The Author(s) *BMC Bioinformatics* 2016, **17**(Suppl 14):416
DOI 10.1186/s12859-016-1267-3

BMC Bioinformatics

RESEARCH Open Access

On the consistency of orthology relationships

Mark Jones[1], Christophe Paul[1] and Céline Scornavacca[2*]

From 14th Annual Research in Computational Molecular Biology (RECOMB) Comparative Genomics Satellite Workshop
Montreal, Canada. 11-14 October 2016

Abstract

Background: Orthologs inference is the starting point of most comparative genomics studies, and a plethora of methods have been designed in the last decade to address this challenging task. In this paper we focus on the problems of deciding consistency with a species tree (known or not) of a *partial* set of orthology/paralogy relationships $\mathcal{C}$ on a collection of *n* genes.

Results: We give the first polynomial algorithm – more precisely a $O(n^3)$ time algorithm – to decide whether $\mathcal{C}$ is consistent, even when the species tree is unknown. We also investigate a biologically meaningful optimization version of these problems, in which we wish to minimize the number of duplication events; unfortunately, we show that all these optimization problems are NP-hard and are unlikely to have good polynomial time approximation algorithms.

Conclusions: Our polynomial algorithm for checking consistency has been implemented in Python and is available at https://github.com/UdeM-LBIT/OrthoPara-ConstraintChecker.

Keywords: Orthology detection, Polynomial-time algorithms, Para-NP hardness, Inapproximability

Background

Two genes from two different species are said to be *orthologous* if they derived from a single gene present in the last common ancestor of the two species via a speciation event, and *paralogous* if they were created by a duplication event [1]. Orthologs inference is the starting point of most comparative genomics studies, and is also a key instrument for functional annotation of new genomes. A plethora of methods have been designed in the last decade to address this challenging task, and can be roughly divided in two groups [2]. The first group of methods use clustering algorithms to detect homologous genes, i.e., genes sharing a common ancestry, and then reconstruct a *gene tree* describing the evolutionary history of this set of genes; orthology relationships are then deduced from this tree by comparing it with the *species tree*, i.e., the tree depicting the history of the species containing those genes, via *reconciliation algorithms* (see [3], among others, and [4] for a review of reconciliation algorithms). The second group of methods use other sources of information, e.g. sequence similarity or synteny, to directly estimate orthology relationships [5, among others]. The first set of methods are considered to be more accurate, but they require a prior knowledge of the species tree, and are very dependent on the accuracy of the gene trees. Unfortunately, the species phylogeny is not always known and gene trees can be highly inaccurate as a result of several kinds of reconstruction artifact, e.g. long-branch attraction (LBA) [6].

The second set of methods does not suffer from these drawbacks but still has an important weakness: given a set of genes V, the set of inferred orthology/paralogy relationships $\mathcal{C}$ for V may fail to be *satisfiable*, i.e., to simultaneously co-exist in any evolutionary history for V, or *consistent* i.e., such that all displayed triplet phylogenies are included in a species tree (formal definitions are given in the next section).

In the last years, the decision problems associated with these questions have been extensively studied, both when

*Correspondence: celine.scornavacca@umontpellier.fr
[2]ISE-M, CNRS, IRD, EPHE, Université Montpellier, Montpellier, France
Full list of author information is available at the end of the article

$\mathcal{C}$ is *full*, i.e., involves a constraint for each pair of genes in V [7, 8], and when it is not [9].

In [9], the authors give $O(n^3)$ time algorithms to decide whether $\mathcal{C}$ is satisfiable and consistent under the assumption that the species tree is known – where $n = |V|$. These results hold whether $\mathcal{C}$ is a full set of constraints or not. They also showed how to decide whether $\mathcal{C}$ is satisfiable when the species tree is unknown but $\mathcal{C}$ is full (this problem was also considered in [10]).

In this paper, we extend the results of [9] by giving a $O(n^3)$ time algorithm to decide whether $\mathcal{C}$ is consistent, even when the species tree is not known and $\mathcal{C}$ is not full, and show an application on real data. Thus the problems of deciding satisfiability, deciding consistency given a species tree, and deciding consistency with an unknown species tree, are all polynomial-time solvable. We also investigate an optimization version of these problems, in which we wish to minimize the number of duplication events in the evolutionary history for V – duplication minimization is a well-known criterion in phylogenomics [11]. Unfortunately, we show that all three problems are NP-hard, even when the maximum number of duplication events is 2, and are unlikely to have good polynomial- time approximation algorithms.

Preliminaries

A *rooted tree* T with arc set $E(T)$ and node set $V(T)$ is a directed acyclic connected graph, in which every node has in-degree 1, except for a single node, the *root* – denoted by $\text{ROOT}(T)$, of in-degree 0, and where the set of nodes in T with out-degree 0 – the *leaves* of T, denoted by $L(T)$ – are univocally labeled. Throughout the paper, we will treat leaves in a tree as synonymous with the labels associated to them. We denote by $I(T)$ the set $V(T) \setminus L(T)$ – the *internal nodes* of T. If all nodes in $I(T)$ have out-degree 2, we say that T is *binary*.

Given two nodes x, y in T, we say that x is an *ancestor of y in T*, and that y is a *descendant of x in T*, if there is a directed path from x to y in T. (Note that any node x is an ancestor and descendant of itself.) If x is not an ancestor of y and y is not a ancestor of x, we say that x, y are *separated in T*. If there is an arc from x to y in T, we say that x is the *parent of y in T* and that y is a *child of x in T*.

Given a node x, let $\text{DESC}_T(x)$ denote the descendants of x in T. Let $\text{CHILD}_T(x)$ denote the set of all children of x in T. Let $\text{LEAF}_T(x) = \text{DESC}(x) \cap L(T)$, i.e. $\text{LEAF}_T(x)$ is the set of leaves in T that are descendants of x. Note that $\text{LEAF}_T(\text{ROOT}(T)) = L(T)$. Given a set A of nodes in T, let $\text{LCA}_T(A)$ denote the *least common ancestor of A in T*, that is, the unique node z such that z is an ancestor of all $x \in A$, and no descendant of z has this property. Given two nodes x, y, we will often write $\text{LCA}_T(x, y)$ as shorthand for $\text{LCA}_T(\{x, y\})$. When T is clear from context, we will often omit "in T" and simply say that x is the ancestor of y, y is the descendant of x, z is a leaf, etc.

Suppressing a non-root node x of out-degree 1 in a tree T consists of removing x and making the unique child of x a new child of the parent of x. Given a set of leaves $L' \subseteq L(T)$, the *restriction of T to L'*, denoted $T|_{L'}$, is the tree derived from T by taking the minimum subtree of T spanning L', and suppressing all non-root nodes of out-degree 1.

A *triplet* is a rooted binary tree T with $|L(T)| = 3$. Given three distinct elements x, y, z, we denote by $xy|z$ the unique triplet T with $L(T) = \{x, y, z\}$ such that $\text{LCA}_T(x, y) \neq \text{ROOT}(T)$ (or equivalently, $\text{LCA}_T(x, y) \neq \text{LCA}_T(x, z) = \text{LCA}_T(y, z)$). We say that a rooted tree T *displays* the triplet $xy|z$ if $T|_{\{x,y,z\}} = xy|z$.

Given a set of edges E over a set of vertices V, and a subset $V' \subseteq V$, we define $E[V'] = \{xy : x, y \in V', xy \in E\}$. Given graphs $G = (V, E)$ and $G' = (V', E')$, we say that G' is an *induced subgraph of G* if $V' \subseteq V$ and $E' = E[V']$, and denote G' by $G[V']$. We define $\overline{E} = \{xy : x, y \in V, xy \notin E\}$ and say that $\overline{G} = (V, \overline{E})$ is the *complement of G*. For any integer $l \geq 1$, a *path P_l* is a graph $(V = \{v_1, \ldots, v_l\}, E = \{v_i v_{i+1} : 1 \leq i \leq l - 1\}$. We note here that if a graph contains an induced P_4, then its complement contains an induced P_4 on the same four vertices.

Species trees and DS-trees. Let Σ denote a set of species. A *species tree S on Σ* is a binary rooted tree such that $L(T) = \Sigma$, used to depict the evolutionary history of the species in Σ.

Genes are said to be *homologous* if they share a common ancestor. Let V denote a set of homologous genes belonging to species in Σ. A *species assignment* of V is a function $s : V \rightarrow \Sigma$, with $s(v) = a$ representing the fact that gene v belongs to species $a \in \Sigma$. For a set $V' \subseteq V$, we define $s(V') = \{a \in \Sigma : \exists x \in V', s(x) = a\}$, and $s_{|V'} : V' \rightarrow s(V')$ such that $s_{|V'}(v) = s(v)$ for all $v \in V'$. A *DS-tree on V* is a pair (T, ℓ), where T is a binary rooted tree with leaf set V and $\ell : I(T) \rightarrow \{Dup, Spec\}$ is a function labeling each internal node x of T as a *speciation node* (if $\ell(x) = Spec$) or a *duplication node* (if $\ell(x) = Dup$). DS-trees are used to depict the evolutionary history of the genes in V. When the function ℓ is clear from context, we will often omit it and speak only of a DS-tree T.

Given two genes x, y in T, we say that x, y are *orthologs with respect to T* if $\text{LCA}_T(x, y)$ is a speciation node, and *paralogs with respect to T* otherwise. Given an undirected graph $G = (V, E)$, a DS-tree (T, ℓ) on V is a *DS-tree for G* (or G is an *orthology graph for T*) if for every $x, y \in V$, $xy \in E \Leftrightarrow \ell(\text{LCA}_T(x, y)) = Spec$. That is, x and y are adjacent in G if and only if they are orthologs with respect to T.

The presence of two homologous genes in the same species can be caused either by duplications or gene transfers [12]. So, in absence of gene transfers, homologous genes from the same species are necessarily paralogs. We formalize this idea in the following assumption.

Assumption 1 *We assume in what follows that whenever we are given a graph $G = (V, E)$ with a species assignment s, two vertices x, y of G are not adjacent if $s(x) = s(y)$.*

Cographs A *cograph* is a graph that can be generated from a single-vertex graph using the operations of *disjoint union* (taking the disjoint union of multiple graphs) and *series composition* (adding all possible edges between vertices of multiple graphs) [13]. This generation scheme yields a representation of a cograph in terms of *cotrees*. A cotree is a rooted tree T, with internal nodes labeled 0 (representing the disjoint union operation) or 1 (representing the series composition). Hence a cotree represents a graph $G = (V, E)$ if $L(T) = V$ and two vertices x and y of G are adjacent if and only if $\text{LCA}_T(x, y) = 1$. Observe that the cotree representation of a cograph is not unique. Also, while a cotree is not necessarily binary, any non-binary cotree can be transformed in linear time into a binary cotree with the same corresponding cograph. There are several characterizations of cographs. Among other characterizations, a cograph is a graph with no induced P_4 [13]. Cographs can also be viewed as graphs where each connected component has diameter at most 2.

Hellmuth et al. [8] noted that all orthology graphs (i.e. graphs for which there exists a DS-tree) can be characterized as symbolic ultrametrics [14], and showed that a graph is an orthology graph if and only if it is a cograph [8, Corollary 4].

Thus we have a useful graph-theoretic framework for deciding on the existence of a DS-tree.

Proposition 1 *For an undirected graph $G = (V, E)$, the following are equivalent:*

1. *There exists a DS-tree for G;*
2. *G contains no induced P_4, i.e. it is P_4-free;*
3. *G is a cograph.*

As cographs can be recognized in linear time [15, 16], deciding whether a graph has a DS-tree, i.e., if it is *satisfiable*, can be achieved within the same time complexity. Note, however, that not every DS-tree represents a possible evolutionary history for a set of genes. In particular, given a species assignment, different parts of a DS-tree may imply conflicting evolutionary histories for the species containing those genes. The concept of *consistency* makes this notion precise.

Consistent DS-trees. Given a DS-tree T on V, a species assignment $s : V \to \Sigma$ and a species tree S on Σ, we say that (T, s) is *consistent with S* (or *S-consistent*) if for every speciation node z in T, and distinct children x, y of z, $\text{LCA}_S(s(\text{LEAF}_T(x)))$ and $\text{LCA}_S(s(\text{LEAF}_T(y)))$ are separated in S. Given a graph $G = (V, E)$ and the species assignment s, the pair (G, s) is *consistent with S* if there exists a DS-tree T for G such that (T,s) is consistent with S. We say that G (resp. T) along with the species assignment s, is *consistent* if there exists a species tree S such that (G,s) (resp. (T,s)) is consistent with S [9].

Given a DS-tree T on V and a species assignment $s : V \to \Sigma$, let $tr(T, s)$ be the set of triplets $s(x)s(y)|s(z)$ for which the triplet $xy|z$ is displayed by T with a speciation node as the root, and for which $s(x) \neq s(y)$.

Hernandez-Rosales et al. [7] showed that (T, s) is consistent with a species tree S if and only S displays all triplets in $tr(T, s)$. In light of this result, Hellmuth et al. [10] gave a framework for finding the DS-tree and species tree for which the maximum number of triplets are displayed, using Integer Linear Programming. Lafond and El-Mabrouk [9] improved the result of [7] by showing that it is enough to consider only the triplets in $tr(T, s)$ that have a speciation node as the root node and a duplication node as the other internal node. This can expressed in terms of the consistency of an orthology graph in the following way.

Given a graph $G = (V, E)$ and species assignment $s : V \to \Sigma$, define the set of triplets $P_3(G, s) = \{s(x)s(y)|s(z) : xz, zy \in E \text{ and } xy \notin E \text{ and } s(x) \neq s(y)\}$. Note that as a consequence of Assumption 1, if $s(x)s(y)|s(z) \in P_3(G, s)$, then $s(z) \neq s(y)$ and $s(z) \neq s(x)$.

By Theorem 5 in [9], we have the following theorem (in fact, Theorem 5 in [9] only states that (G, s) is consistent if and only if there *exists* a species tree S which displays all triplets in $P_3(G, s)$, but their proof shows that (G, s) is indeed consistent with such an S):

Theorem 1 [9] *Let $G = (V, E)$ have a DS-tree and let $s : V \to \Sigma$ be a species assignment. Let S be a species tree on Σ. Then (G, s) is consistent with S if and only if S displays all triplets in $P_3(G, s)$.*

Theorem 1 directly provides a polynomial time algorithm to decide whether a graph and a species assignment are consistent with a given species tree. The following proposition reformulates Theorem 1 in a convenient way:

Proposition 2 *Given a graph $G = (V, E)$, a species assignment $s : V \to \Sigma$, and a species tree S, (G, s) is consistent with S if and only if the following holds:*

1. *G does not contain an induced P_4;*
2. *Every triplet in $P_3(G, s)$ is displayed by S.*

As both of the properties in Proposition 2 are hereditary, we also have:

Corollary 1 *Given a graph $G = (V, E)$, a species assignment s and a subset $V' \subseteq V$, if (G, s) is consistent with a species tree S then $(G[V'], s_{|V'})$ is consistent with the species tree $S|_{s(V')}$.*

Constraint graphs. A *constraint graph* is a pair (G, s) where $G = (V, M \uplus U)$ is an edge-bicolored graph and s is a species assignment on V. A constraint graph aims at representing the partial knowledge about the orthology or paralogy relations between genes from V. The edges in M are *mandatory edges*, representing the pairs of genes xy for which we know that x and y are orthologs. The non-edges of G (i.e. the set of unordered pairs uv for which $uv \notin M \uplus U$) represent the pairs of genes xy for which we know that x and y are paralogs. The edges in U are *unknown edges*, for which we do not know if x and y are orthologs or paralogs. From Assumption 1, we have that $xy \notin M \uplus U$ for any pair of genes x, y such that $s(x) = s(y)$ (in absence of gene transfers, homologous genes from the same species are necessarily paralogs). Note that an orthology graph is a constraint graph where $U = \emptyset$. A *sandwich* of a constraint graph (G, s), with $G = (V, M \uplus U)$, is a graph $H = (V, E)$ such that $M \subseteq E \subseteq M \cup U$.

As a gene is always associated with the species it belongs to, throughout this paper we will always present a DS-tree T together with a species assignment s. Thus we will speak of a DS-tree (T, s). Similarly, we will always present an orthology graph G together with its species assignment s, and speak of an orthology graph (G, s). A sandwich graph G' will be presented on its own without a species assignment, as a sandwich graph is defined relative to a constraint graph $(G = (V, M \uplus U), s)$, and so the species assignment s will always be clear from context.

Methods

Computing a consistent DS-tree

In this section, we describe a polynomial time algorithm for the following problem:

CONSISTENT ORTHOLOGY GRAPH SANDWICH problem
Input: a constraint graph (G, s), with $G = (V, M \uplus U)$ and $s : V \to \Sigma$ a species assignment;
Output: a sandwich graph H for (G, s) such that (H, s) is consistent (if any exists).

Observe that by Proposition 2, the CONSISTENT ORTHOLOGY GRAPH SANDWICH problem amounts to computing a sandwich cograph satisfying extra properties. The sandwich cograph problem is known to be polynomial time solvable [17]. Our algorithm can be seen as a combination of the sandwich cograph algorithm and the BUILD algorithm [18] for checking consistency of a set of triplets.

Let $G = (V, M \uplus U)$ be an edge-bicolored graph and for $F \subseteq U$, define the graph $G(F) = (V, M \cup F)$.

The first lemma proves that unknown edges between connected components of $G(\emptyset)$ can be removed (i.e. freezed as paralogy relations between genes).

Lemma 1 *Let (G, s) be a constraint graph with $G = (V, M \uplus U)$. Let CC be the connected components of $G(\emptyset)$, and let $U_{CC} = \bigcup_{C \in CC} U[C]$. There exists a consistent sandwich graph of (G, s) if and only if there exists a consistent sandwich graph of $(G_{CC} = (V, M \uplus U_{CC}), s)$.*

Proof Suppose first that there exists a consistent sandwich graph $G' = (V, E')$ of (G, s) and let S be a species tree such that (G', s) is S-consistent. For every $C \in CC$, by Corollary 1 $(G'[C], s_{|C})$ is consistent with $S|_{s(C)}$ and hence with S. Then the disjoint union G'' of the $G'[C]$ is a sandwich cograph of (G_{CC}, s). Moreover we clearly have $P_3(G'', s) = \cup_C P_3(G'[C], s_{|C})$, implying that (G'', s) is also consistent with S. The converse is symmetric. □

Reduction Rule 1 *Let (G, s) be a constraint graph with $G = (V, M \uplus U)$. Remove from U every edge xy such that x and y belong to distinct connected components of $G(\emptyset)$.*

As an example, consider the constraint graph (G, s) in Fig. 1. The genes a_1, b_1, c_1, d_1 form one connected component of $G(\emptyset)$, and a_2, b_2, c_2, d_2 form the other. Thus Reduction Rule 1 will remove the unknown edge d_1a_2 from U.

Note that although we remove all edges between connected components of $G(\emptyset)$, we cannot solve the problem on each connected component independently, and so we cannot assume that $G(\emptyset)$ is connected. The reason is that for two connected components C, D of $G(\emptyset)$, a solution for $(G[C], s_{|C})$ may be consistent with a different species tree than a solution for $(G[D], s_{|D})$. To avoid conflicts between solutions on different subgraphs, we must split the graph into subgraphs on disjoint sets of species.

From now on, we may assume that $|s(V)| > 1$. Otherwise, Assumption 1 implies that $M = U = \emptyset$, and thereby (G, s) is a trivial positive instance. For the sake of the algorithm, we define an auxiliary graph $H_{G,s} = (\Sigma, F)$ on the species set, called hereafter the *species graph*. For each pair of distinct species $a, b \in \Sigma$, add ab to F if there exist $x, y \in V$ such that x and y are in the same connected component of $G(\emptyset)$, x and y are not adjacent in $G(U)$ and $s(x) = a$, $s(y) = b$.

Lemma 2 *Let (G, s) be a constraint graph reduced by Reduction Rule 1. If the species graph $H_{G,s}$ is connected, then (G, s) does not have a consistent sandwich graph.*

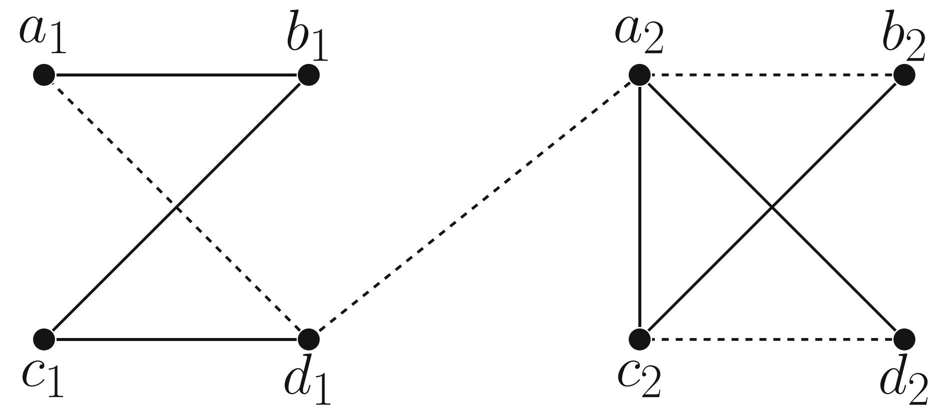

Fig. 1 Example of an constraint graph $(G = (V, M \uplus U), s)$, with $s(a_1) = s(a_2) = A, s(b_1) = s(b_2) = B, s(c_1) = s(c_2) = C, s(d_1) = s(d_2) = D$. The mandatory edges M are *solid*; the unknown edges U are *dashed*. Reduction Rule 1 will delete the unknown edge d_1a_2

Proof Consider an arbitrary binary species tree S, and an arbitrary sandwich graph $G' = (V, E')$ of (G, s). We show that $P_3(G', s)$ contains a triplet not displayed by S.

Let $A = \text{LEAF}_S(u_A)$ and $B = \text{LEAF}_S(u_B)$ where u_A and u_B are the children of $\text{ROOT}(S)$. Note that A and B partition the set of species Σ. As $H_{G,s}$ is connected, there exists $a \in A, b \in B$ such that $ab \in F$. Therefore there exist $x, y \in V$ such that x, y are in the same connected component C of $G(\emptyset), s(x) = a, s(y) = b$ and $xy \notin M \cup U$.

As $G'[C]$ is connected, there exists a chordless path P from x to y in G'. By Proposition 2, G' is P_4-free. This implies that P contains, in addition to x and y, a third vertex z such that $xz \in E'$ and $zy \in E'$.

Assume without loss of generality that $s(z) \in A$ (the case $s(z) \in B$ is symmetric). Then we have $s(x)s(y)|s(z) \in P_3(G')$. Note however that $\text{LCA}_S(s(y), s(z)) = \text{ROOT}(S)$ (as $s(z) \in A, s(y) \in B$), while $\text{LCA}_S(s(x), s(z))$ is a descendant of $\text{LCA}_S(A)$. It follows that $\text{LCA}_S(s(x), s(z))$ is different from $\text{LCA}_S(s(y), s(z))$, and so $s(x)s(y)|s(z)$ is not displayed by S. □

The next lemma shows how to use connected components of the species graph in order to freeze some unknown edges to orthology relations between genes.

Lemma 3 *Let (G, s) be a constraint graph reduced by Reduction Rule 1 such that the species graph $H_{G,s}$ is not connected.*

Let A be the vertices of a connected component of the species graph $H_{G,s}$ and let $B = \Sigma \setminus A$. Let $G_A = (V_A, M[V_A] \uplus U[V_A])$ and $G_B(V_B, M[V_B] \uplus U[V_B])$, where $V_A = s^{-1}(A)$ and $V_B = s^{-1}(B)$. There exists a consistent sandwich graph of (G, s) if and only if there exist consistent sandwich graphs of $(G_A, s_{|V_A})$ and of $(G_B, s_{|V_B})$.

Proof Let G'_A and G'_B be respectively consistent sandwich graphs of $(G_A, s_{|V_A})$ and of $(G_B, s_{|V_B})$. Suppose that G'_A is consistent with the species tree S_A and G'_B with S_B. For every connected component C of $G(\emptyset)$, let G'_C be the series composition of $G'_A[C]$ and $G'_B[C]$ and let $G' = (V, E')$ be the disjoint union of all G'_C's. We now show that (G', s) is a consistent sandwich graph of (G, s).

As G'_A and G'_B are cographs, by construction G' is a cograph too. Now, as G'_A and G'_B are respectively sandwich graphs of $(G_A, s_{|V_A})$ and $(G_B, s_{|V_B})$, and as there is no edge in M between different connected components of $G(\emptyset)$, we have that $M \subseteq E'$. By construction of $H_{G,s}$ and the fact that $H_{G,s}$ has no edges between A and B, for every connected component C of $G(\emptyset)$, if $x \in V_A \cap C$ and $y \in V_B \cap C$, then $xy \in M \cup U$. As G'_A and G'_B are respectively sandwich graphs of $(G_A, s_{|V_A})$ and $(G_B, s_{|V_B})$, this implies that $E' \subseteq M \cup U$. It follows that G' is a sandwich graph of G.

Now consider the species tree S obtained from S_A and S_B by adding a root whose children are $\text{ROOT}(S_A)$ and $\text{ROOT}(S_B)$. We claim that (G', s) is consistent with S. Consider a triplet $s(x)s(y)|s(z) \in P_3(G', s)$. We distinguish two cases:

- If $\{s(x), s(y), s(z)\} \subseteq A$ (the case $\{s(x), s(y), s(z)\} \subseteq B$ is symmetric), then $s(x)s(y)|s(z) \in P_3(G_A)$ and is displayed by S_A and thereby by S as well.
- Otherwise, as $xz, yz \in E', x$ and y are connected in G' and so by construction of G', we have that $x, y \in C$ for some connected component C of $G(\emptyset)$. As $xy \notin E'$, by construction of G'_C either $\{s(x), s(y)\} \subseteq A$ or $\{s(x), s(y)\} \subseteq B$. Suppose w.l.o.g that the former holds, implying $s(z) \in B$. Observe then that $s(x)s(y)|s(z)$ is displayed by S. Indeed, we have $\text{LCA}_S(s(x), s(z)) = \text{LCA}_S(s(y), s(z)) = \text{ROOT}(S)$, and $\text{LCA}_S(s(x)s(y))$ is a descendant of $\text{ROOT}(S_A)$.

The converse follows from Corollary 1. □

The correctness of the next branching rule follows from Lemma's 2 and 3.

Branching Rule 1 *Let (G, s) be a constraint graph reduced by Reduction Rule 1 such that the species graph $H_{G,s}$ is not connected. Let A be a connected component of the species graph $H_{G,s}$ and let $B = \Sigma \setminus A$. Solve* CONSISTENT SANDWICH SUBGRAPH *on $(G_A, s_{|V_A})$ and $(G_B, s_{|V_B})$ where $V_A = s^{-1}(A)$ and $V_B = s^{-1}(B)$. If there exist $G'_A = (V_A, E'_A)$ and $G'_B = (V_B, E'_B)$ that are respectively consistent sandwich graphs of $(G_A, s_{|V_A})$ and $(G_B, s_{|V_B})$, then return $G' = (V, E'_A \cup E'_B \cup M')$, where $M' = \{xy \in M \cup U : x \in V_A, y \in V_B\}$. Otherwise, return* NULL.

Consider again the example of Fig. 1, after the unknown edge d_1a_2 has been removed by Reduction Rule 1. Because one connected component has non-edges a_1c_1, b_1d_1 and the other has non-edge b_2d_2, the edges in $H_{G,s}$ will be AC and BD (see Fig. 2). Thus, Branching Rule 1 will split the constraint graph into two parts, one restricted to a_1, c_1, a_2, c_2, and one restricted to b_1, d_1, b_2, d_2.

We can now give the pseudocode of the algorithm, which essentially consists of alternately applying Reduction Rule 1 and Branching Rule 1.

Theorem 2 *Given a constraint graph (G, s), the* CONSISTENT ORTHOLOGY GRAPH SANDWICH *problem can be solved in $O(n^3)$ time, where n is the number of genes in G.*

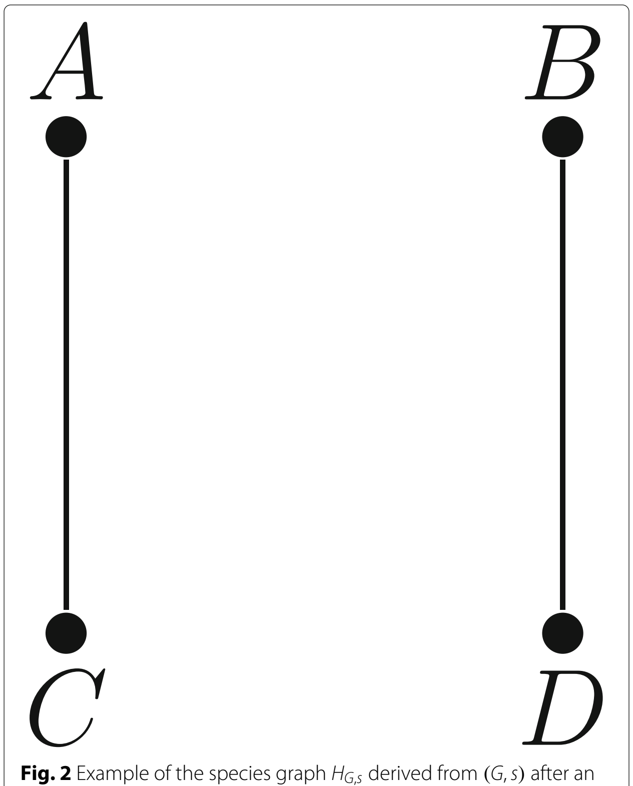

Fig. 2 Example of the species graph $H_{G,s}$ derived from (G, s) after an application of Reduction Rule 1. The edge *AC* is due to non-edge a_1c_1; edge *BD* is due to non-edges b_1d_1 and b_2d_2. As $H_{G,s}$ is not connected, we can apply Branching Rule 1

Algorithm 1: COSG(G, s)

```
input : A constraint graph $(G, s)$ with $G = (V, M \uplus U)$
        and a species assignment $s : V \to \Sigma$;
output: A consistent sandwich graph $G' = (V, E')$
        such that $M \subseteq E' \subseteq M \uplus U$, if one exists;
        otherwise return NULL.
Apply Reduction Rule 1;
if $|s(V)| = 1$ then
    return $G$;
else
    Compute the species graph $H_{G,s}$;
    if $H_{G,s}$ is connected then
        return NULL;
    else
        Let $G_A, G_B$ be the graphs and $M'$ the set of
        edges defined in Branching Rule 1;
        $G'_A \leftarrow COSG(G_A, s_{|V_A})$ and
        $G'_B \leftarrow$ COSG$(G_B, s_{|V_B})$;
        if $G'_A$ is not NULL and $G'_B$ is not NULL then
            return $(G, E(G'_A) \cup E(G'_B) \cup M')$;
        else return NULL
```

Proof The correctness of Algorithm 1 follows from the correctness of Reduction Rule 1 (Lemma 1) and Branching Rule 1 (Lemma's 2 and 3).

To analyze the running time of Algorithm 1, we simply observe that the recursive calls define a binary tree structure with at most $O(|\Sigma|) = 0(n)$ nodes. As each step of the recursion can clearly be performed in quadratic time, so the complexity follows. □

We can adapt the algorithm to cases when the species tree S is partially known, by adjusting the construction of $H_{G,s}$. In particular, for any $x, y, z \in V$ for which it is known that S displays the triplet $s(x)s(y)|s(z)$, we add $s(x)s(y)$ as an edge in $H_{G,s}$.

Algorithm 1 has important applications. When the species tree is not known, it allows us to differentiate constraint graphs that are consistent with a species tree from those that are not; the latter cannot be depicted by a consistent DS-tree, and should be considered as phylogenetically irrelevant and discarded. When the species tree S is known and a given constraint graph C is not consistent with it, the sandwich graph returned by Algorithm 1 shows to what extent C and S are in contradiction. Furthermore if S contains some uncertainties, it allows us to see if the contradictions between C and S lie in the "uncertainty zone" of S. This may help to correct the species tree.

As an example of the last appplication, suppose that we have the species tree given in Fig. 3(a), but the relative

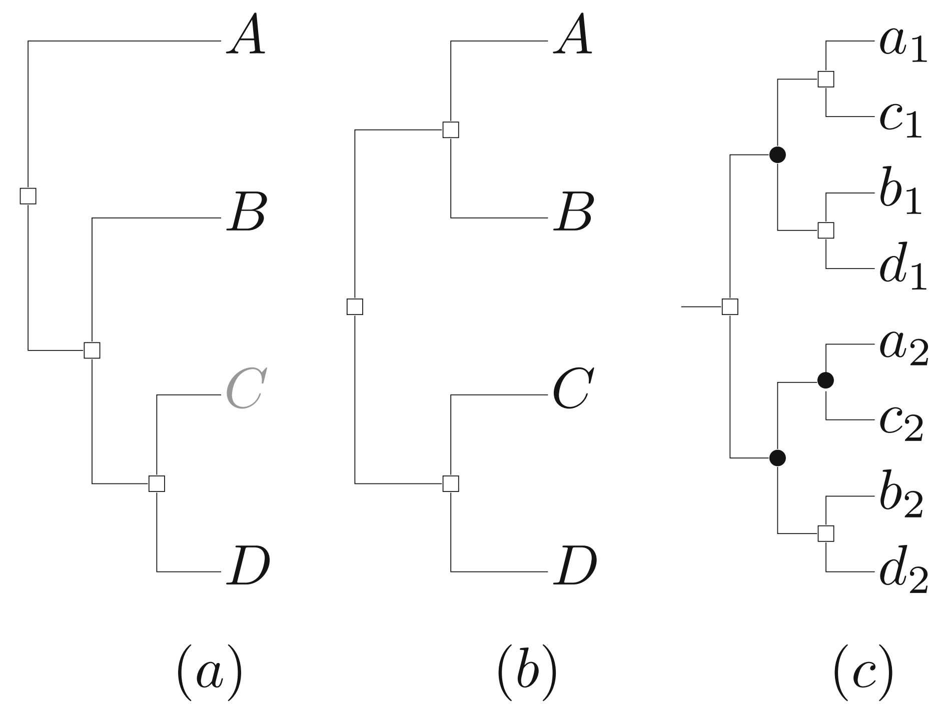

Fig. 3 Example of (**a**) a species tree where the placement of *C* is uncertain, and **b**) another species tree that can be derived from the first by changing the position of *C*. The DS-tree in (**c**) is not consistent with the species tree in (**a**)(assuming $s(a_1) = s(a_2) = A, s(b_1) = s(b_2) = B, s(c_1) = s(c_2) = C, s(d_1) = s(d_2) = D$), but it is consistent with the species tree in (**b**). In (**c**), *circles* represent speciation events, and *squares* represent duplication events

position of species C in this tree is uncertain. Suppose in addition we are given the constraint graph (G, s) given in Fig. 1. The DS-tree in Fig. 3(c) is a DS-tree for (G, s), but is not consistent with the Fig. 3(a). However, it is consistent with the species tree in Fig. 3(b), which can be derived from Fig. 3(a) by moving species C.

See the "Results and Discussion" section for an example of application on real data.

Hardness of optimizing the duplication nodes

Given a constraint graph (G, s) for which there exist several possible DS-trees, we may be interested in finding one minimizing the number of duplication nodes. Duplication minimization is a well-known criterion in phylogenomics [4, 11]; for example, it is used to resolve polytomies in gene trees in [19] and to estimate the species tree in [20].

In this section, we consider the following three optimization variants of the ORTHOLOGY GRAPH SANDWICH problem in which the number of duplication nodes has to be minimized. We prove hardness results for each of these problems.

k-DUPLICATION ORTHOLOGY GRAPH SANDWICH problem (k-DOGS)

Input: a constraint graph (G, s) and an integer k;

Output: does there exists a DS-tree (T, s) containing at most k duplication nodes, whose orthology graph is a sandwich of G?

The above problem is equivalent to asking if (G, s) is satisfiable and there exists a DS-tree for (G, s) containing at most k duplication nodes.

SPECIES TREE CONSISTENT k-DUPLICATION ORTHOLOGY GRAPH SANDWICH problem (S-CONS-k-DOGS)

Input: a constraint graph (G, s), with $G = (V, M \uplus U)$ and $s : V \to \Sigma$ a species assignment, a species tree S on Σ and an integer k;

Question: does there exist a DS-tree (T, s) containing at most k duplication nodes, whose orthology graph is a sandwich of G, and is consistent with S?

CONSISTENT k-DUPLICATION ORTHOLOGY GRAPH SANDWICH Problem (CONS-k-DOGS)

Input: a constraint graph (G, s), with $G = (V, M \uplus U)$ and $s : V \to \Sigma$ a species assignment, and an integer k;

Question: does there exist a DS-tree (T, s) containing at most k duplication nodes and a species tree S, such that the orthology graph of (T, s) is a sandwich of G and is consistent with S?

We first provide a reduction from 3-Coloring that proves that k-DOGS is para-NP-hard [21] with respect to the number of duplication nodes k (that is, k-DOGS is NP-hard for some fixed k). This implies that k-DOGS does not belong to the complexity class **XP**, meaning that the problem cannot be solved in time $O(n^{f(k)})$ for some function $f(.)$. In what follows, $[k]$ denotes the set $\{1, \cdots, k\}$.

k-Coloring Problem
Input: a (connected) graph $G = (V, E)$;
Question: does there exist a k-coloring $c : V \rightarrow [k]$ such that for every $xy \in E$, $c(x) \neq c(y)$?

The following lemma will be useful in this section. An equivalent version of this lemma could be written in terms of cographs, and we believe a proof for such a lemma should already exist in the literature. However, as we were unable to find such a proof, we give one here.

Lemma 4 *Let (G, s) be an orthology graph with a DS-tree containing at most k duplication nodes. Then we can find a $k + 1$ coloring of its complement $\overline{G}$ in polynomial time.*

Proof Let $(G = (V, E), s)$ be an orthology graph. We prove the claim by induction on $|V|$.

If $|V| = 1$, then there are 0 duplication nodes in a DS-tree for (G, s), and $\overline{G}$ has a 1-coloring, as required.

So now suppose the claim holds for all orthology graphs $(G' = (V', E'), s')$ with $|V'| < |V|$. Let (T, σ) be a DS-tree for (G, s) with at most k duplication nodes. Consider $\text{ROOT}(T)$. If $\text{ROOT}(T)$ is a duplication node, then G is disconnected, and we can find a partition $V = V_A \uplus V_B$ such that there is no edge between V_A and V_B in G. Moreover, the number of duplication nodes in T is $k_A + k_B + 1$, where k_A is the number of duplication nodes in a DS-tree for $G[V_A]$, and k_B is the number of duplication nodes in a DS-tree for $G[V_B]$. By the inductive hypothesis, there exists a $k_A + 1$ coloring for $\overline{G[V_A]}$, and a $k_B + 1$ coloring for $\overline{G[V_B]}$. It is clear that we can combine these colorings into a $k_A + 1 + k_B + 1 \leq k + 1$ coloring of $\overline{G}$.

If $\text{ROOT}(T)$ is a speciation node, then $\overline{G}$ is disconnected, and we can find a partition $V = V_A \uplus V_B$ such that there are no edges between V_A and V_B in $\overline{G}$. Moreover, the number of duplication nodes in a DS-tree for $G[V_A]$ ($G[V_B]$, respectively) is at most k. By the inductive hypothesis, there exists a $k+1$ coloring for $\overline{G[V_A]}$ and a $k+1$ coloring for $\overline{G[B]}$ and these can be combined into a $k + 1$ coloring for G.

This proof can be turned into a polynomial time algorithm as follows. If G is disconnected, find a partition $V = V_A \uplus V_B$ with no edges between V_A and V_B in G, and recursively find colorings for $\overline{G[V_A]}$ and $\overline{G[V_B]}$, adjusting the coloring on $\overline{G[V_B]}$ to assign different values from those assigned by the coloring on $\overline{G[V_A]}$.

Otherwise, find a partition $V = V_A \uplus V_B$ with no edges between V_A and V_B in $\overline{G}$, and recursively find colorings for $\overline{G[V_A]}$ and $\overline{G[V_B]}$. As each recursion splits the set of vertices and each recursive step takes polynomial time, the whole algorithm takes polynomial time. □

Lemma 5 *Given a connected graph $G = (V, E)$, define a constraint graph $(H = (V, M \uplus U), s)$ by setting $M = \emptyset$ and $U = \overline{E}$, and letting $s : V \rightarrow \Sigma$ be an arbitrary species assignment such that each gene in V is assigned to a different species. Then for any integer $k > 0$, G is k-colorable if and only if (H, s) has a solution with at most $k - 1$ duplication nodes. Furthermore if such a solution exists then there exists a solution consistent with an arbitrary species tree on Σ.*

Proof Assume that G is k-colorable, with $c : V \rightarrow [k]$ a k-coloring of G. Let $V_i = c^{-1}(i)$ for each $i \in [k]$. Thus $V_1, \ldots V_k$ form a partition of V. For each $i \in [k]$, let $(T_i, s_{|V_i})$ be an arbitrary DS-tree with leaves V_i such that every internal node is a speciation node, and let x_i denote the root of T_i. We now construct a DS-tree (T, s) as follows. Let $z_1, \ldots, z_{k-1}$ be duplication nodes such that $\text{ROOT}(T) = z_1$, such that for each $i \in [k-2]$, z_i has child nodes x_i and z_{i+1}, and the children of z_{k-1} are x_{k-1} and x_k. Now consider the graph $H' = (V, E')$ obtained from the disjoint union of cliques on V_i for $1 \leqslant i \leqslant k$. Observe that H' is a sandwich graph of (H, s). Moreover by construction, we have that $xy \in E'$ if and only if $\text{LCA}_T(x, y)$ is a speciation node. Moreover (T, s) has $k - 1$ duplication nodes, so H' is a solution. To conclude the proof of the first claim, observe that the converse follows from Lemma 4. To see the second claim, observe that as H' is a disjoint union of cliques, $P_3(H', s) = \emptyset$ and therefore (H', s) is consistent with any species tree on Σ. □

As an example of the construction in the proof above, consider the graph $G = (V, E)$ given in Fig. 4. The corresponding constraint graph $(H = (V, M \uplus U), s)$ is given in Fig. 5, and a DS-tree for this constraint graph is given in Fig. 6. As this DS-tree has 2 duplication nodes, G has a 3-coloring. In particular, following the structure of Fig. 6, we observe that there is a 1-coloring of $G[\{a, e\}]$ (as these vertices are not adjacent in G), and a 1-coloring of $G[\{b, f\}]$. Combining these colorings gives a 2-coloring of $G[\{a, b, e, f\}]$, which can then be combined with a 1-coloring of $G[\{c, d\}]$ to give a 3-coloring of G.

We now prove the NP-hardness of 2-DOGS, S-Cons-2-DOGS or Cons-2-DOGS, using Lemma 5 and the fact that 3-Coloring is NP-hard [22].

Theorem 3 2-DOGS *is NP-hard.*

Proof Given an instance $G = (V, E)$ of 3-Coloring, let (H, s) be the constraint graph given by Lemma 5. Then by Lemma 5, $(H, s, 2)$ is a Yes-instance of k-DOGS if and

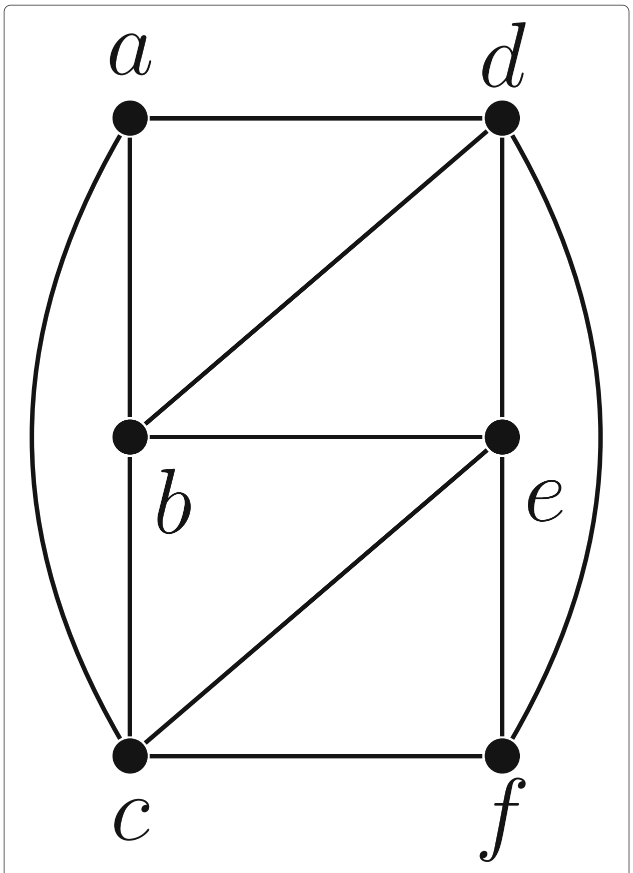

Fig. 4 A graph $G = (V, E)$

only if G is 3-colorable. As 3-COLORING is NP-hard, so is 2-DOGS. □

Using the same technique as for Theorem 3, we can prove the same NP-hardness result for S-CONS-2-DOGS and CONS-2-DOGS. The proofs are identical to that of Theorem 3, except that in the case of Theorem 4 we construct an arbitrary species tree S on Σ in addition to the constraint graph (H, s).

Theorem 4 S-CONS-2-DOGS *is NP-hard.*

Theorem 5 CONS-2-DOGS *is NP-hard.*

Let MINDOGS, S-CONS-MINDOGS, and CONS-MINDOGS denote the minimization versions of k-DOGS, S-CONS-k-DOGS, and CONS-k-DOGS respectively, in which we want to find a solution with the minimum number of duplication nodes. Let GRAPH COLORING denote the minimization version of k-COLORING. As GRAPH COLORING has no polynomial time $n^{1-\epsilon'}$-approximation for any $\epsilon' > 0$, unless P=NP [23], we can prove the following theorem.

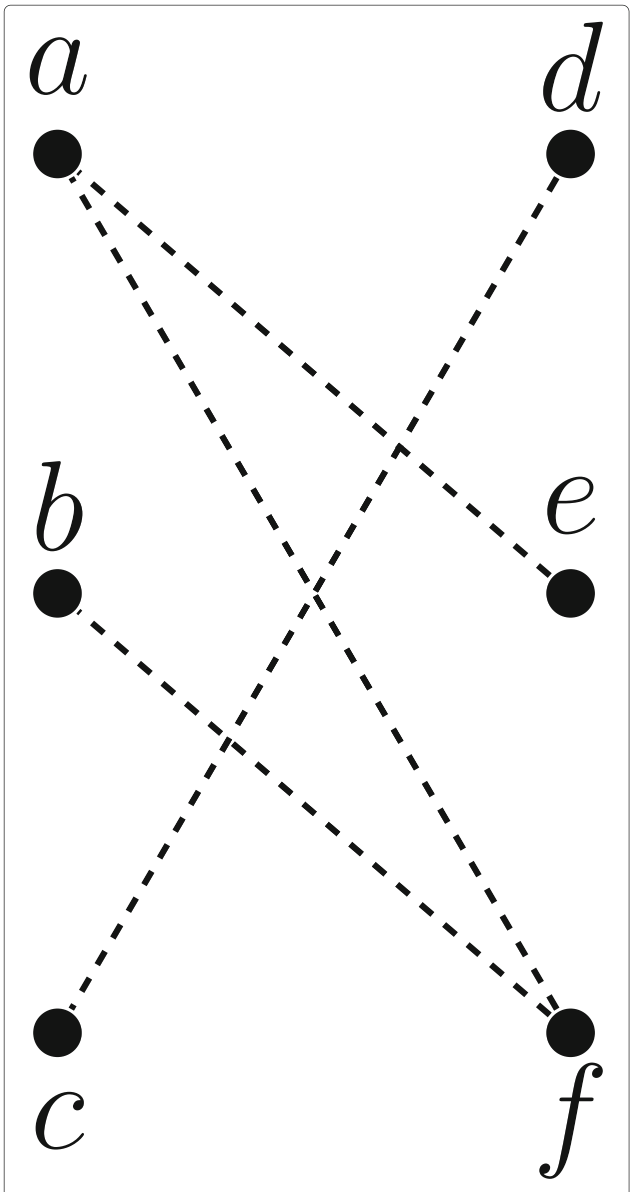

Fig. 5 A constraint graph ($H = (V, M \uplus U), s$) derived from G by setting $M = \emptyset$ and $U = \overline{E}$, and letting $s : V \to \Sigma$ be an arbitrary species assignment such that each gene is mapped to a different species

Theorem 6 *For any $\epsilon > 0$, there is no polynomial time algorithm that takes as input an instance of* MINDOGS, *and returns a solution with at most $n^{1-\epsilon} \cdot k$ duplication nodes if there exists a solution with at most k duplication nodes, unless P = NP.*

Proof Let $G = (V, E)$ be an instance of GRAPH COLORING. Without loss of generality we may assume that G is connected. Let (H, s) be the constraint graph given by Lemma 5.

Now for any $\epsilon > 0$, fix an integer n_0 and $\epsilon' > 0$ such that $n^{1-\epsilon} + 1 < n^{1-\epsilon'}$ for any $n \geq n_0$.

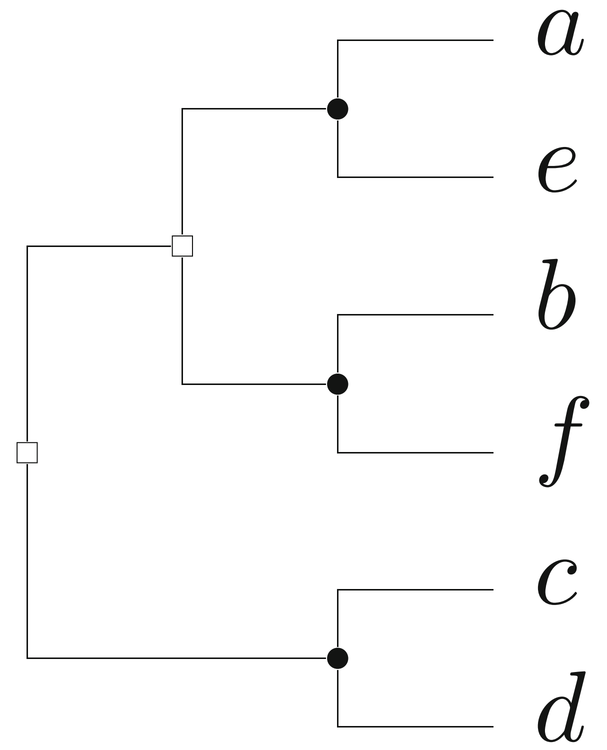

Fig. 6 A DS-tree for $(H = (V, M \uplus U), s)$. Note that the partition $\{a, e\}, \{b, f\}, \{c, d\}$ corresponds to a 3-coloring of *G*. *Circles* represent speciation events, and *squares* represent duplication events

Suppose that there exists a polynomial-time $n^{1-\epsilon}$-approximation for MINDOGS, i.e. an algorithm that for any instance (H, s) with n vertices, finds a solution with at most $n^{1-\epsilon} \cdot k$ duplication nodes if there exists a solution with at most k duplication nodes. We show that there exists a polynomial-time $n^{1-\epsilon'}$-approximation for GRAPH COLORING.

Let G be an instance of GRAPH COLORING with n vertices, and suppose without loss of generality that $n \geq n_0$ (as otherwise the problem can be solved exactly in polynomial time). Let (H, s) be the instance of MINDOGS constructed from G as above. Now run the supposed approximation algorithm for MINDOGS on (H, s). If G is k-colorable for any $k > 1$, then by Lemma 5, there exists a solution for (H, s) with at most $k - 1$ duplication nodes. Therefore if G is k-colorable, the algorithm returns a solution with at most $n^{1-\epsilon} \cdot (k-1)$ duplication nodes. (Note that we may assume the solution contains at least 1 duplication node, as otherwise G would be disconnected). Let (H', s) be the orthology graph for this solution. Then by Lemma 4, we have a $n^{1-\epsilon} \cdot (k-1) + 1$-coloring for $\overline{H'}$. As G is a subgraph of $\overline{H'}$, this is also a $n^{1-\epsilon} \cdot (k-1) + 1$-coloring for G.

As $n \geq n_0, n^{1-\epsilon} \cdot (k-1) + 1 \leq (n^{1-\epsilon} + 1) \cdot k \leq n^{1-\epsilon'} \cdot k$ and so we have a polynomial time $n^{1-\epsilon'}$-approximation for GRAPH COLORING, a contradiction. □

Using the same technique as for Theorem 6, we can prove the same inapproximability result for S-CONS-MINDOGS and CONS-MINDOGS. The proofs are identical to that of Theorem 6, except that in the case of Theorem 7 we construct an arbitrary species tree S on Σ in addition to the constraint graph (H, s).

Theorem 7 *For any $\epsilon > 0$, there is no polynomial time algorithm that takes as input an instance of* S-CONS-MINDOGS, *and returns a solution with at most $n^{1-\epsilon} \cdot k$ duplication nodes if there exists a solution with at most k duplication nodes, unless P = NP.*

Theorem 8 *For any $\epsilon > 0$, there is no polynomial time algorithm that takes as input an instance of* CONS-MINDOGS, *and returns a solution with at most $n^{1-\epsilon} \cdot k$ duplication nodes if there exists a solution with at most k duplication nodes, unless P = NP.*

To summarise the results in this section: given a constraint graph on n vertices, it is NP-hard to find a DS-tree for that graph with at most k duplication nodes, even when $k = 2$. This holds regardless of whether we require the DS-tree to be consistent, or whether we are given a species tree that it should be consistent with. Viewed as a minimization problem, it is NP-hard even to find an $n^{1-\epsilon}$-approximate solution, for any $\epsilon > 0$.

Results and Discussion

We integrated Algorithm 1 to the software provided at [24] by the authors of [9]. Note that the previous version of the program only permitted to check satisfiability and consistency of a constraint graph with respected to a given species tree *S*.

We used the modified software to reanalyze the data set in [9]. This data set was constructed by randomly choosing 265 gene families of vertebrates with more than 20 genes from *Ensembl* [25]. Each gene family was then analysed with ProteinOrtho [26] using 9 different parameter settings, yielding 2385 different constraint graphs. Here *S* is the Ensembl species tree, which can be downloaded at [27].

For this data set we have that, apart from one case, all satisfiable constraint graphs are also consistent. In 533 out of 2385 cases, the constraint graph was found to be consistent, but not consistent with *S*. We were interested in finding out how greatly the graphs in this set (denoted $\mathcal{CG}$) conflicted with *S*. Indeed, some nodes in the Ensembl species tree, for example the position of *Equus*, *Tupaia* and *Cavia*, do not enjoy a consensus in the community, so some contradictions with *S* are expected.

Note that we can use the graph G' outputted by Algorithm 1 to obtain a species tree in the following way: we compute the set $\mathcal{T}$ of all $P_3(G', s)$ and then feed $\mathcal{T}$ to the

BUILD algorithm [18], which will return a species tree displaying all the triplets in $\mathcal{T}$ (in practice, our implementation of Algorithm 1 is able to construct a species tree directly).

This species tree can fail to be binary, if the information contained in $\mathcal{T}$ is sparse (this is actually the case for our data set: the maximum number of internal nodes over all species trees reconstructed by our approach from constraint graphs in $\mathcal{CG}$ was 6, with an average of 1.5).

To estimate the discordancy between the Ensembl species tree S and each of the species trees S' reconstructed by our approach for a constraint graph in $\mathcal{CG}$, we did the following: for each pair (S, S') we constructed a tree S'' displaying the maximum number of triplets of S not contradicting S' using PhySIC_IST [28]. We then computed the number of triplets displayed by S not in S'', as a proportion of the total number of triplets displayed by S: the higher this number is, the higher the conflict between S and S'. This number, denoted $c(S, S')$, can be used to differentiate gene families that are good markers (i.e. markers highly coherent with the given species tree, which will have a low $c(S, S')$) from gene families that are bad markers (with a high $c(S, S')$). The histogram of the values of $c(S, S')$ for our data set is given in Fig. 7. This shows that several constraint graphs, even though not consistent with S, are not in high contradiction with it and thus the corresponding gene families can still be considered as good markers.

Conclusions

In this paper, we extend the results of [9] by giving a $O(n^3)$ time algorithm to decide whether $\mathcal{C}$ is consistent, even when the species tree is not known and $\mathcal{C}$ is not full. We also incorporated this algorithm into the software provided at [24]. The algorithm has important applications in providing evidence for the structure of a species tree when that species tree is unknown. It also allows us to see how much an 'inconsistent' set of constraints is in conflict with a known species tree, as the algorithm returns a species tree for which those constraints are consistent, if any exists. On the negative side, we show that the problem of minimizing duplications nodes in DS-trees is NP-hard even when the number of duplications is very small, and it is also hard to find approximate solutions for this criterion.

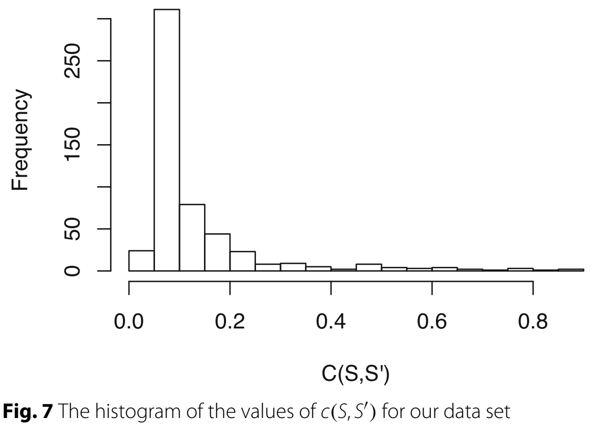

Fig. 7 The histogram of the values of $c(S, S')$ for our data set

Acknowledgements
The authors would like to thank Manuel Lafond for providing the Python script to check consistency with a known species tree.

Declarations
This article has been published as part of BMC Bioinformatics Vol 17 Suppl 14, 2016: Proceedings of the 14th Annual Research in Computational Molecular Biology (RECOMB) Comparative Genomics Satellite Workshop: bioinformatics. The full contents of the supplement are available online at https://bmcbioinformatics.biomedcentral.com/articles/supplements/volume-17-supplement-14.

Funding
Research of MJ was funded by Labex NUMEV (ANR-10-LABX-20).

Availability of data and materials
Our polynomial algorithm for checking consistency has been implemented in Python and is available at [24]. The gene families and Ensembl species tree used in our experiments are available at ensembl.org. The datasets analysed during this study are available from the corresponding author upon request. An implementation of the PhySIC_IST method is available at http://www.atgc-montpellier.fr/physic_ist/.

Authors' contributions
Designed the algorithms: CP MJ CS. Implemented the algorithms: MJ. Designed, performed and analyzed the experiments: MJ CS. Wrote the paper: CP MJ CS. All authors have read and approved the final version of the manuscript.

Competing interests
The authors declare that they have no competing interests.

Consent for publication
Not applicable.

Ethics approval and consent to participate
Not applicable.

Publication funding
Publication charges for this article have been funded by the French Agence Nationale de la Recherche Investissements d'Avenir/ Bioinformatique (ANR-10-BINF-01-02, Ancestrome).

Author details
[1]LIRMM, CNRS, Université de Montpellier, Montpellier, France. [2]ISE-M, CNRS, IRD, EPHE, Université Montpellier, Montpellier, France.

Published: 11 November 2016

References
1. Fitch WM. Distinguishing homologous from analogous proteins. Syst Biol. 1970;19(2):99–113.
2. Altenhoff AM, Dessimoz C. Inferring orthology and paralogy. Evol Genomics Stat Comput Methods. 2012;1:259–79.
3. Schreiber F, Patricio M, Muffato M, Pignatelli M, Bateman A. Treefam v9: a new website, more species and orthology-on-the-fly. Nucleic Acids Res. 2014;42(Database issue):D922–5. doi:10.1093/nar/gkt1055. Epub 2013 Nov 4.
4. Doyon JP, Ranwez V, Daubin V, Berry V. Models, algorithms and programs for phylogeny reconciliation. Brief Bioinform. 2011;12(5):392–400.

5. Li L, Stoeckert CJ, Roos DS. Orthomcl: identification of ortholog groups for eukaryotic genomes. Genome Res. 2003;13(9):2178–89.
6. Bergsten J. A review of long-branch attraction. Cladistics. 2005;21(2): 163–93.
7. Hernandez-Rosales M, Hellmuth M, Wieseke N, Huber KT, Moulton V, Stadler PF. From event-labeled gene trees to species trees. BMC Bioinforma. 2012;13(Suppl 19):6.
8. Hellmuth M, Hernandez-Rosales M, Huber KT, Moulton V, Stadler PF, Wieseke N. Orthology relations, symbolic ultrametrics, and cographs. J Math Biol. 2013;66(1-2):399–420.
9. Lafond M, El-Mabrouk N. Orthology and paralogy constraints: satisfiability and consistency. BMC Genomics. 2014;15(Suppl 6):12.
10. Hellmuth M, Wieseke N, Lechner M, Lenhof HP, Middendorf M, Stadler PF. Phylogenomics with paralogs. Proc Natl Acad Sci U S A. 2015;112(7): 2058–63.
11. Ma B, Li M, Zhang L. From gene trees to species trees. SIAM J Comput. 2000;30(3):729–52.
12. Koonin EV. Orthologs, paralogs, and evolutionary genomics 1. Annu Rev Genet. 2005;39:309–38.
13. Corneil D, Lerchs H, Stewart-Burlingham LK. Complement reducible graphs. Discret Appl Math. 1981;3(1):163–74.
14. Bőcker S, Dress AWM. Recovering symbolically dated, rooted trees from symbolic ultrametrics. Adv Math. 1998;138(1):105–25. doi:10.1006/aima.1998.1743.
15. Bretscher A, Corneil D, Habib M, Paul C. A simple linear time lexbfs cograph recognition algorithm. SIAM J Discret Math. 2008;22(4):1277–96.
16. Corneil D, Perl Y, Stewart LK. A linear time recognition algorithm for cographs. SIAM J Comput. 1985;14(4):926–34.
17. Golumbic M, Kaplan H, Shamir R. Graph sandwich problems. J Algorithm. 1995;19:449–73.
18. Aho AV, Sagiv Y, Szymanski TG, Ullman JD. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. SIAM J Comput. 1981;10(3):405–21.
19. Lafond M, Swenson KM, El-Mabrouk N. An optimal reconciliation algorithm for gene trees with polytomies. In: International Workshop on Algorithms in Bioinformatics. Springer Berlin Heidelberg; 2012. p. 106–22.
20. Boussau B, Szöllősi GJ, Duret L, Gouy M, Tannier E, Daubin V. Genome-scale coestimation of species and gene trees. Genome Res. 2013;23(2):323–30.
21. Downey RG, Fellows MR. Parameterized Complexity. New York, NY, USA: Springer; 1999, p. 530.
22. Garey MR, Johnson DS. Computers and Intractability: A Guide to the Theory of NP-Completeness. New York, NY, USA: W. H. Freeman & Co.; 1979.
23. Zuckerman D. Linear degree extractors and the inapproximability of max clique and chromatic number. Theory Comput. 2007;3(6):103–28. doi:10.4086/toc.2007.v003a006.
24. Lafond M, Jones M. GitHub - UdeM-LBIT/OrthoPara-ConstraintChecker: A program to check if a given set of orthology/paralogy relations, with possible unknowns, is satisfiable or consistent with a species tree. https://github.com/UdeM-LBIT/OrthoPara-ConstraintChecker. Accessed 22 July 2016.
25. Flicek P, Ahmed I, Amode MR, Barrell D, Beal K, Brent S, Carvalho-Silva D, Clapham P, Coates G, Fairley S, et al. Ensembl 2013. Nucleic Acids Res. 2012;41(Database issue):D48–D55. Published online 2012 Nov 30. doi:10.1093/nar/gks1236.
26. Lechner M, Findeiß S, Steiner L, Marz M, Stadler PF, Prohaska SJ. Proteinortho: Detection of (co-) orthologs in large-scale analysis. BMC Bioinforma. 2011;12(1):1.
27. Ensembl. Ensembl Species Tree. http://www.ensembl.org/info/about/speciestree.html. Accessed 22 July 2016.
28. Scornavacca C, Berry V, Lefort V, Douzery EJ, Ranwez V. Physic_ist: cleaning source trees to infer more informative supertrees. BMC Bioinforma. 2008;9(1):413.