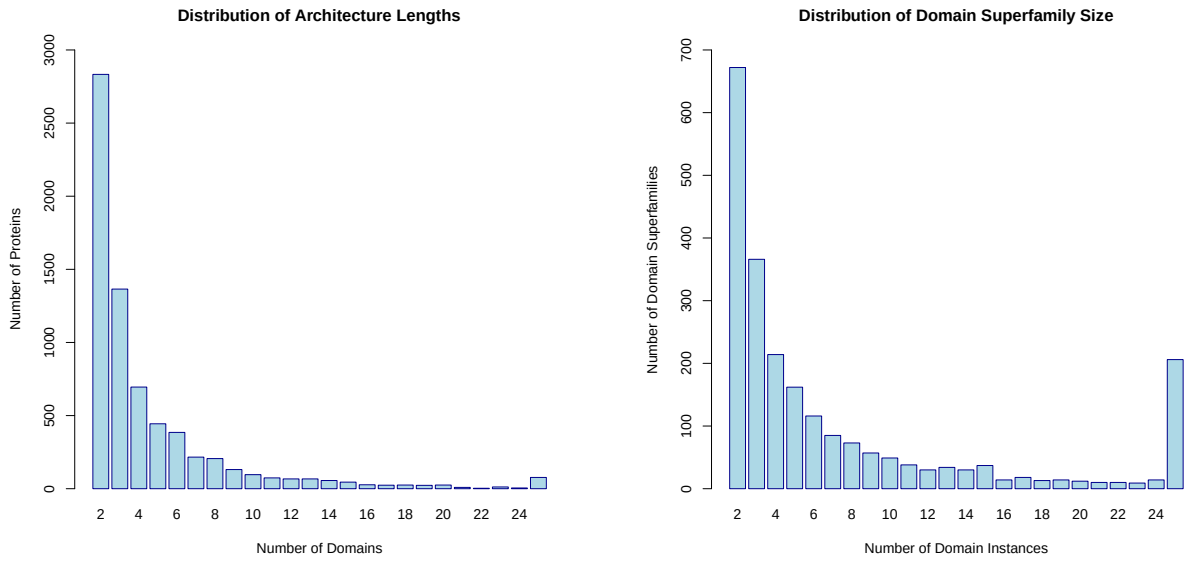
In order to demonstrate the utility of my methods, I carried out an analysis of all pairs of comparable domain superfamilies in the human genome. Domain shuffling activity between pairs of co-occurring domains can be assessed by reconciling their respective domain trees. Pair-wise domain tree reconciliation provides information about the relative shuffling propensity of various domain superfamilies and does not require family predictions or gene family tree inference.

The computational pipeline for this study is summarized in Figs. 6.2a and 6.2b. Amino acid sequences for all genes in the human genome were obtained from the Panther 7.0 database [315] (`http://www.patherdb.org`). Domain architectures and domain boundaries for instances of each domain superfamily were identified by scanning the genomic sequences with the set of identifying HMMs from the Pfam database [61, 62, 316]. Each domain instance is referenced by: *(1)* its domain superfamily identifier; *(2)* the identifier for the gene in which it occurred; and *(3)* the domain position in that gene (i.e., a number i indicating the domain was the ith domain in the domain architecture). For example, Mpp2_PDZ3 identifies the PDZ domain as the third domain in the architecture of protein Mpp2. Of the roughly 20,000 genes in the human genome, 15,733 had at least one PFAM domain identified. Of these, 6,562 (just over 40%) contained at least two domains, representing more than 2,300 different domain architectures. These architectures range in size from two domain instances to 330 domain instances (see Fig. 6.1a). Table 6.1 describes the 11 largest architectures in terms of the number of domain instances in the architecture.

Longest Multidomain Architectures in Human

Protein name	Uniprot ID	Length
Titin	Q8WZ42	330
Nebulin	P20929	145
Low-density lipoprotein receptor-related protein 2	P98164	75
Low-density lipoprotein receptor-related protein 1B	Q9NZR2	74
Prolow-density lipoprotein receptor-related protein 1	Q07954	73
Obscurin	Q5VST9	62
Hemicentin-1	Q96RW7	59
Mucin-16	Q8WXI7	55
Fibrillin-1	P35555	52
Fibrillin-2	P35556	52
Fibrillin-3	Q75N90	52

Table 6.1: The 11 longest domain architectures, in number of domain instances, of multidomain proteins in human.



(a) (b)

Figure 6.1: **(a)** The distribution of architecture lengths for all multidomain proteins in the human genome. **(b)** Distribution of the sizes of domain superfamilies, by the number of domain instances identified in the human genome. In both histograms, the final column represents all instances greater than 25. A list of the longest architectures and largest families can be found in Tables 6.1 and 6.2, respectively.

Largest Domain Superfamilies in Human

Domain Superfamily	Pfam name	Pfam ID	Size
Zinc finger, C2H2 type	zf-C2H2	PF00096	6,111
Ankyrin repeat	Ank	PF00023	1,308
WD40, G-beta repeat	WD40	PF00400	959
Leucine-rich repeat	LRR_1	PF00560	844
Fibronectin type III	fn3	PF00041	705
7 transmembrane receptor (rhodopsin family)	7tm_1	PF00001	670
Cadherin	Cadherin	PF00028	619
Collagen triple helix	Collagen	PF01391	584
Immunoglobulin I-set	I-set	PF07679	541
EGF-like	EGF	PF00008	463
Immunoglobulin	ig	PF00047	420
Tetratricopeptide	TPR_1	PF00515	405
Protein kinase	Pkinase	PF00069	378
Calcium-binding EGF	EGF_CA	PF07645	344
RNA recognition motif	RRM_1	PF00076	319
Sushi	Sushi	PF00084	317
Kruppel associated box	KRAB	PF01352	311
Kelch motif	Kelch_1	PF01344	296
Spectrin repeat	Spectrin	PF00435	290
Pleckstrin homology	PH	PF00169	273

Table 6.2: The 20 largest domain superfamilies in human, as defined by the number of domain instances identified in the human genome.

Once the entire genome was scanned and domain boundaries were identified, the amino acid sequence of every domain instance was extracted from the genomic data. In this dataset, there were 3,891 different domain superfamilies ranging in size from one instance to over 6,000 instances. The distribution of superfamily sizes, by the number of domain instance found in the human genome can be seen in Fig. 6.1b. The largest families are shown in Table 6.2

These files were then processed following the tree construction pipeline, shown in Fig. 6.2a. MSAs were built for all domain superfamilies with three or more representatives in the data set using the MAFFT program [317]. Each alignment was then trimmed to remove poorly aligned and uninformative columns using the TrimAl program [318]. The trimmed MSA was then used to reconstruct a domain tree for each superfamily using the Neighbor-Joining (distance-based) method provided in the Phylip package [288] and bootstrapped 100 times.

Once trees for each superfamily were constructed, reconciliation was performed following the

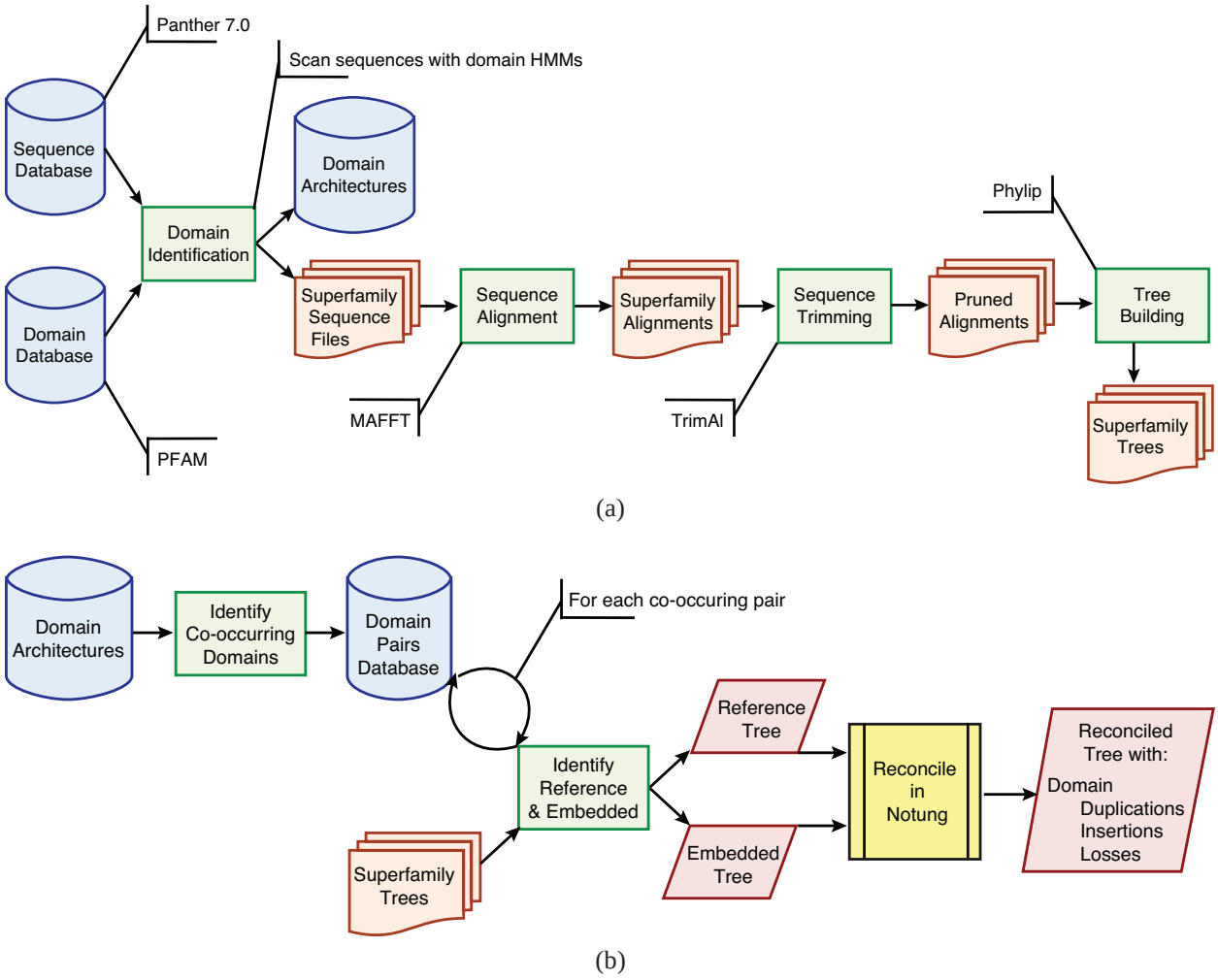


Figure 6.2: **(a)** Tree Construction Pipeline. First, domains are identified from the amino acid sequences and the domain model database, providing domain architectures and domain instance boundaries, which are used to extract domain sequences. A multiple sequence alignment is constructed for each domain superfamily and then trimmed to remove poorly aligned regions. Finally, a domain tree is constructed for each domain superfamily. **(b)** Reconciliation Pipeline. Domain architectures are analyzed to produce a list of all co-occurring domain pairs. Reconcilable domain pairs are identified, and the trees corresponding to these pairs are then reconciled in NOTUNG, providing data on domain shuffling events.

pipeline in Fig. 6.2b. I used the identified domain architectures to compile a list of co-occurring domain pairs. Trees for all comparable domain pairs were then reconciled. Given a candidate reference domain D_R and a candidate embedded domain D_E, D_R and D_E are *comparable* if

1. D_R and D_E co-occur in a protein more than once,
2. D_R appears exactly once in each protein in which they co-occur.

Recall that the reconciliation algorithm takes as input a rooted reference tree, a rooted embedded tree and a mapping between the leaves. In this case, the mapping is established by protein co-occurrence. Each embedded domain instance is mapped to the reference domain instance with which it co-occurs. The requirement that D_R occur only once in each protein ensures that this mapping is unique and unambiguous. Note that given a comparable pair of domains, D_1 and D_2, each of which occurs only once in every protein in which they co-occur (a one-to-one pair), either member of the pair can take the role of the reference domain. In this case, the corresponding domain trees are reconciled twice, once with $D_R = D_1$ and once with $D_R = D_2$. If one member of the pair appears more than once in one or more proteins (a many-to-one pair), then the pair can only be reconciled in one direction. If *both* domains appear more than once in a protein (a many-to-many pair), then the pair is not comparable and cannot be reconciled at all. For example, in Fig. 6.3a the blue domain co-occurs with the red domain in g_1 and g_2. Because the blue domain only appears once in each protein, the blue domain is D_R and the red domain is D_E, as in Fig. 6.3b. In this case, the domains are many-to-one and a mapping is easily established. However, we can not treat the red domain as D_R, with the blue domain as D_E. As seen in Fig. 6.3b, g_1_b maps to both g_1_r1 and g_1_r2, and the mapping of the parent of g_1_b cannot be determined.

Once all comparable pairs were identified and the reference and embedded roles assigned, these pairs were reconciled using NOTUNG's command-line functionality with the duplication, insertion, and loss algorithm. Of the 4,097 different co-occurring pairs, 2,914 domain pairs were reconciled (see Table 6.3). Of the 1893 comparisons in one direction, 1,287 inferred at least one event. This indicates that 606, or roughly 68%, of reconcilable domain pairs co-evolved by vertical descent with no additional shuffling once the initial pair was formed. Thus, more than half of comparable pairs either formed a pair at least twice in evolutionary history or sustained a duplication. Fig. 6.4 shows the distribution of the number of duplication and insertions events inferred in a reconciliation. Most reconciliations contained only a few events, implying that most domain pairs evolve largely through co-divergence. Some pairs, however, show a high amount of domain shuffling, such as the reconciliations of Nebulin with SH3_1 (143 duplications and 6 insertions) and Cadherin with Cadherin_2 (5 duplications and 137 insertions).

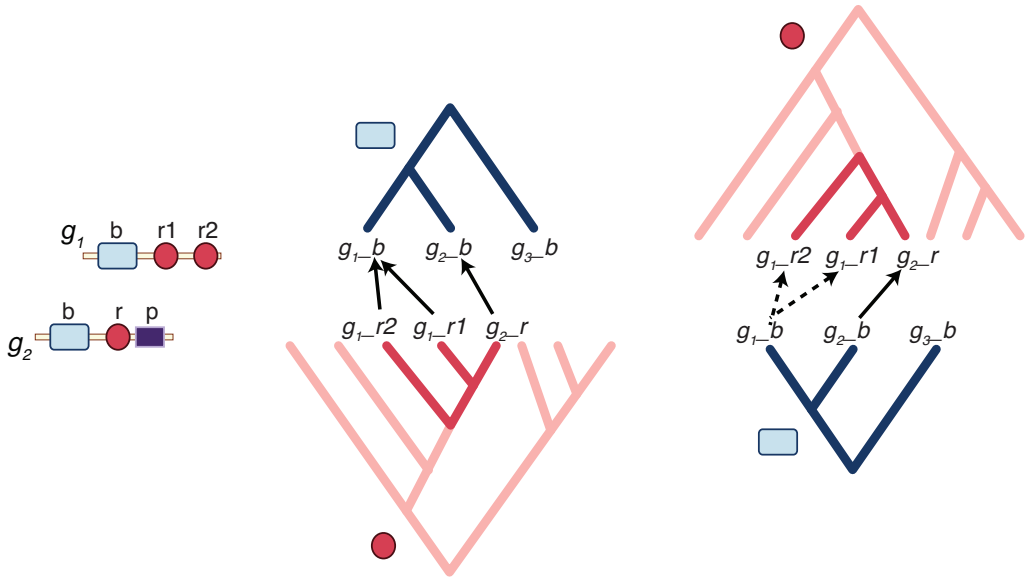


Figure 6.3: Explanation of *x*-to-*y* mapping, based on the example family in Fig. 1.2. **(a)** Domain architecture of the blue and red domain-containing proteins. **(b)** A many-to-one mapping of the leaves of the red domain tree to leaves of the blue domain tree, because more than one leaf in the red tree is mapped to the same leaf in the blue tree. **(c)** A one-to-many mapping of the leaves of the blue domain tree to leaves of the red domain tree, there exists a leaf in the blue tree that is mapped to more than one leaf in the red tree. Thus, when reconciling the red and blue domains, only the blue domain can act as the reference.

Summary of Domain Pairs in Human.

	Pairs	Reconciliations
Pairs of distinct domains	4,097	NA
Only one leaf in both trees	180	0
Many-to-many	2024	0
Many-to-one	872	872
One-to-one	1,021	2042
Total		2,914

Table 6.3: The number of different co-occurring pairs of domains in the human genome, broken down by their *x*-to-*y* relationships.

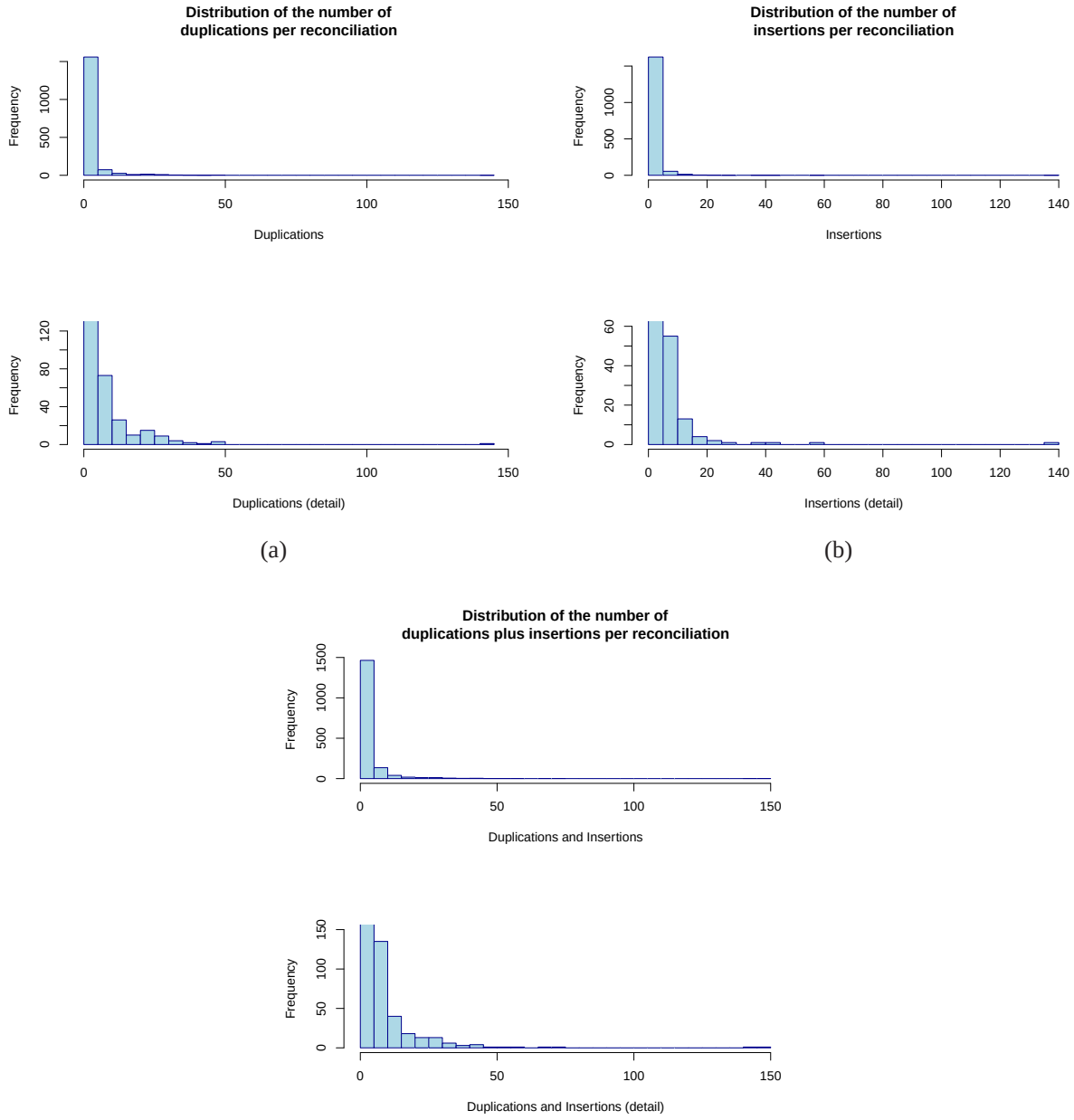


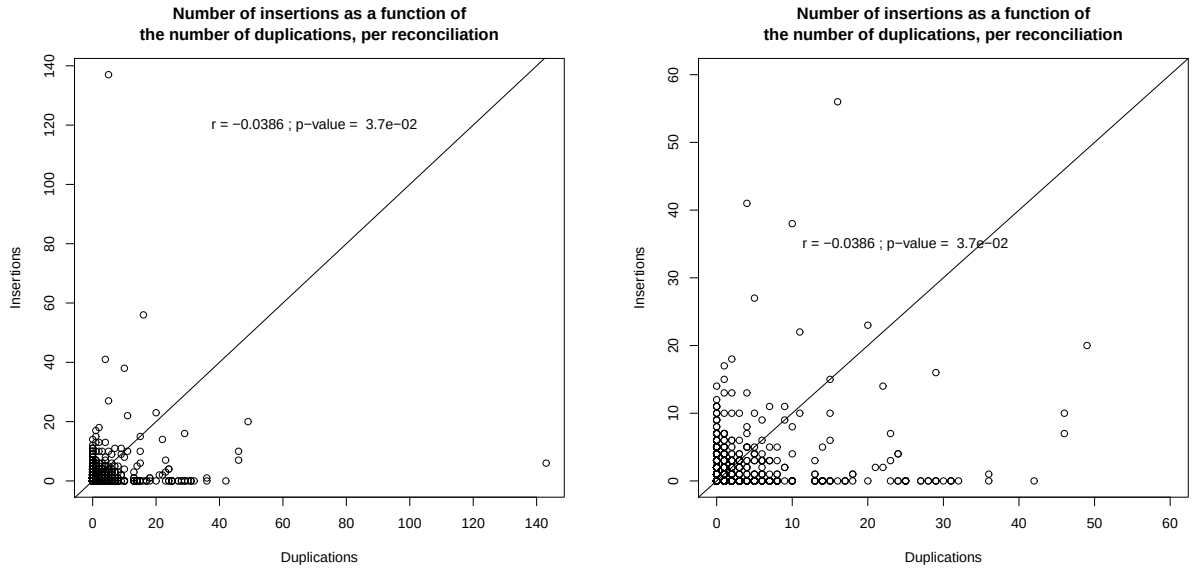
Figure 6.4: Distributions of the number of **(a)** duplications, **(b)** insertions, and **(c)** duplications *and* insertions for a single reconciliation. Reconciliations containing no events were not included.

Of the reconciliations resulting in at least one event, 576 contained only duplications and losses and 862 contained only insertions and losses; the remaining 265 contained both duplications and insertions. These numbers suggest that most reconciliations are dominated by either duplication or insertion, but not both. Fig. 6.5 explores this possibility further. As seen in the figure, duplications and insertions do seem to be negatively correlated — reconciled pairs that have many duplications, tend to have few insertions, and vice versa. Spearman's rank correlation test [319] reveals that duplications and insertions are slightly, negatively correlated, with a significant p-value. Overall, as suggested by this figure, the number of domains that were replicated by duplication is greater than the number replicated by insertions (the ratio of duplications to insertions is 1.23). Note that my methods do not consider events involving more than one domain at a time — *compound events*. While the total number of events may be overestimated if compound events are common, the *number of domains* affected by duplication or insertion are still the same. Thus, these results suggest that the effect of domain duplication is more common than domain insertion.

In addition to investigating trends in inferred events per reconciliation, I also analyzed the total number of events inferred per domain superfamily. When examining events on a per superfamily basis, we can further break this down into superfamilies acting as the embedded or reference domain. For each domain superfamily, D, the total number of events is calculated twice: first, by summing the events for all reconciliations in which $D = D_E$ and second, by summing the events for all reconciliations in which the $D = D_R$. Out of the 1611 superfamilies with more than two children, there are 914 different embedded domain superfamilies, 110 of which only act as a embedded domain; and 897 reference superfamilies, 93 of which only act as a reference domain. The remaining 804 superfamilies can play both roles.

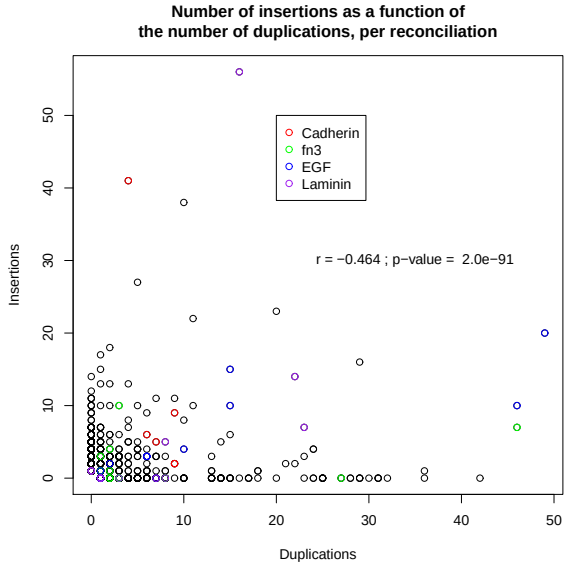
When the aggregate behavior of domain superfamilies is considered, most families do not participate in a large number of domain shuffling events, as seen in Figs. 6.6 and 6.7. However, some superfamilies appear to be more mobile than others. These superfamilies are presented in Table 6.4.

As seen in Figs. 6.8 and 6.9, when the number of events per family are considered, the number of insertions versus duplications per superfamily is weakly, but significantly, positively correlated. It is interesting to note the difference in trends when comparing events per reconciliation with events per family. This indicates that while a domain pair tends to *either* duplicate or insert on a per reconciliation basis, superfamilies as a whole do not seem to continue that trend. Rather, superfamilies will participate in both types of events, and if a family is involved with a number of insertions it is also likely to be involved in a number of duplications. This also suggests that some superfamilies are generally active (i.e., participate in both types of events) and others are



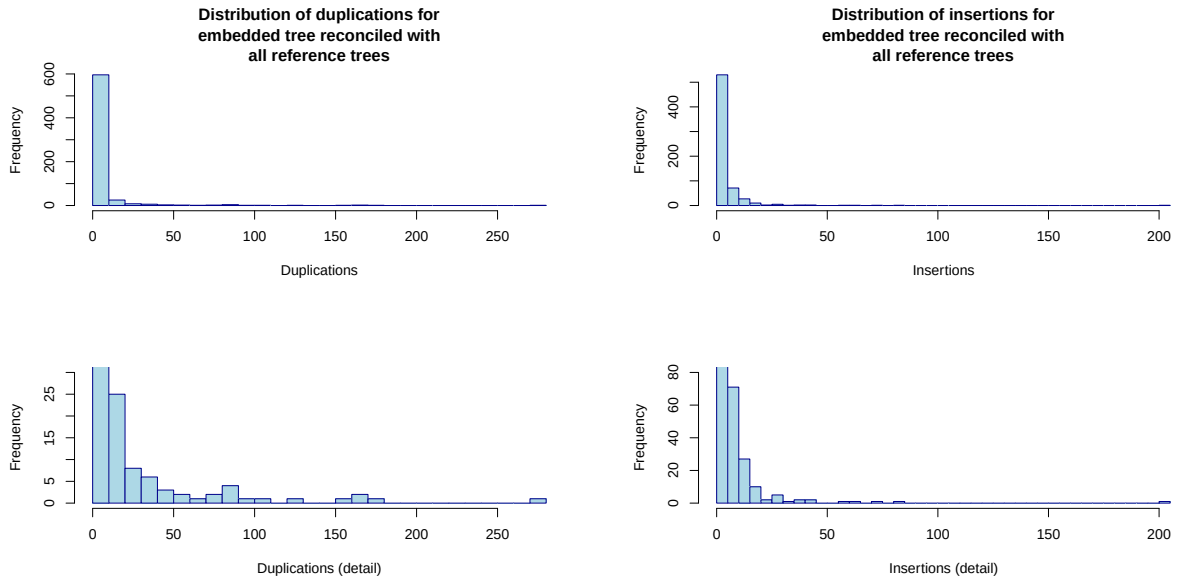
(a)

(b)

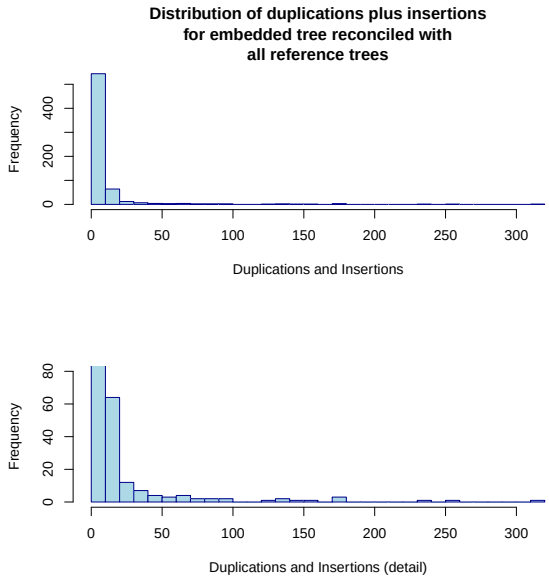


(c)

Figure 6.5: **(a)** Comparison of the number of duplications and insertions per reconciliation. Each dot represents a reconciliation of an embedded tree and reference tree. **(b)** The same as in (a), but zoomed in to present details. **(c)** The same as in (b), but with outliers removed and reconciliations with cadherin, fn3, EGF, and laminin domains highlighted. Spearman's rank correlation, r, between duplications and insertions is shown with its p-value. The solid line represents $y = x$.

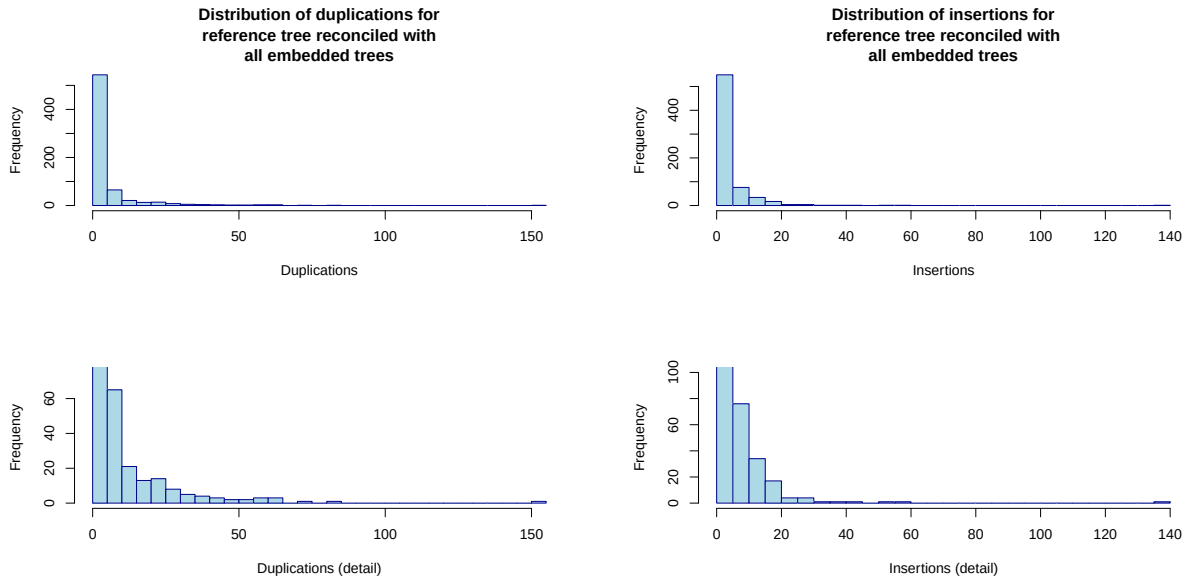


(a) (b)

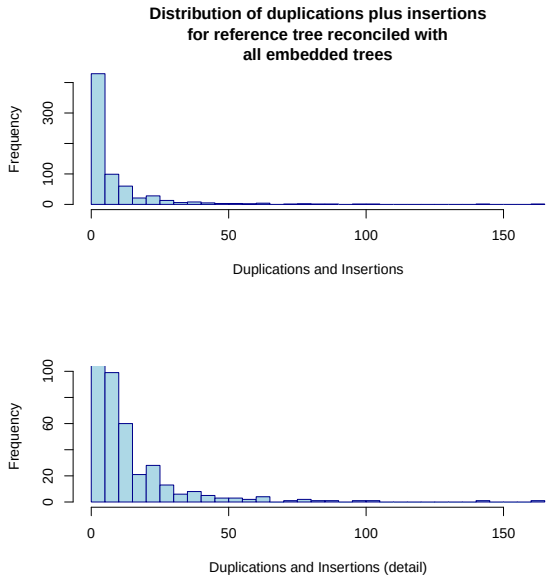


(c)

Figure 6.6: Summary information for embedded domains. Event numbers are totals over all reconciliation in which the domain participates. Distributions of the number of **(a)** duplications; **(b)** insertions; and **(c)** duplications *and* insertions. Reconciliations containing no events were not included.



(a) (b)



(c)

Figure 6.7: Summary information for reference domains. Event numbers are totals over all reconciliation in which the domain participates. Distributions of the number of **(a)** duplications; **(b)** insertions; and **(c)** duplications *and* insertions. Reconciliations containing no events were not included.

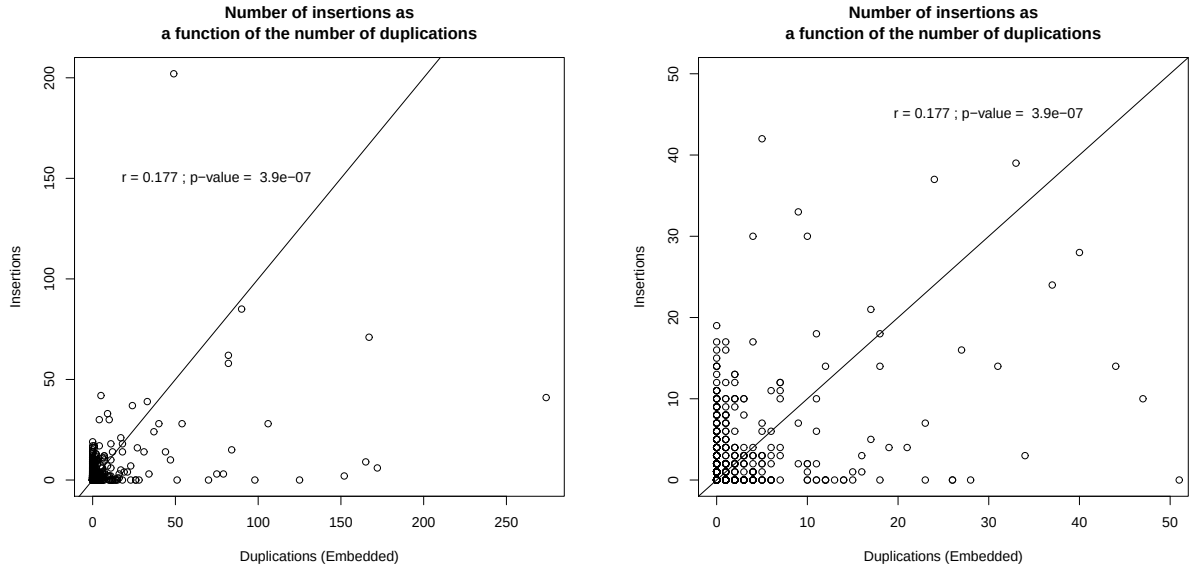
Outlier domains

Embedded domain	Reference domain	Duplications	Insertions	Duplication + insertions
LRR_1		82	58	140
TSP_1		82	62	144
I-set		152	2	154
Spectrin		165	9	174
Laminin_EGF		90	85	175
Nebulin		172	6	178
EGF		167	71	238
Cadherin		49	202	251
Ank		274	41	315
	NODP	61	37	98
	Laminin_N	48	57	105
	Cadherin_2	5	138	143
	SH3_1	152	10	162

Table 6.4: Domain superfamilies with a high number of duplication and/or insertion events when summarized over the embedded and reference. Events are totaled over all reconciliations in which the domain participates.

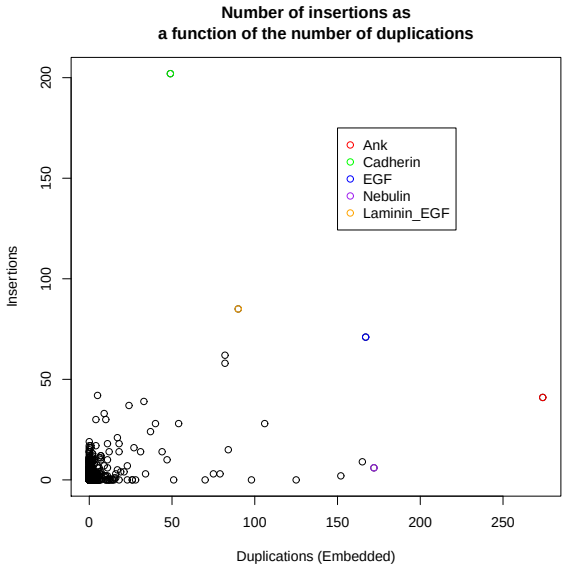
more static. This correlation effect is slightly stronger for reference than embedded superfamilies. One possible reason for this is that there is a maximum number of events that can be inferred on an embedded tree, but there is no such limit for the reference tree. This is discussed further in the upcoming paragraphs.

I also considered how the size of the embedded and reference trees influences the number of inferred events. Larger trees could allow for more shuffling because there are more domain instances and more chances to diverge. That is exactly what is observed in Figs. 6.10 and 6.11. The number of events inferred during a reconciliation is significantly, highly correlated with the size of the embedded tree, and significantly, but not as highly, correlated with the size of the reference tree. This intuitively makes sense. Domain shuffling could be contributing to the growth of larger domain superfamilies, in addition to gene duplication. Thus, larger families are more likely to have been involved in domain shuffling events. Note that the maximal number of events, N_E, that can be inferred given the size of the embedded tree. Because leaf nodes in the embedded tree do not represent any event, the maximum number of nodes that can be assigned an event is limited by the number of internal nodes. Thus, the maximal number of duplications and insertions in the embedded tree is $N_E = (\frac{|V_E|}{2} - 1)$. While the number of events in some reconciliations reaches N_E through a combination of duplications and insertions (Fig. 6.11a), it is interesting that several achieve N_E with only duplication events (Fig. 6.10a), but only a few reach N_E with insertion events



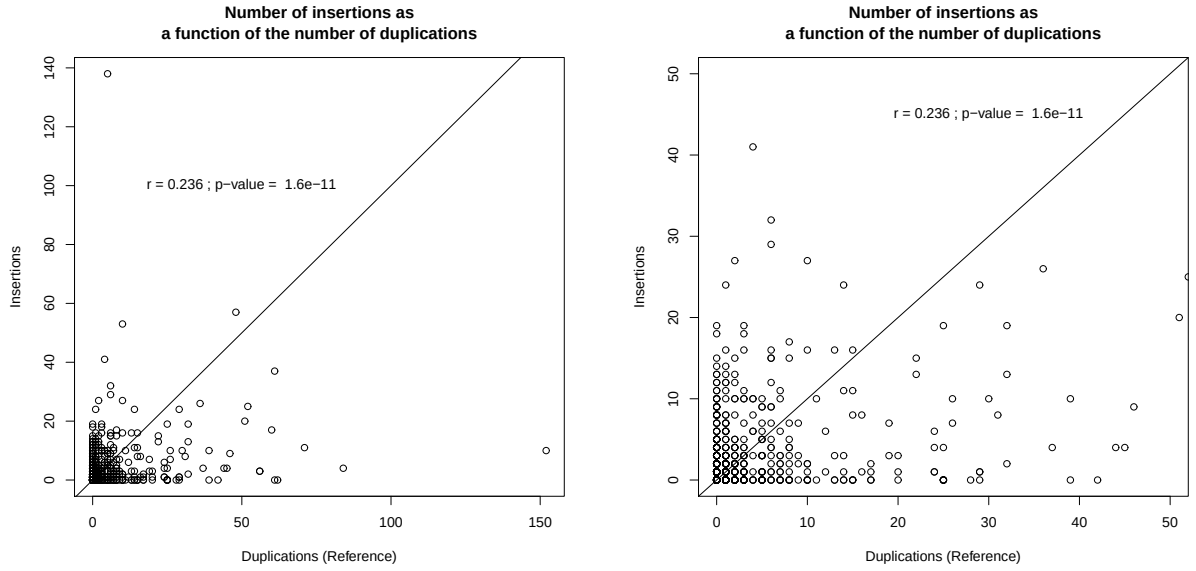
(a)

(b)



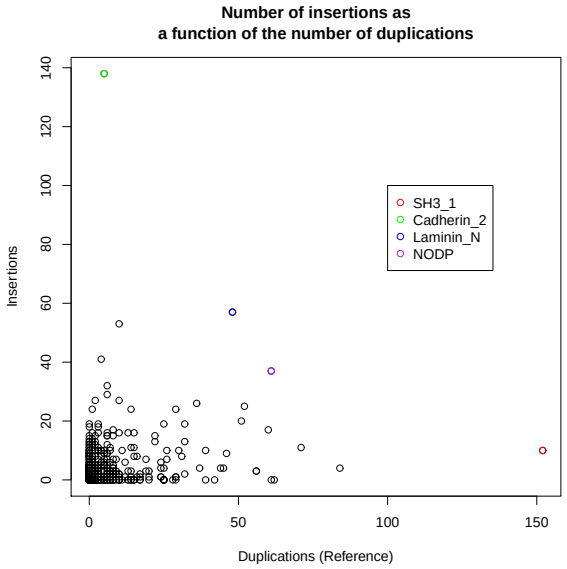
(c)

Figure 6.8: **(a)** Comparison of the number of duplications and insertions summarized over each embedded domain superfamily. Each dot represents a domain superfamily and all the events inferred when that domain acts as the embedded. **(b)** The same as in (a), but zoomed in to present details. **(c)** The same as in (b), but without the correlation and with Ank, cadherin, EGF, Nebulin, and laminin_EGF superfamilies highlighted. Spearman's rank correlation, r, between duplications and insertions is shown with its p-value. The solid line represents $y = x$.



(a)

(b)



(c)

Figure 6.9: **(a)** Comparison of the number of duplications and insertions summarized over each reference domain superfamily. Each dot represents a domain superfamily and all the events inferred when that domain acts as the reference. **(b)** The same as in (a), but zoomed in to present details. **(c)** The same as in (b), but without the correlation and with SH3_1, cadherin_2 laminin_N, and NODP superfamilies highlighted. Spearman's rank correlation, r, between duplications and insertions is shown with its p-value. The solid line represents $y = x$.

Near-Maximal Number of Events

Embedded domain	Reference domain	Duplications	Insertions	$\|V_E\|$	$\|V_R\|$
ig	RhoGEF	17	0	35	1
Ank	FYVE	20	0	41	1
PPAK	Pkinase	23	0	47	1
Ank	MutS_V	24	0	49	1
TSP_1	F5_F8_type_C	24	0	49	1
RNA_pol_Rpb1_R	RNA_pol_Rpb1_2	25	0	51	1
RNA_pol_Rpb1_R	RNA_pol_Rpb1_3	25	0	51	1
RNA_pol_Rpb1_R	RNA_pol_Rpb1_1	25	0	51	1
RNA_pol_Rpb1_R	RNA_pol_Rpb1_5	25	0	51	1
RNA_pol_Rpb1_R	RNA_pol_Rpb1_4	25	0	51	1
fn3	Laminin_N	27	0	55	1
LSPR	Cu-oxidase_3	28	0	57	1
Ldl_recept_a	NHL	29	0	59	1
Ldl_recept_a	EGF_2	29	0	59	1
Ldl_recept_b	NHL	31	0	63	1
Ldl_recept_b	EGF_2	31	0	63	1
Nebulin	SH3_1	143	6	321	5
Sushi	Pentaxin	31	0	63	1
CheC	PDZ	32	0	65	1
I-set	IQ	36	0	73	1
I-set	G2F	42	0	85	1

Table 6.5: Reconciliation pairs that meet or nearly meet the maximum number of events on a tree, based on tree size. Events are for the specified reconciliation.

only (Fig. 6.10c). The reconciliations that reach N_E through duplications plus insertions are also the domains that meet, or nearly meet, the maximum through duplications alone (see Table 6.5). One explanation for this phenomenon is the presence of genes in the human genome that contain long strings of domain repeats, which likely evolved through repeated domain duplication, and a lack of genes which evolved domains solely through domain insertions.

In contrast to the embedded tree, the reference tree may have more than one event associated with any node. The red lines in Figs. 6.10b, d and 6.11b represent the reconciliations that have as many events as nodes in the reference tree. Points above this line have more events than nodes in the reference tree, indicating that some nodes in the reference tree *must* be associated with more than one event. Points below this line indicate reconciliations where nodes in the reference tree *may* not be associated with more than one event. Note that there are a large number of reconciliations that fall above this line.

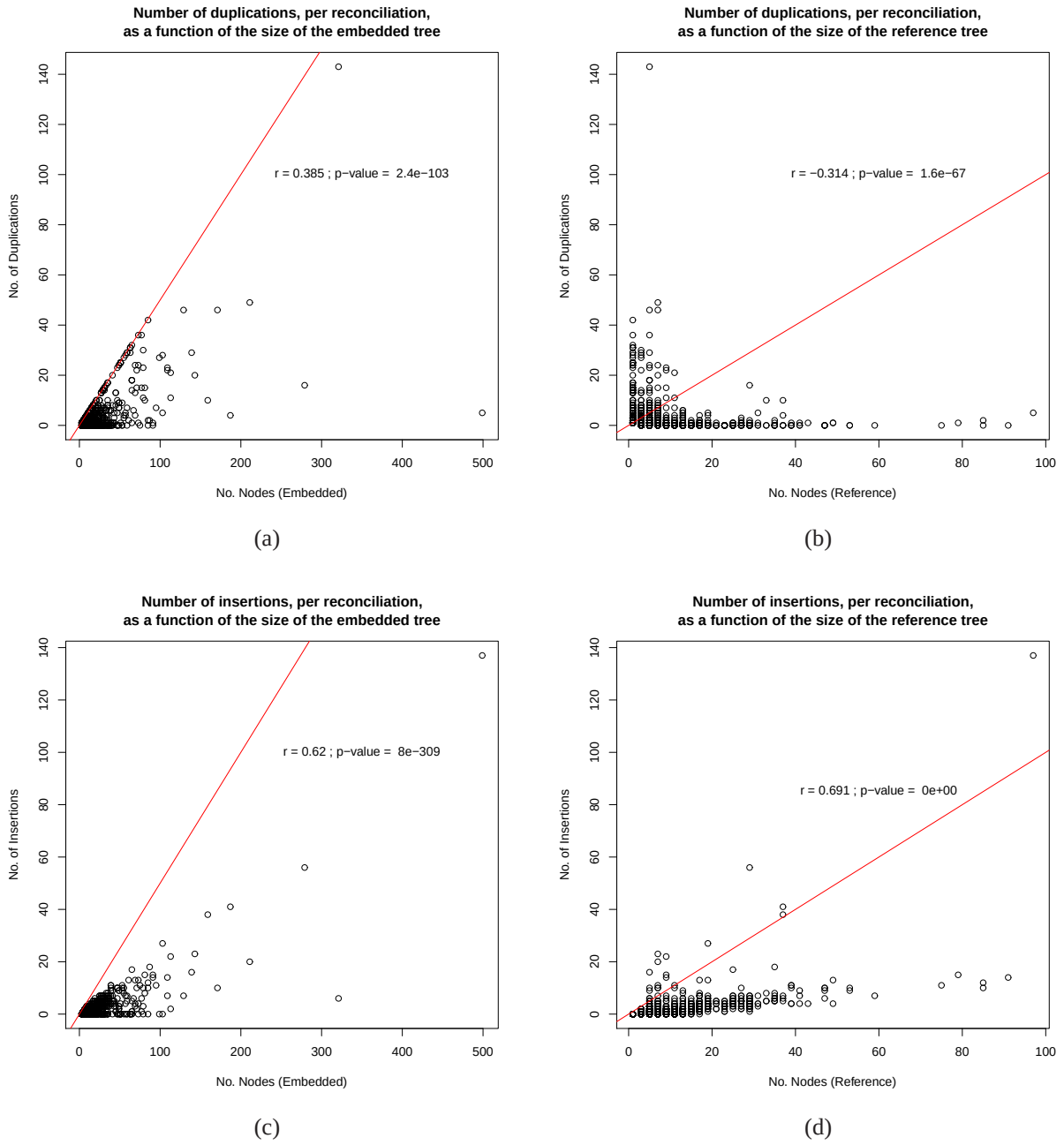
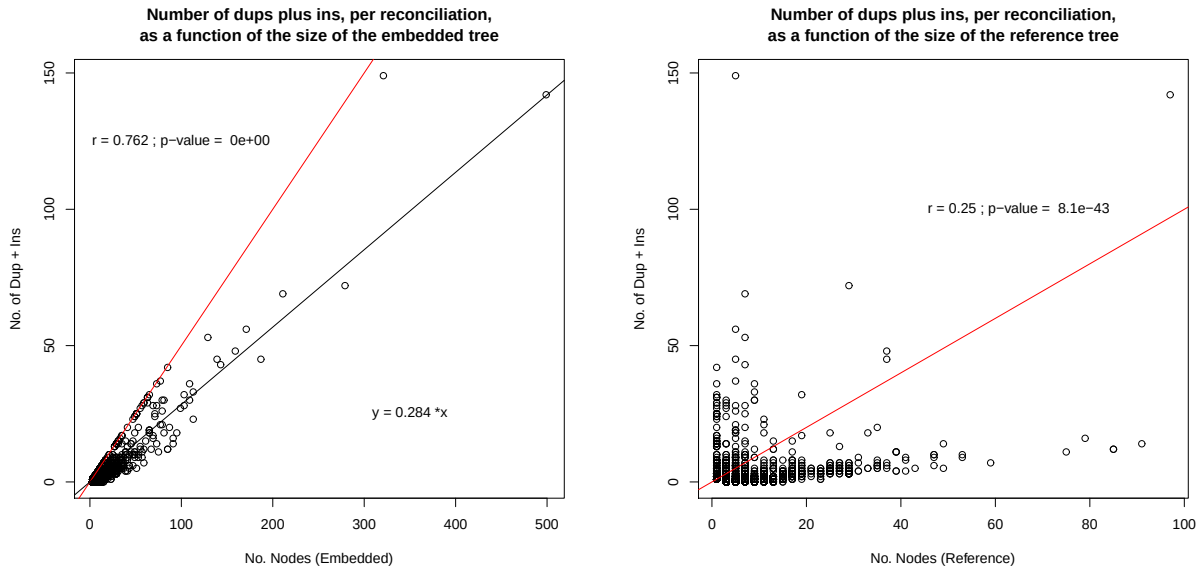


Figure 6.10: Comparing the number of inferred events in a reconciliation with the size of the trees, $|V_i|$ participating in the reconciliation. Each dot represents a single reconciliation. **(a)** The number of duplications and the V_E are significantly, but not strongly, correlated. **(b)** In contrast, the number of duplications and $|V_R$ have a weak, negative correlation that is significant. **(c)** Insertions are more correlated than duplications with $|V_E|$. **(d)** Insertions are also positively (rather than negatively) correlated with $|V_R|$. Spearman's rank correlation, r, between duplications or insertions and $|V_i|$ is shown with its p-value. The red solid lines in (a) and (c) represent number of events $= 0.5 * |V_E| - 1$, while the red solid lines in (b) and (d) represent number of events $= |V_R|$.



(a) (b)

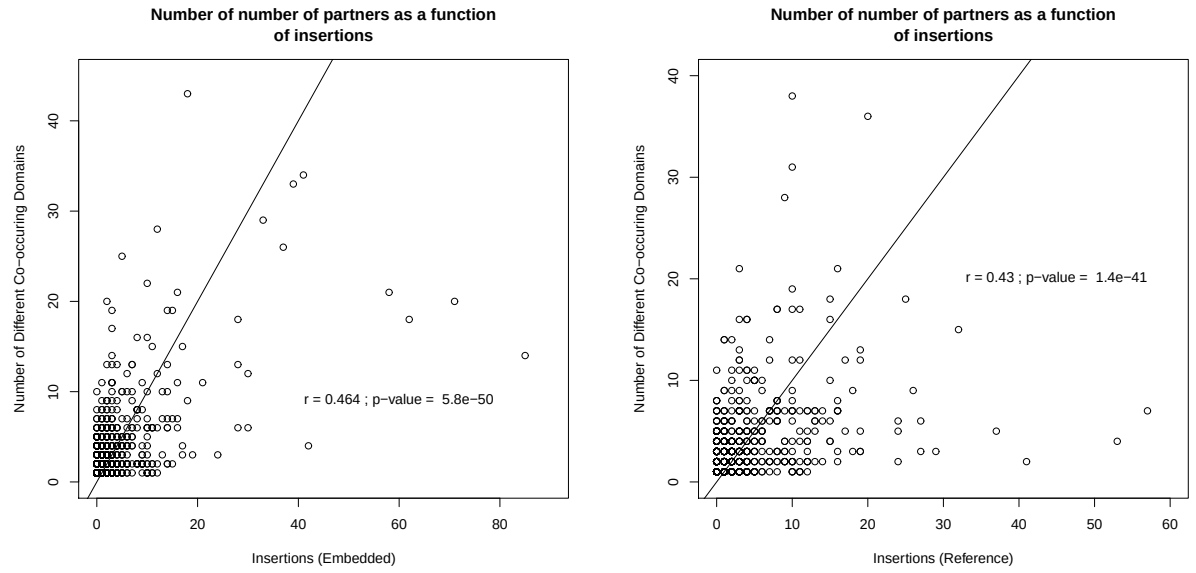
Figure 6.11: Comparing the number of inferred events in a reconciliation with $|V_i|$. Each dot represents a single reconciliation. **(a)** The number of duplications and insertions are highly, and significantly, correlated with $|V_E|$. **(b)** In contrast, the duplications plus insertions, are less, but still significantly, correlated with $|V_R|$. Spearman's rank correlation, r, between duplications plus insertions and tree size is shown with its p-value. The red solid line in (a) represents number of events $= 0.5 * |V_E| - 1$, while the black solid line represents the least squares best fit of a line: number of events $= 0.284 * |V_E|$. The red solid line in (b) represents number of events $= |V_R|$.

I also considered the correlation between the number of inferred events and the promiscuity of a domain, quantified by the number of other superfamilies with which the domain co-occurs. Domain promiscuity is often assumed to also be a measure of a domain's "mobility." In general, the thought is that a domain has more partners because it has been inserted into more contexts. However, because models currently in use do not capture dynamic properties, little evidence is available to refute or support this assumption. Fig. 6.12 shows scatter plots of the number of co-occurrences with distinct domains as a function of the number of domain insertions. If promiscuity arises from increased mobility, we would expect co-occurrences to increase with insertions. While this general trend is seen, with a statistically significant, positive correlation, Fig. 6.12 shows that some domains co-occur with many other domains, yet have experienced relatively few insertions while others are involved in many insertions, yet have relatively few partners (see also Table 6.6 for an example of these domains). These are potential attractor/mobile domains and are good candidates for a detailed study for evidence of mobility (or lack thereof), which is beyond the scope of this thesis.

Domains with Poor Correlation between Partners and Insertions

Embedded domain	Reference domain	Number partners	Duplications	Insertions
Cadherin		7	49	202
Neur_chan_memb		1	0	12
Connexin_CCC		1	0	11
Connexin		1	0	11
FG-GAP		4	5	42
Laminin_EGF		14	90	85
EGF		20	167	71
TSP_1		18	82	62
LRR_1		21	82	58
AT_hook		11	15	1
I-set		20	152	2
dsrm		9	10	1
RasGEF_N		8	0	1
SAM_2		8	1	1
PH		43	11	18
efhand		25	17	5
RhoGEF_N		19	2	3
	Cadherin_2	2	5	138
	Cadherin_C	2	4	41
	Reprolysin	4	10	53
	Neur_chan_LBD	1	0	12
	Integrin_alpha	2	1	24
	Laminin_N	7	48	57
	Bromodomain	14	5	1
	efhand	14	16	1
	I-set	9	5	1
	zf-CXXC	9	5	1
	RhoGEF	21	56	3
	PH	38	26	10
	Pkinase	36	51	20
	SH3_1	31	152	10

Table 6.6: Domain superfamilies with the weakest correlation between promiscuity (number of different partners) and the number of inferred events, including either many partners and few events, or few partners but many events. Events are totaled over all reconciliations in which the domain participates as either the embedded or reference tree.



(a) (b)

Figure 6.12: Comparing the number of different superfamilies with which a domain co-occurs to the number of inferred events all reconciliations. Each dot represents the summarized data for a domain superfamily. The number of insertions summed over all reconciliations for a domain acting as **(a)** the embedded and **(b)** the reference are positively correlated, with a high-level of significance. Spearman's rank correlation, r, between insertions and the number of partners is shown with its p-value. The solid black lines represent $y = x$.

This analysis also provides evidence for the *convergent evolution* of domain architectures. In the scatter plot in Fig. 6.12, the domains with the most insertions have relatively few partners, suggesting that domains are repeatedly inserted adjacent to the same partners. This could be evidence that certain domain architectures are particularly advantageous and suggests that these are the most mobile domains, but not the most promiscuous domains. This observation also speaks to the question: is it unusual for the same domain architecture to arise more than once through independent events? The presence of domains with high insertion to co-occurrence ratios suggests that same domain pairs must arise more than once. Otherwise, we would expect to see (at least) as many partners as insertions (as represented by the black lines in the figures). Roughly 20% of domain superfamilies in this study have more insertions than partners.

6.0.1. Significance of results

Through this high-throughput analysis, I have demonstrated the utility of the algorithms described in Chapter 4 for genome-scale analyses. All 2,914 tree pairs were reconciled in NOTUNG in only 3 hours, 1 minute, and 28.33 seconds, on a 3.2ghz OptiPlex GX620 computer. In addition, I have presented results describing general patterns of domain shuffling in the human genome. First, more than half of comparable domain pairs participated in some type of domain shuffling event. Second, domain proliferation via duplication events is generally more common (≈ 1.25 times) than proliferation by insertion. In addition, events do not occur with equal frequency across all families, and some families are more "mobile" than others. Third, comparison of the number of inferred events with tree size and the number of co-occurring domains sheds further light on domain mobility and the convergent evolution of domain architectures. First, larger trees, and thus larger families, are involved in more domain shuffling, which may have contributed to the larger size of these families. In addition, while there is a positive correlation between the number of partners for a domain and the number of events in which that domain is involved, there are still a number of superfamilies that do not fit this generalization. Both domains with many partners but few events, and domains with few partners involved in many events were observed. This provides indirect evidence that mobility is not synonymous with promiscuity (as measured by number of partners). However, it is not possible to determine from pairwise domain tree comparisons which of the two domains was inserted. Similarly, these results suggest that the same domain pairs form more than once and that convergent evolution of the same domain pair may be more common than previously thought.

Chapter 7

Discussion

Over the past thirty years, molecular phylogenetics has grown into a well-established and essential field in the life sciences and is invaluable for many applications. Although there is an abundance of research on phylogenetic methods for single-domain genes, current approaches are not appropriate for the analysis of multidomain genes. When researchers first recognized that some genes are a mosaic of sequence fragments [52, 53], the consensus was that these types of genes were rare exceptions. However, with the advent of whole genome sequencing, the extent of such genes has become readily apparent: the percentage of genes with two or more domains has been estimated to range from 27% of all genes in prokaryotes and 40% in metazoans [32] to as much as 60% in prokaryotes and 80% in eukaryotes [80]. Gene families containing multidomain members not only evolve via sequence substitution and gene duplication, transfer, and loss, but also through domain shuffling, a process in which individual domains are duplicated, deleted, and inserted into new contexts. Because multidomain families can consist of multiple domain superfamilies that also occur in otherwise unrelated gene families, different domain superfamilies in the same multidomain family may have different evolutionary histories [27–31]. Yet, standard phylogenetic methods do not recognize this possibility and will not infer domain shuffling events.

Ironically, these gene families that elude current phylogenetic approach are of particular evolutionary and functional importance. They played transformative roles in key evolutionary transitions, including the emergence of multicellularity in animals and chordate and early vertebrate evolution [33, 35, 81]. Many multidomain families are associated with fundamental molecular functions such as cell signaling and cell adhesion.

7.1. Summary of results

In this thesis, I have presented my work on: *(1)* the development of a model of co-evolution of domains with the locus of the gene family; *(2)* the design of algorithms to realize this model; *(3)* the implementation of these algorithms into software capable of in-depth and/or high-throughput analyses; *(4)* a detailed empirical analysis of a set of multidomain families from the literature, demonstrating the power of my approach; and *(5)* a study of domain shuffling in the human genome, providing information on genome-level patterns of domain shuffling and demonstrating the suitability of my approach for genomic-scale studies.

In Chapter 3, I discussed the classical single-domain model [181] of gene family evolution and the locus model of multidomain gene family evolution [151]. A multidomain family is the set of genes descended from a common locus in the genome. Under this model, a tree structure is an appropriate representation of multidomain family's evolution; domain shuffling events are annotations on this tree. I developed a model of the evolution of multidomain families, in which a multidomain protein is represented as a co-evolving set or sequence of domains. The domain shuffling events and ancestral domain architectures in the history of the family can all be inferred by comparing trees representing these co-evolving domains. I further developed this model in the context of reconciliation to define the abstract domain shuffling events consistent with known molecular mechanisms for shuffling.

My integrated model of multidomain family evolution comprises the evolutionary history of each constituent domain (including the sequence mutation that gave rise to it), the evolutionary history of the locus as a whole, the domain shuffling events that gave rise to the domain architectures, and the ancestral domain architectures. The use of a comprehensive, formal framework to investigate co-evolutionary relationships in a multidomain context is a novel and dramatic departure from previous approaches. While several studies have focused on some aspect of these co-evolutionary relationships (domain-species evolution [129,138,140,142,320] or domain-domain evolution [139]), an explicit and comprehensive formal framework that models the co-evolution of domains, genes, and species has not been proposed.

To realize this model, I developed four algorithms (presented in Chapter 4) to infer the evolutionary history of a multidomain family:

1. A reconciliation algorithm for duplication, heuristic loss, and incomplete lineage sorting, given a binary embedded tree and a non-binary reference tree. The details of this

algorithm have been published in *The Journal of Computational Biology* [252] and implemented in the NOTUNG package by Ben Vernot.

2. A reconciliation algorithm for horizontal transfer, duplication, and loss, given binary embedded and reference trees. I implemented this algorithm in NOTUNG and extended the GUI to handle aspects specific to multidomain evolution.

3. A reconciliation algorithm for duplication, heuristic loss, incomplete lineage sorting, and transfer given a binary embedded tree and a non-binary reference tree. This algorithm is based on algorithms (1) and (3) that I developed. It has been implemented in NOTUNG by Han Lai, under my supervision.

4. An algorithm to infer ancestral states of the reference tree and assign events to the reference tree, given a set of reconciled embedded trees. This algorithm was implemented in NOTUNG by a Master's student, Ravi Chinoy, under my supervision.

I used my algorithms in NOTUNG to apply my novel approach to two data sets: a set of multidomain families discussed in the literature and the set of all domains in the human genome.

A few studies have considered domain trees in a reconciliation framework. These studies used an *ad hoc*, informal version of this approach to explore the differing histories of various domains in the same multidomain family [27–30]. In this body of work, disagreement between domain trees is inferred by inspection, and taken as general evidence of domain shuffling, although the specific domain shuffling events are not inferred. Inference by visual inspection is error prone and not feasible for large data sets. A limitation of this work is that no models, algorithms, or software to infer multidomain gene family trees or domain shuffling events have been proposed, until now. My algorithms and software will allow researchers to perform phylogenetic analysis of such families consistently and on a much larger scale. A great advantage of the tree comparison method is that it incorporates both sequence and domain architecture information. Moreover, unlike the domain architecture model, this representation makes it possible not only to infer domain shuffling events, but also to infer which specific domain in one gene was duplicated and then inserted into a different gene.

In Chapter 5, I selected three multidomain families from the literature (the protein tyrosine kinases [27], the Notch-related genes [28], and the membrane-associated guanylate kinases [30]) to demonstrate how my work can be used to study the evolution of such families. Over the last decade, a body of work based on domain architecture parsimony has established several hypotheses about

the processes of multidomain evolution that are often treated as accepted theory. However, I have demonstrated that the domain gain-loss, based on architecture parsimony alone, may underestimate the extent of domain shuffling. In particular, my approach is able to infer domain insertions that may not be apparent to domain architecture parsimony methods. As a result, my approach has the potential to overturn prior results based on domain architecture parsimony, such as rates of domain shuffling events and the assertion that convergent evolution of the same architectures is rare [142]. My novel methodology, which captures both domain architecture and sequence information, provides a platform upon which to test whether a more informative model will lead to different conclusions. Specifically, in my empirical analyses, I repeatedly observed that the same assemblage of domains may have evolved independently, multiple times. As a result, domains in the same position in paralogous genes may not be close homologs, as previously assumed. These results suggest that my new method has the potential to provide a deeper, more detailed glimpse into the area of multidomain evolution.

In Chapter 6, I describe the application of the developed methods and software to all pairs of comparable domain superfamilies in the human genome. This study revealed a number of interesting observations about the pattern of domain shuffling in humans. Specifically, there was evidence of domain shuffling in more than half of all comparable domain pairs, indicating that the pair either formed at least twice in their evolutionary history or included a domain duplication. Also of note was the observation that domain duplications are 1.23 times more common than insertions. Summarizing results over all instances of a particular domain provided further insights into the behavior of domain superfamilies as a whole: A co-occurring pair of domains tends to proliferate *either* by duplication or by insertion. In contrast, families as a whole will participate in both types of events, and if a family is involved with a number of insertions it is also likely to be involved in a number of duplications.

Comparing the inferred number of domain shuffling events with properties of domain superfamilies indicated two interesting correlations. First, the number of events inferred during a reconciliation is significantly and highly correlated with the size of the embedded tree, which is representative of the size of the domain family. This suggests that larger families are larger because they were involved in more domain shuffling with gene duplication. Of note was the observation that a few domain pairs reach the maximal number of events that can be inferred from tree discordance given the size of the embedded tree. These are mostly the result of genes that contain long strings of domain repeats, which likely evolved through repeated domain duplication. In addition, a number have reference trees containing nodes associated with more than one event. This observation,

plus the distribution of events per reconciliation, indicates that there is a *gradient* of "mobility" in which some superfamilies are much more mobile than others. This was further investigated by the comparison of the number of different co-occurring partners of a domain, a widely used measure of promiscuity, with the amount of shuffling in which that domain was involved. While this comparison revealed a correlation between these two quantities, there were noted exceptions. Some domains co-occur with many other domains, yet have experienced relatively few insertions while others are involved in many insertions, yet have relatively few partners. This indicates that there may be a distinction between mobility and co-occurrence, although these definitions were previously assumed to be synonymous (see [154]). The presence of domains with high insertion to co-occurrence ratios also suggests that domains are repeatedly inserted adjacent to the same partners and is possible evidence that convergent evolution of domain architectures is not infrequent.

Limitations. While the method presented here has an advantage over previously proposed methods for analyzing the evolution of multidomain proteins, there are still some limitations. First, while my method does take sequence information from domains into account, through the construction of domain trees, this does not cover the full sequence of every gene: *linker sequences*, the sequence between domains, are not included. Thus the history of some parts of the gene are ignored. However, this omission may not pose a significant problem. Domain boundaries tend to coincide with exon boundaries [102, 112, 321]. Also, domains correspond to sequences that have a fold and/or function found in many contexts. As a result, they are more likely to be shuffled than arbitrary fragments lacking these characteristics. Second, this method assumes that all domain instances in the multidomain family have been accurately annotated; yet this assumption is not necessarily valid, as false negative errors are not uncommon [322]. While the effect of domain misannotation on event inference has not been largely studied, Weiner et al. [127], in their study on domain losses, reports that misannotations accounted for only a small fraction of false positive domain loss predictions. In addition, with increasing amounts of genomic and structural data, as well as improved computational techniques and continued manual curation, the number of false negative domain misannotations will continue to decline [322]. Third, the reconciliation of multidomain families is based on an event parsimony approach, and does not include a probabilistic model of uncertainty, etc. Several problems remain for future work. Parsimony approaches are well-suited to data sets in which events are rare, due to selective pressure. Probabilistic models would complement the parsimony framework presented here. Bayesian approaches [147, 159, 215, 219, 300, 323], which assume homogeneous rates, are appropriate for data sets in which duplication and loss are neutral, stochastic processes. A probabilistic framework provides a natural setting for incorporat-

ing sequence data directly into the reconciliation process. As such, a complete phylogenetic toolkit should include both approaches. However a probabilistic model has the disadvantage that it is both computationally intensive and that it requires enough data to learn parameters, which may not be possible with domain sequences since they are very short. Finally, as discussed in Sec. 5.5, inferred domain insertion with domain loss events may be the result of gene conversion on a domain level, rather than domain shuffling events. The hypothesis for gene conversion can be ruled out by screening members of multidomain gene families for gene conversion in a pre-processing step, such as with the OrgConv [312] or GeneConv [313, 314] software. Alternatively, my work may prove to be a useful approach for detecting gene conversion.

7.2. Future work

The novel approach to the phylogenetic analysis of multidomain proteins, presented in this thesis, represents a major step towards furthering our understanding of multidomain evolution. It further suggests a number of open research directions, including with further algorithmic development of this work and the application of these methods to other data sets and to answer specific evolutionary questions.

7.2.1. Future directions for algorithmic development

I have developed and implemented reconciliation algorithms for inferring events and composing evolutionary histories of multidomain gene families. My results suggest several important directions for further development: *(1)* identification of possible compound events; *(2)* identification of insertions originating outside the gene family; *(3)* improved identification and avoidance of temporal inconsistencies; *(4)* parameter selection; and *(5)* development of algorithms for the tree inference problem. Many of these are unique challenges that do not arise in other reconciliation frameworks.

Compound events. My event model, like all reconciliation approaches, assumes that each event modifies one entity (i.e., domain or gene) at a time. However, it is possible that a fragment encoding two or more domains could be the target of a single insertion, duplication, or deletion — a *compound event*. If such compound events do, in fact, occur, the reconciliation will overestimate

the amount of shuffling that takes place. Current evidence suggests that domain insertions and losses tend to involve a single domain (rather than a set of domains) [130, 131, 133, 134], but it has been observed that tandem repeats can expand through internal duplication of several domains at once [170, 324, 325]. Current methods to identify compound events are limited to tandem repeats involving the same number of domain instances from the same domain superfamily in each duplication. Compound insertions have not been considered. Development of improved methods for the detection of compound events would be beneficial to this field. Specifically, domain tree structure and reconciliations could be exploited to identify compound events.

The origin of insertions. The ability to infer the source of insertions is a particular strength of our approach. My algorithms, like *all* reconciliation algorithms that include horizontal events such as insertion, are based on the assumption that insertions or transfers originate *within* the gene family. A promising approach to infer insertions that originate from outside a given multidomain family is to consider all instances of the domain superfamily in the genomes of interest. Reconciliation methods for full domain superfamilies trees, not just trees constructed from the domain instance in the given multidomain family, could provide a solution for inferring external domain insertions in such cases.

Temporal constraints and cycles. To be biologically relevant, inferred insertions must obey temporal constraints; insertions can only occur between contemporaneous taxa. However, because the relative timing of nodes in different lineages in the reference tree is, in general, unknown, the algorithms presented here only identify temporal violations that create cycles. If reference tree nodes are partitioned into sets of contemporaneous taxa, a reconciliation algorithm could be developed that only considers insertions that occur within the same set [326]. An improved reconciliation technique that incorporates species information could be developed along these lines, by exploiting divergence time estimates available from other sources; e.g., the fossil record. If the reference tree is a gene tree, prior reconciliation with a species tree could be used to transfer time estimates from the species tree to the reference gene tree nodes, via the mappings.

Selecting event costs. Different values for event cost parameters may result in different optimal event histories. As I demonstrated in Sec. 5.4, the set of different solutions and the region in parameter space that they occupy can be determined by sampling the parameter space and reconciling with these samples. The development of a methodology to systematically enumerate all possible

histories and sample the space of reconciliations would be of great benefit to genomic-scale studies where sampling the parameter space is not feasible.

In addition, further genomic-scale analyses using a maximum likelihood approach could provide estimates for domain shuffling event rates. These rates are expressed in terms of the number of events that occur per unit of time. These rates could then be converted into event costs — those events with higher rates should be assigned lower costs because they occur more often. While a simple inversion of rates would seem appropriate, it is not statistically consistent. Statistical analyses of parsimony methods have revealed that the appropriate rate-to-cost function is the negative natural log of the rate (i.e., $-\ln r_i$ for rate r_i of event i) [24].

7.2.2. Future directions for biological analysis

The goal of my thesis was to develop methods to help bridge the gap between the large amount of available multidomain protein data and our lack of efficient, sufficiently detailed methods to analyze this data. The models, algorithms, and software I developed to meet this goal promise to provide the research community with tools to analyze a wide range of multidomain gene families, which could not be studied with standard phylogenetic methods. My methodology can also be used to investigate and evaluate phylogenetic hypotheses. Open scientific questions that can be investigated, with my methods:

1. Do different types of domains (e.g., domains associated with adhesion versus signaling) have different shuffling propensities?
2. Does shuffling occur more frequently within or between multidomain gene families?
3. Are shuffling rates lineage specific, and, if so, how much variation is observed in domain shuffling rates across species lineages?
4. How do inferred species-specific and family-specific rates compare to gain and loss rates reported in previous studies of gene and domain families [272, 300, 327, 328]?
5. How often do domain insertions cross species boundaries?
6. Has the mobility of a given domain changed over time?
7. How does the emergence of new domains or new architectures correlate with major evolutionary transitions, such as multicellularity, or the emergence of metazoans, vertebrates, and primates?

The origin of inserted domains. Almost nothing is known about the *origins of inserted domains* because prior studies were based on models in which domain instances are indistinguishable. Under such a model, all members of the domain family are equally likely donors. In order to infer the most likely source of a domain insertion, it is necessary to consider variation within a domain family. My algorithm infers the donor and the recipient gene of each insertion, making it possible to investigate the extent to which domain shuffling occurs within a gene family, between distinct gene families, and to what extent domain shuffling is contained within in species boundaries.

Differences across lineages. Is multidomain evolution a universal process with similar behavior in all lineages, or does the interplay between gene duplication, domain shuffling and sequence evolution differ between lineages? Since at this point multidomain evolution has been studied in much greater detail in bacteria and animals than in plants and fungi, it is difficult to know to what extent results from those studies can be generalized. In addition, many studies have been carried out on the complete set of whole genome data available at the time of that study, which typically include hundreds of bacterial genomes and only a handful of eukaryotic genomes. Possible lineage specific differences would likely be obscured by this disparity in numbers and the use of combined analyses. Moreover, it has been observed that the increased complexity of multidomain families in metazoans coincided with the advent of multicellular animals. It is an intriguing question whether the same patterns of gene duplication and domain insertion that prompted the evolution of metazoan signal transduction families also dominate in other lineages.

Domain promiscuity. A domain may be promiscuous because it is mobile (i.e., frequently inserted into new contexts) or because it is an attractor (i.e., insertions into the neighborhood of this domain tend to be selectively advantageous). Numerous measures have been proposed for quantifying domain promiscuity [32,120,121,123,155,173,174,329], each of which are based on domain architecture statistics, not on a explicit model of domain shuffling events. These measures represent the domain content in contemporary proteins without considering events that gave rise to that domain content. Measures based on domain architecture statistics may not give a good estimate of mobility, and thus may include domains that are attractors, but not mobile. My methodology can be used to estimate domain mobility and compare these estimates with previous measures of promiscuity, as reported in Chapter 6. Two important questions this can help answer are: What is the relationship between family-specific shuffling rates, inferred using an explicit, event-based model, and traditional measures of promiscuity? Can event models distinguish between actively mobile domains and attractors, and is such a dichotomy appropriate?

Convergent evolution. Several studies have considered *convergent evolution of domain architectures*, that is, the propensity for the same architecture to arise more than once through independent events. Convergent evolution is generally thought to occur rarely [116, 139], although more recent evidence is less clear [139, 143]. The assumption driving these studies is that a domain combination that is selectively advantageous, once formed, is unlikely to separate. Thus, identical domain architectures in contemporary proteins are assumed to be related through vertical descent. However, prior studies were based on methods that do not explicitly model events and have treated all domain instances as indistinguishable. These models could obscure evidence of repeated formation of the same domain combinations. Several recent studies of specific multidomain protein families have commented on instances where convergent evolution of domain architectures is the most compelling explanation for similar architectures in distantly related species [178, 179]. As shown, my methodology is capable of recognizing identical or very similar architectures that arose through independent events. Thus, it provides an excellent platform to investigate whether an approach that captures sequence similarity between domain instances uncovers evidence that the same domain architecture forms more than once.

Bibliography

[1] Searls, D. (**Aug 2003**). "Pharmacophylogenomics: genes, evolution and drug targets." *Nat Rev Drug Discov*, **2**(8): 613.

[2] Bull, J. and H. Wichman (**Sep 1998**). "A revolution in evolution." *Science*, **281**: 1959.

[3] Mathema, B., N. Kurepina, D. Fallows, and B. Kreiswirth (**Oct 2008**). "Lessons from molecular epidemiology and comparative genomics." *Semin Respir Crit Care Med*, **29**: 467.

[4] Campoccia, D., L. Montanaro, and C. Arciola (**Sep 2009**). "Current methods for molecular epidemiology studies of implant infections." *Int J Artif Organs*, **32**: 642.

[5] Shibata, D. (**Jan 2008**). "Stem cells as common ancestors in a colorectal cancer ancestral tree." *Curr Opin Gastroenterol*, **24**: 59.

[6] Navin, N. and J. Hicks (**Jun 2010**). "Tracing the tumor lineage." *Mol Oncol*, **4**: 267.

[7] Semrau, J., A. DiSpirito, and S. Yoon (**Jul 2010**). "Methanotrophs and copper." *FEMS Microbiol Rev*, **34**: 496.

[8] Lillis, L., N. Clipson, and E. Doyle (**Aug 2010**). "Quantification of catechol dioxygenase gene expression in soil during degradation of 2,4-dichlorophenol." *FEMS Microbiol Ecol*, **73**: 363.

[9] Bernard, E., Y. Azad, A. Vandamme, M. Weait, and A. Geretti (**Sep 2007**). "Hiv forensics: pitfalls and acceptable standards in the use of phylogenetic analysis as evidence in criminal investigations of HIV transmission." *HIV Med*, **8**: 382.

[10] Metzker, M., D. Mindell, X. Liu, *et al.* (**Oct 2002**). "Molecular evidence of HIV-1 transmission in a criminal case." *PNAS*, **99**: 14292.

[11] Nugent, K. and B. Saville (**Mar 2004**). "Forensic analysis of hallucinogenic fungi: a DNA-based approach." *Forensic Sci Int*, **140**: 147.

[12] Salas, A., H. Bandelt, V. Macaulay, and M. Richards (**May 2007**). "Phylogeographic investigations: the role of trees in forensic genetics." *Forensic Sci Int*, **168**: 1.

[13] Budowle, B., M. Johnson, C. Fraser, *et al.* (**2005**). "Genetic analysis and attribution of microbial forensics evidence." *Crit Rev Microbiol*, **31**: 233.

[14] Alizon, S., V. von Wyl, T. Stadler, *et al.* (**Sep 2010**). "Phylogenetic approach reveals that virus genotype largely determines HIV set-point viral load." *PLoS Pathog*, **6**.

[15] Tebit, D., M. Lobritz, M. Lalonde, *et al.* (**Oct 2010**). "Divergent evolution in reverse transcriptase (RT) of HIV-1 group O and M lineages: impact on structure, fitness, and sensitivity to nonnucleoside RT inhibitors." *J Virol*, **84**: 9817.

[16] Lamping, E., P. Baret, A. Holmes, *et al.* (**Feb 2010**). "Fungal PDR transporters: Phylogeny, topology, motifs and function." *Fungal Genet Biol*, **47**: 127.

[17] Aminov, R. (**Dec 2009**). "The role of antibiotics and antibiotic resistance in nature." *Environ Microbiol*, **11**: 2970.

[18] Thomas, P., H. Mi, and S. Lewis (**Feb 2007**). "Ontology annotation: mapping genomic regions to biological function." *Curr Opin Chem Biol*, **11**: 4.

[19] Mi, H., J. Vandergriff, M. Campbell, *et al.* (**Sep 2003**). "Assessment of genome-wide protein function classification for *Drosophila melanogaster*." *Genome Res*, **13**: 2118.

[20] Engelhardt, B., M. Jordan, K. Muratore, and S. Brenner (**Oct 2005**). "Protein molecular function prediction by Bayesian phylogenomics". *PLoS Comput Biol*, **1**(5): e45.

[21] Horner, D., W. Pirovano, and G. Pesole (**Jan 2008**). "Correlated substitution analysis and the prediction of amino acid structural contacts." *Brief Bioinform*, **9**: 46.

[22] Moreland, J. L., A. Gramada, O. V. Buzko, Q. Zhang, and P. E. Bourne (**Feb 2005**). "The molecular biology toolkit (mbt): a modular platform for developing molecular visualization applications." *BMC Bioinformatics*, **6**: 21.

[23] Xu, W., S. C. Harrison, and M. J. Eck (**Feb 1997**). "Three-dimensional structure of the tyrosine kinase c-src." *Nature*, **385**: 595.

[24] Felsenstein, J. (**2003**). *Inferring Phylogenies*. Sinauer.

[25] Lemey, P., M. Salemi, and A.-M. Vandamme (**2009**). *The Phylogenetic Handbook*. Cambridge, 2nd edition.

[26] Page, R. and E. Holmes (**1998**). *Molecular Evolution: A phylogenetic approach*. Blackwell Science.

[27] Nars, M. and M. Vihinen (**2001**). "Coevolution of the domains of cytoplasmic tyrosine kinases". *Mol Biol Evol*, **18**: 312.

[28] Maine, E., J. Lissemore, and W. Starmer (**Jun 1995**). "A phylogenetic analysis of vertebrate and invertebrate Notch-related genes". *Mol Phylogenet Evol*, **4**(2): 139.

[29] Thornton, J. and R. DeSalle (**Jun 2000**). "A new method to localize and test the significance of incongruence: detecting domain shuffling in the nuclear receptor superfamily." *Syst Biol*, **49**(2): 183.

[30] te Velthuis, A., J. Admiraal, and C. Bagowski (**Aug 2007**). "Molecular evolution of the MAGUK family in metazoan genomes." *BMC Evol Biol*, **7**: 129.

[31] Chalkia, D., N. Nikolaidis, W. Makalowski, J. Klein, and M. Nei (**Dec 2008**). "Origins and evolution of the formin multigene family that is involved in the formation of actin filaments." *Mol Biol Evol*, **25**: 2717.

[32] Tordai, H., A. Nagy, K. Farkas, L. Banyai, and L. Patthy (**Oct 2005**). "Modules, multidomain proteins and organismic complexity". *FEBS J*, **272**(19): 5064.

[33] Patthy, L. (**Jul 2003**). "Modular assembly of genes and the evolution of new functions". *Genetica*, **118**(2-3): 217.

[34] Aravind, L., V. Dixit, and E. Koonin (**Feb 2001**). "Apoptotic molecular machinery: vastly increased complexity in vertebrates revealed by genome comparisons." *Science*, **291**(5507): 1279.

[35] Ben-Shlomo, I., S. Yu Hsu, R. Rauch, H. Kowalski, and A. Hsueh (**Jun 2003**). "Signaling receptome: a genomic and evolutionary perspective of plasma membrane receptors involved in signal transduction". *Sci STKE*, **2003**(187): RE9.

[36] Stratton, M., P. Campbell, and P. Futreal (**Apr 2009**). "The cancer genome." *Nature*, **458**: 719.

[37] Arena, S., S. Benvenuti, and A. Bardelli (**Sep 2005**). "Genetic analysis of the kinome and phosphatome in cancer." *Cell Mol Life Sci*, **62**: 2092.

[38] Domazet-Loso, T. and D. Tautz (**May 2010**). "Phylostratigraphic tracking of cancer genes suggests a link to the emergence of multicellularity in metazoa." *BMC Biol*, **8**: 66.

[39] Greenman, C., P. Stephens, R. Smith, *et al.* (**Mar 2007**). "Patterns of somatic mutation in human cancer genomes." *Nature*, **446**: 153.

[40] Delsuc, F., H. Brinkmann, and H. Philippe (**May 2005**). "Phylogenomics and the reconstruction of the tree of life." *Nat Rev Genet*, **6**: 361.

[41] Fitch, W. (**Jun 1970**). "Distinguishing homologous from analogous proteins". *Syst Zool*, **19**(2): 99.

[42] Day, W. H. E., D. Johnson, and D. Sankoff (**1986**). "The computational complexity of inferring rooted phylogenies by parsimony". *Math Biosci*, **81**(33): 33.

[43] Fitch, W. (**1971**). "Toward defining the course of evolution: Minimum change for a specified tree topology." *Syst Zool*, **20**: 406.

[44] Huelsenbeck, J. and F. Ronquist (**Aug 2001**). "MRBAYES: Bayesian inference of phylogenetic trees." *Bioinformatics*, **17**(8): 754.

[45] Huelsenbeck, J., F. Ronquist, R. Nielsen, and J. Bollback (**Dec 2001**). "Bayesian inference of phylogeny and its impact on evolutionary biology." *Science*, **294**: 2310.

[46] Strimmer, K. and A. von Haeseler (**2009**). "Genetic distances and nucleotide substitution models – theory". In *The Phylogenetic Handbook: A Practical Approach to Phylogenetic Analysis and Hypothesis Testing*, edited by P. Lemey, M. Salemi, and A.-M. Vandamme, chapter 4, pp. 111 – 125. Cambridge University Press, Cambridge, UK.

[47] Jukes, T. and C. Cantor (**1969**). *Evolution of protein molecules*. Academic Press.

[48] Posada, D. (**2009**). "Selecting models of evolution – theory". In *The Phylogenetic Handbook: A Practical Approach to Phylogenetic Analysis and Hypothesis Testing*, edited by

P. Lemey, M. Salemi, and A.-M. Vandamme, chapter 10, pp. 345 – 354. Cambridge University Press, Cambridge, UK.

[49] Abascal, F., R. Zardoya, and D. Posada (**May 2005**). "ProtTest: selection of best-fit models of protein evolution." *Bioinformatics*, **21**: 2104.

[50] Keane, T. M., C. J. Creevey, M. M. Pentony, T. J. Naughton, and J. O. Mclnerney (**Mar 2006**). "Assessment of methods for amino acid matrix selection and their use on empirical data shows that ad hoc assumptions for choice of matrix are not justified." *BMC Evol Biol*, **6**: 29.

[51] Saitou, N. and M. Nei (**1987**). "The neighbor-joining method: A new method for reconstructing phylogenetic trees". *Molecular Biology and Evolution*, **4**: 406.

[52] Wetlaufer, D. (**Mar 1973**). "Nucleation, rapid folding, and globular intrachain regions in proteins." *Proc Natl Acad Sci U S A*, **70**: 697.

[53] Gilbert, W. (**Feb 1978**). "Why genes in pieces?" *Nature*, **271**(5645): 501.

[54] Allen, W., C. Smith, and P. Porter (**Jul 1973**). "Localization of intracellular immunoglobulin a in porcine intestinal mucosa using enzyme-labelled antibody. an ultrastructural study." *Immunology*, **25**: 55.

[55] Edelman, G. (**May 1973**). "Antibody structure and molecular immunology." *Science*, **180**: 830.

[56] Rossmann, M., D. Moras, and K. Olsen (**Jul 1974**). "Chemical and biological evolution of nucleotide-binding protein." *Nature*, **250**: 194.

[57] Cheng, J., M. Sweredoski, and P. Baldi (**2006**). "DOMpro: Protein domain prediction using profiles, secondary structure, relative solvent accessibility, and recursive neural networks". *Data Mining and Knowledge Discovery*, **13**(1): 1.

[58] Holm, L. and C. Sander (**Sep 1994**). "The FSSP database of structurally aligned protein fold families." *Nucleic Acids Res*, **22**: 3600.

[59] Swindells, M. (**Jan 1995**). "A procedure for detecting structural domains in proteins." *Protein Sci*, **4**: 103.

[60] Finn, R., J. Mistry, J. Tate, *et al.* (**Jan 2010**). "The Pfam protein families database." *Nucleic Acids Res*, **38**: D211.

[61] Bateman, A., E. Birney, R. Durbin, *et al.* (**Jan 1999**). "Pfam 3.1: 1313 multiple alignments and profile HMMs match the majority of proteins." *Nucleic Acids Res*, **27**: 260.

[62] Bateman, A., E. Birney, L. Cerruti, *et al.* (**Jan 2002**). "The Pfam protein families database." *Nucleic Acids Res*, **30**(1): 276.

[63] Schultz, J., F. Milpetz, P. Bork, and C. Ponting (**May 1998**). "SMART, a simple modular architecture research tool: identification of signaling domains." *PNAS*, **95**: 5857.

[64] Schultz, J., R. Copley, T. Doerks, C. Ponting, and P. Bork (**Jan 2000**). "SMART: a web-based tool for the study of genetically mobile domains." *Nucleic Acids Res*, **28**: 231.

[65] Murzin, A., S. Brenner, T. Hubbard, and C. Chothia (**Apr 1995**). "SCOP: a structural classification of proteins database for the investigation of sequences and structures." *J Mol Biol*, **247**(4): 536.

[66] Apweiler, R., T. Attwood, A. Bairoch, *et al.* (**Jan 2001**). "The InterPro database, an integrated documentation resource for protein families, domains and functional sites". *Nucleic Acids Res*, **29**(1): 37.

[67] Marchler-Bauer, A., J. Anderson, P. Cherukuri, *et al.* (**Jan 2005**). "CDD: a Conserved Domain Database for protein classification." *Nucleic Acids Res*, **33**(Database issue): D192.

[68] Stormo, G. (**1990**). "Consensus patterns in DNA". In *Methods in Enzymology*, volume 183, pp. 211 – 221. Academic Press.

[69] Krogh, A., M. Brown, I. Mian, K. Sjolander, and D. Haussler (**Feb 1994**). "Hidden markov models in computational biology. applications to protein modeling". *J Mol Biol*, **235**(5): 1501.

[70] Schultz, J., R. Copley, T. Doerks, C. Ponting, and P. Bork (**Jan 2000**). "Smart: a web-based tool for the study of genetically mobile domains". *Nucleic Acids Res*, **28**(1): 231.

[71] Schultz, J., F. Milpetz, P. Bork, and C. Ponting (**May 1998**). "Smart, a simple modular architecture research tool: identification of signaling domains". *Proc Natl Acad Sci U S A*, **95**(11): 5857.

[72] Marchler-Bauer, A., J. Anderson, P. Cherukuri, *et al.* (**Jan 2005**). "CDD: a Conserved Domain Database for protein classification." *Nucleic Acids Res*, **33**: D192.

[73] Marchler-Bauer, A., A. R. Panchenko, B. A. Shoemaker, *et al.* (**Jan 2002**). "CDD: a database of conserved domain alignments with links to domain three-dimensional structure." *Nucleic Acids Res*, **30**: 281.

[74] Attwood, T. (**Sep 2002**). "The PRINTS database: a resource for identification of protein families." *Brief Bioinform*, **3**: 252.

[75] Pearl, F., D. Lee, J. Bray, *et al.* (**Jan 2000**). "Assigning genomic sequences to cath". *Nucleic Acids Res*, **28**(1): 277.

[76] Pearl, F., C. Bennett, J. Bray, *et al.* (**Jan 2003**). "The CATH database: an extended protein family resource for structural and functional genomics". *Nucleic Acids Res*, **31**(1): 452.

[77] Tatusov, R., N. Fedorova, J. Jackson, *et al.* (**Sep 2003**). "The COG database: an updated version includes eukaryotes". *BMC Bioinformatics*, **4**(1): 41.

[78] Darren Natale, T. T. U. S. B. R. B. K. M. G. N. F., Igor Garkavtsev and E. Koonin (**Jan 2001**). "The COG database: new developments in phylogenetic classification of proteins from complete genomes". *Nucleic Acids Res*, **29**(1): 22.

[79] Mulder, N., R. Apweiler, T. Attwood, *et al.* (**Jan 2007**). "New developments in the InterPro database." *Nucleic Acids Res*, **35**: D224.

[80] Han, J., S. Batey, A. Nickson, S. Teichmann, and J. Clarke (**Apr 2007**). "The folding and evolution of multidomain proteins." *Nat Rev Mol Cell Biol*, **8**: 319.

[81] Miyata, T. and H. Suga (**Nov 2001**). "Divergence pattern of animal gene families and relationship with the Cambrian explosion." *Bioessays*, **23**(11): 1018.

[82] Sayah, D., E. Sokolskaja, L. Berthoux, and J. Luban (**Jul 2004**). "Cyclophilin A retrotransposition into TRIM5 explains owl monkey resistance to HIV-1." *Nature*, **430**(6999): 569.

[83] Long, M., E. Betran, K. Thornton, and W. Wang (**Nov 2003**). "The origin of new genes: glimpses from the young and old." *Nat Rev Genet*, **4**(11): 865.

[84] Long, M. and K. Thornton (**Aug 2001**). "Gene duplication and evolution". *Science*, **293**(5535).

[85] Vinckenbosch, N., I. Dupanloup, and H. Kaessmann (**Feb 2006**). "Evolutionary fate of retroposed gene copies in the human genome". *PNAS*, **103**(9): 3220.

[86] Finnegan, D. (**Apr 1989**). "Eukaryotic transposable elements and genome evolution". *Trends Genet*, **5**(4): 103.

[87] Begun, D. (**Feb 1997**). "Origin and evolution of a new gene descended from *alcohol dehydrogenase* in *Drosophila*". *Genetics*, **145**(2): 375.

[88] Jones, C., A. Custer, and D. Begun (**May 2005**). "Origin and evolution of a chimeric fusion gene in *Drosophila subobscura*, *D. madeirensis* and *D. guanche*". *Genetics*, **170**(1): 207.

[89] Finta, C. and P. Zaphiropoulos (**Dec 2000**). "The human cytochrome P450 3A locus. Gene evolution by capture of downstream exons". *Gene*, **260**(1-2): 13.

[90] Hurles, M. (**Jul 2004**). "Gene duplication: the genomic trade in spare parts". *PLoS Biol*, **2**(7): E206.

[91] Gilbert, W. (**1987**). "The exon theory of genes". *Cold Spring Harb Symp Quant Biol*, **52**: 901.

[92] Eichler, E. (**Nov 2001**). "Recent duplication, domain accretion and the dynamic mutation of the human genome". *Trends Genet*, **17**(11): 661.

[93] Emanuel, B. and T. Shaikh (**Oct 2001**). "Segmental duplications: an 'expanding' role in genomic instability and disease". *Nat Rev Genet*, **2**(10): 791.

[94] Kaessmann, H., S. Zollner, A. Nekrutenko, and W. Li (**Nov 2002**). "Signatures of domain shuffling in the human genome". *Genome Res*, **12**(11): 1642.

[95] Mighell, A., N. Smith, P. Robinson, and A. Markham (**Feb 2000**). "Vertebrate pseudogenes". *FEBS Lett*, **468**(2-3): 109.

[96] Buljan, M., A. Frankish, and A. Bateman (**Jul 2010**). "Quantifying the mechanisms of domain gain in animal proteins." *Genome Biol*, **11**: R74.

[97] Babushok, D., E. Ostertag, and H. Kazazian (**Mar 2007**). "Current topics in genome evolution: molecular mechanisms of new gene formation." *Cell Mol Life Sci*, **64**: 542.

[98] Ciccarelli, F., C. Mering, M. Suyama, *et al*. (**Mar 2005**). "Complex genomic rearrangements lead to novel primate gene function." *Genome Res*, **15**(3): 343.

[99] Betrán, E., K. Thornton, and M. Long (**Dec 2002**). "Retroposed new genes out of the X in *Drosophila*". *Genome Res*, **12**(12): 1854.

[100] van Rijk, A. and H. Bloemendal (**Jul 2003**). "Molecular mechanisms of exon shuffling: illegitimate recombination." *Genetica*, **118**: 245.

[101] Lorenc, A. and W. Makalowski (**Jul 2003**). "Transposable elements and vertebrate protein diversity". *Genetica*, **118**(2-3): 183.

[102] Liu, M., S. Wu, H. Walch, and A. Grigoriev (**Aug 2005**). "Exon-domain correlation and its corollaries". *Bioinformatics*, **21**(15): 3213.

[103] Schmidt, E. and C. Davies (**Mar 2007**). "The origins of polypeptide domains." *Bioessays*, **29**: 262.

[104] Wang, M. and G. Caetano-Anollés (**Jan 2009**). "The evolutionary mechanics of domain organization in proteomes and the rise of modularity in the protein world." *Structure*, **17**: 66.

[105] Collinet, B., M. Herve, F. Pecorari, *et al*. (**Jun 2000**). "Functionally accepted insertions of proteins within protein domains." *J Biol Chem*, **275**: 17428.

[106] Moore, A. (**May 2008**). "Science teaching must evolve." *Nature*, **453**: 31.

[107] Jun, J., P. Ryvkin, E. Hemphill, and C. Nelson (**Sep 2009**). "Duplication mechanism and disruptions in flanking regions determine the fate of mammalian gene duplicates." *J Comput Biol*, **16**(9): 1253.

[108] Jun, J., P. Ryvkin, E. Hemphill, I. Mandoiu, and C. Nelson (**Oct 2009**). "The birth of new genes by RNA- and DNA-mediated duplication during mammalian evolution." *J Comput Biol*, **16**(10): 1429.

[109] Schacherer, J., Y. Tourrette, J. Souciet, S. Potier, and J. De Montigny (**Jul 2004**). "Recovery of a function involving gene duplication by retroposition in *Saccharomyces cerevisiae*." *Genome Res*, **14**: 1291.

[110] Schacherer, J., J. de Montigny, A. Welcker, J. Souciet, and S. Potier (**Nov 2005**). "Duplication processes in *Saccharomyces cerevisiae* haploid strains." *Nucleic Acids Res*, **33**: 6319.

[111] Gilbert, W. (**May 1985**). "Genes-in-pieces revisited." *Science*, **228**(4701): 823.

[112] Liu, M. and A. Grigoriev (**Sep 2004**). "Protein domains correlate strongly with exons in multiple eukaryotic genomes–evidence of exon shuffling?" *Trends Genet*, **20**(9): 399.

[113] Patthy, L. (**Apr 1987**). "Intron-dependent evolution: preferred types of exons and introns". *FEBS Lett*, **214**(1): 1.

[114] de Souza., S., M. Long, and W. Gilbert (**Jun 1996**). "Introns and gene evolution". *Genes Cells*, **1**(6): 493.

[115] Patthy, L. (**Sep 1999**). "Genome evolution and the evolution of exon-shuffling–a review". *Gene*, **238**(1): 103.

[116] Vogel, C., S. Teichmann, and J. Pereira-Leal (**Feb 2005**). "The relationship between domain duplication and recombination". *J Mol Biol*, **346**(1): 355.

[117] Karev, G., Y. Wolf, A. Rzhetsky, F. Berezovskaya, and E. Koonin (**2002**). "Birth and death of protein domains: a simple model of evolution explains power law behavior". *BMC Evol Biol*, **2**(1): 18.

[118] Vogel, C., M. Bashton, N. Kerrison, C. Chothia, and S. Teichmann (**Apr 2004**). "Structure, function and evolution of multidomain proteins". *Curr Opin Struct Biol*, **14**(2): 208.

[119] Vogel, C., C. Berzuini, M. Bashton, J. Gough, and S. Teichmann (**Feb 2004**). "Supra-domains: evolutionary units larger than single protein domains". *J Mol Biol*, **336**(3): 809.

[120] Apic, G., W. Huber, and S. Teichmann (**2003**). "Multi-domain protein families and domain pairs: comparison with known structures and a random model of domain recombination". *J Struct Funct Genomics*, **4**(2-3): 67.

[121] Wuchty, S. (**2001**). "Scale-free behavior in protein domain networks". *Mol Biol Evol*, **18**: 1694.

[122] Koonin, E., Y. Wolf, and G. Karev (**2002**). "The structure of protein universe and genome evolution". *Nature*, **420**: 218.

[123] Ye, Y. and A. Godzik (**Mar 2004**). "Comparative analysis of protein domain organization". *Genome Res*, **14**(3): 343.

[124] Qian, J., N. Luscombe, and M. Gerstein (**2001**). "Protein family and fold occurrence in genomes: powerlaw behaviour and evolutionary model". *J Mol Biol*, **313**: 679.

[125] Apic, G., J. Gough, and S. Teichmann (**Jul 2001**). "Domain combinations in archaeal, eubacterial and eukaryotic proteomes". *J Mol Biol*, **310**(2): 311.

[126] Huynen, M. and P. Bork (**May 1998**). "Measuring genome evolution". *PNAS*, **95**(11): 5849.

[127] Weiner, J. and E. Bornberg-Bauer (**Apr 2006**). "Evolution of circular permutations in multi-domain proteins". *Mol Biol Evol*, **23**(4): 734.

[128] Bashton, M. and C. Chothia (**2002**). "The geometry of domain combination in proteins". *J Mol Biol*, **315**: 927.

[129] Kummerfeld, S. and S. Teichman. (**2005**). "Relative rates of gene fusion and fission in mutli-domain proteins". *Trends in Genetics*, **21**: 25.

[130] Fong, J., L. Geer, A. Panchenko, and S. Bryant (**Feb 2007**). "Modeling the evolution of protein domain architectures using maximum parsimony". *J Mol Biol*, **366**: 307.

[131] Weiner, J., F. Beaussart, and E. Bornberg-Bauer (**May 2006**). "Domain deletions and substitutions in the modular protein evolution". *FEBS J*, **273**(9): 2037.

[132] Snel, B., P. Bork, and M. Huynen (**2000**). "Genome evolution. Gene fusion versus gene fission." *Trends Genet*, **16**: 9.

[133] Pasek, S., J. Risler, and P. Brezellec (**Jun 2006**). "Gene fusion/fission is a major contributor to evolution of multi-domain bacterial proteins". *Bioinformatics*, **22**(12): 1418.

[134] Bjorklund, A., D. Ekman, S. Light, J. Frey-Skott, and A. Elofsson (**Nov 2005**). "Domain rearrangements in protein evolution". *J Mol Biol*, **353**(4): 911.

[135] Yanai, I., Y. Wolf, and E. Koonin (**2002**). "Evolution of gene fusions: horizontal transfer versus independent events". *Genome Biol*, **3**(5): research0024.

[136] Kummerfeld, S. and S. Teichmann (**Jan 2009**). "Protein domain organisation: adding order." *BMC Bioinformatics*, **10**: 39.

[137] Liu, Q., J. Huang, H. Liu, *et al.* (**Mar 2009**). "Analyses of domains and domain fusions in human proto-oncogenes." *BMC Bioinformatics*, **10**: 88.

[138] Behzadi, B. and M. Vingron (**2006**). "Reconstructing domain compositions of ancestral multi-domain proteins". In *Comparative Genomics*, edited by G. Bourque and M. El-Mabrouk, *LNCS*, volume 4205, pp. 1–10. Springer.

[139] Forslund, K., A. Henricson, V. Hollich, and E. Sonnhammer (**Feb 2008**). "Domain tree-based analysis of protein architecture evolution." *Mol Biol Evol*, **25**: 254.

[140] Fong, J., L. Geer, A. Panchenko, and S. Bryant (**Feb 2007**). "Modeling the evolution of protein domain architectures using maximum parsimony". *J Mol Biol*, **366**(1): 307.

[141] Snel, B., P. Bork, and M. Huynen (**2002**). "Genomes in flux: the evolution of archaeal and proteobacterial gene content." *Genome Res*, **12**: 17.

[142] Gough, J. (**Apr 2005**). "Convergent evolution of domain architectures (is rare)". *Bioinformatics*, **21**(8): 1464.

[143] Przytycka, T., G. Davis, N. Song, and D. Durand (**2006**). "Graph theoretical insights into evolution of multidomain proteins". *J Comput Biol*, **13**(2): 351.

[144] Wiedenhoeft, J., R. Krause, and O. Eulenstein (**2010**). "Inferring evolutionary scenarios for protein domain compositions". In *Bioinformatics Research and Applications*, edited by M. Borodovsky, J. Gogarten, T. Przytycka, and S. Rajasekaran, *LNCS*, volume 6053, pp. 179–190. Springer.

[145] Qian, J., N. Luscombe, and M. Gerstein (**Nov 2001**). "Protein family and fold occurrence in genomes: power-law behaviour and evolutionary model." *J Mol Biol*, **313**: 673.

[146] Karev, G., Y. Wolf, and E. Koonin (**Oct 2003**). "Simple stochastic birth and death models of genome evolution: was there enough time for us to evolve?" *Bioinformatics*, **19**: 1889.

[147] Karev, G., Y. Wolf, F. Berezovskaya, and E. Koonin (**Sep 2004**). "Gene family evolution: an in-depth theoretical and simulation analysis of non-linear birth-death-innovation models". *BMC Evol Biol*, **4**: 32.

[148] Novozhilov, A., G. Karev, and E. Koonin (**Mar 2006**). "Biological applications of the theory of birth-and-death processes." *Brief Bioinform*, **7**(1): 70.

[149] Rzhetsky, A. and S. Gomez (**Oct 2001**). "Birth of scale-free molecular networks and the number of distinct DNA and protein domains per genome". *Bioinformatics*, **17**(10): 988.

[150] Przytycka, T. and Y. Yu (**Oct 2004**). "Scale-free networks versus evolutionary drift." *Comput Biol Chem*, **28**: 257.

[151] Song, N., J. Joseph, G. Davis, and D. Durand (**Apr 2008**). "Sequence similarity network reveals common ancestry of multidomain proteins." *PLoS Comput Biol*, **4**: e1000063.

[152] Przytycka, T., G. Davis, N. Song, and D. Durand (**2005**). "Graph theoretical insights into evolution of multidomain proteins". In *RECOMB 2005: Proceedings of the Ninth International Conference on Research in Computational Biology*, edited by S. M. et al., LNCS 3500, pp. 311–325. Springer.

[153] Marcotte, E., M. Pellegrini, H. Ng, *et al.* (**Jul 1999**). "Detecting protein function and protein-protein interactions from genome sequences". *Science*, **285**(5428): 751.

[154] Basu, M., E. Poliakov, and I. Rogozin (**Jan 2009**). "Domain mobility in proteins: functional and evolutionary implications." *Brief Bioinform*, **10**: 205.

[155] Basu, M., L. Carmel, I. Rogozin, and E. Koonin (**Jan 2008**). "Evolution of protein domain promiscuity in eukaryotes." *Genome Res*.

[156] Moore, A., A. Björklund, D. Ekman, E. Bornberg-Bauer, and A. Elofsson (**Sep 2008**). "Arrangements in the modular evolution of proteins." *Trends Biochem Sci*, **33**: 444.

[157] Chothia, C. and J. Gough (**Apr 2009**). "Genomic and structural aspects of protein evolution." *Biochem J*, **419**: 15.

[158] Bornberg-Bauer, E., F. Beaussart, S. Kummerfeld, S. Teichmann, and J. Weiner (**Feb 2005**). "The evolution of domain arrangements in proteins and interaction networks". *Cell Mol Life Sci*, **62**(4): 435.

[159] Karev, G., F. Berezovskaya, and E. Koonin (**Nov 2005**). "Modeling genome evolution with a diffusion approximation of a birth-and-death process." *Bioinformatics*, **21**(Suppl 3): iii12.

[160] Pastor-Satorras, R., E. Smith, and R. Solé (**May 2003**). "Evolving protein interaction networks through gene duplication." *J Theor Biol*, **222**(2): 199.

[161] Wagner, A. (**Mar 2003**). "How the global structure of protein interaction networks evolves." *Proc Biol Sci*, **270**(1514): 457.

[162] Apic, G., J. Gough, and S. Teichmann (**2001**). "An insight into domain combinations". *Bioinformatics*, **17 Suppl 1**: 83.

[163] Liu, Y., M. Gerstein, and D. Engelman (**Mar 2004**). "Transmembrane protein domains rarely use covalent domain recombination as an evolutionary mechanism." *PNAS*, **101**: 3495.

[164] Fliess, A., B. Motro, and R. Unger (**Aug 2002**). "Swaps in protein sequences." *Proteins*, **48**: 377.

[165] Ekman, D., A. Björklund, and A. Elofsson (**Oct 2007**). "Quantification of the elevated rate of domain rearrangements in metazoa." *J Mol Biol*, **372**: 1337.

[166] Consortium, I. H. G. S. (**Oct 2004**). "Finishing the euchromatic sequence of the human genome". *Nature*, **431**(7011): 931.

[167] Aravind, L., R. Mazumder, S. Vasudevan, and E. Koonin (**Jun 2002**). "Trends in protein evolution inferred from sequence and structure analysis." *Curr Opin Struct Biol*, **12**: 392.

[168] Pawson, T. and P. Nash (**Apr 2003**). "Assembly of cell regulatory systems through protein interaction domains." *Science*, **300**: 445.

[169] Kawashima, T., S. Kawashima, C. Tanaka, *et al.* (**Aug 2009**). "Domain shuffling and the evolution of vertebrates." *Genome Res*, **19**: 1393.

[170] Bjorklund, A., D. Ekman, and A. Elofsson (**Aug 2006**). "Expansion of protein domain repeats". *PLoS Comput Biol*, **2**(8): e114.

[171] Björklund, A., S. Light, R. Sagit, and A. Elofsson (**Sep 2010**). "Nebulin: a study of protein repeat evolution." *J Mol Biol*, **402**: 38.

[172] Wright, C., S. Teichmann, J. Clarke, and C. Dobson (**Dec 2005**). "The importance of sequence diversity in the aggregation and evolution of proteins." *Nature*, **438**: 878.

[173] Song, N., R. Sedgewick, and D. Durand (**May 2007**). "Domain architecture comparison for multidomain homology identification". *J Comput Biol*, **14**(4): 496.

[174] Weiner, J., A. Moore, and E. Bornberg-Bauer (**Oct 2008**). "Just how versatile are domains?" *BMC Evol Biol*, **8**: 285.

[175] Huynen, M. and E. Nimwegen (**1998**). "The frequency distribution of gene family sizes in complete genomes". *Mol Biol Evol*, **15**(5): 583.

[176] Barabasi, A. (**Oct 1999**). "Emergence of scaling in random networks". *Science*, **286**: 509.

[177] Lynch, M. and J. Conery (**Nov 2000**). "The evolutionary fate and consequences of duplicate genes". *Science*, **290**(5494): 1151.

[178] Grunt, M., V. Zárský, and F. Cvrcková (**Apr 2008**). "Roots of angiosperm formins: the evolutionary history of plant FH2 domain-containing proteins." *BMC Evol Biol*, **8**: 115.

[179] Alves, R., E. Vilaprinyo, A. Sorribas, and E. Herrero (**Mar 2009**). "Evolution based on domain combinations: the case of glutaredoxins." *BMC Evol Biol*, **9**: 66.

[180] Middendorf, M., E. Ziv, and C. Wiggins (**Mar 2005**). "Inferring network mechanisms: the *Drosophila melanogaster protein interaction network*". *PNAS*, **102**(9): 3192.

[181] Fitch, W. (**May 2000**). "Homology: a personal view on some of the problems". *Trends Genet*, **16**(5): 227.

[182] Joseph, J. and D. Durand (**Jun 2009**). "Family classification without domain chaining." *Bioinformatics*, **25**: i45.

[183] Huson, D. H., R. Rupp, and C. Scornavacca (**2010**). *Phylogenetic Networks: Concepts, Algorithms and Applications*. Cambridge University Press, Cambridge, UK.

[184] Page, R. and M. Charleston (**1997**). "From gene to organismal phylogeny: Reconciled trees and the gene tree/species tree problem." *Molecular Phylogenetics and Evolution*, **7**: 231.

[185] Ronquist, F. (**2002**). *Parsimony analysis of coevolving species associations*, chapter 2, pp. 23–64. University of Chicago Press.

[186] —— (**1997**). "Phylogenetic approaches in coevolution and biogeography". *Zool Scr*, **26**(4): 313.

[187] Goodman, M., J. Czelusniak, G. Moore, A. Romero-Herrera, and G. Matsuda (**1979**). "Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences". *Syst Zool*, **28**: 132.

[188] Ronquist, F. (**2002**). "Parsimony analysis of coevolving species associations". In *Tangled Trees: Phylogeny, Cospeciation and Coevolution*, edited by R. D. M. Page, chapter 2, pp. 22–64. Univ of Chicago Press.

[189] Ma, B., M. Li, and L. Zhang (**2000**). "From gene trees to species trees". *SIAM J Comput*, **30**(3): 729.

[190] Bansal, M. S. and R. Shamir (**May/Jun 2011**). "A note on the fixed parameter tractability of the gene-duplication problem." *IEEE/ACM Trans Comput Biol Bioinform*, **8**: 848.

[191] Górecki, P. and J. Tiuryn (**Jan 2007**). "Inferring phylogeny from whole genomes." *Bioinformatics*, **23**: e116.

[192] Chaudhary, R., M. Bansal, A. Wehe, D. Fernández-Baca, and O. Eulenstein (**Nov 2010**). "iGTP: a software package for large-scale gene tree parsimony analysis." *BMC Bioinformatics*, **11**: 574.

[193] Allman, E., J. Degnan, and J. Rhodes (**Jul 2010**). "Identifying the rooted species tree from the distribution of unrooted gene trees under the coalescent." *J Math Biol*, **DOI 10.1007/s00285-010-0355-7**.

[194] Liu, L. and D. Pearl (**Jun 2007**). "Species trees from gene trees: reconstructing bayesian posterior distributions of a species phylogeny using estimated gene tree distributions". *Syst Biol*, **56**(3): 504.

[195] Liu, L., L. Yu, L. Kubatko, D. Pearl, and S. Edwards (**Oct 2009**). "Coalescent methods for estimating phylogenetic trees." *Mol Phylogenet Evol*, **53**: 320.

[196] Huson, D., T. Klopper, P. Lockhart, and M. Steel (**2005**). "Reconstruction of reticulate networks from gene trees". In *RECOMB 2005: Proceedings of the Ninth International Conference on Research in Computational Biology*, edited by S. M. et al., *LNCS 3500*, volume 3500, pp. 233–249. Springer.

[197] Mossel, E. and S. Roch (**Jan/Mar 2010**). "Incomplete lineage sorting: consistent phylogeny estimation from multiple loci." *IEEE/ACM Trans Comput Biol Bioinform*, **7**: 166.

[198] Page, R. (**Mar 1994**). "Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas". *Syst Biol*, **43**(1): 58.

[199] Guigo, R., I. Muchnik, and T. Smith (**Oct 1996**). "Reconstruction of ancient molecular phylogeny". *Molecular Phylogenetics and Evolution*, **6**(2): 189.

[200] Hallett, M. and J. Lagergren (**2000**). "New algorithms for the duplication-loss model". In *RECOMB 2000: Proceedings of the Fourth International Conference on Research in Computational Biology*, pp. 138–146. ACM, New York, NY, USA.

[201] Mirkin, B., I. Muchnik, and T. Smith (**1995**). "A biologically consistent model for comparing molecular phylogenies". *J Comput Biol*, **2**: 493.

[202] Stege, U. (**1999**). "Gene trees and species trees: The gene-duplication problem is fixed-parameter tractable". In *Proceedings of the 6th International Workshop on Algorithms and Data Structures (WADS)*, LNCS 1663, pp. 288–293.

[203] Zhang, L. (**Summer 1997**). "On a Mirkin–Muchnik-Smith conjecture for comparing molecular phylogenies". *J Comput Biol*, **4**(2): 177.

[204] Eulenstein, O., B. Mirkin, and M. Vingron (**Spring 1998**). "Duplication-based measures of difference between gene and species trees". *J Comput Biol*, **5**(1): 135.

[205] C. Chauve, J. and N. El-Mabrouk (**2007**). "Inferring a duplication, speciation and loss history from a gene tree". In *RECOMB-CG 2007: Proceedings of the International Workshop on Comparative Genomics*, edited by G. Tesler and D. Durand, LNCS 4751, pp. 45–57. Springer.

[206] Page, R. (**1998**). "GeneTree: comparing gene and species phylogenies using reconciled trees." *Bioinformatics*, **14**(9): 819.

[207] Zmasek, C. and S. Eddy (**Sep 2001**). "A simple algorithm to infer gene duplication and speciation events on a gene tree." *Bioinformatics*, **17**(9): 821.

[208] —— (**Apr 2001**). "ATV: display and manipulation of annotated phylogenetic trees." *Bioinformatics*, **17**(4): 383.

[209] Berglund-Sonnhammer, A., P. Steffansson, M. Betts, and D. Liberles (**Aug 2006**). "Optimal gene trees from sequences and species trees using a soft interpretation of parsimony". *J Mol Evol*, **63**(2): 240.

[210] Sennblad, B., E. Schreil, A. Berglund Sonnhammer, J. Lagergren, and L. Arvestad (**May 2007**). "primetv: a viewer for reconciled trees." *BMC Bioinformatics*, **8**: 148.

[211] Dufayard, J., L. Duret, S. Penel, *et al.* (**Jun 2005**). "Tree pattern matching in phylogenetic trees: automatic search for orthologs or paralogs in homologous gene sequence databases." *Bioinformatics*, **21**(11): 2596.

[212] Roth, C., M. Betts, P. Steffansson, G. Sælensminde, and D. Liberles (**2005**). "The adaptive evolution database (TAED): a phylogeny based tool for comparative genomics". *Nucleic Acids Research*, **33**: D495.

[213] Durand, D. and R. Hoberman (**Mar 2006**). "Diagnosing duplications: can it be done?" *Trends Genet*, **22**(3): 156.

[214] Page, R. (**1994**). "Parallel phylogenies: Reconstructing the history of host-parasite assemblages". *Cladistics*, **10**: 155.

[215] Arvestad, L., A. Berglund, J. Lagergren, and B. Sennblad (**2003**). "Bayesian gene/species tree reconciliation and orthology analysis using MCMC." *Bioinformatics*, **19 Suppl 1**: i7.

[216] —— (**2004**). "Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution". In *RECOMB 2004: Proceedings of the Eighth Annual International Conference on Computational Molecular Biology*, pp. 326–335. ACM.

[217] Arvestad, L., J. Lagergren, and B. Sennblad (**2009**). "The gene evolution model and computing its associated probabilities." *J ACM*, **56**(2): 1.

[218] Akerborg, O., B. Sennblad, L. Arvestad, and J. Lagergren (**Apr 2009**). "Simultaneous Bayesian gene tree reconstruction and reconciliation analysis." *PNAS*, **106**: 5714.

[219] De Bie, T., N. Cristianini, J. Demuth, and M. Hahn (**May 2006**). "CAFE: a computational tool for the study of gene family evolution." *Bioinformatics*, **22**: 1269.

[220] Górecki, P., G. Burleigh, and O. Eulenstein (**Feb 2011**). "Maximum likelihood models and algorithms for gene tree evolution with duplications and losses." *BMC Bioinformatics*, **12 Suppl 1**: S15.

[221] Ane, C., B. Larget, D. Baum, S. Smith, and A. Rokas (**Feb 2007**). "Bayesian estimation of concordance among gene trees". *Mol Biol Evol*, **24**(2): 412.

[222] Kubatko, L. S. (**Oct 2009**). "Identifying hybridization events in the presence of coalescence via model selection." *Syst Biol*, **58**: 478.

[223] Meng, C. and L. Kubatko (**Feb 2009**). "Detecting hybrid speciation in the presence of incomplete lineage sorting using gene tree incongruence: a model." *Theor Popul Biol*, **75**: 35.

[224] Hallett, M. and J. Lagergren (**2000**). "New algorithms for the duplication-loss model". In *RECOMB 2000*, pp. 138–146. ACM, New York, NY, USA.

[225] Gorecki, P. and J. Tiuryn (**2006**). "DLS-trees: A model of evolutionary scenarios". *Theor Comp Sci*, **359**: 378.

[226] Han, M. V. and C. M. Zmasek (**Oct 2009**). "phyloXML: XML for evolutionary biology and comparative genomics." *BMC Bioinformatics*, **10**: 356.

[227] Chauve, C., J. Doyon, and N. El-Mabrouk (**Oct 2008**). "Gene family evolution by duplication, speciation, and loss." *J Comput Biol*, **15**: 1043.

[228] Chauve, C. and N. El-Mabrouk (**2009**). "New perspectives on gene family evolution: Losses in reconciliation and a link with supertrees". In *RECOMB 2009: Proceedings of the Thirteenth International Conference on Research in Computational Biology*, edited by S. Batzoglou, *LNCS 5541*, volume 5541, pp. 46–58. Springer.

[229] Hallett, M. and J. Lagergren (**2001**). "Efficient algorithms for lateral gene transfer problems". In *RECOMB '01: Proceedings of the Fifth Annual International Conference on Computational Biology*, pp. 149–156. ACM Press.

[230] Nakhleh, L., T. Warnow, C. Linder, and K. St John (**Jul/Aug 2005**). "Reconstructing reticulate evolution in species-theory and practice." *J Comput Biol*, **12**: 796.

[231] Nakhleh, L., D. Ruths, and W. Li-San (**2005**). "RIATA-HGT: A fast and accurate heuristic for reconstructing horizontal gene transfer". In *Proc. 11th International Computing and Combinatorics Conference (COCOON)*, edited by L. Wang, LNCS 3595, pp. 84–93. Springer, Berlin Heidelberg.

[232] Beiko, R. and N. Hamilton (**Feb 2006**). "Phylogenetic identification of lateral genetic transfer events." *BMC Evol Biol*, **6**: 15.

[233] Birin, H., Z. Gal-Or, I. Elias, and T. Tuller (**Mar 2008**). "Inferring horizontal transfers in the presence of rearrangements by the minimum evolution criterion." *Bioinformatics*, **24**: 826.

[234] Hallett, M., J. Lagergren, and A. Tofigh (**2004**). "Simultaneous identification of duplications and lateral transfers". In *RECOMB 2004: Proceedings of the Eigth International Conference on Research in Computational Biology*, pp. 347–356. ACM, New York, NY, USA.

[235] Tofigh, A., M. Hallett, and J. Lagergren (**Feb 2010**). "Simultaneous identification of duplications and lateral gene transfers". *TCBB*.

[236] Gorecki, P. (**2004**). "Reconciliation problems for duplication, loss and horizontal gene transfer". In *RECOMB 2004: Proceedings of the Eigth International Conference on Research in Computational Biology*, pp. 316–325. ACM.

[237] Pamilo, P. and M. Nei (**1988**). "Relationships between gene trees and species trees". *Mol Biol Evol*, **5**(5): 568.

[238] Takahata, N. and M. Nei (**Jun 1985**). "Gene genealogy and variance of interpopulational nucleotide differences". *Genetics*, **110**(2): 325.

[239] Maddison, W. (**1997**). "Gene trees in species trees". *Syst Biol*, **46**(3): 523.

[240] Tajima, F. (**Oct 1983**). "Evolutionary relationship of dna sequences in finite populations". *Genetics*, **105**(2): 437.

[241] Hudson, R. (**1990**). "Gene genealogies and the coalescent process". In *Oxford surveys in evolutionary biology*, volume 7, pp. 1–44. Oxford University Press.

[242] Pollard, D., V. Iyer, A. Moses, and M. Eisen (**Oct 2006**). "Widespread discordance of gene trees with species tree in *Drosophila*: evidence for incomplete lineage sorting". *PLoS Genet*, **2**(10): e173.

[243] Maddison, W. and L. Knowles (**Feb 2006**). "Inferring phylogeny despite incomplete lineage sorting". *Syst Biol*, **55**(1): 21.

[244] Maddison, W. and D. Maddison (**2010**). "Mesquite: A modular system for evolutionary analysis".

[245] Lyons-Weiler, J. and M. Milinkovitch (**1997**). "A phylogenetic approach to the problem of differential lineage sorting". *Mol Biol Evol*, **14**(9): 968.

[246] Rosenberg, N. (**Apr 2007**). "Counting coalescent histories." *J Comput Biol*, **14**: 360.

[247] Rosenberg, N. and J. Degnan (**May 2010**). "Coalescent histories for discordant gene trees and species trees." *Theor Popul Biol*, **77**: 145.

[248] Than, C. and L. Nakhleh (**Sep 2009**). "Species tree inference by minimizing deep coalescences." *PLoS Comput Biol*, **5**: e1000501.

[249] Zhang, L. (**Apr 2011**). "From gene trees to species trees ii: Species tree inference by minimizing deep coalescence events." *IEEE/ACM Trans Comput Biol Bioinform*.

[250] Yu, Y., C. Than, J. Degnan, and L. Nakhleh (**Jan 2011**). "Coalescent histories on phylogenetic networks and detection of hybridization despite incomplete lineage sorting." *Syst Biol*.

[251] Ané, C. (**Jan 2011**). "Detecting phylogenetic breakpoints and discordance from genome-wide alignments for species tree reconstruction." *Genome Biol Evol*, **3**: 246.

[252] Vernot, B., M. Stolzer, A. Goldman, and D. Durand (**Oct 2008**). "Reconciliation with non-binary species trees." *J Comput Biol*, **15**: 981.

[253] Jin, G., L. Nakhleh, S. Snir, and T. Tuller (**Nov 2006**). "Maximum likelihood of phylogenetic networks." *Bioinformatics*, **22**(21): 2604.

[254] MacLeod, D., R. Charlebois, F. Doolittle, and E. Bapteste (**Apr 2005**). "Deduction of probable events of lateral gene transfer through comparison of phylogenetic trees by recursive consolidation and rearrangement." *BMC Evol Biol*, **5**(1): 27.

[255] Bansal, M., J. Burleigh, and O. Eulenstein (**Jan 2010**). "Efficient genome-scale phylogenetic analysis under the duplication-loss and deep coalescence cost models." *BMC Bioinformatics*, **11 Suppl 1**: S42.

[256] Cheek, S., H. Zhang, and N. Grishin (**Jul 2002**). "Sequence and structure classification of kinases." *J Mol Biol*, **320**(4): 855.

[257] Scheeff, E. D. and P. E. Bourne (**Oct 2005**). "Structural evolution of the protein kinase-like superfamily". *PLoS Comput Biol*, **1**(5): e49.

[258] Manning, G., D. Whyte, R. Martinez, T. Hunter, and S. Sudarsanam (**Dec 2002**). "The protein kinase complement of the human genome". *Science*, **298**(5600): 1912.

[259] Chen, K., D. Durand, and M. Farach-Colton (**2000**). "Notung: Dating gene duplications using gene family trees". In *RECOMB 2000: Proceedings of the Fourth International Conference on Research in Computational Biology*, pp. 96–106. ACM Press, New York, NY, USA.

[260] Durand, D., B. Halldorsson, and B. Vernot (**2006**). "A hybrid micro-macroevolutionary approach to gene tree reconstruction." *J Comput Biol*, **13**(2): 320.

[261] Yang, X., U. Kalluri, S. Jawdy, *et al.* (**Nov 2008**). "The F-box gene family is expanded in herbaceous annual plants relative to woody perennial plants." *Plant Physiol*, **148**: 1189.

[262] Hanada, K., C. Zou, M. Lehti-Shiu, K. Shinozaki, and S. Shiu (**Oct 2008**). "Importance of lineage-specific expansion of plant tandem duplicates in the adaptive response to environmental stimuli." *Plant Physiol*, **148**: 993.

[263] Somogyi, K., B. Sipos, Z. Pénzes, *et al.* (**Nov 2008**). "Evolution of genes and repeats in the Nimrod superfamily." *Mol Biol Evol*, **25**: 2337.

[264] Fisher, K. (**2008**). "Bayesian reconstruction of ancestral expression of the LEA gene families reveals propagule-derived desiccation tolerance in resurrection plants". *American Journal of Botany*, **95**: 506.

[265] Gardiner, A., D. Barker, R. Butlin, W. Jordan, and M. Ritchie (**Jan 2008**). "Evolution of a complex locus: exon gain, loss and divergence at the Gr39a locus in *Drosophila*". *PLoS ONE*, **3**: e1513.

[266] Pinney, J., G. Amoutzias, M. Rattray, and D. Robertson (**Dec 2007**). "Reconstruction of ancestral protein interaction networks for the bZIP transcription factors." *Proc Natl Acad Sci U S A*, **104**: 20449.

[267] Plachetzki, D., B. Degnan, and T. Oakley (**Oct 2007**). "The origins of novel protein interactions during animal opsin evolution." *PLoS One*, **2**: e1054.

[268] Hahn, M., M. Han, and S. Han (**Nov 2007**). "Gene family evolution across 12 *Drosophila* genomes." *PLoS Genet*, **3**: e197.

[269] Sakarya, O., K. Armstrong, M. Adamska, *et al.* (**Jun 2007**). "A post-synaptic scaffold at the origin of the animal kingdom." *PLoS One*, **2**: e506.

[270] Hahn, M. (**Jul 2007**). "Bias in phylogenetic tree reconciliation methods: Implications for vertebrate genome evolution". *Genome Biol*, **8**(7): R141.

[271] Jackson, A. (**2007**). "Origins of amino acid transporter loci in trypanosomatid parasites". *BMC Evol Biol*, **7**: 26.

[272] Demuth, J., T. Bie, J. Stajich, N. Cristianini, and M. Hahn (**2006**). "The evolution of mammalian gene families". *PLoS ONE*, **1**: e85.

[273] Floyd, S. and J. Bowman (**2007**). "The ancestral developmental tool kit of land plants". *Int J Plant Sci*, **168**(1): 1.

[274] Li, W. (**1997**). *Molecular Evolution*. Sinauer Associates Inc., Sunderland, MA.

[275] Bender, M. and M. Farach-Colton (**2000**). "Least common ancestors revisited". *Latin*, pp. 88–94.

[276] Bourgon, R., M. Delorenzi, T. Sargeant, *et al.* (**Nov 2004**). "The serine repeat antigen SERA gene family phylogeny in *Plasmodium*: the impact of GC content and reconciliation of gene and species trees". *Mol Biol Evol*, **21**(11): 2161.

[277] Hughes, A. (**1998**). "Phylogenetic tests of the hypothesis of block duplication of homologous genes on human chromosomes 6, 9, and 1". *Mol Biol Evol*, **15**(7): 854.

[278] Pebusque, M., F. Coulier, D. Birnbaum, and P. Pontarotti (**1998**). "Ancient large-scale genome duplications: phylogenetic and linkage analyses shed light on chordate genome evolution." *Mol Biol Evol*, **15**(9): 1145.

[279] Ruvinsky, I. and L. Silver (**1997**). "Newly indentified paralogous groups on mouse chromosomes 5 and 11 reveal the age of a T-box cluster duplication". *Genomics*, **40**: 262.

[280] Jackman, T., A. Larson, K. De Queiroz, and J. Losos (**1999**). "Phylogenetic relationships and tempo of early diversification in *Anolis* lizards". *Syst Biol*, **48**(2): 254.

[281] Walsh, H., M. Kidd, T. Moum, and V. Friesen (**1999**). "Polytomies and the power of phylogenetic inference". *Evolution*, **53**(3): 932.

[282] Poe, S. and A. Chubb (**2004**). "Birds in a bush: five genes indicate explosive evolution of avian orders". *Evolution*, **58**(2): 404.

[283] Wheeler, D. *et al.* (**Jan 2005**). "Database resources of the National Center for Biotechnology Information." *Nucleic Acids Res*, **33**(Database issue): D39.

[284] Notredame, C., D. Higgins, and J. Heringa (**2000**). "T-Coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment". *J Mol Biol*, **302**: 205.

[285] Li, H. *et al.* (**Jan 2006**). "TreeFam: a curated database of phylogenetic trees of animal gene families." *Nucleic Acids Res*, **34**(Database issue): 572.

[286] Notredame, C., D. Higgins, and J. Heringa (**Sep 2000**). "T-Coffee: A novel method for fast and accurate multiple sequence alignment." *J Mol Biol*, **302**(1): 205.

[287] Nicholas, K., H. Nicholas, and D. Deerfield (**1997**). "Genedoc: Analysis and visualization of genetic variation". *EMBNEWNEWS*, **4**: 14.

[288] Felsenstein, J. (**1989**). "PHYLIP - phylogeny inference package (version 3.2)". *Cladistics*, **5**: 164.

[289] Ehebauer, M., P. Hayward, and A. Martinez-Arias (**Dec 2006**). "Notch signaling pathway." *Sci STKE*, **2006**(364): cm7.

[290] Higgins, D. G., A. J. Bleasby, and R. Fuchs (**Apr 1992**). "Clustal V: improved software for multiple sequence alignment." *Comput Appl Biosci*, **8**(2): 189.

[291] Robinson, D., Y. Wu, and S. Lin (**Nov 2000**). "The protein tyrosine kinase family of the human genome". *Oncogene*, **19**(49): 5548.

[292] Devereux, J., P. Haeberli, and O. Smithies (**Jan 1984**). "A comprehensive set of sequence analysis programs for the vax." *Nucleic Acids Res*, **12**(1): 387.

[293] Swofford, D. L. (**1998**). *PAUP*. Phylogenetic analysis using parsimony (*and other methods).* Sinauer, Sunderland, Mass, version 4. edition.

[294] Adell, T., V. Gamulin, S. Perović-Ottstadt, *et al.* (**Jul 2004**). "Evolution of metazoan cell junction proteins: the scaffold protein magi and the transmembrane receptor tetraspanin in the demosponge suberites domuncula." *J Mol Evol*, **59**(1): 41.

[295] Ruiz-Trillo, I., A. J. Roger, G. Burger, M. W. Gray, and B. F. Lang (**Apr 2008**). "A phylogenomic investigation into the origin of metazoa." *Mol Biol Evol*, **25**(4): 664.

[296] de Mendoza, A., H. Suga, and I. Ruiz-Trillo (**Apr 2010**). "Evolution of the MaGuK protein gene family in premetazoan lineages." *BMC Evol Biol*, **10**: 93.

[297] Funke, L., S. Dakoji, and D. Bredt (**2005**). "Membrane-associated guanylate kinases regulate adhesion and plasticity at cell junctions." *Annu Rev Biochem*, **74**: 219.

[298] Thompson, J. D., T. J. Gibson, F. Plewniak, F. Jeanmougin, and D. G. Higgins (**Dec 1997**). "The Clustal_X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools." *Nucleic Acids Res*, **25**(24): 4876.

[299] Guindon, S. and O. Gascuel (**2003**). "A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood". *Syst Biol*, **52**(5): 696.

[300] Stolzer, M. (**Jan 2011**). *Data Analysis Project: Inferring Rates of Domain Shuffling Using a Birth-Death and Gain Model*. Ph.D. thesis.

[301] Cheng, Z., M. Ventura, X. She, *et al.* (**Sep 2005**). "A genome-wide comparison of recent chimpanzee and human segmental duplications." *Nature*, **437**(7055): 88.

[302] McGrath, C., C. Casola, and M. Hahn (**Jun 2009**). "Minimal effect of ectopic gene conversion among recent duplicates in four mammalian genomes." *Genetics*, **182**: 615.

[303] Vàzquez-Salat, N., N. Yuhki, T. Beck, S. O'Brien, and W. Murphy (**Aug 2007**). "Gene conversion between mammalian CCR2 and CCR5 chemokine receptor genes: a potential mechanism for receptor dimerization." *Genomics*, **90**: 213.

[304] Winter, E. and C. Ponting (**Oct 2005**). "Mammalian BEX, WEX and GASP genes: coding and non-coding chimaerism sustained by gene conversion events." *BMC Evol Biol*, **5**: 54.

[305] Walsh, J. (**Jan 1985**). "Interaction of selection and biased gene conversion in a multigene family." *Proc Natl Acad Sci U S A*, **82**: 153.

[306] Angers, B., K. Gharbi, and A. Estoup (**Apr 2002**). "Evidence of gene conversion events between paralogous sequences produced by tetraploidization in Salmoninae fish." *J Mol Evol*, **54**(4): 501.

[307] Lin, Y., J. Byrnes, J. Hwang, and W. Li (**Sep 2006**). "Codon-usage bias versus gene conversion in the evolution of yeast duplicate genes." *Proc Natl Acad Sci U S A*, **103**: 14412.

[308] Nagylaki, T. (**Mar 1984**). "The evolution of multigene families under intrachromosomal gene conversion." *Genetics*, **106**: 529.

[309] Ohta, T. (**Mar 1984**). "Some models of gene conversion for treating the evolution of multigene families." *Genetics*, **106**: 517.

[310] Semple, C. and K. Wolfe (**1999**). "Gene duplication and gene conversion in the *Caenorhabditis elegans* genome". *J Mol Evol*, **48**(5): 555.

[311] Noonan, J., J. Grimwood, J. Schmutz, M. Dickson, and R. Myers (**Mar 2004**). "Gene conversion and the evolution of protocadherin gene cluster diversity." *Genome Res*, **14**: 354.

[312] Hao, W. (**Mar 2010**). "Orgconv: detection of gene conversion using consensus sequences and its application in plant mitochondrial and chloroplast homologs." *BMC Bioinformatics*, **11**: 114.

[313] Sawyer, S. (**Sep 1989**). "Statistical tests for detecting gene conversion." *Mol Biol Evol*, **6**: 526.

[314] Sawyer, S. A. (**1999**). *GENECONV: A computer package for the statistical detection of gene conversion.* Department of Mathematics, Washington University in St. Louis.

[315] Thomas, P., A. Kejariwal, M. Campbell, *et al.* (**Jan 2003**). "PANTHER: a browsable database of gene products organized by biological function, using curated protein family and subfamily classification." *Nucleic Acids Res*, **31**(1): 334.

[316] Bateman, A., L. Coin, R. Durbin, *et al.* (**Jan 2004**). "The Pfam protein families database." *Nucleic Acids Res*, **32**(Database issue): D138.

[317] Katoh, K., K. Misawa, K. Kuma, and T. Miyata (**Jul 2002**). "MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform." *Nucleic Acids Res*, **30**(14): 3059.

[318] Capella-Gutiérrez, S., J. Silla-Martínez, and T. Gabaldón (**Aug 2009**). "trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses." *Bioinformatics*, **25**(15): 1972.

[319] Spearman, C. (**1904**). "The proof and measurement of association between two things". *Amer J Psychol*, **15**: 72.

[320] Snel, B., G. Lehmann, P. Bork, and M. Huynen (**2000**). "STRING: a web-server to retrieve and display the repeatedly occurring neighbourhood of a gene". *Nucleic Acids Res*, **28**(18): 3442.

[321] Liu, M., H. Walch, S. Wu, and A. Grigoriev (**2005**). "Significant expansion of exon-bordering protein domains during animal proteome evolution". *Nucleic Acids Res*, **33**(1): 95.

[322] Yeats, C., O. C. Redfern, and C. Orengo (**Mar 2010**). "A fast and automated solution for accurately resolving protein domain architectures." *Bioinformatics*, **26**(6): 745.

[323] Arvestad, L., A. Berglund, J. Lagergren, and B. Sennblad (**2004**). "Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution." In *RECOMB 2004: Proceedings of the Eigth International Conference on Research in Computational Biology*, pp. 326–335. ACM.

[324] Nacher, J., M. Hayashida, and T. Akutsu (**Aug 2010**). "The role of internal duplication in the evolution of multi-domain proteins." *Biosystems*, **101**: 127.

[325] Reshef, D., Z. Itzhaki, and O. Schueler-Furman (**Sep 2010**). "Increased sequence conservation of domain repeats in prokaryotic proteins." *Trends Genet*, **26**(9): 383.

[326] Doyon, J., C. Scornavacca, K. Gorbunov, *et al.* (**2010**). "An efficient algorithm for gene/species trees parsimonious reconciliation with losses, duplications and transfers". In *RECOMB-CG 2010: Proceedings of the International Workshop on Comparative Genomics*, edited by E. Tannier, pp. 93–108. Springer.

[327] Hahn, M., J. Demuth, and S. Han (**Nov 2007**). "Accelerated rate of gene gain and loss in primates." *Genetics*, **177**: 1941.

[328] Graur, D. and W. Li (**2000**). *Fundamentals of Molecular Evolution*, chapter 6, pp. 249–322. Sinaure Associates, 2nd edition.

[329] Marcotte, E., M. Pellegrini, M. Thompson, T. Yeates, and D. Eisenberg (**Nov 1999**). "A combined algorithm for genome-wide prediction of protein function". *Nature*, **402**(6757): 83.