Bioinformatics, 34, 2018, i366–i375
doi: 10.1093/bioinformatics/bty242
ISMB 2018

Accurate prediction of orthologs in the presence of divergence after duplication

Manuel Lafond[1,2,*], Mona Meghdari Miardan[1] and David Sankoff[1]

[1]Department of Mathematics and Statistics, University of Ottawa, Ottawa, ON K1N 6N5, Canada and [2]Department of Computer Science, Université de Sherbrooke, Sherbrooke, QC J1K 2R1, Canada

*To whom correspondence should be addressed

Abstract

Motivation: When gene duplication occurs, one of the copies may become free of selective pressure and evolve at an accelerated pace. This has important consequences on the prediction of orthology relationships, since two orthologous genes separated by divergence after duplication may differ in both sequence and function. In this work, we make the distinction between the *primary* orthologs, which have not been affected by accelerated mutation rates on their evolutionary path, and the *secondary* orthologs, which have. Similarity-based prediction methods will tend to miss secondary orthologs, whereas phylogeny-based methods cannot separate primary and secondary orthologs. However, both types of orthology have applications in important areas such as gene function prediction and phylogenetic reconstruction, motivating the need for methods that can distinguish the two types.

Results: We formalize the notion of divergence after duplication and provide a theoretical basis for the inference of primary and secondary orthologs. We then put these ideas to practice with the Hybrid Prediction of Paralogs and Orthologs (HyPPO) framework, which combines ideas from both similarity and phylogeny approaches. We apply our method to simulated and empirical datasets and show that we achieve superior accuracy in predicting primary orthologs, secondary orthologs and paralogs.

Availability and implementation: HyPPO is a modular framework with a core developed in Python and is provided with a variety of C++ modules. The source code is available at https://github.com/manuellafond/HyPPO.

Contact: mlafond2@uOttawa.ca

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

During the course of evolution, speciation and duplication create pairs of homologous genes, which can be classified into two categories. Two genes are orthologs if they descend from an ancestral gene that has undergone speciation and paralogs if they result from duplication. Distinguishing orthologs from paralogs is of considerable importance in biology, owing to their functional and evolutionary implications (Gabaldón and Koonin, 2013; Koonin, 2005). Notably, the well-known 'orthologs conjecture' states that orthologs tend to perform similar functions, in contrast with paralogs, which tend to diverge in their functional role. The reasoning underlying this conjecture is that when a gene creates a copy in the same genome, then this genome has two genes that are able to perform the same function. Hence, one of these copies may become free of selective pressure, resulting in a higher rate of mutations and facilitating a possible change in functionality. We refer the reader to (Altenhoff *et al.*, 2012; Chen and Zhang, 2012; Nehrt *et al.*, 2011; Studer and Robinson-Rechavi, 2009; Thomas *et al.*, 2012) for some recent developments on this conjecture.

The two traditional methods to predict orthology relations are *similarity-* and *phylogeny-based* (see Altenhoff and Dessimoz, 2012; Kristensen *et al.*, 2011 for a survey). Similarity-based methods aim to partition the set of genes into 'groups' of orthologs under the assumption that genes that share similar sequences are orthologous. The precise notion of a group depends on the particular method used, since in some cases, these groups include multiple genes from the same species (e.g. recent paralogs). For instance in OrthoMCL (Li *et al.*, 2003), Markov clustering is used to identify groups of orthologs and close paralogs. A similar idea is implemented in Proteinortho using spectral clustering techniques (Lechner *et al.*, 2011). The OMA software is somewhat more stringent and aims to find cliques of pairwise orthologous genes (Roth *et al.*, 2008). This was extended in later versions with the addition of the GETHOGS algorithm (Altenhoff *et al.*, 2013; Train *et al.*, 2017). The program is able to potentially find all

orthologous gene pairs by grouping genes into *hierarchical orthologous groups* (HOGs), which are the orthologs that belong to a set of species within a given taxonomic range. Some other similarity-based grouping methods and databases include COG (Tatusov *et al.*, 2003), EggNOG (Powell *et al.*, 2012), InParanoid (O'Brien *et al.*, 2004) and OrthoFinder (Emms and Kelly, 2015). The OMG approach (Zheng *et al.*, 2011) identifies sets of pairwise *orthogonal* genes from a homology graph inferred from synteny.

On the other hand, phylogeny-based methods do not attempt in the first instance to group genes together but rather aim to identify orthologs by inferring the lowest common ancestor (lca) relationships between genes directly. This can be done by inferring a gene tree and identifying its speciation and duplication events using *reconciliation* with a species tree (Stolzer *et al.*, 2012; Ullah *et al.*, 2015). This approach is however sensitive to errors in the gene tree and the species tree. Methods of this type include notably LOFT (Van der Heijden *et al.*, 2007) and COCO-CL (Jothi *et al.*, 2006). Recently, a number of methods exploit a fundamental property of orthology graphs, in which vertices are genes and edges depict orthology: a graph G is a valid orthology graph if and only if it is P_4-free (Dondi *et al.*, 2017a; Dondi *et al.*, 2017b; Hernandez-Rosales *et al.*, 2012; Hellmuth *et al.*, 2013; Hellmuth *et al.*, 2015; Jones *et al.*, 2016; Lafond and El-Mabrouk, 2014; Lafond *et al.*, 2016), i.e. G has no path on four vertices without a shortcut. The essential reason underlying this characterization is that P_4-free graphs have a special tree representation called a *co-tree*, which turns out to be interpretable as a gene tree that depicts all the lca relationships prescribed by G. These methods usually construct an approximate putative orthology graph and perform a minimum number of modifications on the graph so that it becomes P_4-free.

The output from similarity- and phylogeny-based methods can be different, as some orthologies may be irrelevant from the perspective of grouping genes with similar functions, but will be inferred by phylogenetic methods. Consider the example of Figure 1. The node x represents an ancestral duplication event in which *neofunctionalization* hypothetically occurred, where one copy (here the left child of x) retains its original functionality, whereas the other copy (the right child of x) diverges and starts accumulating mutations at a higher rate. As a result of this event, the a and b genes remain similar, as no major divergence event separates the two. However, a and c will have differentiated significantly, even though the two genes are orthologous (since their lca is a speciation). Consequently, similarity-based methods are unlikely to mark a and c as orthologs, as the two genes do not share sequence similarity. From the point of view of functional annotation, this is not problematic, since the a, c orthology pair essentially behave as paralogs and may differ in function.

This raises the question as to which type of inference is the most relevant: should we focus on the groups of similar orthologs, or do we need *all* the orthologs? Ideally, both types of orthologies should be predicted, since similarity groups are useful for function-related applications (Doyle *et al.*, 2010), while the complete set of pairwise relations is useful for phylogenetic reconstructions (Hellmuth *et al.*, 2015).

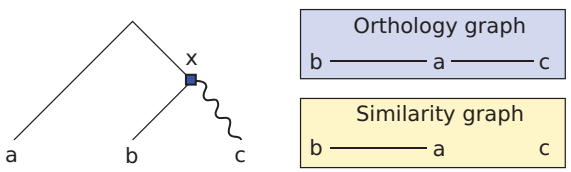

Fig. 1. An example of gene tree for the gene family {*a*, *b*, *c*}, along with the underlying orthology and similarity graphs. The root of the tree is a speciation, the square is a duplication, and the wiggly edge represents an event of divergence after duplication. The gene pairs *ab* and *ac* are orthologs. However, *a* and *c* will not appear as 'similar', as there was a significant divergence event on their evolutionary path

Of course, post-duplication phenomena other than neofunctionalization as described above can arise. For instance, *bifunctionalization* occurs when both copies preserve the parental function without diverging, and *subfunctionalization* occurs when the functions of the common ancestor are partitioned among the descending paralogs (Zhang, 2003), in which case it is possible for both copies to undergo a new rate of mutations. Nevertheless, the phenomenon of a single copy undergoing functional change has been reported to occur frequently (Cardoso-Moreira *et al.*, 2016; Jordan *et al.*, 2004; Lynch and Conery, 2000; Woods *et al.*, 2013; Soria *et al.*, 2014), as duplication models other than neofunctionalization, for instance, *adaptive radiation* or *modified duplication*, also predict this type of divergence after duplication. We redirect the reader to (Innan and Kondrashov, 2010; Table 1) for more details.

In this work, we are interested in how the task of predicting orthology is affected when divergence after duplication does occur. We propose an orthology prediction framework that makes the distinction between the *primary orthologs*, the orthologous gene pairs that have not been separated by an event of duplication followed by an increased rate of mutation, and the *secondary orthologs*, which consist of the pairs of orthologs that have had at least one such event along their evolutionary path. For example in Figure 1, {*a*, *b*} are primary orthologs, whereas {*a*, *c*} are secondary orthologs. Note that the relevance of making this distinction was also discussed by Swenson and El-Mabrouk (2012), where the primary orthologs are called *isoorthologs*. Also observe that this categorization is different from the co-orthology relationship, which is usually defined with respect to a given node of a gene tree. We formalize this notion of primary orthology and show that they must form cliques in the orthology graph, thereby providing a formal justification of the clustering steps often performed by orthology prediction methods.

Our algorithmic framework is a hybrid approach that makes use of both similarity clustering and phylogenetic reconstruction. More specifically, we use exact graph partitioning techniques to find the primary orthologs, construct a species tree from the orthology clusters found and use this species tree to obtain the secondary orthologs. The method also puts to practical use the theory of P_4-free orthology graphs developed recently. The framework, which we call Hybrid Prediction of Paralogs and Orthologs (HyPPO), is implemented as a fully modular pipeline in which each task can have its own independent implementation.

We evaluate HyPPO on both simulated and real empirical datasets. We compare our method with OrthoMCL and OMA-GETHOGS—to the best of our knowledge, the latter is the only other program that can distinguish primary and secondary orthologs. We show that HyPPO achieves superior accuracy in the set of predicted pairwise relations and is better at finding the primary orthology clusters. Table 1 presents a summary of our results.

Table 1. Average accuracy, precision, recall and cluster scores, taken over the set of all simulated gene trees (see the Results section for the definitions)

	Accuracy	Precision	Recall	Cluster score
HyPPO	0.940	0.911	0.875	0.915
HyPPO + Species tree	0.949	0.924	0.905	0.915
OMA-GETHOGS	0.877	0.940	0.699	0.831
OrthoMCL	0.812	0.845	0.496	0.690

2 Materials and methods

In this section, we first introduce the preliminary notions necessary for the understanding of this paper. We then formalize the notion of divergence after duplication in gene trees and its consequences on distinguishing primary and secondary orthologs. We then present the main algorithmic components of the HyPPO framework. Note that due to space constraints, most of the proofs are relegated to the Supplementary Material.

2.1 Preliminary notions

In this paper, every tree T is assumed to be rooted and binary, i.e. each internal node has two children, and each node v has a unique parent $p(v)$, with the exception of the root, denoted $r(T)$, which has no parent. The set of vertices of T is denoted by $V(T)$ and its set of leaves by $L(T)$. A node u of T is an *ancestor* of a node v if u is on the path from v to $r(T)$. Then v is a *descendant* of u, in which case we write $v \leq u$. If $v \neq u$, we write $v < u$. If none of $v \leq u$ nor $u \leq v$ holds, then u and v are *incomparable*, and we write $u \oplus v$. The *least common ancestor* of a set of leaves $X \subseteq L(T)$, denoted $lca_T(X)$, is the node u of T that is the ancestor of every node in X, and that is the most distant from the root. For convenience, we may write $lca_T(a, b)$ instead of $lca_T(\{a, b\})$. For a graph G, a *clique* is a set of vertices of G in which each pair shares an edge. A P_k is a path on k vertices with no chord.

A *gene family* Γ is a set of genes related by homology. A *gene tree* for Γ is a tree T in which $L(T) = \Gamma$. Likewise, if Σ is a set of species, a *species tree* for Σ is a tree S satisfying $L(S) = \Sigma$. Each gene $g \in \Gamma$ belongs to a species denoted $\sigma(g)$. If $X \subseteq \Gamma$ is a set of genes, then we write $\sigma(X) = \{\sigma(g) : g \in X\}$.

A *DS*-tree (T, ℓ) is a pair in which T is a gene tree, and $\ell : V(T) \backslash L(T) \rightarrow \{\mathbb{D}, \mathbb{S}\}$ is a function labeling internal nodes by either $\mathbb{D}$ or $\mathbb{S}$, respectively standing for duplication and speciation. We will often omit mentioning ℓ explicitly and say that T is a *DS*-tree with the understanding that its internal nodes are labeled by ℓ. Given a species tree S, we map each node of T to a node of S with the *lca-mapping* $s : V(T) \rightarrow V(S)$ defined as follows: if $g \in L(T)$, then $s(g) = \sigma(g)$, and otherwise $s(g) = lca_S(\{\sigma(l) : l \in L(T) \text{ and } l \leq g\})$. By a slight abuse of notation, if $C \subseteq \Gamma$, we denote $s(C) = lca_S(\sigma(C))$. Note that s depends on the species tree S. In case of ambiguity, we may write $s(g, S)$ or $s(C, S)$ to denote the lca-mapping with respect to S. An example of lca-mapping is presented in Figure 3.

We use the definition of Fitch (2000) for orthology and paralogy.

Definition 1. With respect to a DS-tree T, two genes a and b are orthologs if $\ell(lca_T(a, b)) = \mathbb{S}$, and paralogs if $\ell(lca_T(a, b)) = \mathbb{D}$.

An *orthology graph* $G = (V, E)$ is a graph in which V is a set of genes and $uv \in E$ if and only if u and v are orthologs. An *orthology cluster* C is a set of genes that are pairwise orthologous. In particular, C may contain at most one gene from any given species.

2.2 Orthology and divergence after duplication

As mentioned in the Section 1, duplication may introduce divergence when one of the child copies is redundant. In this section, our goal is to investigate the implications of this phenomenon on the structure of orthology relations. That is, if divergence after duplication occurs, what are the orthologs that are expected to be recovered using similarity-based methods? Or using phylogeny-based methods? We present the tools that we will use to devise methods tailored for this type of evolution. As we will demonstrate in the Section 3, the methodology developed here can also be used in case that divergence after duplication does not always occur, or even when if it does not happen at all.

We formalize divergence after duplication as follows. Let T be a *DS*-tree. Suppose that for each $\mathbb{D}$ node v of T, v has one child v_1 that evolved at a normal rate and another child v_2 that underwent an accelerated pace of evolutionary changes. We will say that the vv_2 edge is a *divergent duplication edge*. Now, we will assume that if, for two leaves g_1 and g_2 of $L(T)$, there is a divergent duplication edge on the path between g_1 and g_2, then g_1 and g_2 will be considered 'not similar'. Conversely, we will assume that if there is no such edge on the path between g_1 and g_2, then no unexpected differentiation between g_1 and g_2 will be observed, and thus g_1 and g_2 will be considered 'similar'.

Note that here, we have not defined 'similarity' in a quantitative manner. For our purposes, we prefer to leave this notion abstract, and we will assume that divergence after duplication leaves behind some traces that can be used to detect similarity versus non-similarity (for instance, one could compare the sequence of a gene to all its homologs in a given species). This is because our goal is to evaluate the consequences of divergence in the orthology inference process, and not necessarily to detect such divergence. We thus prefer to keep the notion of similarity as general as possible. We do note however that in the future, further investigation will be needed to establish concrete methods to recognize similarity.

We now introduce divergence after duplication (DAD)-trees.

Definition 2. A DAD-tree T for Γ is a DS-tree for Γ in which each $\mathbb{D}$ node v has exactly one child v' such that the edge vv' is marked as divergent, and the other edges of T are marked as non-divergent.

Two genes $g_1, g_2 \in L(T)$ are called *similar* if there is no divergent edge on the path between g_1 and g_2 in T. Otherwise, g_1 and g_2 are called *non-similar*.

It then becomes quite easy to characterize similarity graphs, using the above notion of 'similarity'.

Proposition 3. Let G be the graph in which $V(G) = \Gamma$, and two genes share an edge if and only if they are similar with respect to a DAD-tree T. Then each connected component of G is a clique.

Therefore, unlike orthology, this notion of similarity is a transitive relation. Proposition 3 then tells us precisely what we should expect from similarity graphs when divergence after duplication always occurs: we should obtain cliques of genes. Moreover, the genes in these cliques are pairwise orthologous, as two paralogous genes cannot appear as 'similar' under the above definition. Hence, the primary orthologs resulting from a DAD-tree form the cliques in the similarity graph. For this reason, we may sometimes call *orthology clusters* the cliques of primary orthologs. This motivates the first essential step of our orthology prediction framework, which is to transform the similarity graph into a cluster graph. However, this will only identify a subset of all orthologies, as there may be inter-cluster orthologies.

One idea might be to infer orthology between the 'closest' inter-cluster genes. However, this may introduce invalid relations or inconsistencies. Here, as by Hellmuth *et al.*, (2015) and Lafond *et al.* (2016), we take the viewpoint that for a set of relations to be correct, there must be an evolutionary scenario that could have given rise to the observed relations. In other words, there should exist a *DS*-tree that displays the inferred orthologies, and the speciation nodes on this *DS*-tree should agree with the species history.

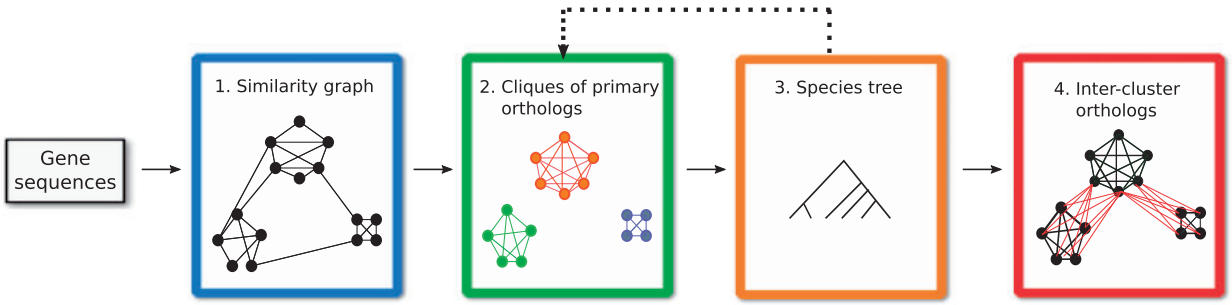

Fig. 2. The four essential steps of the HyPPO pipeline. First, a similarity graph is constructed from the gene sequences. This graph is refined into orthology clusters, which are cliques of primary orthologs. These clusters are used to infer a species tree, which is then used to find the secondary orthologs (which we also call inter-cluster orthologs)

This notion of agreement with a species tree is defined as in (Jones *et al.*, 2016; Lafond and El-Mabrouk, 2014).

Definition 4. Let T be a DS-tree and S be a species tree. Moreover, let $v \in V(T) \backslash L(T)$ be an $\mathbb{S}$ node of T, and let v_1 and v_2 be its children. Then we say that v is S-consistent if $s(v) \neq s(v_1)$ and $s(v) \neq s(v_2)$.

Furthermore, we say that T is S-consistent if each of its $\mathbb{S}$ nodes is S-consistent.

In the context of our work, the tree T is usually unknown, and only the orthology graph G is known. We use the existence/non-existence of a tree T corresponding to the relations in G to validate the orthology graph.

Definition 5. An orthology graph G is S-consistent if there exists a DS-tree T such that the two following conditions hold:

1. $ab \in E(G)$ if and only if a and b are orthologs in T;
2. T is S-consistent.

The property of S-consistency has been investigated in depth in the recent years and admits a convenient graph-theoretical characterization. First, we need to define a *triplet*. Let T be a tree, and let $x, y, z \in L(T)$ be three distinct leaves. We say that T contains the triplet $xy|z$ if $lca_T(x, y) < lca_T(x, z) = lca_T(y, z)$.

Theorem 6. An orthology graph G is S-consistent if and only if the two following conditions hold (Lafond and El-Mabrouk (2014):

1. G is P_4-free, i.e. has no path on four vertices without a chord;
2. for every $x, y, z \in V(G)$ such that $\sigma(x), \sigma(y)$ and $\sigma(z)$ are distinct and such that $xy, yz \in E(G)$ but $xz \notin E(G)$, we have that $\sigma(x)\sigma(z)|\sigma(y)$ is a triplet of S.

Therefore, S-consistent graphs can be characterized by the absence of P_4's and a set of forbidden P_3's given by the structure of the species tree. Given the set of primary orthology clusters, our goal is to infer the inter-cluster orthologs in a way that the resulting relations could be generated from a DAD-tree in a S-consistent manner.

2.3 The HyPPO orthology framework

We may now describe the main ingredients of our orthology inference framework. The flow of the pipeline is illustrated in Figure 2. Starting only from the DNA or amino acid sequences of a given set of genes (or proteins), our ultimate goal is to identify the cliques of primary orthologs, and the orthology relations between these cliques. Recall that this framework is modular, and that an implementation can be provided for each step individually. We first provide an outline of the tasks that each step must perform and then proceed with the algorithmic details of how steps 2–4 were realized in the default implementation of our framework.

Step 1. We start by building a tentative *similarity graph*, in which two genes share an edge if they are 'similar'. The pairwise similarity scores between the genes are typically used for this step. For instance, the graph may contain an edge xy iff the BLAST score between x and y is above a certain threshold. The edges can also be weighted by this score.

Step 2. As stated in Proposition 3, we expect the above similarity graph to contain only cliques. In practice, this might of course not always be the case and so in this step, we partition the graph into cliques of pairwise orthologous genes, which are assumed to be primary orthologs. Note that virtually every similarity-based inference method performs this task, although it should be observed that our requirement of pairwise orthology is strict, and no in-paralogs are allowed in the inferred groups.

Step 3. In order to obtain the inter-cluster relations in a S-consistent manner, a species tree S is needed. If known, this step can be omitted, but otherwise, a species tree can be inferred from the orthology clusters. Indeed, as we will show, some phylogenetic signal can be extracted from these clusters.

Note that in Figure 2, step 3 points back to step 2. This is because the framework allows reinferring the orthology clusters, but this time with the additional information of the species tree. The new clusters can then be used to reinfer the species tree, and this process can be looped until convergence. Although our framework supports such a loop, we did not use it in our experimentations and leave this exploration for future work.

Step 4. In this final step, we assume that the orthology clusters and the species tree are known, but the orthologous gene pairs containing genes from distinct clusters are missing. Our goal is to recover these inter-cluster relations in a way to ensure S-consistency while preserving the properties of a DAD-tree. That is, we assume that in their history, two genes from two distinct clusters are separated by a divergent duplication edge, and that the predicted orthology relations can be explained by this type of history.

In the rest of this section, we describe how steps 2, 3 and 4 were implemented in HyPPO.

2.3.1 Prediction of orthology clusters

As partitioning a graph into cliques is a difficult algorithmic problem, most of the orthology prediction methods that perform clustering use a heuristic in order to deal with thousands, or even millions, of genes. In our case, we assume that genes are partitioned into families, which do not usually exceed a few hundred genes. In some

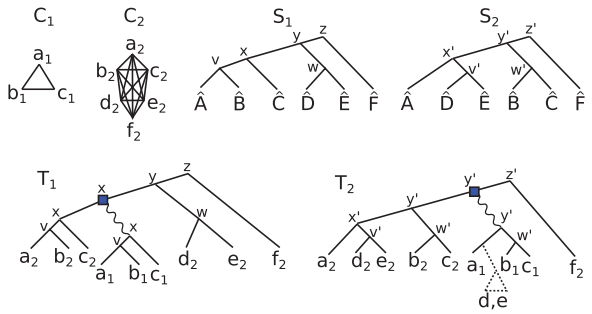

Fig. 3. Two orthology clusters C_1 and C_2, and two putative species trees S_1 and S_2. The genes are named according to their species, e.g. $\sigma(a_1) = \widehat{A}$. Each internal node u of T_1 (respectively, T_2) is labeled by its lca-mapping $s(u, S_1)$ (resp. $s(u, S_2)$). The tree T_1 is the DAD-tree that could have given rise to C_1 and C_2 if S_1 was the true species tree, assuming that C_1 was born during the history of the C_2 genes. T_2 is the DAD-tree for C_1 and C_2 under the assumption that S_2 is the true species tree. In the case of T_2, a loss in the clade $\{\widehat{D}, \widehat{E}\}$ is required to explain the birth of the C_1 cluster at a time prior to $lca_{S_2}(\sigma(C_1)) = y'$

cases, this makes it possible to use exact algorithms instead of heuristics. In fact, we provide two implementations for gene clustering: an exact method and a greedy heuristic. The exact method can be used on gene families with a reasonable number of genes (up to 200) and the heuristic for the other families.

The exact algorithm solves the cluster editing problem, which asks for the minimum number of edges to add or delete in a given graph so that every connected component of the resulting graph is a clique. We used the BBH graph, in which vertices are genes and edges are bidirectional best hits (BBH), where two genes x, y form a BBH if y is the gene of $\sigma(y)$ of maximum score with x, and vice-versa. The cluster editing problem is well-studied, and there are multiple methods that can find an optimal solution in an acceptable amount of time, including integer linear programming (ILP) formulations (Böcker *et al.*, 2011; Hartung and Hoos, 2015) and fixed-parameter algorithms (Böcker *et al.*, 2009). We used the ILP method (see Böcker *et al.* 2011 for details), ensuring that genes a and b from the same species would never belong to the same cluster by making the cost of adding the ab edge infinite.

The greedy heuristic proceeds in an agglomerative fashion using the pairwise scores. This can be seen as an adaptation of UPGMA with the restriction that clusters with a common species can never be joined. More precisely, at the start, we treat each individual gene as a cluster. Then, we merge the two clusters C_1 and C_2 if $\sigma(C_1) \cap \sigma(C_2) = \varnothing$ and if they maximize $\max_{a \in C_1, b \in C_2} \text{score}(a, b)$ (i.e. C_1 and C_2 have the two closest genes among all possible pairs of clusters). The algorithm stops when no more clusters can be merged. Note that we also tested other join criteria, for instance, joining the two clusters that minimize the average distance as in UPGMA, but the above algorithm yielded the best results.

2.3.2 Inference of a species tree from orthology clusters

We now show how orthology clusters may guide the construction of a species tree. In order to avoid confusion between species and clusters, we shall denote species with a *hat* symbol, e.g. $\widehat{C}$ is a species. Consider the example provided in Figure 3. Here, the set of species $\sigma(C_1)$ appearing in cluster C_1 forms a subset of the species $\sigma(C_2)$ from cluster C_2. This suggests that C_1 was given birth somewhere during the evolution of the C_2 cluster. Moreover, the duplication that gave rise to C_1 should have occurred in a species that existed at least as far back in time as the lca of $\sigma(C_1)$ in the true species tree S. This means that in the true gene tree for $C_1 \cup C_2$, the duplication node d must satisfy $s(d) \geq s(C_1)$ (where here s is with respect to the true species tree S). We argue that this suggests that the true species tree S contains the clade $\{\widehat{A}, \widehat{B}, \widehat{C}\}$, meaning that some node u of S has exactly $\widehat{A}, \widehat{B}$ and $\widehat{C}$ as its descendant leaves.

To justify this claim, observe that some species trees may require *losses* to explain the clusters, in a way that is analogous to gene tree and species tree *reconciliation* (see Doyon *et al.* 2011 for more details on this topic). Consider the two species tree S_1 and S_2 from Figure 3. If S_1 is the true species tree and d is the duplication node that gave birth to the C_1 cluster, then $s(d, S_1) \geq s(C_1, S_1) = x$, as argued in the previous paragraph. T_1 is an example of an S_1-consistent DAD-tree that satisfies this condition. If S_2 is the true species tree, then T_1 does not meet this requirement anymore. The tree T_2 does satisfy $s(d, S_2) \geq s(C_1, S_2) = y'$. However, $s(C_1, S_2)$ has $\widehat{A}, \widehat{B}, \widehat{C}, \widehat{D}$ and $\widehat{E}$ as its descendants, but $\widehat{D}$ and $\widehat{E}$ do not appear in C_1. This can be explained by a gene loss prior to the ancestor of $\widehat{D}$ and $\widehat{E}$, resulting in a loss in the $\{\widehat{D}, \widehat{E}\}$ clade. The tree T_1 does not require a loss to be explained, and therefore its corresponding species tree is preferred. Observe that any species tree that contains the $\{\widehat{A}, \widehat{B}, \widehat{C}\}$ clade leads to a loss-less history, suggesting that this clade should be in the species tree.

Of course, the situation gets more difficult when $\sigma(C_1)$ is not a subset of $\sigma(C_2)$, or when there are more than two clusters. Below, we formalize the problem of reconstructing a species tree from for an arbitrary set of clusters while minimizing losses.

Let C be an orthology cluster. For convenience, we will make no distinction between C and its set of species $\sigma(C)$, as this has no bearing on the inference procedure. For a species tree S and $x \in V(S)$, the *clade* of x is defined as $\text{clade}(x) = \{l \in L(S) : l \text{ is a descendant of } x\}$. The set of clades of S is $\text{clades}(S) = \{\text{clade}(x) : x \in V(S)\}$. For a given cluster C, let S_C be the subtree of S rooted at $lca_S(C)$. The number of losses of C with respect to S, denoted $l_S(C)$, is defined as the minimum number of clades to remove from $\text{clades}(S_C)$ so that the union of the remaining clades is exactly C. Another way to view $l_S(C)$ is as follows. Call a non-root node v of S_C *maximal C-free* if $\text{clade}(v) \cap C = \varnothing$ but $\text{clade}(p(v)) \cap C \neq \varnothing$. Then $l_S(C)$ is the number of maximal C-free nodes in S_C. As an example, the reader can verify in Figure 3 that $l_{S_2}(C_1) = 1$ and $l_{S_2}(C_2) = 0$. We may now formally define our species tree reconstruction problem.

The species tree using clusters (STUC) problem:
Given: a set of orthology clusters $\mathscr{C} = \{C_1, \ldots, C_k\}$.
Find: a species tree S that minimizes $\sum_{i=1}^{k} l_S(C_i)$.

Denote by $OPT(\mathscr{C})$ the minimum number of losses of a species tree for the STUC instance $\mathscr{C}$. Note that $\mathscr{C}$ might not be sufficient to determine the complete species tree exactly, as in Figure 3 where any species tree containing the $\{\widehat{A}, \widehat{B}, \widehat{C}\}$ clade is optimal. However, the more clusters there are, the more precisely S can be inferred.

We do not know whether the STUC problem is NP-hard, although we suspect that is it. Nevertheless, we borrow ideas from (Mirarab *et al.*, 2014) in order to limit the space of possible species trees to those containing a predefined set of splits. For $X \subset \Sigma$, a *split* $\{A, B\}$ for X is a partition of X into two non-empty subsets. We say that S *contains* the split $\{A, B\}$ if there is a node v with children v_1, v_2 such that $A = \text{clade}(v_1)$ and $B = \text{clade}(v_2)$. Suppose that we are given a set of *allowable splits* $\mathbb{A} = \{A_i, B_i\}_{i=1}^{r}$, where for each i, A_i and B_i are disjoint subsets of Σ. Then we may ask for a species tree S that solves the STUC problem under the condition that every split

contained in S appears in $\mathbb{A}$. In the Supplementary Material, we show that this restricted version can be solved in polynomial time by dynamic programming, and then we discuss how a reasonable set $\mathbb{A}$ can be constructed.

THEOREM 7. The STUC problem with species trees restricted to the allowable splits $\mathbb{A}$ can be solved in time $O(|\mathbb{A}|^2|\mathscr{C}||\Sigma|)$.

2.3.3 Inference of inter-cluster orthology relations

In this final step, we assume that the orthology clusters $\mathscr{C}$ and the species tree S are known, but the orthologous gene pairs containing genes from distinct clusters are missing. Our goal is to recover these inter-cluster relations in a way to ensure S-consistency while preserving the properties of evolution under divergence after duplication. Note that unlike the last section, here we do not confound a cluster C and its set of species $\sigma(C)$.

Consider the orthology clusters C_1 and C_2 in Figure 3. As we have argued in the previous section, the ancestral gene of C_1 was born off a duplication node d satisfying $s(d) \geq s(C_1)$. One implication of this is that if $\sigma(g_2) < s(C_1)$, where $g_2 \in C_2$, then it is not be possible for a gene of C_1 to be orthologous to g_2. As an example, take the gene $a_2 \in C_2$ in Figure 3 and assume that S_1 and T_1 are the true species tree and gene tree, respectively. We have $\sigma(a_2) < s(C_1)$, and so a_2 is orthologous with no gene of C_1. However, this is not true for $d_2 \in C_2$, for example, because d_2 is a secondary ortholog with all of C_1. This is because $\sigma(d_2) \oplus s(C_1)$.

In our implementation, when inferring inter-cluster relations between C_1 and C_2, we simply make as many pairs of genes orthologous whilst preserving S-consistency. As earlier, we also require that if $\sigma(g_1) < s(C_2)$, then g_1 is paralogous to all the genes in C_2 (and vice-versa). This results in Algorithm 1. This works as described above in the case of $k = 2$ clusters. For larger k, we merge the clusters iteratively in a greedy manner.

More specifically, we first pick the cluster C for which $s(C)$ is the the lowest in the species tree, breaking ties arbitrarily. We then ask: what other cluster C_j could have spawned the C cluster? We call C_j the *favorite cluster* of C and assume that we are given a function *fav* $(C, \mathscr{C})$ that finds the favorite cluster of C among a set of clusters $\mathscr{C}$. We then merge C with C_j by making all possible orthologies - namely, we make $c_j \in C_j$ orthologous to every gene of C iff $\sigma(c_j) \oplus s(C)$. Afterwards, C_j is redefined as $C_j \cup C$, C is emptied and we proceed with the next cluster. We show that Algorithm 1 preserves S-consistency as desired.

Algorithm 1 Algorithm to infer inter-cluster orthologs.

```
1: procedure inferInterClusters(G, S, 𝒞 = {C_1, ..., C_k}, fav)  G is the initial orthology graph, S is the species tree, 𝒞 is the set of clusters, and fav is a function that finds favorites. We assume that in G, two vertices share an edge iff they belong to the same cluster. The output is an orthology graph obtained from G.
2:     while 𝒞 has at least two non-empty clusters do
3:         Let C ∈ 𝒞 such that s(C) ≤ s(C_i) or s(C)⊕s(C_i) holds for every C_i ∈ 𝒞
4:         Let C_j = fav(C, 𝒞).
5:         for g_j ∈ C_j such that σ(g_j)⊕s(C) do
6:             Add an edge in G between g_j and every vertex in C.
7:         C_j ← C_j ∪ C and C ← ∅
```

THEOREM 8. The orthology graph obtained from Algorithm 1 is S-consistent.

In our implementation, we used the following simple *fav* function: for each $x \in C$, find the gene $y \in \Gamma \backslash C$ of maximum score. Then x votes for the cluster that contains y. The favorite cluster of C is then the one with the highest number of votes. Only the members of the genes initially in C, as given in the input, are allowed to vote.

3 Results

We compare the default implementation of the HyPPO pipeline with two other orthology prediction methods: OMA-GETHOGS and OrthoMCL. To our knowledge, OMA-GETHOGS is the method that is the most similar to ours, as it outputs both the 1-to-1 orthology groups, and the pairwise orthology relations. The groups can be interpreted as the primary orthologs. As for OrthoMCL, it is a similarity-based clustering method that focuses on grouping orthologs, and so it is not expected to find all the secondary orthologs. Nevertheless, we include it for the sake of comparison, as OrthoMCL is one of the most popular orthology prediction tools. HyPPO was tested under two settings: one in which the species tree is unknown, and the other in which the true simulated species tree is provided. OMA-GETHOGS was provided the true species tree in all experiments. (The program does not require a species tree as input, but OMA-GETHOGS produced an error message when not given one—for reasons that are yet to be determined. Nevertheless, better results are expected when the true species tree is known.)

We performed our experiments on both simulated and real empirical datasets. In the simulated datasets, the true orthologs and the primary orthologs are known. This is not true for the real datasets. As in the study by Altenhoff *et al.* (2013), we used a set of gold standard, manually curated, gene trees from the SwissTree database (Wapinski *et al.*, 2007). These trees are annotated with duplication and speciation events, and so the underlying orthology/paralogy relations can be considered as 'gold standard' relations. We did not, however, include the analysis of the primary vs secondary orthologs, as those are not determined in the real datasets. Note that in both simulated and real datasets, the genes were already separated into homologous families. All the programs were executed on the separated gene families, thereby avoiding predictions between non-homologous genes. For HyPPO, the sequences were aligned with MAFFT (Katoh *et al.*, 2002), and the similarity values for the first step were obtained from the pairwise identity percentages computed from these alignments (number of identical base pairs divided by the length of the longest sequence, not counting gaps).

3.1 Performance metrics

In our experiments, we address two questions: is the set of all pairwise relations predicted correct? And, can the method distinguish the primary orthologs from the secondary orthologs? For the pairwise relations, we used precision, recall and accuracy as our performance metrics. The precision *Prec* and recall *Rec* values are calculated from the orthology relations as follows: $\text{Prec} = \frac{\text{TP}}{\text{TP+FP}}$ and $\text{Rec} = \frac{\text{TP}}{\text{TP+FN}}$. Here, *TP* means True Positive (the number of relations correctly predicted as orthology), *FP* means False Positive (a paralogy relation incorrectly predicted as orthology) and *FN* means False Negative (an orthology relation left undetermined or predicted as paralogy). As for accuracy, it consists in the total number of relations predicted correctly (including both orthologs and paralogs) divided by $\binom{n}{2}$, the total number of relations. (Note that

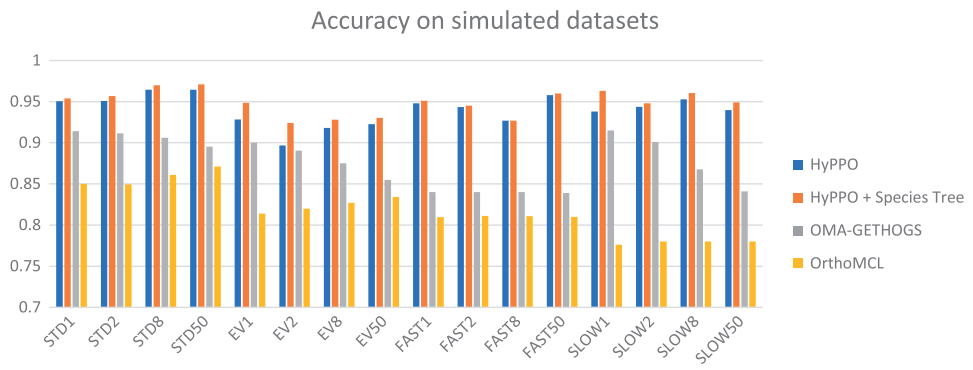

Fig. 4. Accuracy of the methods for each dataset and each branch multiplication factor $\ell \in \{1, 2, 8, 50\}$. STD stands for Standard and EV for Eventful

Ortho-MCL does not predict paralogs, aside from the recent ones. In our computation of accuracy, we interpreted as paralogy the set relations not predicted as orthology—otherwise, leaving these relations as undetermined made the accuracy too low for comparison.)

In the simulated datasets containing divergent duplication edges, we also evaluated the ability to recover the primary orthologs. The metric we used is the *cluster score* defined as follows: let P_1 and P_2 be two partitions of a given set of elements. Here, P_1 would be the predicted primary orthologs, and P_2 the 'true' simulated primary orthologs. Take a graph in which the vertices are $P_1 \cup P_2$ (i.e. one vertex for each subset), add an edge X_1X_2 between each pair $X_1 \in P_1, X_2 \in P_2$ and weigh this edge by $|X_1 \cap X_2|$. Then the cluster score between P_1 and P_2 is the value of a maximum matching in this graph, divided by the number of genes. Hence the cluster score is 1 iff P_1 and P_2 are equal. The idea behind this score is that this value yields the proportion of elements from P_1 that must be moved from one set to another in order to obtain P_2 (if the score is c, then a fraction of $1 - c$ elements from P_1 need to be moved).

3.2 Simulated datasets

For our simulations, we used SimPhy (Mallo *et al.*, 2016) to generate a set of species trees along with a set of gene trees evolving within these species trees. As SimPhy incorporates speciation and duplication events in its simulations, the true sets of orthologs and paralogs are known. We generated 40 species trees and 400 gene trees—10 gene families for each species tree. The number of species in each species tree was chosen uniformly at random between 30 and 50. The gene trees were subject to the events of speciation, duplication and losses. We then used the INDELiBLE (Fletcher and Yang, 2009) module provided with SimPhy to simulate the evolution of gene sequences on each gene tree. The nucleotide evolutionary model was selected at random for each gene family, each evolving under a general-time reversible (GTR) model with rates sampled from a six-dimensional Dirichlet distribution. The only input to the programs was the set of extant gene sequences and, depending on the method, the true simulated species tree.

In order to evaluate the impact of the rates of duplications, losses and substitutions, we used four sets of parameters for the simulations, yielding four datasets named as follows: Standard, Eventful, Fast and Slow. Standard uses default parameters, Eventful has an increased rate of duplications and losses, Fast has high mutation rates and Slow has low mutation rates. For each dataset, 10 species trees were generated. In both the Standard and Eventful datasets, the tree-wide substitution rate was set to 0.000005 (the default value in SimPhy). In the Default dataset, the duplication and loss rates were set to 0.0000005 each and to 0.000001 in the Eventful dataset. Higher duplication rates led to trees with too many genes in the trees (averaging in the thousands, making our analysis too computationally intensive) while lower values yielded gene trees with too few paralogs. In the Fast dataset, the duplication/loss rates were set as in the Standard setting, but the tree-wide substitution rate was increased to 0.00005, In the Slow dataset, this parameter took value 0.0000005. The Fast dataset led to difficulties in alignments, due to the presence of many substitutions and very large gaps, whereas the Slow dataset had a small amount of substitutions, resulting in sequences with fewer differences.

These simulations do not consider the phenomenon of divergence after duplication. That is, the children of $\mathbb{D}$ nodes in the generated gene trees do not differentiate from each other, and both copies continue to evolve under the same model. Therefore, we also simulated divergent edges by choosing, for every $\mathbb{D}$ node in the simulated gene trees, one child edge arbitrarily and multiplying its length by some factor ℓ. Thus for each gene tree and each $\ell \in \{1, 2, 8, 50\}$, we used INDELiBLE to generate a set of extant gene sequences obtained by multiplying the length each divergent edge by ℓ. Each gene family gave rise to four sets of sequences, resulting in a total of 1600 gene families to analyze. In the cases where $\ell > 1$, the primary orthologs can be deduced from the divergent duplication edges.

The average accuracy, precision, recall and cluster scores, taken over the 1600 simulated gene families, are presented in Table 1. Figure 4 shows the average accuracy per gene family, for each dataset and possible value of ℓ. HyPPO outperforms the other methods on every dataset, even when not given a species tree. Moreover, the accuracy of HyPPO always improves when the true species tree is known. We observe that the accuracy of OMA tends to decrease as ℓ is increased, whereas for HyPPO and OrthoMCL, the impact of changing ℓ seems to vary.

Figure 5 shows the average cluster scores per dataset. Note that the scores for the trees having multiplication factor $\ell = 1$ are not presented, as in these cases the primary orthologs cannot be determined with certainty. Here again, we see that HyPPO is better at identifying the primary orthologs. Observe that knowledge of the species tree is inconsequential here, as the species tree does not impact the gene clustering step in our pipeline. We also observe that for all methods, the best cluster scores are achieved at $\ell = 50$, which is expected since this creates a greater separation between the primary and secondary orthologs.

The comparison of precision and recall is shown in Figure 6 for $\ell = 8$. The other values of ℓ yielded similar results. One can see that OMA typically attains better precision values. This is at the cost of a much lower recall, as HyPPO outperforms the other methods in this

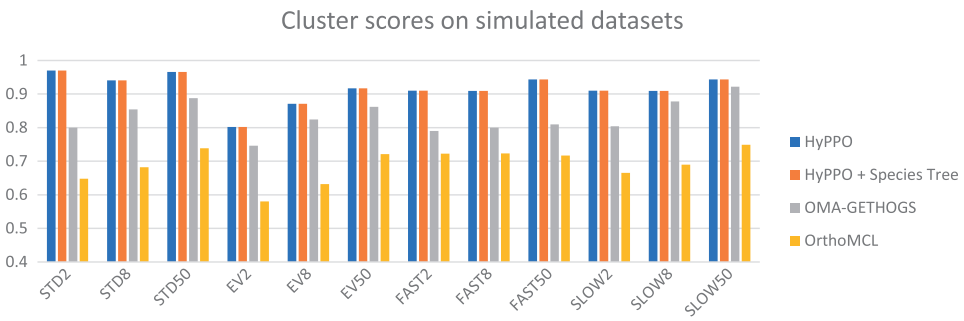

Fig. 5. Cluster scores of the methods for each dataset and each branch multiplication factor $\ell \in \{2, 8, 50\}$

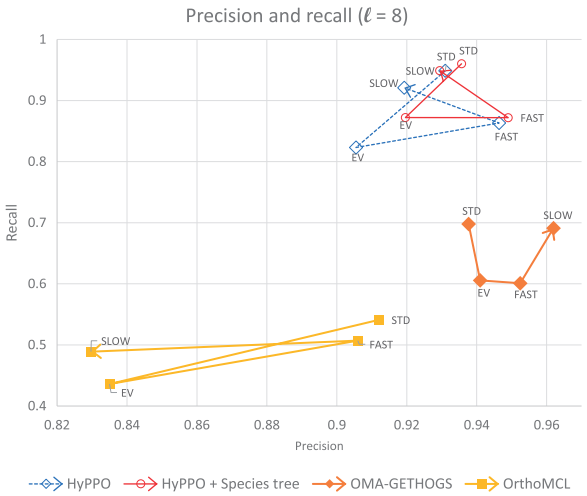

Fig. 6. Precision versus recall values of each method and each dataset for $\ell = 8$. Each method has four corresponding points, one for each dataset, and are ordered as follows: Standard—Eventful—Fast - Slow. For example the second point of each path for the analysis of the Eventful dataset

aspect. Therefore, the orthologs predicted by OMA are truly orthologs slightly more often, but the method tends to miss more of the orthology relations.

3.3 Empirical datasets

We used eight gene trees from the gold standard SwissTree database. This included the Popeye domain family (POP), the NOX 'ancestral-type' subfamily NADPH oxidases (NOX), the V-type ATPase beta subunit (VATB), the Serine incorporator family (SERC), the Sulfatase-modifying factor 1 family (SUMF), the HOX cluster genes family 9-14 (HOX), the Asterix family (ARX) and the Cited family (CITE). These gene families were chosen because they belong to Eukaryotes and, according to the curators, are not suspected to have undergone horizontal gene transfer.

Figure 7 shows the accuracy of each method for each gene tree. Contrary to the simulated datasets, HyPPO does not outperform OMA on every single dataset, as it performs poorly on the VATB and SERC families. Notably, Ortho-MCL attains almost perfect accuracy on the VATB family (0.96), whereas HyPPO is not much more than half as accurate (0.59). The reason appears to be that most gene pairs are orthologous (95.7%), since the duplications are low in this tree. HyPPO infers multiple orthology clusters joined together by duplications, resulting in many false paralogs (we predict 47% pairs as orthologous), whereas OrthoMCL puts almost all the genes into one big orthology cluster, making its predictions more accurate. A similar phenomenon appears to occur with SERC. Nevertheless, HyPPO achieves similar or superior accuracy on the other trees, and so it remains competitive. Observe that as in the simulations, when HyPPO is given the species tree (provided by SwissTree), the accuracy increases.

As for precision and recall, the four methods attained similar values. Once again, OMA offers slightly better precision at the cost of lower recall. The averages over all eight trees are presented in Table 2.

4 Discussion

Although the results of HyPPO are promising, there is still room for improvements. Each step of the pipeline could be implemented in a more sophisticated manner, and the DAD model around which HyPPO is built could be more realistic. When a duplication node is not followed by accelerated mutations, there is no special post-duplication divergent edge, and it could be argued that the orthology clusters should contain in-paralogs. In the future, we might thus consider merging the clusters that are share a high enough degree of similarity. We have also ignored the effects of convergent evolution, even though it is possible that two independent divergent duplication events lead to similar genes. If this occurs, one cannot guarantee that similar genes are orthologs, and that primary orthologs form cliques.

On the practical side, we assume that the genes are already partitioned into homologous families, which allow the computation of a multiple sequence alignement and eases the clustering phase. Future versions of the framework should be able to predict orthologs and paralogs from whole proteomes, as OrthoMCL and OMA-GETHOGS do. However, one could also argue that partitioning genes into homologous families and predicting orthology are two different tasks. The aforementioned programs perform both simultaneously. A better approach might be to use methods dedicated to homology clustering and then refine the orthology/paralogy relations within each family using HyPPO.

Also, the reconstruction of a species tree from the clusters is the main bottleneck of HyPPO in terms of time. To give a rough idea, in the worst case, some datasets resulted in a set of about 3000 allowable splits, which took our algorithm 4–5 h to solve. This motivates the need to improve the complexity of the dynamic programming formulation given in this paper. Also, in the near future, we plan to evaluate the quality of the predicted species tree. Note however that our current algorithm may take many arbitrary decisions, and we must refine this process before performing a careful species tree analysis. Finally, it will be interesting to include a wider range of orthology prediction software in a future comparative study and to establish a procedure to analyze primary orthologs on real

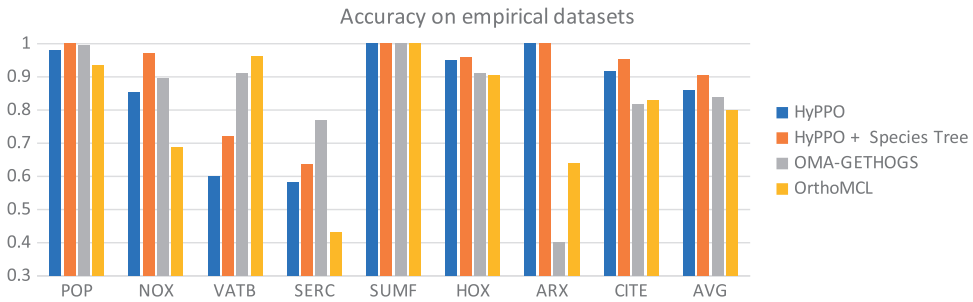

Fig. 7. Accuracy of the methods for each tree in our empirical dataset. The last column shows the average accuracy of the methods

Table 2. Average accuracy, precision and recall over the set of eight trees from SwissTree

	Accuracy	Precision	Recall
HyPPO	0.860	0.941	0.785
HyPPO + Species Tree	0.905	0.945	0.881
OMA-GETHOGS	0.837	0.950	0.711
OrthoMCL	0.800	0.859	0.680

datasets—for instance, by predicting these orthologs by comparing the ratio of branch lengths following a duplication.

5 Conclusion

In this work, we have introduced the notion of orthology in the presence of divergence after duplication and have addressed the problem that some orthologs may behave as paralogs from a functional point of view. We therefore propose to make a distinction between the orthologs that have or have not been separated by a divergent duplication event. As we have shown, the HyPPO framework is able to distinguish paralogs, primary orthologs and secondary orthologs with better accuracy than its competitors. This is true even if divergence after duplication does not always occur. Our work is far from over though, as HyPPO offers many opportunities to improve these results even further. Notably, it remains to evaluate alternative algorithms for each step in our pipeline, and to identify which ones yield the best results. Finally, our work raises some interesting questions from a theoretical standpoint. The complexity of inferring a species tree from gene clusters is left open, and it remains to evaluate how many clusters are necessary to reconstruct a high quality species tree.

Funding

M.L. was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

Conflict of Interest: none declared.

References

Altenhoff,A.M. *et al.* (2012) Resolving the ortholog conjecture: orthologs tend to be weakly, but significantly, more similar in function than paralogs. *PLoS Comput. Biol.*, **8**, e1002514.

Altenhoff,A.M. and Dessimoz,C. (2012) Inferring orthology and paralogy. *Methods Mol. Biol.*, **855**, 259–279.

Altenhoff,A.M. *et al.* (2013) Inferring hierarchical orthologous groups from orthologous gene pairs. *PLoS One*, **8**, e53786.

Böcker,S. *et al.* (2009) Going weighted: parameterized algorithms for cluster editing. *Theor. Comput. Sci.*, **410**, 5467–5480.

Böcker,S. *et al.* (2011) Exact algorithms for cluster editing: evaluation and experiments. *Algorithmica*, **60**, 316–334.

Cardoso-Moreira,M. *et al.* (2016) Evidence for the fixation of gene duplications by positive selection in drosophila. *Genome Res.*, **26**, 787–798.

Chen,X. and Zhang,J. (2012) The ortholog conjecture is untestable by the current gene ontology but is supported by rna sequencing data. *PLoS Comput. Biol.*, **8**, e1002784.

Dondi,R. *et al.* (2017a) Approximating the correction of weighted and unweighted orthology and paralogy relations. *Algorithms Mol. Biol.*, **12**, 4.

Dondi,R. *et al.* (2017b) Orthology correction for gene tree reconstruction: Theoretical and experimental results. *Proc. Comput. Sci.*, **108**, 1115–1124.

Doyle,M.A. *et al.* (2010) Drug target prediction and prioritization: using orthology to predict essentiality in parasite genomes. *BMC Genomics*, **11**, 222.

Doyon,J.-P. *et al.* (2011) Models, algorithms and programs for phylogeny reconciliation. *Brief. Bioinformatics*, **12**, 392.

Emms,D.M. and Kelly,S. (2015) Orthofinder: solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy. *Genome Biol.*, **16**, 157.

Fitch,W.M. (2000) Homology: a personal view on some of the problems. *Trends Genet.*, **16**, 227–231.

Fletcher,W. and Yang,Z. (2009) Indelible: a flexible simulator of biological sequence evolution. *Mol. Biol. Evol.*, **26**, 1879–1888.

Gabaldón,T. and Koonin,E.V. (2013) Functional and evolutionary implications of gene orthology. *Nat. Rev. Genet.*, **14**, 360–366.

Hartung,S. and Hoos,H.H. (2015). Programming by optimisation meets parameterised algorithmics: a case study for cluster editing. In *International Conference on Learning and Intelligent Optimization*, pp. 43–58. Springer.

Hellmuth,M. *et al.* (2013) Orthology relations, symbolic ultrametrics, and cographs. *J. Math. Biol.*, **66**, 399–420.

Hellmuth,M. *et al.* (2015) Phylogenomics with paralogs. *Proc. Natl. Acad. Sci.USA*, **112**, 2058–2063.

Hernandez-Rosales,M. *et al.* (2012) From event-labeled gene trees to species trees. *BMC Bioinformatics*, **13**, S6.

Innan,H. and Kondrashov,F. (2010) The evolution of gene duplications: classifying and distinguishing between models. *Nat. Rev. Genet.*, **11**, 97.

Jones,M. *et al.* (2016) On the consistency of orthology relationships. *BMC Bioinformatics*, **17**, 416.

Jordan,I.K. *et al.* (2004) Duplicated genes evolve slower than singletons despite the initial rate increase. *BMC Evol. Biol.*, **4**, 1–22.

Jothi,R. *et al.* (2006) Coco-cl: hierarchical clustering of homology relations based on evolutionary correlations. *Bioinformatics*, **22**, 779–788.

Katoh,K. *et al.* (2002) Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic Acids Res.*, **30**, 3059–3066.

Koonin,E.V. (2005) Orthologs, paralogs, and evolutionary genomics. *Ann. Rev. Genet.*, **39**, 309–338.

Kristensen,D.M. *et al.* (2011) Computational methods for gene orthology inference. *Brief. Bioinformatics*, **12**, 379–391.

Lafond,M. *et al.* (2016) The link between orthology relations and gene trees: a correction perspective. *Algorithms Mol. Biol.*, **11**, 4.

Lafond,M. and El-Mabrouk,N. (2014) Orthology and paralogy constraints: satisfiability and consistency. *BMC Genomics*, **15**, S12.

Lechner,M. *et al*. (2011) Proteinortho: detection of (co-) orthologs in large-scale analysis. *BMC Bioinformatics*, **12**, 124.

Li,L. *et al*. (2003) Orthomcl: identification of ortholog groups for eukaryotic genomes. *Genome Res*., **13**, 2178–2189.

Lynch,M. and Conery,J.S. (2000) The evolutionary fate and consequences of duplicate genes. *Science*, **290**, 1151–1155.

Mallo,D. *et al*. (2016) Simphy: phylogenomic simulation of gene, locus, and species trees. *Syst. Biol*., **65**, 334–344.

Mirarab,S. *et al*. (2014) Astral: genome-scale coalescent-based species tree estimation. *Bioinformatics*, **30**, i541–i548.

Nehrt,N.L. *et al*. (2011) Testing the ortholog conjecture with comparative functional genomic data from mammals. *PLoS Comput. Biol*., **7**, e1002073.

O'Brien,K.P. *et al*. (2004) Inparanoid: a comprehensive database of eukaryotic orthologs. *Nucleic Acids Res*., **33**, D476.

Powell,S. *et al*. (2012) eggnog v3. 0: orthologous groups covering 1133 organisms at 41 different taxonomic ranges. *Nucleic Acids Res*., **40**, D284.

Roth,A.C. *et al*. (2008) Algorithm of oma for large-scale orthology inference. *BMC Bioinformatics*, **9**, 518.

Soria,P.S. *et al*. (2014) Functional divergence for every paralog. *Mol. Biol. Evol*., **31**, 984–992.

Stolzer,M. *et al*. (2012) Inferring duplications, losses, transfers and incomplete lineage sorting with nonbinary species trees. *Bioinformatics*, **28**, i409–i415.

Studer,R.A. and Robinson-Rechavi,M. (2009) How confident can we be that orthologs are similar, but paralogs differ? *Trends Genet*., **25**, 210–216.

Swenson,K.M. and El-Mabrouk,N. (2012) Gene trees and species trees: irreconcilable differences. *BMC Bioinformatics*, **13**, S15.

Tatusov,R.L. *et al*. (2003) The cog database: an updated version includes eukaryotes. *BMC Bioinformatics*, **4**, 41.

Thomas,P.D. *et al*. (2012) On the use of gene ontology annotations to assess functional similarity among orthologs and paralogs: a short report. *PLoS Comput. Biol*., **8**, e1002386.

Train,C.-M. *et al*. (2017) Orthologous matrix (oma) algorithm 2.0: more robust to asymmetric evolutionary rates and more scalable hierarchical orthologous group inference. *Bioinformatics*, **33**, i75–i82.

Ullah,I. *et al*. (2015) Integrating sequence evolution into probabilistic orthology analysis. *Syst. Biol*., **64**, 969–982.

Van der Heijden,R.T. *et al*. (2007) Orthology prediction at scalable resolution by phylogenetic tree analysis. *BMC Bioinformatics*, **8**, 83.

Wapinski,I. *et al*. (2007) Automatic genome-wide reconstruction of phylogenetic gene trees. *Bioinformatics*, **23**, i549–i558.

Woods,S. *et al*. (2013) Duplication and retention biases of essential and non-essential genes revealed by systematic knockdown analyses. *PLoS Genet*., **9**, e1003330.

Zhang,J. (2003) Evolution by gene duplication: an update. *Trends Ecol. Evol*., **18**, 292–298.

Zheng,C. *et al*. (2011) Omg! orthologs in multiple genomes–competing graph-theoretical formulations. In *International Workshop on Algorithms in Bioinformatics*. Springer, pp. 364–375.