An Integrated Reconciliation Framework for Domain, Gene, and Species Level Evolution

Lei Li and Mukul S. Bansal

Abstract—The majority of genes in eukaryotes consist of one or more *protein domains* that can be independently lost or gained during evolution. This gain and loss of protein domains, through domain duplications, transfers, or losses, has important evolutionary and functional consequences. Yet, most computational methods for studying gene evolution view genes as the basic unit of evolution and assume that evolutionary processes such as duplications and losses act on entire genes, rather than on parts of genes. Specifically, even though it is well understood that domains evolve inside genes and genes inside species, there do not exist any computational frameworks to simultaneously model the evolution of domains, genes, and species and account for their inter-dependency.

Here, we develop an integrated model of domain evolution that explicitly captures the interdependence of domain-, gene-, and species-level evolution. Our model extends the classical phylogenetic reconciliation framework, which infers gene family evolution by comparing gene trees and species tree, by explicitly considering domain-level evolution and decoupling domain-level events from gene-level events. In this paper, we (i) introduce the new integrated reconciliation framework, (ii) prove that the associated optimization problem is NP-hard, (iii) devise an efficient heuristic solution for the problem, (iv) apply our algorithm to a large dataset of over 3700 domain trees and 7000 gene trees from 12 fly species, and (v) demonstrate the impact of using our new computational framework by comparing the inferred evolutionary histories against those obtained using existing approaches. The implemented software is freely available from http://compbio.engr.uconn.edu/software/seadog/.

1 Introduction

Gene families evolve via complex evolutionary processes such as gene duplication, gene loss, horizontal gene transfer, and incomplete lineage sorting, and understanding the role of these processes in the evolution of any gene family has many important applications throughout biology. As a result, many methods exist for studying the role of these processes in gene family evolution. Yet, most previous methods view the gene as the basic unit of evolution and assume that the evolutionary processes act on entire genes, rather than on parts of genes. It has been estimated that up to 60% of genes in multicellular organisms and 40% of genes in unicellular organisms [19] consist of multiple *protein domains* (well-characterized functional units) that can be independently lost or gained during evolution; i.e., that events such as duplication, transfer, and loss can act upon domains, instead of on entire genes [25]. Thus, previous methods for inferring gene family evolution ignore one of its primary evolutionary mechanisms.

The gain or loss of domains has important functional consequences for any gene [2], [24], [39], [40], and there exists a large body of work studying the evolutionary and functional dynamics of domains and of multi-domain genes. However, even though it is well understood that domains evolve inside genes and genes inside species, e.g., [35], there do not exist any computational frameworks to simultaneously model the evolution of domains, genes, and species and account for their inter-dependency. This is a major limitation of the existing models of domain and gene evolution, with implications on their accuracy and capability.

In this work, we develop a *three-tree* model of domain evolution that explicitly captures the interdependence of domain-, gene-, and species-level evolution. Our model decouples domain-level events from gene-level events and provides a much more fine-grained view of gene family and domain family evolution that is both more accurate and easy to interpret. Our three-tree model builds upon the classical *phylogenetic reconciliation* framework, which compares a *gene tree* (evolutionary tree of a gene family) with its *species tree* (evolutionary tree of the corresponding species) to infer the evolutionary events that shaped that gene family. Our new, three-tree reconciliation model takes as input a domain tree, its gene trees, and a species tree, and jointly optimizes the reconciliation of the domain tree with the gene trees and gene trees with the species tree. We develop our framework in the context of eukaryotic gene families where gene families evolve primarily through gene duplications and gene losses, and domain families through domain duplications, domain transfers (from one gene to another within the same species), and domain losses. Our model captures these primary elementary events of domain and gene evolution, and we formulate the optimization problem as a parsimony problem where the goal is to minimize the weighted sum of the evolutionary events invoked. An illustration appears in Figure 1(a).

Previous work on domain analysis. Previous computational work on domain evolution can be divided into three categories: (i) methods for identifying domains in gene sequences, (ii) methods for studying the dynamics of domain content in proteins (genes), and (iii) methods that explicitly study domain evolution. Methods in the first

• *Lei Li is with the Department of Computer Science & Engineering at the University of Connecticut, Storrs, USA.* lei.li@uconn.edu
• *Mukul S. Bansal is with the Department of Computer Science & Engineering and the Institute for Systems Genomics at the University of Connecticut, Storrs, USA.* mukul.bansal@uconn.edu

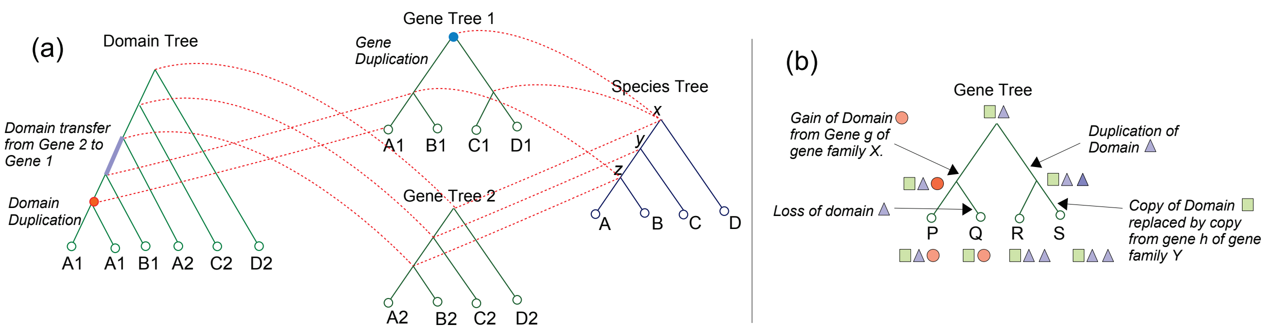

Fig. 1. **The proposed three-tree model.** (a) The figure depicts a simple example with a domain tree whose domains come from two gene trees (gene trees 1 and 2). The Domain-Gene-Species (DGS) reconciliation framework simultaneously optimizes the mapping (shown by the red dotted lines) of the domain tree into the gene trees and of the gene trees into the species tree. The domain-to-gene leaf associations are specified by shared leaf labels, and the gene-to-species leaf associations are specified by shared letters (A, B, C, or D). The depicted DGS reconciliation scenario shows how gene trees 1 and 2 evolved inside the species tree and how the domain tree evolved inside the two gene trees. In the gene-to-species reconciliation, a gene-duplication event (highlighted in blue) is invoked at the root of gene tree 1 and all other internal nodes of the gene trees represent speciation events. In the domain-to-gene reconciliation, a domain duplication event is invoked at the marked node (highlighted in orange) as shown in the figure, and a domain-transfer event is invoked at the bolded edge in the domain tree where the domain is copied from gene tree 2 to gene tree 1. Observe that the donor gene from gene tree 2 and the recipient gene from gene tree 1 both map to the same species tree node z. Thus, this DGS reconciliation allows for detailed inference of the evolution of the domain tree and of loss/gain of domains inside the gene trees. (b) A consolidated view of domain gain and loss inside a gene tree, obtained by combining the evolutionary histories for all domains in that gene family. Each colored square, triangle, and circle represents a domain from a distinct domain family present in the considered gene tree.

category are focused on identifying known and unknown domains in gene sequences and classifying them into distinct domain families [5], [9], [36]. These methods have led to the identification of tens of thousands of domain families, knowledge about the prevalence of multi-domain proteins, and the creation of many databases for domain sequences like Pfam [13], SMART [30], etc. Methods in the second category [7], [10], [14], [15], [27] focus primarily on problems related to the emergence and preservation of domain combinations in genes, rates of various domain shuffling events (such as domain insertions, duplications, and deletions), inference of ancestral domain architectures, etc. These methods generally use simple evolutionary models that often do not consider the phylogenetic history of domains and are not designed for inferring evolutionary scenarios for domains or genes.

The methods most closely related to our work are those that belong to category (iii) and whose focus is on explicitly reconstructing the evolutionary histories of domain families [6], [33], [35], [42], [45]. The methods of Behzadi and Vingron [6], Weidenhoeft et al. [42] and Wu et al. [45] take as input a collection of domain trees along with domain compositions or architectures for the genes from which the domains were sampled. The method of Wu et al. [45] also requires a species tree. Both methods combine domains parsimoniously in a bottom up fashion based on the given domain compositions/architectures to recreate the compositions/architectures for ancestral genes, along with events such as gene fusions and fissions. Neither of these methods captures the interdependence of domain, gene, and species evolution, and neither uses gene trees. Explicit use of gene trees is important since gene trees are widely used for functional studies and in the study of genome evolution. Thus, using gene trees greatly improves the utility and interpretability of domain evolution. Moreover, without using gene trees one cannot properly model the interdependence between domain, gene, and species evolution. The previous approach conceptually most similar to ours is that of Stolzer et al. [33], [35], which uses the well-established Duplication-Transfer-Loss reconciliation model to reconcile domain trees with either gene trees or species trees to gain evolutionary insight, and can also reconcile domain trees with gene trees and gene trees with species tree. The approach of Stolzer et al. is ground-breaking in its application of phylogenetic reconciliation to infer domain-level evolution. However, Stolzer et al. use a simpler problem formulation that assumes a fixed gene to species mapping and does not seek a *joint* reconciliation of the domain, gene, and species trees. Thus, their approach does not model the interdependence of domain, gene, and species evolution. Furthermore, while our approach models the evolution of a domain tree in multiple gene trees simultaneously, the approach of Stolzer et al. only allows for the reconciliation of a domain tree with a single gene tree and therefore cannot infer a complete history of the evolution of domain families.

Previous work on phylogenetic reconciliation. Phylogenetic reconciliation is one of the most powerful and most widely used techniques for studying gene family evolution and involves the systematic comparison of a gene tree with its species tree. The technique is based on the observation that the evolutionary processes responsible for gene family evolution create incongruence between the topology of the gene tree and that of the underlying species tree. Thus, by comparing the gene tree with the species tree one can infer the evolutionary events required to explain their incongruence. The inference problem may be formulated either probabilistically, where the goal is to find the most likely reconciliation, or based on the parsimony principle, where the goal is to find a reconciliation with the smallest weighted total cost of events. Most formulations are based on parsimony, which is conceptually simpler, makes fewer assumptions about the rates of evolutionary events, admits more efficient algorithms, and is highly accurate in practice. Based on the evolutionary events considered several reconciliation models exist in the literature. The *duplication-loss* model, e.g., [1], [8], [12], [16], [18], [22], [23], [26], [46], and *Duplication-Transfer-Loss* model, e.g., [3], [11], [17], [31], [34], [37], [38] are especially well-studied. None of the previous reconciliation models consider domain-level evolutionary events, and take as input only two trees.

Our Contributions. Here, we introduce our three-tree model of domain evolution that explicitly captures the interdependence of domain-, gene-, and species-level evolution, lay down its methodological and algorithmic foundations, and study the impact of the new model on inferring domain- and gene-level evolution in practice. Specifically, our contributions are as follows: We (i) introduce our new computational framework, the Domain-Gene-Species (DGS) reconciliation model, (ii) prove that the associated optimization problem is NP-hard, (iii) devise an efficient and effective heuristic algorithm for the problem, (iv) apply our algorithm to a large dataset of over 3700 domain trees and 7000 gene trees from 12 fly species, and (v) demonstrate the significant impact of using our new computational framework by comparing the inferred evolutionary histories of domains and genes against those obtained using existing approaches.

The remainder of this manuscript is organized as follows: The next section starts with definitions and preliminaries, introduces the DGS reconciliation model and associated optimization problem, and discusses basic properties, assumptions, and limitations of the DGS reconciliation model. In Section 3 we prove that the associated optimization problem is NP-hard. Our heuristic algorithm for the problem is described in Section 4. Experimental results appear in Section 5, and concluding remarks in Section 6.

2 Definitions and Preliminaries

Preliminaries. Throughout this manuscript, the term *tree* refers to rooted trees. Given a tree T, we denote its node, edge, and leaf sets by $V(T)$, $E(T)$, and $Le(T)$ respectively. The root node of T is denoted by $rt(T)$, the parent of a node $v \in V(T)$ by $pa_T(v)$, its set of children by $Ch_T(v)$, and the (maximal) subtree of T rooted at v by $T(v)$. The set of *internal nodes* of T, denoted $I(T)$, is defined to be $V(T) \setminus Le(T)$. We define $\leq_T$ to be the partial order on $V(T)$ where $x \leq_T y$ if y is a node on the path between $rt(T)$ and x. The partial order $\geq_T$ is defined analogously, i.e., $x \geq_T y$ if x is a node on the path between $rt(T)$ and y. We say that y is an *ancestor* of x, or that x is a *descendant* of y, if $x \leq_T y$ (note that, under this definition, every node is a descendant as well as ancestor of itself). We say that x and y are *incomparable* if neither $x \leq_T y$ nor $y \leq_T x$. Given a non-empty subset $L \subseteq Le(T)$, we denote by $lca_T(L)$ the least common ancestor (LCA) of all the leaves in L in tree T; i.e., $lca_T(L)$ is the unique smallest upper bound of L under $\leq_T$.

Our three-tree framework takes as input a domain tree D, a collection of gene trees $\mathcal{G}$, and a species tree S. For the core framework, all trees are assumed to be rooted (unrooted trees can be rooted by locally optimizing the reconciliation over all possible roots). The *species tree* is a tree showing the evolutionary history for a chosen set of species. Each *gene tree* is a tree showing the evolutionary history for a set of genes related by common ancestry, called a *gene family*, restricted to the species represented in the species tree. Similarly, a *domain tree* shows the evolutionary history of a set of domains related by common ancestry, called a *domain family*, restricted to the chosen gene families.

The leaves in each of the three types of trees represent existing entities (species, or gene sequences, or domain sequences), while internal nodes represent hypothetical ancestral species or sequences. Each leaf in a gene tree is labeled by the species from which that leaf (gene) was sampled. Similarly, each leaf in a domain tree is labeled with the gene from which that leaf (domain) was taken. This defines a leaf-to-leaf mapping from the domain trees to the gene trees, and from the gene trees to the species tree. Since a gene may have multiple domains, there may be multiple domains (possibly from different domain trees) mapping to the same gene. Similarly, since domains from the same domain family may be present in multiple gene families, different leaves of a single domain tree may map to genes from different gene families. This is illustrated in Figure 1(a) and (b).

2.1 The domain-gene-species reconciliation model

We define the *domain-gene-species (DGS)* reconciliation model where the goal is to find a reconciliation of the given gene trees with the species tree, and of the given domain tree with the gene trees. The reconciliation of a gene tree with a species tree models the primary evolutionary events that shape gene family evolution within species; in the case of multi-cellular organisms these are *speciation*, *gene duplication*, and *gene loss*. Similarly, the reconciliation of a domain tree with one or more gene trees models the elementary evolutionary events that shape domain family evolution within genes; in this case *co-divergence*, *domain transfer*, *domain duplication*, and *domain loss*. Our reconciliation model is based on the parsimony principle, where each event is assigned a cost and we seek a DGS reconciliation of minimum total cost.

The DGS reconciliation model requires *joint* optimization of gene-species and domain-gene reconciliations since the domain-gene reconciliation depends on the gene-species reconciliation. Thus, the two reconciliation problems cannot be solved individually. A valid DGS reconciliation for a given domain tree D, a set of gene trees $\mathcal{G}$ in which the domains of D are represented, and a species tree S, can be formally defined as follows.

Definition 2.1 (DGS-reconciliation). *Given a domain tree D, collection of gene trees $\mathcal{G}$, a species tree S, and leaf-mappings $\mathcal{L}^D \colon Le(D) \to Le(\mathcal{G})$ and $\mathcal{L}^{\mathcal{G}} \colon Le(\mathcal{G}) \to Le(S)$, a DGS-reconciliation for $D, \mathcal{G}$ and S is a nine-tuple $\langle \mathcal{M}^D, \mathcal{M}^{\mathcal{G}}, \Sigma^D, \Sigma^{\mathcal{G}}, \Delta^D, \Delta^{\mathcal{G}}, \Theta, \Xi, \tau \rangle$, where $\mathcal{M}^D \colon V(D) \to V(\mathcal{G})$ and $\mathcal{M}^{\mathcal{G}} \colon V(\mathcal{G}) \to V(S)$ map each node of D to a node from $\mathcal{G}$ and each node from $\mathcal{G}$ to a node of S, respectively, the sets Σ^D, Δ^D, and Θ partition $I(D)$ into co-divergence, domain-duplication, and domain-transfer nodes, respectively, the sets $\Sigma^{\mathcal{G}}$ and $\Delta^{\mathcal{G}}$ partition $I(\mathcal{G})$ into speciation and gene-duplication nodes, respectively, Ξ is a subset of domain tree edges that represent domain-transfer events, and $\tau \colon \Theta \to V(\mathcal{G})$ specifies the recipient gene for each domain-transfer event, subject to:*

Gene-Species constraints:
1) *If $g \in Le(\mathcal{G})$, then $\mathcal{M}^{\mathcal{G}}(g) = \mathcal{L}^{\mathcal{G}}(g)$.*
2) *If $g \in I(\mathcal{G})$ and g' and g'' denote the children of g, then,*
   a) *$\mathcal{M}^{\mathcal{G}}(g) \geq_S lca(M^{\mathcal{G}}(g'), M^{\mathcal{G}}(g''))$,*

b) $g \in \Sigma^{\mathcal{G}}$ if and only if $\mathcal{M}^{\mathcal{G}}(g) = lca(\mathcal{M}^{\mathcal{G}}(g'), \mathcal{M}^{\mathcal{G}}(g''))$ and $\mathcal{M}^{\mathcal{G}}(g')$ and $\mathcal{M}^{\mathcal{G}}(g'')$ are incomparable,
c) $g \in \Delta^{\mathcal{G}}$ only if $\mathcal{M}^{\mathcal{G}}(g) \geq_S lca(\mathcal{M}^{\mathcal{G}}(g'), \mathcal{M}^{\mathcal{G}}(g''))$.

Domain-Gene constraints:

3) If $d \in Le(D)$, then $\mathcal{M}^D(d) = \mathcal{L}^D(d)$.
4) If $d \in I(D)$ and d' and d'' denote the children of d, then,
  a) $\mathcal{M}^D(d) \not\leq_{\mathcal{G}} \mathcal{M}^D(d')$ and $\mathcal{M}^D(d) \not\leq_{\mathcal{G}} \mathcal{M}^D(d'')$,
  b) At least one of $\mathcal{M}^D(d')$ and $\mathcal{M}^D(d'')$ is a descendant of $\mathcal{M}^D(d)$ (in the same gene tree).
5) Given any edge $(d, d') \in E(D)$, $(d, d') \in \Xi$ if and only if $\mathcal{M}^D(d)$ and $\mathcal{M}^D(d')$ are in different gene trees or incomparable in the same gene tree.
6) If $d \in I(D)$ and d' and d'' denote the children of d, then,
  a) $d \in \Sigma^D$ if and only if $\mathcal{M}^D(d) = lca(\mathcal{M}^D(d'), \mathcal{M}^D(d''))$ (in the same gene tree) and $\mathcal{M}^D(d')$ and $\mathcal{M}^D(d'')$ are incomparable,
  b) $d \in \Delta^D$ only if $\mathcal{M}^D(d) \geq_{\mathcal{G}} lca(\mathcal{M}^D(d'), \mathcal{M}^D(d''))$ (in the same gene tree),
  c) $d \in \Theta$ if and only if either $(d, d') \in \Xi$ or $(d, d'') \in \Xi$.
  d) If $d \in \Theta$ and $(d, d') \in \Xi$, then $\mathcal{M}^D(d)$ and $\tau(d)$ must either be in different gene trees or incomparable in the same gene tree, $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(d)) = \mathcal{M}^{\mathcal{G}}(\tau(d))$, and $\mathcal{M}^D(d') \leq_{\mathcal{G}} \tau(d)$.

Constraints 1 and 2 above apply to the reconciliation of the gene trees with the species tree. Essentially, we define each gene tree node to be speciation or gene-duplication based on the classical *Duplication-Loss* model [16], [26], allowing for suboptimal gene-species reconciliations. Constraints 3, 4, 5, and 6 apply to the reconciliation of the domain tree with the gene trees. This domain tree to gene trees reconciliation is similar to the well-studied *Duplication-Transfer-Loss (DTL)* reconciliation model for reconciling gene trees and species tree in the presence of horizontal gene transfer, e.g., [3], [11], [17], [34], [38], with a few key differences: First, the speciation, duplication, transfer, and loss events from DTL-reconciliation correspond to co-divergence, domain-duplications, domain-transfers (which modeling the copying of a domain from one gene to another gene, and domain-losses in the domain tree D. Second, the reconciliation between D and $\mathcal{G}$ may span more than one gene tree from $\mathcal{G}$. And third, domain transfers can only occur between genes present in the same genome/species, and these transfers may occur between genes from either the same gene family or different gene families. Constraint 3 ensures that the mapping $\mathcal{M}^D$ is consistent with the leaf-mapping $\mathcal{L}^{\mathcal{D}}$. Constraint 4a imposes on $\mathcal{M}^D$ the temporal constraints (ancestor-descendant relationships) implied by the gene trees. Constraint 4b implies that any internal node in D may represent at most one domain-transfer event. Constraint 5 determines the edges of D that are domain-transfer edges. Constraints 6a, 6b, and 6c state the conditions under which an internal node of $\mathcal{G}$ may represent a co-divergence, domain-duplication, and domain-transfer respectively. Constraint 6d specifies which genes may be designated as the recipient gene for any given domain-transfer event. Note that, in the absence of horizontal gene transfer, the transfer of a domain from one gene to another can only happen within the same genome. Thus, we explicitly enforce that the donor gene and recipient gene for any domain transfer event must map to the same species in the species tree (constraint 6d). Figure 1(a) shows an example of a valid DGS-reconciliation.

In our parsimony-based reconciliation framework, each evolutionary event other than speciation and co-divergence is assigned a positive cost. $P^{\mathcal{G}}_{\Delta}$ and $P^{\mathcal{G}}_{loss}$ denote the gene-duplication and gene-loss costs, while P^D_{Δ}, P^D_{Θ}, and $P^{\mathcal{G}}_{loss}$ denote domain-duplication, domain-transfer, and domain-loss costs. We use two separate costs $P^D_{\Theta 1}$ and $P^D_{\Theta 2}$ instead of a single P^D_{Θ}, so that we can distinguish between domain transfers that remain within the same gene family from those that cross gene family boundaries.

The reconciliation cost of a given DGS-reconciliation is defined as follows.

Definition 2.2 (Reconciliation cost of DGS-reconciliation)**.** *Given a DGS-reconciliation α, the* reconciliation cost *for α is the total cost of all events invoked by α.*

Note that, while domain-duplication, domain-transfer, and gene-duplication events are directly specified in the DGS-reconciliation, domain-losses and gene-losses are not. However, given a DGS-reconciliation, one can directly count the minimum number of gene-losses and domain-losses, in accordance with the DTL and duplication-loss reconciliation models [3]. For completeness, we provide formal definitions of the minimum number of gene-losses and domain-losses in Section S1 in the supplement.

The computational objective is to find an optimal, or most parsimonious, reconciliation, i.e., a DGS-reconciliation that has minimum reconciliation cost. More formally:

Definition 2.3 (Optimal DGS-Reconciliation (ODGS) Problem)**.** *Given D, $\mathcal{G}$ and S, along with $P^{\mathcal{G}}_{\Delta}$, $P^{\mathcal{G}}_{loss}$, P^D_{Δ}, $P^D_{\Theta 1}$, $P^D_{\Theta 2}$, and P^D_{loss}, the* ODGS *problem is to find a DGS-reconciliation for D, $\mathcal{G}$ and S with minimum reconciliation cost.*

Relationship to previous approaches. The only other reconciliation-based approach to modeling domain evolution is that of Stolzer et al. [33], [35], which uses the well-established Duplication-Transfer-Loss reconciliation model to reconcile domain trees with either gene trees or species trees to gain evolutionary insight, and can also reconcile domain trees with gene trees and gene trees with species tree. A crucial difference between the DGS reconciliation model and the approach of Stolzer et al. is that they use a simpler problem formulation that assumes a fixed gene to species mapping and does not seek a *joint* reconciliation of the domain, gene, and species trees. Thus, their approach does not fully model the interdependence of domain, gene, and species evolution. Another difference is that while our approach models the evolution of a domain tree in multiple gene trees simultaneously, the approach of Stolzer et al. only allows for the reconciliation of a domain tree with a single gene tree and therefore is unable infer a complete history of the evolution of domain families.

2.2 Existence of DGS-reconciliations

Given the constraints on DGS reconciliation, it may not be immediately clear if a DGS-reconciliation always exists. As the following claim shows, there always exists a valid DGS-reconciliation if each gene tree in $\mathcal{G}$ has at least two leaves.

Claim 1. *Given D, $\mathcal{G}$ and S, along with $P^{\mathcal{G}}_{\Delta}$, $P^{\mathcal{G}}_{loss}$, P^D_{Δ}, $P^D_{\Theta 1}$, $P^D_{\Theta 2}$, and P^D_{loss}, if each $G \in \mathcal{G}$ is such that $|V(G)| \geq 2$, then there exists a valid DGS-reconciliation for D, $\mathcal{G}$ and S.*

Proof. Consider any mapping of D to the gene trees in $\mathcal{G}$ that is valid under the DTL reconciliation model (such a mapping always exists). We will show how to construct a mapping from $\mathcal{G}$ to S that allows for this mapping from D to $\mathcal{G}$: Simply map every internal node of every gene tree in $\mathcal{G}$ to the root node of S. The resulting domain-to-gene and gene-to-species mappings constitute a valid DGS-reconciliation and satisfy all constraints of Definition 1. $\square$

Thus, a DGS reconciliation is guaranteed to exist as long as a very basic condition is met. In fact, even if $\mathcal{G}$ contains single-leaf gene trees, a DGS reconciliation should still exist, under reasonable evolutionary assumptions, unless there are errors in the domain tree or the gene family is incomplete.

2.3 Temporal consistency of DGS-reconciliations

Our model allows for the transfer of domains from one gene to another within the same genome/species. In models where such horizontal transfer of genetic material is allowed (e.g., in the DTL reconciliation model), there is the possibility that an inferred evolutionary history might invoke transfer events that are temporally infeasible [3], [11], [34], [38]. This happens when the inferred transfer events imply contradictory temporal constraints, making them inconsistent with any temporal ordering (or dating) of the internal nodes of the underlying tree. A desirable property of our DGS reconciliation model is that any DGS-reconciliation is guaranteed to be temporally consistent (or temporally feasible) with respect to the species tree. This is because, in any DGS-reconciliation, domain transfers can only occur between genes that co-exist in the same genome/species on the species tree. Thus, irrespective of the temporal ordering of the nodes of the species tree, any invoked domain-transfer events each occur within the same species node. However, DGS-reconciliations need not always be temporally consistent with respect to the gene trees. This can happen when there are multiple nodes on the gene trees that map to the same species node. Domain transfers involving those gene tree nodes could, potentially, be temporally inconsistent. Temporal inconsistency is a well studied problem in the DTL reconciliation literature and is known to affect only a small fraction of optimal reconciliations [38]. In fact, since DGS-reconciliations allow domain-transfers only between genes that map to the same node on the species tree, the possibility for temporal inconsistency is further reduced. Moreover, even when reconciliations are allowed to be temporally inconsistent, the reconciliations show very high accuracy overall [4]. As a result, we do not explicitly enforce time consistency of domain-transfer events in our DGS-reconciliation model. However, if desired, a requirement for time consistency could be easily added to the definition of DGS-reconciliation.

2.4 Simplifying assumptions in DGS-reconciliation

The DGS reconciliation model makes some of the same assumptions as other phylogeny-based methods for studying domain evolution [6], [33], [35], [42], [45]. In particular, the DGS reconciliation framework assumes that domain trees, gene trees, and species trees can all be reconstructed with reasonable accuracy and provided as input. This is a safe assumption for species trees, whose accurate reconstruction benefits from the availability of well-behaved core genes or from whole-genome data. A potential problem with the construction of multi-domain gene trees is that different genes may have different domain contents or architectures, which complicates the building of those gene families due to domain chaining (caused by sharing of domains from the same domain family between genes from different gene families) [20], and of aligning their sequences for reconstructing gene trees [28]. These complications can lead to situations where two or more separate gene families are collapsed into a single gene family or to gene trees that have poor sequence support or incorrect tree topologies. However, some over-clustering of gene families is not a major confounding factor for DGS-reconciliation. Moreover, gene trees can generally be reconstructed with good accuracy, especially when error-correction techniques are used, e.g., [29], [44]. Indeed, the use of gene trees and species trees is ubiquitous throughout evolutionary genomics. The identification of protein domains and the construction of domain families is also a well-studied problem and gene/protein sequences are routinely annotated with their protein domains [13], [21]. However, the accurate reconstruction of domain trees can be difficult in many cases. This is primarily due to the shorter sequence lengths of protein domains, which makes it difficult to compute a well-supported domain tree topology. However, accurate domain trees can still be constructed for domains that have sufficient sequence length or sufficient diversity in their sequences. Furthermore, as we demonstrate in Section 5, it should be possible to error-correct incorrect domain trees topologies using techniques similar to those used for gene tree error correction. This limitation of obtaining reasonably accurate domain trees is one that our model shares will all other methods for studying domain evolution that use domain trees in their analysis [6], [33], [35], [42], [45]. Finally, as we discuss in Section 6, the DGS reconciliation framework may itself be used to error-correct domain trees by computing DGS-reconciliations for alternative domain tree topologies.

The current DGS reconciliation framework also makes some other simplifying assumptions. These include: (i) reconciliation of only a single domain family at a time, where simultaneous reconciliation of multiple domain trees may yield more accurate results by making it easier to identify events that affect multiple domains simultaneously, and (ii) no consideration of domain architectures or of the biological mechanisms, e.g., [40], [41], for domain duplication, transfer, or loss, consideration of which may lead to more accurate modeling of the domain gain/loss process. Despite these assumptions, the DGS reconciliation framework represents a significant advancement over existing phylogenetic approaches for studying domain family and gene family evolution and lays the foundation for more complex models of domain and gene evolution.

3 NP-hardness of the ODGS Problem

We now show that the ODGS problem is NP-hard. Specifically, let *D-ODGS* denote the decision version of the ODGS problem where, given $D, \mathcal{G}$ and S along with event costs $P_{\Delta}^{\mathcal{G}}$,

$P^{\mathcal{G}}_{loss}$, P^D_Δ, $P^D_{\Theta 1}$, $P^D_{\Theta 2}$, P^D_{loss} and a nonnegative integer N, the question is to decide if there exists a DSG reconciliation of D, $\mathcal{G}$ and S with reconciliation cost at most N.

Theorem 3.1. *The D-ODGS problem is NP-Complete.*

The D-ODGS problem is clearly in NP. We will show that D-ODGS is NP-hard using a polynomial-time reduction from the NP-Complete independent set problem. Given a graph $G = (V, E)$ and a nonnegative integer k, the *Independent Set (IS)* problem asks if there exists an independent set of size at least k in G.

Given instance $\langle G = (V, E), k\rangle$ of IS, let $\{v_1, \ldots, v_n\}$ denote the n vertices in V. We use E_{ij}, where $i < j$, to denote the edge between vertices v_i and v_j (assuming it exists). We will assume, without any loss of generality, that G is connected.

Consider an instance ϕ of the IS problem with $G = (V, E)$, and k given. We will show how to transform ϕ into an instance λ of the D-ODGS problem by constructing domain tree D, collection of gene trees $\mathcal{G}$, species tree S, and setting all the event costs in such a way that there exists a YES answer to the IS instance ϕ if and only if there exists a YES answer to the D-ODGS instance λ with $N = 5n + 4m + 4 - 4k$, where $n = |V|$ and $m = |E|$. The transformation of Φ into λ proceeds as follows:

Domain Tree. The domain tree D consists of m subtrees, labeled $T_{E_{ij}}$, each corresponding to a different edge $E_{ij} \in E$, connected together to form a caterpillar tree on these m subtrees. This is illustrated in Figure 2. The subtree $T_{E_{ij}}$ consists of four leaf nodes labeled d_{ij}, d_{ji}, d'_{ij}, and d'_{ji}. The topology of $T_{E_{ij}}$ is set to be $((d_{ij}, d_{ji}), (d'_{ij}, d'_{ji}))$. The ordering of these m subtrees in the caterpillar backbone is unimportant.
Gene Trees. $\mathcal{G}$ contains $n + 1$ gene trees denoted $G_0, G_1, \ldots, G_n$, each with only two leaves. The leaf nodes of G_0 are labeled g_1 and g_2, and those of G_i, for $i \geq 1$, are labeled g'_i and g''_i. Thus, gene trees correspond, in a sense, to vertices in the graph and each gene tree has the same topology.
Species Tree. The species tree S has four leaves labeled r_1, r_2, l, and r_s, and topology $((r_1, r_2), (l, r_s))$. Here l is a redundant node to which no gene node maps.
Leaf-Mappings $\mathcal{L}^D$ and $\mathcal{L}^{\mathcal{G}}$. We first describe $\mathcal{L}^D$. Domain tree leaves labeled d_{ij}, where $i < j$ map to node g_1 and those labeled d_{ji}, where $i < j$, map to node g_2. Domain tree leaves labeled d'_{ij} (resp. d'_{ji}) map to node g'_i (resp. g'_j). None of the domain nodes map to gene nodes labeled g''_i, for any $i \geq 1$. We now describe $\mathcal{L}^{\mathcal{G}}$. The gene nodes g_1 and g_2 from G_0 map to r_1 and r_2, respectively. For all other gene trees, G_i, for $i \geq 1$, leaf nodes g'_i and g''_i both map to r_s.
Event Costs. We set $P^{\mathcal{G}}_\Delta$, $P^{\mathcal{G}}_{loss}$, P^D_Δ, $P^D_{\Theta 1}$, $P^D_{\Theta 2}$, P^D_{loss} to all be 1.

The gadget described above is illustrated in Figure 2.
Main idea of the reduction. While the actual NP-completeness proof is non-trivial, the main idea of the reduction is as follows. Consider the relationship between the IS problem and D-ODGS problem. The structure of the domain tree and gene trees is such that a large number of the internal nodes of the domain tree must be domain transfer events. These domain transfer events require that their donor and recipient genes map to the same species in the species tree, which is only possible if some of the gene nodes deviate from their optimal mapping under the duplication-loss model. This incurs an additional cost in the gene-species reconciliation. However, this additional cost can be minimized by assigning the mappings of the internal nodes of the domain tree in such a way that the required domain transfer events are able to "share" the additional cost, rather than impose an additional cost of their own. The larger the independent set in the IS instance, the more the number of domain transfer events that can share the additional cost, resulting in a smaller DGS reconciliation cost overall.

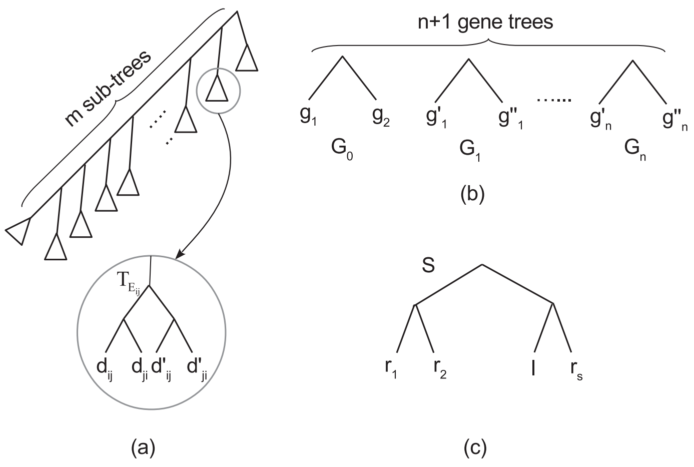

Fig. 2. This figure illustrates the construction of the D-ODGS instance given an instance of the IS problem. The domain tree D, gene trees in $\mathcal{G}$, and species tree S are shown.

More formally, the proof of Theorem 3.1 is based on the next claim.

Claim 2. *There exists a YES answer to the independent set instance ϕ if and only if there exists a YES answer to the D-ODGS instance λ with parameter $N = 5n + 4m + 4 - 4k$.*

3.1 Proof of Claim 2

Forward Direction: We first prove the forward direction of the claim, i.e., if there exists a YES answer to the independent set instance ϕ then there exists a YES answer to the D-ODGS instance λ with parameter $N = 5n + 4m + 4 - 4k$. Given an independent set V' of size at least k in G, we will show how to construct a DGS reconciliation α with reconciliation cost at most $5n + 4m + 4 - 4k$. The reconciliation α is defined by the following domain-gene and gene-species mappings.

$$\mathcal{M}^D : \begin{cases} pa(d'_{ij}) = \begin{cases} g'_i, & \text{if } v_i \in V \backslash V' \\ g'_j, & \text{if } v_i \in V' \end{cases} \\ pa(d_{ij}) = rt(G_0) \\ rt(T_{E_{ij}}) = rt(G_0) \\ pa(rt(T_{E_{ij}})) = rt(G_0) \end{cases}$$

$$\mathcal{M}^{\mathcal{G}} : \begin{cases} rt(G_0) = rt(S) \\ rt(G_i) = \begin{cases} rt(S), & \text{if } v_i \in V \backslash V' \\ r_s, & \text{if } v_i \in V' \end{cases} \end{cases}$$

Lemma 1. *Reconciliation α is a valid DGS reconciliation.*

Proof. It suffices to show that every internal node in D and $\mathcal{G}$ represents a valid reconciliation event. Based on the gadget and the reconciliation α, there are five types of internal nodes:

1) $pa(d'_{ij}) \in \Theta$, for all $T_{E_{ij}}$. Consider an internal node $pa(d'_{ij}) \in T_{E_{ij}}$, its two children d'_{ij} and d'_{ji} map to G_i and G_j, respectively. First, assume $v_i \in V \backslash V'$ and $\mathcal{M}^D(pa(d'_{ij})) = g'_i$. Note that nodes g'_i and g'_j are from different gene trees and map to the same species node r_s. Let $\tau(pa(d'_{ij})) = g'_j$. Since $pa(d'_{ij})$ and g'_j map to the sam species node, and $\mathcal{M}^D(d'_{ji}) \leq_{\mathcal{G}} \tau(pa(d'_{ij}))$, $pa(d'_{ij})$ is a valid domain transfer event. Now, assume that $v_i \in V'$ and $\mathcal{M}^D(pa(d'_{ij})) = g'_j$. Since (i, j) is an edge in the graph, it immediately follows that $j \in V \backslash V'$. The proof for this case is thus analogous to the proof for the previous case.
2) $pa(d_{ij}) \in \Sigma^D$, for all $T_{E_{ij}}$. This follows easily since $\mathcal{M}^D(pa(d_{ij})) = lca(\mathcal{M}^D(d_{ij}), \mathcal{M}^D(d_{ji}))$, and $\mathcal{M}^D(d_{ij})$ and $\mathcal{M}^D(d_{ji})$ are incomparable.
3) $rt(T_{E_{ij}}) \in \Theta$, for all $T_{E_{ij}}$. There are two cases: $\mathcal{M}^D(pa(d'_{ij})) = g'_i$ or $\mathcal{M}^D(pa(d'_{ij})) = g'_j$. Suppose $\mathcal{M}^D(pa(d'_{ij})) = g'_i$. Then, observe that $\mathcal{M}^D(pa(d'_{ij}))$ and $\mathcal{M}^D(pa(d_{ij}))$ are in different gene trees, and we can assign $\tau(rt(T_{E_{ij}})) = rt(G_i)$ so that $\mathcal{M}^D(pa(d'_{ij})) \leq_{\mathcal{G}} \tau(rt(T_{E_{ij}}))$. Now, since $\mathcal{M}^D(pa(d'_{ij})) = g'_i$, we must have $v_i \in V \backslash V'$, which further implies that $\mathcal{M}^{\mathcal{G}}(rt(G_i)) = rt(S)$. Thus, since $rt(G_0)$ and $rt(G_i)$ both map to the same species node $rt(S)$, $rt(T_{E_{ij}})$ must be a valid transfer event. The case when $\mathcal{M}^D(pa(d'_{ij})) = g'_j$ is analogous.
4) $pa(rt(T_{E_{ij}})) \in \Delta^D$, for all $T_{E_{ij}}$. Observe that both children of $pa(rt(T_{E_{ij}}))$, for any $T_{E_{ij}}$, map to $rt(G_0)$. Furthermore, the node $pa(rt(T_{E_{ij}}))$ also maps to $rt(G_0)$. Thus, by the definition of DGS reconciliation, $pa(rt(T_{E_{ij}}))$ is a valid domain duplication node.
5) $rt(G_i) \in \Delta^{\mathcal{G}}$, for all $i \geq 0$. First, assume that $i > 0$. The two children of $rt(G_i)$ both map to the same species node, r_s, and $rt(G_i)$ itself maps to either $rt(S)$ or r_s. In both these case, by the definition of DGS reconciliation, $rt(G_i)$ is a valid gene duplication node. Now, suppose $i = 0$. Then, the two children of $rt(G_0)$ map to r_1 and r_2, while $rt(G_0)$ itself maps to $rt(S)$. Again, by the definition of DGS reconciliation, $rt(G_i)$ is a valid gene duplication node.

Thus, each internal node in D and $\mathcal{G}$ represents a valid reconciliation event. □

Lemma 2. *The reconciliation cost of reconciliation α is at most $5n + 4m - 4k + 4$.*

Proof. To calculate the reconciliation cost of α we will traverse through D and $\mathcal{G}$, adding up the total cost at each internal node. Since all events have been assigned an identical cost of 1, it suffices to count only the total number of events. We count the number of transfers, duplication, and losses separately, for each class of domain and gene nodes as given in the proof of Lemma 1 above.

Consider any domain node x of the form $pa(d'_{ij})$. This node represents a domain transfer event and, based on the mappings of x, $\tau(x)$, and of x's two children, does not create any domain loss events. Thus, this class of nodes invokes a total of m domain transfer events.

Consider any domain node x of the form $pa(d_{ij})$. This node represents a domain co-divergence event and, based on the mappings of x and its two children, does not create any domain loss events. Thus, this class of nodes invokes no countable events.

Consider any domain node x of the form $rt(T_{E_{ij}})$. This node represents a domain transfer event and, based on the mappings of x, $\tau(x)$, and of x's two children, leads to exact one domain loss event in either the gene tree G_i or G_j. Thus, this class of nodes invokes a total of m domain transfer events and m domain loss events.

Consider any domain node x of the form $pa(rt(T_{E_{ij}}))$. This node represents a domain duplication event and, based on the mappings of x and its two children, does not create any domain losses. Thus, this class of nodes invokes a total of $m - 1$ domain duplication events.

Finally, consider any domain node x of the form $rt(G_i)$, for $i \geq 0$. If $i = 0$, then x is a gene duplication node and, based on the mappings of x and its two children, invokes 4 gene losses on the species tree. If $1 \leq i \leq n$, then x is a gene duplication, but the number of gene losses depends on whether x maps to r_s or to $rt(S)$. For the case when x maps to r_s, there are no gene losses. For the case when x maps to $rt(S)$, there are 4 gene losses based on the mappings of x and its two children. Now, since $|V'| \geq k$, there can be at most $n - k$ vertices in the set $V \backslash V'$. Thus, the total number of distinct x's that map to $rt(S)$ is no more than $n - k$. This gives a total of $n + 1$ gene duplication events and at most $4(n - k)$ gene loss events for the nodes in this class.

Counting events over all domain and gene nodes, we get $2m$ domain transfers, $m-1$ domain duplications, m domain losses, $n+1$ gene duplications, and at most $4+4(n-k)$ gene losses. Thus, the total reconciliation cost is at most $5n+4m-4k+4$. □

Lemmas 1 and 2 together establish the forward direction of our proof.

Backward Direction. We will now prove that if there is a YES answer to the D-ODGS instance λ with parameter $N = 5n + 4m + 4 - 4k$ then there is a YES answer to the independent set instance ϕ with parameter k.

Let α^* denote the DGS reconciliation on instance λ with cost at most $N = 5n + 4m + 4 - 4k$. We will show how to construct an independent set V' of size at least k, based on α^*. Specifically, we add vertex v_i to the set $V \backslash V'$ if and only if there exists $T_{E_{ij}}$ or $T_{E_{ji}}$ in which $(pa(d'_{ij}), d'_{ij}) \notin \Xi$.

We first state several simple but useful lemmas on the structure of any optimal DGS reconciliation for instance λ of D-OGDS in our gadget.

Lemma 3. *Consider any $d \in I(D)$ and let d' and d'' denote its two children. If $\mathcal{M}^D(d') \in G_i$ and $\mathcal{M}^D(d'') \in G_j$, for $i \neq j$, then $d \in \Theta$ in any DGS reconciliation. Furthermore, $\mathcal{M}^D(d) \in G_i$ or $\mathcal{M}^D(d) \in G_j$.*

Proof. Node d is clearly not in Σ^D or Δ^D due to constraints 6(a) and 6(b) in Definition 1. Thus, $d \in \Theta$. By constraint 4(b), $\mathcal{M}^D(d)$ is either ancestral to $\mathcal{M}^D(d')$ or to $\mathcal{M}^D(d'')$, and so $\mathcal{M}^D(d) \in G_i$ or $\mathcal{M}^D(d) \in G_j$. □

Lemma 4. *Consider any $d \in I(D)$ and let d' and d'' denote its two children. If $\mathcal{M}^D(d') = \mathcal{M}^D(d'')$, then $d \in \Delta^D$ in any DGS reconciliation.*

Proof. Node d is clearly not in Σ^D or Θ due to constraints 6(a) and 6(c) in Definition 1, so $d \in \Delta^D$. □

Lemma 5. *Consider any $d \in I(D)$ and let d' and d'' denote its two children. If $\mathcal{M}^D(d') \in G_i$ and $\mathcal{M}^D(d'') \in G_i$, for some i, then $\mathcal{M}^D(d) \in G_i$ in any valid DGS reconciliation.*

Proof. Suppose $\mathcal{M}^D(d) \notin G_i$, then $\mathcal{M}^D(d)$ is incomparable with $lca(\mathcal{M}^D(d'), \mathcal{M}^D(d''))$, and thus $d \notin \Sigma^D$, $d \notin \Delta^D$, and since $\mathcal{M}^D(d)$ is ancestral to neither $\mathcal{M}^D(d')$ nor $\mathcal{M}^D(d'')$, $d \notin \Theta$. This is a contradiction because Σ^D, Δ^D and Θ partition $I(D)$. □

Lemma 6. *Consider any node d of the form $pa(d'_{ij})$. Then, $d \in \Theta$ in any DGS reconciliation.*

Proof. The two children of d, i.e., d'_{ij} and d'_{ji}, are both leaf nodes and map to different gene trees. Following Lemma 3, d must be in Θ in any DGS reconciliation. □

Lemma 7. *Consider any node d of the form $rt(T_{E_{ij}})$. Then, $d \in \Theta$ in any DGS reconciliation.*

Proof. The two children of d are $pa(d'_{ij})$ and $pa(d_{ij})$. By Lemma 6, we know that $pa(d'_{ij})$ must map to a node in either G_i or G_j, where $i, j \neq 0$. Moreover, since both children of $pa(d_{ij})$ map to nodes in G_0, the node $pa(d_{ij})$ itself must also map to a node in G_0. Lemma 3 therefore applies and implies that $d \in \Theta$. □

Lemma 8. *For any edge $E_{ij} \in E$, we must have $\mathcal{M}^D(rt(T_{E_{ij}})) \in \{rt(G_0), rt(G_i), rt(G_j)\}$ in any DGS reconciliation.*

Proof. Since the subtree $T_{E_{ij}}$ only contains leaves that map to nodes from G_0, G_i or G_j, the root of this subtree, $rt(T_{E_{ij}})$, can only map to a node from those three gene trees. Observe that node $pa(d_{ij})$ can only map to $rt(G_0)$, since if it maps to either g_1 or g_2 then $pa(d_{ij})$ must be a transfer event invoking a domain transfer between g_1 and g_2, but g_1 and g_2 map to different species nodes. Thus, if $rt(T_{E_{ij}})$ maps to a node in G_0, the mapping must be to $rt(G_0)$. Now suppose $rt(T_{E_{ij}})$ maps to a node from either G_i or G_j. From Lemma 7 we know that $rt(T_{E_{ij}}) \in \Theta$. Thus, if $rt(T_{E_{ij}})$ maps to a node of G_i or G_j, then $\tau(rt(T_{E_{ij}})) = rt(G_0)$. Now, if $rt(T_{E_{ij}})$ maps to a leaf node of G_i or G_j, then $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(rt(T_{E_{ij}}))) = r_s$. By the species constraint on transfer events, we must have $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(rt(T_{E_{ij}}))) = \mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}})))$. This means that $\mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}}))) = r_s$, which is a contradiction since $\tau(rt(T_{E_{ij}})) = rt(G_0)$ and $rt(G_0)$ can only map to nodes $pa(r_1)$ or $rt(S)$ (i.e., nodes ancestral to r_1 and r_2) in the species tree. Thus, $rt(T_{E_{ij}})$ cannot map to a leaf node of G_i or G_j and the lemma follows. □

Lemma 9. *We must have $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = rt(S)$ in any DGS reconciliation.*

Proof. Observe that the two children of $rt(G_0)$ map to nodes r_1 and r_2 in the species tree. Thus, according to the DGS reconciliation model, $rt(G_0))$ has only two possible mappings: Either $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = pa(r_1)$ or $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = rt(S)$. Suppose $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = pa(r_1)$. Let $E_{ij} \in E$ be any arbitrary edge. Observe that node $pa(d_{ij})$ can only map to $rt(G_0)$, since if it maps to either g_1 or g_2 then $pa(d_{ij})$ must be a transfer event invoking a domain transfer between g_1 and g_2, but g_1 and g_2 map to different species nodes. From Lemma 7 we know that $rt(T_{E_{ij}}) \in \Theta$. From Lemma 8, we know that $\mathcal{M}^D(rt(T_{E_{ij}})) \in \{rt(G_0), rt(G_i), rt(G_j)\}$, and we consider these three cases separately. Suppose $\mathcal{M}^D(rt(T_{E_{ij}})) = rt(G_0)$. In this case, we must have $\tau(rt(T_{E_{ij}})) \in V(G_i) \cup V(G_j)$. By the species constraint for domain transfers, we must also have $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(rt(T_{E_{ij}}))) = \mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}})))$. Since $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(rt(T_{E_{ij}})))$ must be ancestral to the nodes r_1 and r_2 in the species tree, and $\mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}})))$ must be ancestral to r_s in the species tree, the only way $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(rt(T_{E_{ij}}))) = \mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}})))$ is if $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = rt(S)$. Now, suppose $\mathcal{M}^D(rt(T_{E_{ij}})) = rt(G_i)$. In this case, we must have $\tau(rt(T_{E_{ij}})) = rt(G_0)$. Since $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(rt(T_{E_{ij}})))$ must be ancestral to the node r_s in the species tree, and $\mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}})))$ must be ancestral to nodes r_1 and r_2 in the species tree, the only way $\mathcal{M}^{\mathcal{G}}(\mathcal{M}^D(rt(T_{E_{ij}}))) = \mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}})))$ is if $\mathcal{M}^{\mathcal{G}}(\tau(rt(T_{E_{ij}}))) = rt(S)$, i.e., if $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = rt(S)$. The case when $\mathcal{M}^D(rt(T_{E_{ij}})) = rt(G_j)$ is analogous, and the lemma follows. □

Lemma 10. *For any edge $E_{ij} \in E$, we must have $pa(rt(T_{E_{ij}})) \notin \Sigma^D$ in any DGS reconciliation.*

Proof. Consider any $pa(rt(T_{E_{ij}}))$. As shown in Lemma 8, $\mathcal{M}^D(rt(T_{E_{ij}})) \in \{rt(G_0), rt(G_i), rt(G_j)\}$, i.e., $rt(T_{E_{ij}})$ only maps to the root node of a gene tree. Thus, if $pa(rt(T_{E_{ij}})) \in \Sigma^D$, then $\mathcal{M}^D(pa(rt(T_{E_{ij}})))$ must be ancestral to $\mathcal{M}^D(rt(T_{E_{ij}}))$, which is only possible if $\mathcal{M}^D(pa(rt(T_{E_{ij}}))) = \mathcal{M}^D(rt(T_{E_{ij}}))$. However, this is a contradiction since $\mathcal{M}^D(pa(rt(T_{E_{ij}}))) = \mathcal{M}^D(rt(T_{E_{ij}}))$ implies that $pa(rt(T_{E_{ij}})) \in \{\Delta^D, \Theta\}$. The lemma follows. □

Lemma 11. *We must have $rt(G_i) \in \Delta^{\mathcal{G}}$, for $0 \leq i \leq n$, in any DGS reconciliation.*

Proof. Consider the case when $i \neq 0$. In this case, the two children of $rt(G_i)$ map to the same species node r_s. Thus, $rt(G_i) \in \Delta^{\mathcal{G}}$ by Lemma 4. Now, consider the case when $i = 0$. In this case, we know that the two children of $rt(G_0)$ map to leaves r_1 and r_2 of the species tree, and we know from Lemma 9 that $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = rt(S)$. Since $rt(S)$ is ancestral to both r_1 and r_2, but $rt(S) \neq lca(r_1, r_2)$, by Definition 1, $rt(G_0)$ $\not{i}n\Sigma^{\mathcal{G}}$. Consequently, $rt(G_0) \in \Delta^{\mathcal{G}}$ and the lemma follows. □

Recall that we construct the independent set V' by adding vertex v_i to the set $V \backslash V'$ if and only if there either exists a $T_{E_{ij}}$ or a $T_{E_{ji}}$ in which $(pa(d'_{ij}), d'_{ij}) \notin \Xi$. The vertices not added to $V \backslash V'$ are included in V'.

Lemma 12. *We must have $|V'| \geq k$.*

Proof. We will compute a lower bound on the reconciliation cost by summing up the evolutionary events implied by Lemmas 6 through 11. From Lemma 10, we have either a domain duplication or transfer event on each $pa(rt(T_{E_{ij}}))$, giving a total of $m - 1$ events. From Lemmas 6 and 7, we have a total of $2m$ transfer events on nodes of the form $rt(T_{E_{ij}})$ and $pa(d'_{ij})$. From Lemma 11 we have a total of $n + 1$ gene duplications for nodes of the form $rt(G_i)$, and

4 gene losses under the gene duplication event at $rt(G_0)$. Now, consider nodes of the form $pa(d'_{ij})$ in the domain tree. From Lemma 9 we know that each such node is a transfer event. If $pa(d'_{ij})$ maps to a leaf node then there will be no losses associated with that transfer event but, by Lemma 8, it will create one domain loss under the transfer event on $rt(T_{E_{ij}})$. Similarly, if $pa(d'_{ij})$ maps to $rt(G_i)$ or $rt(G_j)$, then there are two domain losses associated with that transfer event. Thus, irrespective of the mapping of $pa(d'_{ij})$, there is at least one domain loss for each node of that form. This gives a lower bound of m domain losses overall for those nodes. Summing up the total number of events counted so far we get a lower bound of $4m+n+4$. Note that this count does not account for any gene losses on G_i, for any $i > 0$. However, since α is a reconciliation whose cost is at most $5n + 4m + 4 - 4k$, the total number of evolutionary events not counted in the sum above can be at most $4(n-k)$. Thus, it now suffices to prove that every distinct vertex in $V \backslash V'$ adds to the uncounted event count by 4.

Let v_i be an arbitrary vertex from $V \backslash V'$. Then, by our construction of the set $V \backslash V'$, there exists either $T_{E_{ij}}$ or $T_{E_{ji}}$ in which $(pa(d'_{ij}), d'_{ij}) \notin \Xi$. Let us assume that there exists $T_{E_{ij}}$ in which $(pa(d'_{ij}), d'_{ij}) \notin \Xi$. Thus, $\mathcal{M}^D(pa(d'_{ij})) \in G_i$. Then, consider the transfer event at node $rt(T_{E_{ij}})$. Based on Lemma 8, we know that that $\mathcal{M}^D(rt(T_{E_{ij}}))$ must be either $rt\,G_0$ or $rt\,G_i$. Observe that if $\mathcal{M}^D(rt(T_{E_{ij}})) = rt\,G_0$, then $\tau(rt(T_{E_{ij}})) = rt\,G_i$. Likewise, if $\mathcal{M}^D(rt(T_{E_{ij}})) = rt\,G_i$, then $\tau(rt(T_{E_{ij}})) = rt\,G_0$. For either of these two possibilities, by the species constraint on transfer events, we must have $\mathcal{M}^{\mathcal{G}}(rt(G_i)) = \mathcal{M}^{\mathcal{G}}(rt(G_0))$. Furthermore, by Lemma 9, we have $\mathcal{M}^{\mathcal{G}}(rt(G_0)) = rt(S)$, and consequently $\mathcal{M}^{\mathcal{G}}(rt(G_i)) = rt(S)$. The mapping of $rt(G_i)$ into the species tree therefore creates 4 losses (not counted before). The case when there exists $T_{E_{ji}}$ instead is analogous. This proves the Lemma. □

Lemma 13. *V' is an independent set.*

Proof. Suppose there is an edge E_{ij} where $v_i, v_j \in V$. Consider the node $pa(d'_{ij}) \in T_{E_{ij}}$. According to the constraint on transfer events, $(pa(d'_{ij}), d'_{ij})$ and $(pa(d'_{ij}), d'_{ji})$ cannot both be transfer edges. Thus either vertex v_i or vertex v_j will be selected into $V \backslash V'$, which is a contradiction to $v_i, v_j \in V$. □

Lemmas 12 and 13 together establish the backward direction of our proof.

4 Solving the ODGS Problem

A possible heuristic for the ODGS problem is to simply compute an optimal DTL reconciliation between D and $\mathcal{G}$, and an optimal reconciliation from $\mathcal{G}$ to S, separately. However, such reconciliations are seldom valid DGS reconciliations since the constraint that donor and recipient genes of any domain transfer event must be from the same species is frequently violated. Another possibility is to first compute an optimal reconciliation between $\mathcal{G}$ and S, and then find a reconciliation between D and $\mathcal{G}$ that satisfies the species constraints imposed by the reconciliation between $\mathcal{G}$ and S. (This is similar to the problem formulation used in [35], restricted to a single gene tree.) However, we observed that this approach does not yield a valid DGS reconciliations for a large fraction of the domain trees in our dataset (detailed experimental results appear in the next section). This is because an optimal duplication-loss reconciliation between $\mathcal{G}$ and S can make it impossible to compute a valid DGS reconciliation by preventing crucial domain transfer events from occurring. This is illustrated in Figure 3. Furthermore, even if a valid DGS reconciliation can be computed using this approach, this DGS reconciliation need not be optimal due to the constraints on domain transfer events imposed by an optimal (or any fixed) reconciliation of $\mathcal{G}$ and S, possibly leading to a highly suboptimal reconciliation between D and $\mathcal{G}$. This is illustrated in Figure 3(b).

To overcome these difficulties, we designed a heuristic that simultaneously optimizes the domain-gene and gene-species mappings. The heuristic uses dynamic programming and is based on the idea that domain transfer events are relatively rare and unlikely to affect the same gene node more than once. The heuristic makes use of an extended version of the traditional DTL reconciliation algorithm [3] to solve an *extended-DTL reconciliation* problem. The extended version of this algorithm, which we will call the *extended-DTL algorithm*, is used to reconcile D with $\mathcal{G}$, given a fixed mapping from $\mathcal{G}$ to S. The extension is required to simultaneously handle multiple gene trees and to only allow reconciliations that respect the species constraint on domain transfer events. Given any $d \in I(D)$, $g \in V(\mathcal{G})$, and mapping $\mathcal{M}^{\mathcal{G}}$ between the gene trees and species tree, we define $c(d, g, \mathcal{M}^{\mathcal{G}})$ to be the cost of an optimal reconciliation of $D(d)$ with $\mathcal{G}$ such that d maps to g and any invoked domain transfer events respect the species constraints imposed by $\mathcal{M}^{\mathcal{G}}$. The extended-DTL algorithm computes the values $c(d, g, \mathcal{M}^{\mathcal{G}})$ using a dynamic programming framework that uses a nested post-order traversal of D and the gene trees in $\mathcal{G}$. The extended-DTL algorithm is built upon the DTL reconciliation algorithm described in [3] (which computes optimal DTL reconciliations between gene trees and species trees), and requires only a few changes. Additional algorithmic details appear in Section S2 in the Supplement.

Dynamic programming heuristic for DGS reconciliation. Our heuristic uses a modified version of the dynamic programming algorithm used to solve the extended-DTL reconciliation problem. Specifically, in the extended-DTL algorithm, while computing any $c(d, g, \mathcal{M}^{\mathcal{G}})$ value, domain transfer events are only considered when the receiver and donor map to the same species node according to the mapping $\mathcal{M}^{\mathcal{G}}$. In the heuristic, this algorithm is modified to allow any transfer event, but with an extra cost penalty equal to the minimum cost of modifying the mapping $\mathcal{M}^{\mathcal{G}}$ to enable that particular transfer event. Thus, the computed $c(d, g, \mathcal{M}^{\mathcal{G}})$ values consider all possible domain transfer events and also include the cost of modifying the LCA mapping to enable the considered domain transfers. Note that this heuristic computes an upper bound on the actual DGS reconciliation cost, i.e., it need not always compute optimal DGS-reconciliations. This is because the heuristic adds the extra gene trees to species tree reconciliation cost individually for each domain transfer event, when in an optimal DGS reconciliation multiple domain transfer events may benefit from the same change in the gene trees to species tree mapping. After all the modified $c(d, g, \mathcal{M}^{\mathcal{G}})$

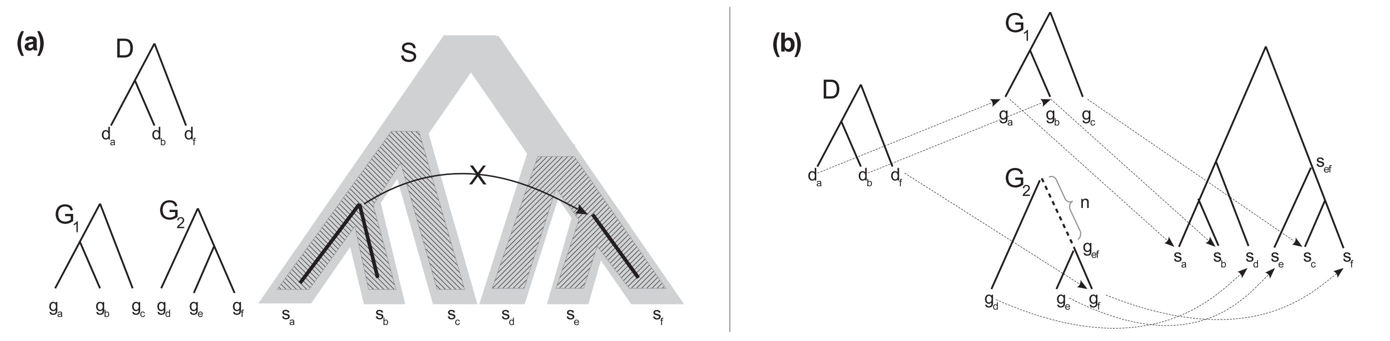

Fig. 3. Part (a) shows a simple example where imposing an optimal reconciliation between $\mathcal{G}$ and S precludes the possibility of computing a DGS reconciliation. The three trees on the left are the domain tree and the two gene trees in which this domain family is found. The solid gray tree on the right is a species tree with six leaf nodes, and the shaded parts inside depict the embeddings of the two gene trees under an optimal (LCA) reconciliation of the gene trees and species tree. The solid lines inside the gene tree embeddings represent the domain tree. The required domain transfer event from one gene tree to the other is not possible because the gene trees have no overlapping species nodes, making it impossible to compute a DGS reconciliation. Part (b) shows an example where imposing an optimal reconciliation between $\mathcal{G}$ and S still allows for a DGS reconciliation to be computed but where the resulting DGS reconciliation is highly suboptimal. The leaf node mappings are shown by dashed lines. The root of the domain tree must be a domain transfer event, and if we fix an optimal (LCA) mapping for both gene trees, there must be x domain losses on G_2 because the node g_{ef} can be neither a receiver nor a donor for this domain transfer. On the other hand if we map g_{ef} to the root of the species tree, then these x domain losses can be avoided.

values have been computed, it may be necessary to fix any conflicting assignments for gene node mappings. Thus, the heuristic also has a second stage in which it traverses the nodes of $\mathcal{G}$ and fixes any conflicting assignments while minimizing the reconciliation cost between $\mathcal{G}$ and S. Observe that this dynamic programming heuristic always find a DGS reconciliation (if one exists). Detailed pseudocode for this heuristic appears in Section S3 in the supplement.

Running time and time complexity. Suppose we are given as input a domain tree D, associated gene trees $\mathcal{G}$, and species tree S. As shown in Section S3 in the supplement, our heuristic has a time complexity of $O(|Le(\mathcal{G})|^2 \cdot (|Le(\mathcal{G})| + |Le(S)| + |Le(D)|))$. This is efficient enough to be applied to even very large trees. For instance, on our data set of 3761 domain trees and 7165 gene trees from 12 fly species (described in the next section), our heuristic runs within a few seconds on almost all trees in the dataset. Details appear in Figure S1 in the supplement. Even on a very large input instance with a domain tree containing 350 leaves and associated gene trees containing a total of over 1000 leaves, the heuristic took only 64 seconds. This running time analysis was performed using a single core on a commodity desktop computer with a 3.40 GHz Quad-Core processor and 8 GB of RAM.

Evaluation of the heuristic. A potential downside of our dynamic programming heuristic is that it may "over-correct" (i.e., over-modify) the gene-species mapping in favor of the domain-gene mapping. Since the ODGS problem is NP-hard, it is not feasible to compute optimal solutions for most domain trees in our dataset. Thus, to evaluate the performance of the dynamic programming heuristic, we designed a second heuristic based on local search and compared its performance (in terms of DGS reconciliation cost) to that of our heuristic. The local search heuristic works as follows: The local search starts with an optimal duplication-loss mapping (also known as a Lowest Common Ancestor (LCA) mapping or Most Recent Common Ancestor (MRCA) mapping), $\mathcal{M}^{\mathcal{G}}$, from $\mathcal{G}$ to S, and iteratively improves this mapping until no better solution can be found. It first computes an optimal extended-DTL reconciliation between D and $\mathcal{G}$, given the LCA mapping $\mathcal{M}^{\mathcal{G}}$, if such a reconciliation exists. It then defines a local search neighborhood around the current mapping $\mathcal{M}^{\mathcal{G}}$ by considering all mappings that can be obtained by moving the mappings of up to two gene nodes upwards towards the species tree root by 1, 2, or 3 edges. This creates a local search neighborhood, denoted $N(\mathcal{M}^{\mathcal{G}})$, of $\Theta(|Le(\mathcal{G})|^2)$ alternative mappings around the current gene-species mapping. The heuristic then computes an extended-DTL reconciliation for each mapping in the set $N(\mathcal{M}^{\mathcal{G}})$, and updates $\mathcal{M}^{\mathcal{G}}$ to be that mapping from $N(\mathcal{M}^{\mathcal{G}})$ that gives lowest total DGS reconciliation cost. The local search stops when a lower cost DGS reconciliation cannot be found in the current local search neighborhood, and the best solution found so far is returned. Thus, this local search heuristic can be expected to perform very well whenever an optimal DGS reconciliation is obtained through a near-optimal mapping between $\mathcal{G}$ and S. By comparing the relative performance of our dynamic programming heuristic with that of the local search method, it is possible to assess if the dynamic programming heuristic is prone to over-correction of the gene-species mapping in favor of the domain-gene mapping.

We found that the dynamic programming heuristic easily outperformed the local search heuristic. Specifically, we found that both heuristics performed equally well on 78.3% of the domain trees in our dataset (described in the next section) and that the dynamic programming heuristic outperformed the local search heuristic on 98% of the remaining domain trees. These results demonstrate that our heuristic is not prone to over-correction of the gene-species mapping in favor of the domain-gene mapping, which suggests that the heuristic should be able to find optimal or near-optimal DGS reconciliations in most cases.

5 Experimental Evaluation

To evaluate the impact of our new model, we constructed a genome-scale dataset of gene families and domain families from 12 fly species [43]. We selected all 7165 gene trees containing at least one fly gene with at least one Pfam A domain (see below) from the set of gene trees constructed in [43], and deleted all non-fly genes from these gene trees. We mapped the Flybase gene IDs used in these gene trees to UniProt gene IDs; overall, 164500 of the 165101 fly genes in the gene trees could be assigned a UniProt ID. We used these UniProt gene IDs to search the Pfam A database [13] for domain sequences within each gene, resulting in a total of 4114 domain families. The resulting domain families contained 55.7 domain sequences on average. Of the 165101 genes with a UniProt ID, 111664 contained at least one domain from the Pfam A database. On average, each gene contained 1.4 domains. Each gene family contained an average of 1.68 domain families, and each domain family is associated with 2.93 gene families on average.

Domain trees, gene trees, and minimizing domain tree error. Errors in domain trees and gene trees directly impact the accuracy of any reconciliation method, so we used state-of-the-art methods to compute these trees as accurately as possible. The gene trees in our dataset were computed using maximum-likelihood phylogeny reconstruction software RAxML [32], applied to amino acid sequences using the JTT substitution model (with gamma distributed rates) and thorough search settings with multiple replicate searches, and error-corrected using the state-of-the-art error-correction technique TreeFix [44]. A detailed description of the construction, error-correction, and rooting of these gene trees appears in [43]. These gene trees can be assumed to be fairly accurate overall. The domain trees were constructed using the same approach as with the gene trees. Specifically, we used RAxML [32], applied again to amino acid sequences using the JTT substitution model (with gamma distributed rates) and thorough search settings with multiple replicate searches, to compute the maximum likelihood estimates of the domain trees, and then error-corrected them using TreeFix [44]. TreeFix uses the species tree to error-correct a given maximum likelihood gene or domain tree topology by maximizing its "fit" with the species tree in terms of the duplication-loss reconciliation cost, while ensuring that the error-corrected tree is equally well-supported by the sequence data, and is known to be among the most effective methods for error-correction [44]. Observe that, with respect to the species tree, domain trees only evolve through domain duplication and domain loss (no domain transfer due to the species constraint on domain transfer events), and TreeFix has been shown to greatly reduce the error-rate in such cases [44]. Given that domain trees are harder to reconstruct accurately than gene trees, the use of an error-correction technique like TreeFix becomes all the more important for domain trees. Thus, we used the set of TreeFix-corrected domain trees as our primary set of domain trees in all our experiments. However, to assess the impact of domain tree error on DGS reconciliation we also used the (uncorrected) maximum likelihood domain trees obtained from RAxML.

After discarding domain families that were either too small (containing less than 3 sequences) or too large to analyze efficiently with RAxML and TreeFix (containing more than 500 domain sequences), and also removing those domain trees that did not admit a valid DGS reconciliation due to the issue of single-leaf gene trees previously discussed in Section 2 (which constituted less than 2% of the domain trees in the dataset), we obtained a set of 3761 domain families for which both RAxML and TreeFix trees were available. Each domain tree (in both the TreeFix and RAxML sets) was rooted by solving the DGS-reconciliation problem for all possible rootings of that domain tree, and identifying the rooting that gave the least DGS-reconciliation cost. Somewhat surprisingly, we found that each of the TreeFix and RAxML domain trees had a unique optimal rooting. We point out that this is among the largest datasets ever analyzed using a phylogenetic approach for understanding domain evolution. For our analysis we used event cost 1 for $P^{\mathcal{G}}_{loss}$ and P^{D}_{loss}, 2 for $P^{\mathcal{G}}_{\Delta}$ and P^{D}_{Δ}, 4 for $P^{D}_{\Theta1}$, and 6 for $P^{D}_{\Theta2}$. This choice of event costs is inspired by the use of similar cost assignments in the DTL reconciliation model.

Unsurprisingly, we observed a large difference in DGS reconciliation costs when using the RAxML domain trees and TreeFix domain trees. Specifically, there was a dramatic 62.2% reduction in DGS reconciliation cost on average when TreeFix domain trees were used instead of RAxML domain trees. Figure S2 in the supplement plots the DGS reconciliation costs for a subset of the dataset using the RAxML, TreeFix, and randomized domain trees. Throughout the remainder of this section we use the TreeFix domain trees as our primary set of domain trees for all experimental analyses, but, for completeness, also provide corresponding results for the RAxML trees in parentheses.

5.1 Results

Impact on inference of domain evolution. We first assessed the impact of our model on understanding domain family evolution. Specifically, we used our heuristic to compute a DGS reconciliation for each domain family with its associated gene trees and the species tree, and compared the inferred domain-gene reconciliation with the domain-gene reconciliation inferred using the standard DTL reconciliation model which does not model the interdependence of domain-, gene-, and species-level evolution (e.g., [33]). We observed that 34.1% of the TreeFix domain trees (72.8% for RAxML domain trees) had different domain-gene reconciliation costs (and hence different reconciliations) under these two models. On average, there was a 5.8% difference in reconciliation costs over the entire TreeFix dataset (30.5% for the RAxML domain trees), and a 19.6% (38.1% for RAxML domain trees) difference when considering only the domain trees with different reconciliation costs. Further details appear in Figure 4(a).

We also measured the impact of DGS reconciliation on the domain-gene mapping. Averaging across all domain trees, we observed that 11.2% of the internal nodes in the TreeFix domain trees (25.4% for RAxML domain trees) were assigned differently under two models. This had a large impact on the inference of domain transfer events, with an average of 66.7% (76.3% for RAxML domain trees) reduction in the number of domain transfer events. In absolute terms, summed over the entire data set, the 20,916 domain transfer events inferred by the DTL reconciliation model on the TreeFix domain trees (58,761 for RAxML domain trees) decreased to just 5,308 domain transfer events (8,641 for

RAxML domain trees) under the DGS reconciliation model, which is reduction of 74.6% (85.6%). This dramatic decrease in the number of inferred domain transfer events illustrates the large impact DGS reconciliation can have in practice.

Impact on inference of gene evolution. Next we assessed the impact of DGS reconciliation on understanding gene family evolution. Specifically, we checked how often an optimal DGS reconciliation did not follow the optimal Duplication-Loss mapping, or LCA mapping, from the gene trees to the species tree. We observed that DGS reconciliations for 24.6% of the TreeFix domain trees (22.8% for RAxML domain trees) deviated from the LCA gene-species mapping. In these cases, imposing an LCA gene-species mapping either makes it impossible to compute a DGS reconciliation or leads to a suboptimal DGS reconciliation. Averaging across all domain trees, there was a 5.4% increase in the reconciliation cost between the gene trees and species tree when using the TreeFix domain trees (6.2% for RAxML domain trees), and a 22% (28.2% for RAxML domain trees) increase when averaging only across the domain trees for which the LCA gene-species mapping was suboptimal. Further details appear in figure 4(c). For the gene trees that do not follow the LCA mapping, there were an average of 1.8 (1.9 for RAxML domain trees) gene nodes that moved, on average, 3.6 (also 3.6 for RAxML domain trees) nodes higher than their LCA mappings on the species tree. This shows that DGS reconciliation can significantly impact the reconciliation between gene trees and species trees, directly affecting the inference of gene family evolution.

Comparison to Existing Methods. The three methods most comparable to this work are STAR-MP [45], the plexus model [42], and the phylogenetic reconciliation based approach of Stolzer et al. [33], [35]. Since these methods all focus on reconstructing different aspects of domain evolution, we focus our attention on comparing those aspects of the methods that can be directly compared against the DGS reconciliation model. All comparisons are based on applying the different methods to the same 3761 TreeFix (and RAxML) domain trees. Among the three methods, we were unable to compare against the plexus model since an implementation is not available.

Comparison to the approach of Stolzer et al. The approach of Stolzer et al. uses the DTL reconciliation model to reconcile a domain tree with a gene tree, given a fixed reconciliation between the gene tree and species tree. In our comparison against this method, we focus on the impact of using this fixed gene-to-species mapping on the computed domain-gene and gene-species reconciliations, as opposed to computing a *joint* reconciliation of the domain, gene, and species trees as in DGS reconciliation. Since the method of Stolzer et al. implements the traditional DTL reconciliation model which can only consider a singe gene tree at a time, we reimplemented their approach using the extended-DTL algorithm. Our reimplementation uses the same approach as Stolzer et al., just appropriately extended to enable it to handle domain trees evolving in multiple gene trees. To enable a direct comparison between the two approaches, we fixed the gene-species reconciliation to be the optimal Duplication-Loss mapping, also called the LCA mapping, as appropriate for our eukaryotic gene families.

We first checked how often imposing the optimal reconciliation (LCA mapping) between the gene trees and species tree in the approach of Stolzer et al. [35], makes it impossible to compute a DGS reconciliation. We found that in 9.8% of the TreeFix domain trees (11.2% RAxML domain trees), it is not possible to reconcile the domain tree with the gene trees (under the species constraint on domain transfer events) because the imposed LCA mapping on gene trees makes it impossible to invoke necessary domain-transfer events. Additionally, in 14.6% of the TreeFix domain trees (11.6% for RAxML domain trees) use of the LCA mapping led to suboptimal DGS reconciliations. On these 14.6% (11.6%) domain trees, we observed a 15.1% (8.8%) decrease in the DGS reconciliation cost when the gene to species mapping is allowed to deviate from the LCA mapping. Further details appear in figure 4(b). Overall, this analysis shows that while the LCA mapping from gene trees to species trees is often optimal even for DGS reconciliation, in many cases (24.6% of TreeFix domain trees, 22.8% for RaxML trees) imposing the LCA mapping either makes it impossible to reconcile the domain trees with the gene trees and species tree (almost 10% of the TreeFix trees) or results in a significantly suboptimal DGS reconciliation. Furthermore, as seen earlier, optimal DGS reconciliations for such domain trees require significant deviations from the LCA gene-species mapping.

Comparison with ancestral domain contents computed by STAR-MP. The STAR-MP method [45] computes evolutionary histories of domain architectures by first mapping the domain trees onto a species tree to infer the domain content of all ancestral species. STAR-MP uses the traditional duplication-loss model (LCA mapping) to compute this mapping and the resulting ancestral domain content. Since DGS-reconciliation does not compute domain architectures, we focused on comparing the impact of using DGS-reconciliation on inferring ancestral domain contents. We therefore computed ancestral domain contents on the species tree using the DGS reconciliation model applied to all 3761 TreeFix (and RAxML) domain trees, and compared the inferred domain contents to those computed using the LCA mapping (used by STAR-MP) on the same datasets. We found that the inferred domain contents (domain to species mappings) are different for 33.0% of our TreeFix domain trees (38.5% for RAxML trees). In fact, for each of these domain trees, there are on average 4.4 (also 4.4 for RAxML domain trees) species nodes, out of the 23 nodes in the species tree, for which the inferred domain contents are different across the two methods. Details appear in Figure 5. These results again highlight the very significant impact of using DGS reconciliation on inferring domain evolution and on reconstructing ancestral domain architectures.

An implementation of our heuristic is freely available from http://compbio.engr.uconn.edu/software/seadog/.

6 Conclusion

In this work, we have introduced a new computation framework, the DGS reconciliation model, for integrated analysis of domain-, gene-, and species-level evolution. Our DGS reconciliation model is the first computational framework that explicitly captures the interdependence of domain-, gene-, and species-level evolution, and simultaneously

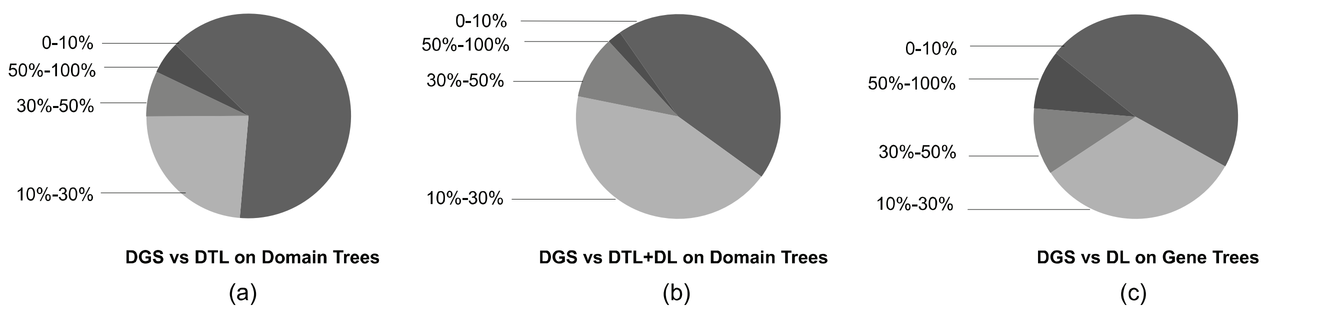

Fig. 4. (a) Distribution of the difference in reconciliation costs for domain-gene reconciliations computed using the conventional DTL reconciliation model and the DGS reconciliation model for all TreeFix domain trees in our dataset. (b) Distribution of the difference in reconciliation costs for domain-gene reconciliations computed using the model of Stolzer et al. (using the extended-DTL reconciliation model with a fixed LCA mapping from the gene trees to the species tree) and the DGS reconciliation model for the 14.6% of TreeFix domain trees for which enforcing the LCA gene-species mapping yields a suboptimal DGS reconciliation. (c) Distribution of the difference in reconciliation costs for gene-species reconciliations computed using the conventional DL reconciliation model and the DGS reconciliation model for all TreeFix domain trees in our dataset.

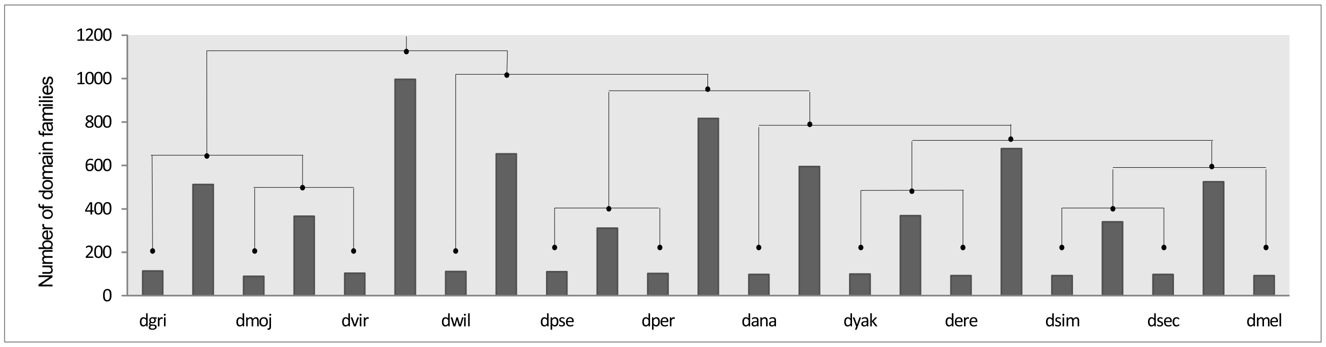

Fig. 5. This figure shows the 12-flies species tree used in this study and, for each node of the species tree, the total number of TreeFix domain trees for which the DGS and DL reconciliation models infer different copy numbers for that domain on that species node.

optimizes the evolution of domains within genes, and genes within species. We show that the underlying computational problem is NP-hard and present an efficient and effective heuristic for the problem. We applied our heuristic to a large genome-scale dataset of thousands of domain and gene families from 12 fly species, and demonstrated the significant impact of our new model on the inference of both domain-level and gene-level evolution. Our results suggest that explicit consideration of the interdependence of domain, gene, and species-level evolution will enable biologists to infer both domain family evolution and gene family evolution more accurately.

Several aspects of the new DGS reconciliation framework need to be explored further. For example, it would be useful to allow horizontal gene transfer so that the framework can be applied to microbial species. It would also be useful to study the prevalence of multiple optimal reconciliations and the effect of varying event cost assignments on evolutionary inferences.

We view the proposed DGS reconciliation model as a foundation on which other more complex and more complete models can be built. For instance, the model can be extended to simultaneously reconcile multiple domain trees with the gene trees and species tree, and doing so may yield more accurate results as well as a better understanding of multi-domain gain and loss. Likewise, consideration of domain architectures (orderings of domains along proteins) and mechanisms of domain duplication, transfer, and loss would further help improve the accuracy of the framework and provide an even more fine-grained view of domain and gene evolution. Finally, the DGS reconciliation model can be leveraged to develop a joint error-correction framework for gene trees and domain trees. This would lead to improved reconstruction of multi-domain gene trees and of domain trees that might otherwise be hard to reconstruct accurately.

Acknowledgments

The authors would like to thank Yi-Chieh Wu for many helpful discussions.

Funding: This work was supported in part by National Science Foundation CAREER award 1553421 and by startup funds from the University of Connecticut to MSB.

References

[1] O. Äkerborg, B. Sennblad, L. Arvestad, and J. Lagergren. Simultaneous Bayesian gene tree reconstruction and reconciliation analysis. *P. Natl. Acad. Sci. USA*, 106(14):5714–5719, 2009.

[2] S. Arena, S. Benvenuti, and A. Bardelli. Genetic analysis of the kinome and phosphatome in cancer. *Cellular and Molecular Life Sciences*, 62(18):2092–2099, 2005.

[3] M. S. Bansal, E. J. Alm, and M. Kellis. Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss. *Bioinformatics*, 28(12):283–291, 2012.

[4] M. S. Bansal, Y.-C. Wu, E. J. Alm, and M. Kellis. Improved gene tree error correction in the presence of horizontal gene transfer. *Bioinformatics*, 31(8):1211–1218, 2015.

[5] A. Bateman, E. Birney, R. Durbin, S. R. Eddy, R. D. Finn, and E. L. L. Sonnhammer. Pfam 3.1: 1313 multiple alignments and profile hmms match the majority of proteins. *Nucleic Acids Research*, 27(1):260–262, 1999.

[6] B. Behzadi and M. Vingron. Reconstructing domain compositions of ancestral multi-domain proteins. In G. Bourque and N. El-Mabrouk, editors, *Comparative Genomics*, volume 4205 of *Lecture Notes in Computer Science*, pages 1–10. Springer Berlin Heidelberg, 2006.

[7] A. K. Bjorklund, D. Ekman, S. Light, J. Frey-Skott, and A. Elofsson. Domain rearrangements in protein evolution. *Journal of Molecular Biology*, 353(4):911 – 923, 2005.

[8] P. Bonizzoni, G. D. Vedova, and R. Dondi. Reconciling a gene tree to a species tree under the duplication cost model. *Theor. Comput. Sci.*, 347(1-2):36–53, 2005.

[9] J. Cheng, M. Sweredoski, and P. Baldi. Dompro: Protein domain prediction using profiles, secondary structure, relative solvent accessibility, and recursive neural networks. *Data Mining and Knowledge Discovery*, 13(1):1–10, 2006.

[10] I. Cohen-Gihon, J. H. Fong, R. Sharan, R. Nussinov, T. M. Przytycka, and A. R. Panchenko. Evolution of domain promiscuity in eukaryotic genomes-a perspective from the inferred ancestral domain architectures. *Mol. BioSyst.*, 7:784–792, 2011.

[11] J.-P. Doyon, C. Scornavacca, K. Y. Gorbunov, G. J. Szöllosi, V. Ranwez, and V. Berry. An efficient algorithm for gene/species trees parsimonious reconciliation with losses, duplications and transfers. In E. Tannier, editor, *RECOMB-CG*, volume 6398 of *Lecture Notes in Computer Science*, pages 93–108. Springer, 2010.

[12] D. Durand, B. V. Halldórsson, and B. Vernot. A hybrid micro-macroevolutionary approach to gene tree reconstruction. *J. Comput. Biol.*, 13(2):320–335, 2006.

[13] R. D. Finn, A. Bateman, J. Clements, P. Coggill, R. Y. Eberhardt, S. R. Eddy, A. Heger, K. Hetherington, L. Holm, J. Mistry, E. L. L. Sonnhammer, J. Tate, and M. Punta. Pfam: the protein families database. *Nucleic Acids Research*, 42(D1):D222–D230, 2014.

[14] J. H. Fong, L. Y. Geer, A. R. Panchenko, and S. H. Bryant. Modeling the evolution of protein domain architectures using maximum parsimony. *Journal of Molecular Biology*, 366(1):307 – 315, 2007.

[15] K. Forslund, A. Henricson, V. Hollich, and E. L. L. Sonnhammer. Domain tree-based analysis of protein architecture evolution. *Molecular Biology and Evolution*, 25(2):254–264, 2008.

[16] M. Goodman, J. Czelusniak, G. W. Moore, A. E. Romero-Herrera, and G. Matsuda. Fitting the gene lineage into its species lineage. a parsimony strategy illustrated by cladograms constructed from globin sequences. *Systematic Zoology*, 28:132–163, 1979.

[17] K. Y. Gorbunov and V. A. Liubetskii. Reconstructing genes evolution along a species tree. *Molekuliarnaia Biologiia*, 43(5):946–958, Oct. 2009.

[18] P. Górecki and J. Tiuryn. Dls-trees: A model of evolutionary scenarios. *Theor. Comput. Sci.*, 359:378–399, 2006.

[19] J.-H. Han, S. Batey, A. A. Nickson, S. A. Teichmann, and J. Clarke. The folding and evolution of multidomain proteins. *Nature Reviews Molecular Cell Biology*, 8:319–330, 2007.

[20] J. M. Joseph and D. Durand. Family classification without domain chaining. *Bioinformatics*, 25(12):i45–i53, 2009.

[21] I. Letunic, T. Doerks, and P. Bork. Smart: recent updates, new developments and status in 2015. *Nucleic Acids Research*, 43(D1):D257–D260, 2015.

[22] B. Ma, M. Li, and L. Zhang. From gene trees to species trees. *SIAM J. Comput.*, 30(3):729–752, 2000.

[23] B. Mirkin, I. Muchnik, and T. F. Smith. A biologically consistent model for comparing molecular phylogenies. *J. Comput. Biol.*, 2(4):493–507, 1995.

[24] T. Miyata and H. Suga. Divergence pattern of animal gene families and relationship with the cambrian explosion. *BioEssays*, 23(11):1018–1027, 2001.

[25] A. D. Moore, A. K. Bjorklund, D. Ekman, E. Bornberg-Bauer, and A. Elofsson. Arrangements in the modular evolution of proteins. *Trends in Biochemical Sciences*, 33:444–451, 2008.

[26] R. D. M. Page. Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. *Syst. Biol.*, 43(1):58–77, 1994.

[27] T. Przytycka, G. Davis, N. Song, and D. Durand. Graph theoretical insights into evolution of multidomain proteins. *Journal of Computational Biology*, 13:351–363, 2006.

[28] B. Raphael, D. Zhi, H. Tang, and P. Pevzner. A novel method for multiple alignment of sequences with repeated and shuffled elements. *Genome Research*, 14(11):2336–2346, 2004.

[29] M. D. Rasmussen and M. Kellis. A bayesian approach for fast and accurate gene tree reconstruction. *Molecular Biology and Evolution*, 28(1):273–290, 2011.

[30] J. Schultz, R. R. Copley, T. Doerks, C. P. Ponting, and P. Bork. Smart: a web-based tool for the study of genetically mobile domains. *Nucleic Acids Research*, 28(1):231–234, 2000.

[31] J. Sjostrand, A. Tofigh, V. Daubin, L. Arvestad, B. Sennblad, and J. Lagergren. A bayesian method for analyzing lateral gene transfer. *Systematic Biology*, 63(3):409–420, 2014.

[32] A. Stamatakis. RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, 22(21):2688–2690, 2006.

[33] M. Stolzer. *Phylogenetic Inference for Multidomain Proteins*. PhD thesis, Carnegie Mellon University, 2011.

[34] M. Stolzer, H. Lai, M. Xu, D. Sathaye, B. Vernot, and D. Durand. Inferring duplications, losses, transfers and incomplete lineage sorting with nonbinary species trees. *Bioinformatics*, 28(18):409–415, 2012.

[35] M. Stolzer, K. Siewert, H. Lai, M. Xu, and D. Durand. Event inference in multidomain families with phylogenetic reconciliation. *BMC Bioinformatics*, 16(14):S8, 2015.

[36] M. B. Swindells. A procedure for detecting structural domains in proteins. *Protein Science*, 4(1):103–112, 1995.

[37] G. J. Szollosi, B. Boussau, S. S. Abby, E. Tannier, and V. Daubin. Phylogenetic modeling of lateral gene transfer reconstructs the pattern and relative timing of speciations. *Proceedings of the National Academy of Sciences*, 109(43):17513–17518, 2012.

[38] A. Tofigh, M. T. Hallett, and J. Lagergren. Simultaneous identification of duplications and lateral gene transfers. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 8(2):517–535, 2011.

[39] H. Tordai, A. Nagy, K. Farkas, L. Banyai, and L. Patthy. Modules, multidomain proteins and organismic complexity. *FEBS Journal*, 272(19):5064–5078, 2005.

[40] C. Vogel, M. Bashton, N. D. Kerrison, C. Chothia, and S. A. Teichmann. Structure, function and evolution of multidomain proteins. *Current Opinion in Structural Biology*, 14(2):208 – 216, 2004.

[41] C. Vogel, S. A. Teichmann, and J. Pereira-Leal. The relationship between domain duplication and recombination. *Journal of Molecular Biology*, 346(1):355 – 365, 2005.

[42] J. Wiedenhoeft, R. Krause, and O. Eulenstein. The plexus model for the inference of ancestral multidomain proteins. *IEEE/ACM Trans. Comp. Biol. Bioinf.*, 8(4):890–901, 2011.

[43] Y.-C. Wu, M. S. Bansal, M. D. Rasmussen, J. Herrero, and M. Kellis. Phylogenetic identification and functional characterization of orthologs and paralogs across human, mouse, fly, and worm. *bioRxiv*, 2014.

[44] Y.-C. Wu, M. D. Rasmussen, M. S. Bansal, and M. Kellis. Treefix: statistically informed gene tree error correction using species trees. *Systematic Biology*, 62(1):110–120, 2013.

[45] Y.-C. Wu, M. D. Rasmussen, and M. Kellis. Evolution at the subgene level: Domain rearrangements in the drosophila phylogeny. *Molecular Biology and Evolution*, 29(2):689–705, 2012.

[46] L. Zhang. On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *J. Comput. Biol.*, 4(2):177–187, 1997.