A practical approximation algorithm for solving massive instances of hybridization number for binary and nonbinary trees

Leo Van Iersel, Steven Kelk, Nela Lekić, Celine Scornavacca

BMC Bioinformatics

RESEARCH ARTICLE Open Access

A practical approximation algorithm for solving massive instances of hybridization number for binary and nonbinary trees

Leo van Iersel[1*], Steven Kelk[2], Nela Lekić[2] and Celine Scornavacca[3]

Abstract

Background: Reticulate events play an important role in determining evolutionary relationships. The problem of computing the minimum number of such events to explain discordance between two phylogenetic trees is a hard computational problem. Even for binary trees, exact solvers struggle to solve instances with reticulation number larger than 40-50.

Results: Here we present CycleKiller and NonbinaryCycleKiller, the first methods to produce solutions verifiably close to optimality for instances with hundreds or even thousands of reticulations.

Conclusions: Using simulations, we demonstrate that these algorithms run quickly for large and difficult instances, producing solutions that are very close to optimality. As a spin-off from our simulations we also present TerminusEst, which is the fastest exact method currently available that can handle nonbinary trees: this is used to measure the accuracy of the NonbinaryCycleKiller algorithm. All three methods are based on extensions of previous theoretical work (SIDMA 26(4):1635-1656, TCBB 10(1):18-25, SIDMA 28(1):49-66) and are publicly available. We also apply our methods to real data.

Keywords: Hybridization number, Phylogenetic networks, Approximation algorithms, Directed feedback vertex set

Background

Phylogenetic trees are used in biology to represent the evolutionary history of a set $\mathcal{X}$ of species (or *taxa*) [1,2]. They are trees whose leaves are bijectively labeled by $\mathcal{X}$ and whose internal vertices represent the ancestors of the species set; they can be rooted or unrooted. Since in a rooted tree edges have a direction, the concepts of indegree and outdegree of a vertex are well defined. *Binary* rooted (phylogenetic) trees are rooted (phylogenetic) trees whose internal vertices have outdegree 2. *Nonbinary* rooted (phylogenetic) trees have no restriction on the outdegree of inner vertices.

Biological events in which a species derives its genes from different ancestors, such as hybridization, recombination and horizontal gene transfer events, cannot be modelled by a tree. To be able to represent such events, a generalization of trees is considered which allows vertices with indegree two or higher, known as *reticulations*. This model, which is called a *rooted phylogenetic network*, is of growing importance to biologists [3]. For detailed background information we refer the reader to [4-6].

Although phylogenetic networks are more general than phylogenetic trees, trees are still often the basic building blocks from which phylogenetic networks are constructed. Specifically, there are many techniques available for constructing gene trees. However, when more genes are analyzed, topological conflicts between individual gene phylogenies can arise for methodological or biological reasons (e.g. aforementioned reticulate phenomena such as hybridization). This has led computational biologists to try and quantify the amount of reticulation that is needed to simultaneously explain two trees.

To state this problem more formally, we have that a phylogenetic tree T on $\mathcal{X}$ is a refinement of a phylogenetic tree T' on the same set $\mathcal{X}$ if T can be obtained from T' by deleting edges and identifying their incident vertices.

*Correspondence: l.j.j.v.iersel@gmail.com
[1]Centrum Wiskunde & Informatica (CWI), P.O. Box 94079, 1090 GB, Amsterdam, The Netherlands
Full list of author information is available at the end of the article

Then, we say that a phylogenetic network N on $\mathcal{X}$ *displays* a phylogenetic tree T on $\mathcal{X}$ if T can be obtained from a subgraph of N by contracting edges. Informally, this means that (a refinement of) T can be obtained from N by, for each reticulation vertex of N, "switching off" all but one of its incoming edges and then suppressing all indegree-1 outdegree-1 vertices (i.e. replacing paths of these vertices by one edge). Given two rooted phylogenetic trees T_1 and T_2 on $\mathcal{X}$, the problem then becomes to determine the minimum number of reticulation events contained in a phylogenetic network N on $\mathcal{X}$ displaying both trees (where an indegree-d reticulation counts as $d-1$ reticulation events). The value we are minimizing is often called the *hybridization number* and instead of the term phylogenetic network, the term *hybridization network* is often used. It is known that the problem of computing hybridization numbers is both NP-hard and APX-hard [7], but it is not known whether it is in APX (i.e. whether it admits a polynomial-time approximation algorithm that achieves a constant approximation ratio).

Until recently, most research on the hybridization number of two phylogenetic trees had focused on the question of how to exactly compute this value using fixed parameter tractable (FPT) algorithms, where the parameter in question is the hybridization number r of the two trees. For an introduction to FPT we refer to [8,9].

For binary trees, algorithmic progress has been considerable in this area, with various authors reporting increasingly sophisticated FPT algorithms [10-13]. The fastest algorithms currently implemented are the algorithm available inside the package DENDROSCOPE [14], based on [15], and the sequence of progressively faster algorithms in the HYBRIDNET family [11,16,17]. The fastest theoretical FPT algorithm has running time $O(3.18^r n)$ [13], where n is the number of taxa in the trees.

Even though in practice it rarely happens that trees are binary, the nonbinary variant of the problem has been less studied. The nonbinary version is also FPT [18,19] and a (non-FPT) algorithm has recently been implemented in DENDROSCOPE [14].

Such (FPT) algorithms do, however, have their limits. The running time still grows exponentially in r, albeit usually at a slower rate than algorithms that have a running time of the form $n^{f(r)}$, where f is some function of r. In practice this means that existing algorithms can only handle instances of binary trees when r is at most 40-50 and instances of nonbinary trees when r is at most 5-10.

These limitations are problematic. Due to ongoing advances in DNA sequencing, more and more species and strains are being sequenced. Consequently, biologists use trees with more and more taxa and software that can handle large trees is required. For such large and/or difficult trees one can try to generate heuristic or approximate solutions, but how far are such solutions from optimality? In [20] we showed that the news is worrying. Indeed, we showed that polynomial-time constant-ratio approximation algorithms exist if and only if such algorithms exist for the problem Directed Feedback Vertex Set (DFVS). However, DFVS is a well-studied problem in combinatorial optimization and to this day it is unknown if it permits such an algorithm. Pending a major breakthrough in computer science, it therefore seems difficult to build polynomial-time algorithms which approximate hybridization number well. On the positive side, we showed that in polynomial time an algorithm with approximation ratio $O(\log r \log\log r)$ is possible. However, this algorithm is purely of theoretical interest and is not useful in practice.

New algorithms: CYCLEKILLER and NONBINARYCYCLEKILLER

In this article we extend the theoretical work of [20] slightly and give it a practical twist to yield a fast approximation algorithm which we have made publicly available as the program CYCLEKILLER. Furthermore, we give an implementation of the algorithm presented in [21], available as NONBINARYCYCLEKILLER.

The worst-case running time of these approximation algorithms is exponential. However, as we demonstrate with experiments, the running time of our algorithms is in practice extremely fast. For large and/or massively discordant binary trees, CYCLEKILLER is typically orders of magnitude faster than the HYBRIDNET algorithms and the algorithm in DENDROSCOPE. The performance gap between NONBINARYCYCLEKILLER and its exact counterparts is less pronounced, but still significant, especially in its fastest mode of operation.

Of course, exact algorithms attempt to compute optimum solutions, whereas our algorithms only give approximate solutions. Nevertheless, our experiments show that when CYCLEKILLER and NONBINARYCYCLEKILLER are run in their most accurate mode of operation, an approximation ratio very close to 1 is not unusual, suggesting that the algorithms often produce solutions close to optimality and well within the worst-case approximation guarantee.

The idea behind the binary and nonbinary algorithm is similar. Specifically, we describe an algorithm with approximation ratio $d(c+1)$ for the hybridization number problem on two binary trees and an algorithm with approximation ratio $d(c+3)$ for the hybridization number problem on two nonbinary trees by combining a c-approximation for the problem MAF (Maximum Agreement Forest) with a d-approximation for the problem DFVS. Both these problems are NP-hard so polynomial-time algorithms attaining $c=1$ or $d=1$ are not realistic. Nevertheless, there exist extremely fast FPT algorithms for solving MAF on binary trees exactly (i.e. $c=1$), the fastest is RSPR by Whidden, Beiko and Zeh

[22,23] although the MAF algorithm inside [17] is also competitive. Moreover, we observe that the type of DFVS instances that arise in practice can easily be solved using Integer Linear Programming (ILP) (and freely-available ILP solver technology such as GLPK), so $d = 1$ is also often possible.

Combining these two exact approaches gives us, in the binary case, an exponential-time approximation algorithm with worst-case approximation ratio 2 that for large instances still runs extremely quickly; this is the `2-approx` option of CycleKiller. In practice, we have observed that the upper bound of 2 is often pessimistic, with much better approximation ratios observed in experiments (1.003 on average for the simulations presented in this article). We find that this algorithm already allows us to cope with much bigger trees than the HybridNet algorithms or the algorithm in Dendroscope.

Nevertheless, for truly massive trees it is often not feasible to have $c = 1$. Fortunately there exist linear-time algorithms which achieve $c = 3$ [13]. This, coupled with the fact that (even for such trees) it remains feasible to use an exact ($d = 1$) solver for DFVS, means that in practice we achieve a 4-approximation for gigantic binary trees; this is the `4-approx` option of CycleKiller. Again, the ratio of 4 is a worst-case bound and we suspect that in practice we are doing much better than 4. However, this cannot be experimentally verified due to the lack of good lower bounds for such massive instances. In any case, the main advantage of this option is that it can, without too much effort, cope with trees with hundreds or thousands of taxa and hybridization number of a similar order of magnitude. An implementation of CycleKiller and accompanying documentation can be downloaded from http://skelk.sdf-eu.org/cyclekiller. Networks created by the algorithm can be viewed in Dendroscope.

For the nonbinary case, there also exist exact and approximation algorithms for MAF [13,21,24]. In case when one of the input trees is binary we can still use the exact (thus $c = 1$) and approximate ($c = 3$) algorithms given in [13] (referred to as rSPR) to obtain respectively a 4-approximation and a 6-approximation of the hybridization number problem for nonbinary trees. When both input trees are nonbinary, then we must use the somewhat less optimized exact ($c = 1$) and approximate ($c = 4$) algorithms described in [21]. We then obtain 4- and 7-approximations (because in the nonbinary case $d = 1$ is still easily attainable using ILP).

To measure the approximation ratios attained by NonbinaryCycleKiller in practice we have also implemented and made publicly available the exact nonbinary algorithm TerminusEst, based on the theoretical results in [18]. TerminusEst will be of independent interest because it is currently the fastest exact nonbinary solver available.

CycleKiller, NonbinaryCycleKiller and TerminusEst can be downloaded respectively from http://skelk.sdf-eu.org/cyclekiller [25], from http://homepages.cwi.nl/~iersel/cyclekiller [26], and from http://skelk.sdf-eu.org/terminusest [27].

Theoretical and practical significance

We have described, implemented and made publicly available two algorithms with two desirable qualities: they terminate quickly even for massive instances of hybridization number and give a non-trivial guarantee of proximity to optimality. These are the first algorithms with such properties. Both algorithms are based on a non-trivial marriage of MAF and DFVS solvers (both exact and approximate), meaning that further advances in solving MAF and DFVS will directly lead to improvements in CycleKiller and NonbinaryCycleKiller.

This article also improves the theoretical work given in [20], which also proposed using DFVS but beginning from a trivial Agreement Forest (AF) known as a *chain forest*. Here we use a smarter starting point: an (approximate) MAF, and it is this insight which makes a 2-approximation (rather than the 6-approximation implied by [20]) possible when using an exact DFVS solver. Other articles have also had the idea of cycle-breaking in AFs: the advanced FPT algorithm of Whidden et al. [13] – which has not been implemented – and the algorithms in the aforementioned HybridNet family. However, both algorithms start the cycle-breaking from many starting points. In contrast, our algorithm requires only a *single* starting point, i.e. a single (approximate) solution to MAF.

Here, we only present the theory behind the binary algorithm. The nonbinary case is more involved and we refer the reader to [21] in which we introduce it. Note that our results for the binary case do not follow from the results for the nonbinary case in [21] because here we obtain a better constant in the approximation ratio. After a presentation of the binary algorithm in Section "The algorithm for binary trees", we will show the results of some experiments with binary trees in Section "Practical experiments with binary trees" and nonbinary trees in Section "Practical experiments with nonbinary trees". Finally, in Section "Practical experiments on biologically relevant trees" we demonstrate that both TerminusEst and NonbinaryCycleKiller are easily capable of generating optimal (respectively, nearly optimal) solutions on a real biological dataset originally obtained from the GreenPhylDB database.

Technical note

At the time the experiments on binary trees were conducted (i.e. for the preliminary version of this article [28]) HybridNet was the fastest algorithm available in its family. It has recently been superceded by the faster

UltraNet [17]. We believe, however, that it is neither necessary nor desirable to re-run the binary experiments, for the following reasons. In the same period the solver rSPR has also increased dramatically in speed (it is now at v1.2), leading to a corresponding speed-up in CycleKiller. In fact, both rSPR and the algorithms in the HybridNet family are constantly in flux and are always being improved, so any experimental setup is prone to age extremely quickly. However, the conclusions that we can derive from these experiments are unlikely to change much over time. Given that the algorithms in the HybridNet family (and the theoretical algorithm in [13]) implicitly have to explore exponentially many optimal and sub-optimal solutions to the MAF problem, the running time of MAF solvers (and thus also CycleKiller) is likely for the foreseeable future to remain much better than the running time of solvers for hybridization number. The central message is stable: approximating hybridization number by splitting it into MAF and DFVS instances yields extremely competitive approximation ratios for instances that exact hybridization number solvers will probably never be able to cope with.

Methods

Preliminaries

Let $\mathcal{X}$ be a finite set (e.g. of species). A *rooted phylogenetic $\mathcal{X}$-tree* is a rooted tree with no vertices with indegree 1 and outdegree 1, a root with indegree 0 and outdegree at least 2, and leaves bijectively labelled by the elements of $\mathcal{X}$. We identify each leaf with its label and use $L(T)$ to refer to the leaf set (or label set) of T. A rooted phylogenetic $\mathcal{X}$-tree is called *binary* if each nonleaf vertex has outdegree two. We henceforth call a rooted, binary phylogenetic $\mathcal{X}$-tree a *tree* for short. For a tree T and a set $\mathcal{X}' \subset \mathcal{X}$, we use the notation $T(\mathcal{X}')$ to denote the minimal subtree of T that contains all elements of $\mathcal{X}'$ and $T|\mathcal{X}'$ denotes the result of suppressing all indegree-1 outdegree-1 vertices in $T(\mathcal{X}')$.

The following definitions apply only to binary trees. Definitions for nonbinary trees are analogous but slightly more technical [21].

We define a *forest* as a set of trees. Each element of a forest is called a *component*. Let T be a tree and $\mathcal{F}$ a forest. We say that $\mathcal{F}$ is a *forest for* T if, for all $F \in \mathcal{F}$, $T|L(F)$ is isomorphic to F and the trees $\{T(L(F)), F \in \mathcal{F}\}$ are vertex-disjoint subtrees of T whose leaf-set union equals $L(T)$. If T_1 and T_2 are two trees, then a forest $\mathcal{F}$ is an *agreement forest* of T_1 and T_2 if it is a forest for T_1 and T_2. The number of components of $\mathcal{F}$ is denoted $|\mathcal{F}|$.

We define *cleaning up* a directed graph as repeatedly suppressing indegree-1 outdegree-1 vertices, removing indegree-0 outdegree-1 vertices and removing unlabelled outdegree-0 vertices until no such operation is possible. Observe that, if $\mathcal{F}$ is a forest for T, $\mathcal{F}$ can be obtained from T by removing $|\mathcal{F}| - 1$ edges and cleaning up. From now on we consider T_1, T_2 as trees on the same taxon set.

Problem: Maximum Agreement Forest (MAF)
Instance: Two rooted, binary phylogenetic trees T_1 and T_2.
Solution: An agreement forest $\mathcal{F}$ of T_1 and T_2.
Objective: Minimize $|\mathcal{F}| - 1$.

The directed graph $IG(T_1, T_2, \mathcal{F})$, called the *inheritance graph*, is the directed graph whose vertices are the components of $\mathcal{F}$ and which has an edge (F, F') precisely if either

- there is a directed path in T_1 from the root of $T_1(L(F))$ to the root of $T_1(L(F'))$ or;
- there is a directed path in T_2 from the root of $T_2(L(F))$ to the root of $T_2(L(F'))$.

An agreement forest $\mathcal{F}$ of T_1 and T_2 is called an *acyclic agreement forest* if the graph $IG(T_1, T_2, \mathcal{F})$ is acyclic. A *maximum acyclic agreement forest (MAAF)* of T_1 and T_2 is an acyclic agreement forest of T_1 and T_2 with a minimum number of components.

Problem: Maximum Acyclic Agreement Forest (MAAF)
Instance: Two rooted, binary phylogenetic trees T_1 and T_2.
Solution: An acyclic agreement forest $\mathcal{F}$ of T_1 and T_2.
Objective: Minimize $|\mathcal{F}| - 1$.

We use $\mathrm{MAF}(T_1, T_2)$ and $\mathrm{MAAF}(T_1, T_2)$ to denote the optimal solution value of the problem MAF and MAAF respectively, for an instance T_1, T_2.

A *rooted phylogenetic network* on $\mathcal{X}$ is a directed acyclic graph with no vertices with indegree 1 and outdegree 1 and leaves bijectively labelled by the elements of $\mathcal{X}$. Rooted phylogenetic networks, which are sometimes also called hybridization networks, will henceforth be called *networks* for short in this paper. A tree T on $\mathcal{X}$ is *displayed* by a network N if T can be obtained from a subtree of N by contracting edges. A *reticulation* is a vertex v with $\delta^-(v) \geq 2$ (with $\delta^-(v)$ denoting the indegree of v). The *reticulation number* (sometimes also called hybridization number) of a network N with root ρ is given by

$$r(N) = \sum_{v \neq \rho} (\delta^-(v) - 1).$$

It was shown that the optimum to MAAF is equal to the optimum of the following problem [29].

Problem: MinimumHybridization
Instance: Two rooted binary phylogenetic trees T_1 and T_2.

Solution: A rooted phylogenetic network N that displays T_1 and T_2.
Objective: Minimize $r(N)$.

Moreover, it was shown that, for two trees T_1, T_2, *any* acyclic agreement forest for T_1 and T_2 with $k+1$ components can be turned into a phylogenetic network that displays T_1 and T_2 and has reticulation number k, and vice versa. Thus, any approximation for MAAF gives an approximation for MINIMUMHYBRIDIZATION.

Finally, a *feedback vertex set* of a directed graph is a subset of the vertices that contains at least one vertex of each directed cycle. Equivalently, a subset of the vertices of a directed graph is a *feedback vertex set* if removing these vertices from the graph makes it acyclic.

Problem: Directed Feedback Vertex Set (DFVS)
Instance: A directed graph D.
Goal: Find a feedback vertex set of D of minimum size.

We note that the definition of MINIMUMHYBRIDIZATION easily generalises to nonbinary trees, since the definition of *display* allows the image of each input tree in the network to be more "resolved" than the original tree. However, the definitions of (acyclic) agreement forests are different in the nonbinary case [21].

The algorithm for binary trees

We show how MAAF can be approximated by combining algorithms for MAF and DFVS. In particular, we will prove the following theorem.

Theorem 1. *If there exists a c-approximation for* MAF *and a d-approx-imation for* DFVS*, then there exists a* $d(c+1)$*-approximation for* MAAF *(and thus for* MINIMUMHYBRIDIZATION*).*

Note that this theorem does not follow from Theorem 2.1 of [21], since there the approximation ratio for MAAF is a $d(c+3)$-approximation.

To prove the theorem, suppose there exists a c-approximation for MAF. Let T_1 and T_2 be two trees and let M be an agreement forest returned by the algorithm. Then,

$$|M| - 1 \leq c \cdot \mathrm{MAF}(T_1, T_2) \leq c \cdot \mathrm{MAAF}(T_1, T_2). \quad (1)$$

An *M-splitting* is an acyclic agreement forest that can be obtained from M by removing edges and cleaning up.

Lemma 2. *Let T_1 and T_2 be two trees and M an agreement forest of T_1 and T_2. Then, there exists an M-splitting of size at most $MAAF(T_1, T_2) + |M|$.*

Proof. Consider a maximum acyclic agreement forest F of T_1 and T_2. For $i \in \{1, 2\}$, F can be obtained from T_i by removing a set of edges, say E_F^i, and cleaning up. Moreover, also M can be obtained from T_i by removing a set of edges, say E_M^i, and cleaning up.

Now consider the forest S obtained from T_1 by removing $E_M^1 \cup E_F^1$ and cleaning up. Then,

- S is an agreement forest of T_1 and T_2 because it can be obtained from T_2 by removing edges $E_M^2 \cup E_F^2$ and cleaning up;
- S is acyclic because it can be obtained by removing edges from F, which is acyclic, and cleaning up;
- S can be obtained from M by removing edges and cleaning up.

Hence, S is an M-splitting. Furthermore, $|S| \leq |E_F^1| + |E_M^1| + 1$. The lemma follows since $|E_F^1| = \mathrm{MAAF}(T_1, T_2)$ and $|M| = |E_M^1| + 1$. □

Let $\mathrm{OptSplitting}_{T_1,T_2}(M)$ denote the size of a minimum-size M-splitting. Combining Lemma 2 and Eq. 1, we obtain

$$\mathrm{OptSplitting}_{T_1,T_2}(M) - 1 \leq (c+1)\mathrm{MAAF}(T_1, T_2) \quad (2)$$

We will now show how to find an approximation for the problem of finding an optimal M-splitting. We do so by reducing the problem to DFVS. We construct an input graph D for DFVS (called the *extended inheritance* graph) as follows. For every vertex of M that has outdegree 2 (in M), we create a vertex in D. There is an edge in D from a vertex u to a vertex v precisely if in either T_1 or T_2 (or in both) there is a directed path from u to v. An example is in Figure 1. We claim the following.

Lemma 3. *A subset V' of the vertices of D is a feedback vertex set of D if and only if removing V' from M makes it an acyclic agreement forest.*

Proof. We show that $D \setminus V'$ has a directed cycle if and only if the inheritance graph of $M \setminus V'$ has a directed cycle.

To prove this, first suppose that there is a cycle $v_1, v_2, \ldots, v_k = v_1$ in the inheritance graph of $M \setminus V'$. The vertices in the inheritance graph of $M \setminus V'$ correspond to the roots of the components of $M \setminus V'$. Since these roots have outdegree 2 in $M \setminus V'$, they had outdegree 2 in M, and are thus vertices of D. So the vertices $v_1, v_2, \ldots, v_k$ that form the cycle are vertices of D. Since these vertices are in the inheritance graph of $M \setminus V'$, they can not be in V' and so they are vertices of $D \setminus V'$. The reachability relation between these vertices in $D \setminus V'$ is the same as in the inheritance graph of $M \setminus V'$. So, the vertices $v_1, v_2, \ldots, v_k$ form a cycle in $D \setminus V'$.

Now suppose that there is a cycle $w_1, w_2, \ldots, w_k = w_1$ in $D \setminus V'$. Each of the vertices $w_1, w_2, \ldots, w_k$ is a vertex with outdegree-2 in M. Some of them might be roots of components, while others are not. However, observe that

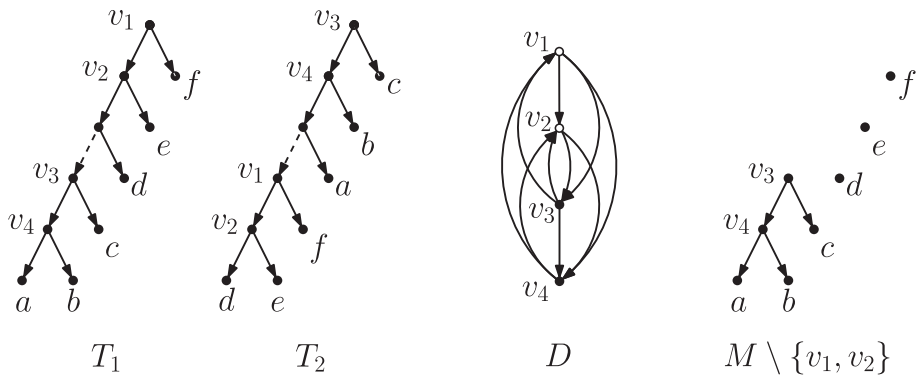

Figure 1 Two binary trees T_1 and T_2 and the auxiliary graph D. A maximum agreement forest M of T_1 and T_2 is obtained by deleting the dashed edges. Graph D can be made acyclic by deleting either both filled or both unfilled vertices. Hence, removing either v_1 and v_2 or v_3 and v_4 from M makes it an acyclic agreement forest for T_1 and T_2, see Lemma 3. The acyclic agreement forest $M \setminus \{v_1, v_2\}$ obtained by removing v_1 and v_2 from M is depicted on the right.

if there is a directed path from a vertex u to a vertex v in T_1 (or in T_2) then there is also a directed path from the root of the component of $M \setminus V'$ that contains u to the root of the component of $M \setminus V'$ that contains v. Hence, there is a directed cycle in the inheritance graph of $M \setminus V'$, formed by the roots of the components of $M \setminus V'$ that contain $w_1, w_2, \ldots, w_k$. □

Proof of Theorem 1. Suppose that there exists a d-approximation for DFVS. Let FVS be a feedback vertex set returned by this algorithm and let MFVS be a minimum feedback vertex set. Then, removing the vertices of MFVS from M gives an optimal M-splitting. Furthermore, $\text{OptSplitting}_{T_1,T_2}(M) = |M| + |\text{MFVS}|$. This is because for every vertex in a cycle C, its parent in M must participate in some cycle that contains elements of C. So if we start by removing the root of the component we are splitting and subsequently remove only those vertices whose parents have already been removed we see that we add at most one component per vertex. In fact, because vertices of D all have out-degree 2 in M, we add exactly one component per vertex.

By removing the vertices of FVS from M, we obtain an acyclic agreement forest $\mathcal{F}$ such that

$$\begin{aligned} |\mathcal{F}| - 1 &= |M| + |\text{FVS}| - 1 \\ &\leq |M| + d \cdot |\text{MFVS}| - 1 \\ &\leq d(|M| + |\text{MFVS}| - 1) \\ &= d(\text{OptSplitting}_{T_1,T_2}(M) - 1) \\ &\leq d(c+1)\text{MAAF}(T_1, T_2), \end{aligned}$$

where the last inequality follows from Eq. 2. Thus, $\mathcal{F}$ is a $d(c+1)$-approximation to MAAF, which concludes the proof of Theorem 1. □

Theorem 1 implies that a solution to the MAAF problem for a given instance can be constructed by (i) finding a solution $\mathcal{F}$ to the MAF problem for the same instance (ii) constructing the extended inheritance graph D for $\mathcal{F}$ (iii) finding a solution V for the DFVS problem on the graph D and (iv) modifying $\mathcal{F}$ accordantly to V.

Results and discussion

Practical experiments with binary trees

To assess the performance of CycleKiller, a simulation study was undertaken. We generated 3 synthetic datasets, an *easy*, a *medium* and a *hard* one, containing respectively 800, 640 and 640 pairs of rooted binary phylogenetic trees.

The easy data set was created by varying two parameters, namely the number of taxa n and the number of rSPR-moves k used to obtain the second tree from the first (note that this number is an upper bound on the actual rSPR distance). The 800 pairs of rooted binary phylogenetic trees were created by varying n in $\{20, 50, 100, 200\}$ and k in $\{5, 10, ..., 25\}$, and then creating 40 different instances per each combination of parameters. Each pair (T_1, T_2) of rooted binary phylogenetic trees for a given set of parameters n and k is created as follows: The first tree T_1 on $\mathcal{X} = \{x_1, \ldots, x_n\}$ is generated by first creating a set of n leaf vertices bijectively labeled by the set $\mathcal{X}$. Then, two vertices u and v, both with indegree 0, are randomly picked and a new vertex w, along with two new edges (w, u) and (w, v), is created. This is done until only one vertex with no ancestor, the root, is present. The second tree T_2 is obtained from T_1 by applying k rSPR-moves. The medium and the hard data sets were generated in the same way as the easy one, but for different choices of the parameters: n in $\{50, 100, 200, 300\}$ and k in $\{15, 25, 40, 55\}$ for the medium one and n in $\{100, 200, 400, 500\}$ and k in $\{40, 60, 80, 100\}$ for the hard one.

The exact hybridization number has been computed by HybridNet [11], available from http://www.cs.cityu.edu.hk/~lwang/software/Hn/treeComp.html or with Dendroscope [14], available from http://dendroscope.org/. We will refer to these algorithms as the *exact algorithms*. Each instance has been run on a single core of an Intel Xeon E5506 processor.

Each run that took more than one hour was aborted. For each instance, we ran our program with the option `2-approx`, and, in case the latter did not finish within one hour, we ran it again, this time using the option `4-approx`, always with a one-hour limit (see Section "New algorithms: CycleKiller and NonbinaryCycleKiller"). We used the program rSPR v1.03 [22,23] to solve or approximate MAF and GLPK v4.47 (http://www.gnu.org/software/glpk/) to solve the following simple polynomial-size ILP formulation of DFVS:

$$\begin{aligned} &\min \sum_{v \in V} x_v \\ &\text{s.t.} \\ &\quad 0 \leq \ell_v \leq |V| - 1 && \text{for all } v \in V \\ &\quad \ell_v \geq \ell_u + 1 - |V| x_u - |V| x_v && \text{for all } e = (u, v) \in E \\ &\quad \ell_v \in \mathbb{Z} && \text{for all } v \in V \\ &\quad x_v \in \{0, 1\} && \text{for all } v \in V \end{aligned}$$

Given a directed graph $D = (V, E)$, the binary variables x_v model whether a vertex is in the feedback vertex set, and the integer variables ℓ_v model the positions of the surviving vertices in the induced topological order. The edge constraints enforce the topological order. Note that an edge constraint is essentially eliminated if one or both endpoints of the edge are in the feedback vertex set.

For all instances of the easy data set, CycleKiller finished with the `2-approx` option within the one hour limit, while for 33 instances the exact algorithms were unable to compute the hybridization number. Note that, even for "easy" instances, computing the exact hybridization number can take a very long time. To give the reader an idea, for 9 runs of the easy data, Dendroscope and HybridNet did not complete within 10 days. Table 1 shows a summary of the results. It can be seen that CycleKiller was much faster than the exact algorithms. Moreover, for 96.6% of the instances for which an exact algorithm could find a solution, CycleKiller also found an optimal solution. While the theoretical worst-case approximation ratio of the `2-approx` option of CycleKiller is 2, in our experiments it performed very close to a 1-approximation.

For the medium data set, CycleKiller finished with the `2-approx` option for 613 instances, and for the remaining ones with the `4-approx` option. The exact algorithms could compute the hybridization number for only 199 instances (out of 640). For 97.5% of these instances, CycleKiller also found an optimal solution, but with a much better running time. Regarding the hard data set, 444 runs were completed with the `2-approx` option and for the remaining ones we were able to use the `4-approx` option within the given time constraint. Unfortunately, the exact algorithms were unable to compute the hybridization number for any tree-pair of this data set and hence we could not compute the average approximation ratios. Over all our experiments, the maximum hybridization number that the exact algorithms could handle was 25[a]. In contrast, the `2-approx` option of CycleKiller could be used for instances for which the size of a MAF was up to 97, and thus for instances for which the hybridization number was at least 97.

To find the limits of the `4-approx` option of CycleKiller, we also tested it on randomly generated trees. On a normal laptop, it could construct networks with up to 10,000 leaves and up to 10,000 reticulations within 10 minutes. Since the number of reticulations found is at most four times the optimal hybridization number, this implies that the `4-approx` option of CycleKiller can handle hybridization numbers up to at least 2,500. These randomly generated trees are, however, biologically meaningless and, therefore, we conducted the extensive experiment described above on trees generated by rSPR moves. Finally we note that over all experiments the worst approximation ratio we encountered was 1.2.

Practical experiments with nonbinary trees

To run the simulations with NonbinaryCycleKiller, we used a subset of the trees from the easy set of binary experiments. We then applied random edge contractions in order to obtain nonbinary trees. Hence, we have the same two parameters as before, namely the number of taxa $n \in \{20, 50, 100\}$ and the number of rSPR-moves $k \in \{5, 10, 15, 20\}$, and an additional parameter $\rho \in \{25, 50, 75\}$ which measures the percentage of the edges of an original binary tree that were contracted in order to obtain a nonbinary tree. We could only use smaller values of n and k from the easy set of experiments because exact solvers for nonbinary MAF (upon which NonbinaryCycleKiller is built) and exact solvers for nonbinary MinimumHybridization (which is important to measure the accuracy of NonbinaryCycleKiller in practice) are slower than their binary counterparts.

We performed two runs of experiments[b]. One run with instances consisting of one binary and one nonbinary tree, and one run with instances consisting of two nonbinary trees.

For the experiments with one binary and one nonbinary tree, we were still able to use the rSPR algorithm [13,22], which has a better running time and approximation ratio compared to the available algo-

Table 1 Experimental results for instances with two binary trees

	Exact algorithms			CycleKiller					
Dataset	Total runs	Completed	Time	2-approx		4-approx		Ratio	Opt. found
				Compl.	Time	Compl.	Time		
Easy	800	767	798	800	3	-	-	1.003	96.6%
Medium	640	199	2572	613	212	27	<1	1.002	97.5%
Hard	640	0	3600	440	1271	200	1.5	-	-

The third column indicates for how many instances at least one exact algorithm finished within one hour. The fifth column indicates for how many instances the `2-approx` option of CycleKiller finished within one hour. For the remaining instances, the `4-approx` option finished within one hour, as can be seen from the seventh column. The average running time for the `2-approx` and the `4-approx` in seconds are reported respectively in the sixth and eighth column. The average approximation ratio (ninth column) is taken over all instances for which at least one exact method finished. The last column indicates the percentage of those instances for which CycleKiller found an optimal solution.

rithm for two nonbinary trees. When rSPR is used in exact mode, NonbinaryCycleKiller yields a theoretical worst-case approximation ratio of 4. When rSPR is used in its 3-approximation mode, NonbinaryCycleKiller yields a theoretical worst-case approximation ratio of 6 (see Section "New algorithms: CycleKiller and NonbinaryCycleKiller"). The results of this run are summarized in Table 2.

For the experiments with two nonbinary trees, the rSPR software can no longer be used, and instead we used the exact and 4-approximate MAF algorithm described in [21]. This makes NonbinaryCycleKiller behave as a 4-approximation and 7-approximation respectively (see Section "New algorithms: CycleKiller and NonbinaryCycleKiller"). Note that the exact algorithm [21] is considerably slower than rSPR, meaning that in practice NonbinaryCycleKiller struggles with two nonbinary trees more than when at most one of the trees is nonbinary. The results for this run are summarized in Table 3.

The exact hybridization number in both runs was computed by TerminusEst [27].

Each instance that took longer than 10 minutes to compute was aborted and the running time was set to 600 seconds. The averages of the running-times are taken over all instances, with running-time taken to be 600 if the program timed out for that instance. (We used a shorter time-out than in the binary experiments because of the observation that, in the nonbinary case, exact algorithms running longer than 10 minutes almost always took longer than 60 minutes too.)

Note that we did not compare the performance of NonbinaryCycleKiller to Dendroscope because TerminusEst has better running times than the exact nonbinary MinimumHybridization solver inside Dendroscope (data not shown).

To enable a clearer analysis we divided the trees into representative "simple" and "tricky" ones based on two parameters, n and k. Parameter values for the simple set were $n \in \{20, 50\}$, $k \in \{5, 10, 15\}$ and for the tricky set $n \in \{50, 100\}$, $k = 20$. In addition we varied the percentage of contracted edges (in a single tree in the first run and in both trees in the second run).

In Table 2 we show running times and solution quality of our algorithm when one of the input trees is binary. For the simple set of instances (regardless of the percentage of edge-contractions) we see that the more accurate version of our algorithm, the 4-approximation, had

Table 2 Summary of results for instances with one binary and one nonbinary tree

		TerminusEst		NonbinaryCycleKiller					
				4-approx			6-approx		
Contr.	Dataset	*opt*	Time	*r*(*N*)	Time	Ratio	*r*(*N*)	Time	Ratio
25%	Simple	7.504	8.004	7.567	0.967	1.007	11.421	0.996	1.532
	Tricky	17.000	203.650	17.288	3.675	1.003	27.238	3.638	1.600
50%	Simple	6.736	9.896	6.829	0.942	1.008	10.900	0.925	1.639
	Tricky	14.976	374.263	16.288	3.388	1.006	26.413	3.438	1.640
75%	Simple	5.139	12.304	5.263	0.867	1.011	8.692	0.963	1.659
	Tricky	10.500	391.575	13.475	3.263	1.006	23.200	3.275	1.633
Worst case	20	600	22	15	1.75	37	13	3	

We list the average hybridization number found (*opt* and *r*(*N*)), the average running time in seconds (Time) and where applicable the average approximation ratio (Ratio) for the three algorithms.

Table 3 Summary of results for instances with two nonbinary trees

		TerminusEst		NonbinaryCycleKiller					
				`4-approx`			`7-approx`		
Contr.	**Dataset**	***opt***	**Time**	***r*(*N*)**	**Time**	**Ratio**	***r*(*N*)**	**Time**	**Ratio**
25%	Simple	7.168	12.971	7.240	43.967	1.032	16.338	2.463	2.343
	Tricky	16.148	279.100	-	-	-	35.638	7.000	2.193
50%	Simple	5.933	11.150	5.900	41.325	1.030	13.721	2.004	2.405
	Tricky	13.216	379.238	-	-	-	32.363	7.200	2.331
75%	Simple	3.654	1.121	3.729	4.208	1.015	9.075	1.483	2.590
	Tricky	8.672	183.150	-	-	-	21.950	5.800	2.294
Worst case		20	600	29	600	1.5	56	22	4

The layout of the table is the same as that of Table 2.

a better running time than the exact algorithm, and at the same time had an average approximation ratio very close to 1. Far more interesting is to see what happens with tricky instances. As predicted, the running time of the exact algorithm is much higher for tricky instances due to the higher hybridization numbers. On the other hand, the running time of the 4-approximation does not rise significantly at all, whilst still attaining an approximation ratio again very close to 1. Another thing to note is that the percentage of contraction only seems to affect the running time of the exact algorithm. The practical worst-case approximation ratio observed in these experiments was 1.75 for the 4-approximation and 3 for the 6-approximation.

Table 3 shows our results on instances with two nonbinary trees. The exact algorithm for MAF is in this case much slower and this affects the running times even for the simple set. While the 4-approximation version has an average approximation ratio very close to 1 again, the running time is in this case worse than that of TerminusEst. For the tricky set the situation is even more significant; the exact MAF algorithm cannot deal with reticulation numbers above 15, while TerminusEst can get slightly further. On the other hand, the 7-approximation still runs much faster than TerminusEst, both for simple and tricky instances, while having an average approximation ratio of less than 2.6. The practical worst-case approximation ratio observed in these experiments was 1.5 for the 4-approximation and 4 for the 7-approximation.

It is worth noting that, for the 4-approximation, the running time for the 75%-contraction trees is considerably lower than the one for the 50%-contraction trees. This is due to the fact that a high contraction in both trees causes the hybridization number of the instance to drop, and a lower hybridization number leads to a better running time. Also note that the exact solver TerminusEst seems more able to cope with the tricky 25%-contraction instances than the tricky 50%-contraction instances. This is probably because, although low contraction rates yield a higher hybridization number, the trees remain "relatively binary" and this can induce more efficient branching in the underlying FPT algorithm [18]. It is plausible that with 50%-contraction the instances suffer from the disadvantage of relatively high hybridization number without the branching advantages associated with (relatively) binary trees.

To find the limits of the `7-approx` option of NBCK, we also tested it on huge, biologically meaningless, randomly generated trees. Below some results:

- 1000 leaves, 25%-contraction, on average 995 reticulations in 63 sec.
- 1000 leaves, 50%-contraction, on average 989 reticulations in 82 sec.
- 1000 leaves, 75%-contraction, on average 840 reticulations, in 656 sec.

Computation times of this last run of experiments do not include the network construction.

Practical experiments on biologically relevant trees

Finally, we tested our methods on phylogenetic trees obtained from GreenPhylDB [30] – version 3, a database containing twenty-two full genomes of members of the plantae kingdom, ranging from algae to angiosperms. We were able to retrieve from the database the 9903 rooted phylogenetic trees associated to the gene families contained in the database (the gene trees), along with the rooted phylogenetic tree describing the history of the twenty-two species contained in GreenPhylDB (the species tree). Note that the species tree for these species is not completely resolved, i.e. it is nonbinary. Among the gene trees, 2769 contain less than 3 species and they were discarded. Of the remaining 7134 trees, only 204 were directly usable for testing our methods. Indeed, because of

Table 4 Summary of results for dataset F_1 (204 gene trees) originally obtained from GreenPhylDB database

	MIN	AVG	MAX
Common taxa	3	5.235	20
opt	0	0.873	7
Ratio 4-approx	1	1.002	1.2
Ratio 6-approx	1	1.088	3
Gap (T-EST - MAF)	0	0.010	1
Gap (4-approx - MAF)	0	0.020	2
Time T-EST	0	0.221	3
Time 4-approx	0	0.270	1

Common taxa is the number of taxa after restricting the gene tree and the species tree to common taxa. *opt* is the exact hybridization number, as computed by TerminusEst. Ratio 4-approx (resp. 6-approx) is the ratio of the solution obtained by NonbinaryCycleKiller (running in 4-approx, resp. 6-approx mode) to the solution obtained by TerminusEst. Gap (T-EST - MAF) is the absolute gap between the optimum MAF solution (here computed with rSPR) and the exact hybridization number, as computed by TerminusEst. Gap (4-approx - MAF) is the absolute gap between the optimum MAF solution and the reticulation number of the solution generated by NonbinaryCycleKiller running in its 4-approx mode. Time T-EST is the running time (in seconds) of TerminusEst, and Time 4-approx is the running time (in seconds) of NonbinaryCycleKiller running in its 4-approx mode. In 202 instances TerminusEst returned the same size solution as rSPR, in 202 cases TerminusEst returned the same size solution as NonbinaryCycleKiller (running in 4-approx mode), and in 201 cases NonbinaryCycleKiller (running in 4-approx mode) returned the same size solution as rSPR.

gene duplication events arising in genomes, some species host several copies of the same gene, hence individual gene trees usually have several leaves labeled with identical species names. Unfortunately, our methods do not handle such multi-labeled gene trees (MUL trees). We thus transformed the MUL trees into trees containing single copies of labels, applying the tools described in [31,32] to the forest F of 7134 trees. As in Section 4.1 of [31], we obtained four data sets: F_1, F_2, F_3^p and F_3^s, respectively containing 204, 1003, 5924 and 5789 trees. Note that only F_3^s contains nonbinary trees. Finally, for each single labeled tree $G \in (F_1 \cup F_2 \cup F_3^p \cup F_3^s)$, we restricted the species tree S (containing 22 taxa) to the leaves of G and we applied our methods to all so obtained pairs (restricted S, G). The results are presented in Tables 4, 5, 6 and 7. For all four datasets both TerminusEst and NonbinaryCycleKiller ran extremely quickly, rarely taking more than a couple of seconds for each species-gene tree pair. Moreover, the clear conclusion with this dataset is that, although the species-gene pairs are often incompatible, there are rarely many cycles to kill and optimum solutions to the hybridization number problem are generally extremely close to optimal solutions to MAF.

Table 5 Summary of results for dataset F_2 (1003 gene trees) originally obtained from GreenPhylDB database

	MIN	AVG	MAX
Common taxa	3	11.704	22
opt	0	2.854	10
Ratio 4-approx	1	1.025	2
Ratio 6-approx	1	1.264	3
Gap (T-EST - MAF)	0	0.048	1
Gap (4-approx - MAF)	0	0.165	3
Time T-EST	0	0.576	7
Time 4-approx	0	0.605	3

In 955 instances TerminusEst returned the same size solution as rSPR, in 911 cases TerminusEst returned the same size solution as NonbinaryCycleKiller (running in 4-approx mode), and in 880 cases NonbinaryCycleKiller (running in 4-approx mode) returned the same size solution as rSPR.

Table 6 Summary of results for dataset F_3^p (5924 gene trees) originally obtained from GreenPhylDB database

	MIN	AVG	MAX
Common taxa	2	14.206	22
opt	0	3.613	12
Ratio 4-approx	1	1.027	2
Ratio 6-approx	1	1.277	3
Gap (T-EST - MAF)	0	0.065	2
Gap (4-approx - MAF)	0	0.195	4
Time T-EST	0	0.689	21
Time 4-approx	0	0.729	3

In 5553 instances TerminusEst returned the same size solution as rSPR, in 5297 cases TerminusEst returned the same size solution as NonbinaryCycleKiller (running in 4-approx mode), and in 5030 cases NonbinaryCycleKiller (running in 4-approx mode) returned the same size solution as rSPR.

Table 7 Summary of results for dataset F_3^s (5789 gene trees) originally obtained from GreenPhylDB database

	MIN	AVG	MAX
Common taxa	3	17.319	22
opt	0	1.560	12
Ratio 4-approx	1	1.021	2
Ratio 7-approx	1	1.704	4
Gap (T-EST - MAF)	0	0.053	4
Gap (4-approx - MAF)	0	0.132	5
Time T-EST	0	0.422	15
Time 4-approx	0	1.182	14

In 5552 instances TerminusEst returned the same size solution as rSPR, in 5415 cases TerminusEst returned the same size solution as NonbinaryCycleKiller (running in 4-approx mode), and in 5209 cases NonbinaryCycleKiller (running in 4-approx mode) returned the same size solution as MAF. In this dataset the gene trees were also nonbinary, meaning that NonbinaryCycleKiller had to use the MAF algorithm described in [21] instead of rSPR.

Conclusions

Our experiments with binary trees show that CycleKiller is much faster than available exact methods once the input trees become sufficiently large and/or discordant. In over 96% of the cases CycleKiller finds the optimal solution and in the remaining cases it finds a solution very close to the optimum. We have shown that the most accurate mode of the program produces solutions that are at most a factor 2 from the optimum. In practice, the average-case approximation ratio that we observed was 1.003. The fastest mode of the algorithm can be used on trees with thousands of leaves and provably constructs networks that are at most a factor of 4 from the optimum.

Our experiments with nonbinary trees highlight once again that the cycle-breaking technique described in this article is intrinsically linked to the current state-of-the-art in MAF algorithms. TerminusEst is faster than the most accurate mode of NonbinaryCycleKiller when both trees are nonbinary due to the fact that MAF solvers for two nonbinary trees have not yet been optimized to the same extent as their binary counterparts. In fact, TerminusEst is the best avaible exact method for nonbinary trees and can handle instances for which the optimum is up to 15-20. For other instances, NonbinaryCycleKiller in its fastest mode is much faster than TerminusEst and produces solutions that are at most a factor 4 from the optimum (less than 2.6 on average).

Finally, for instances with one binary and one nonbinary tree, the most accurate mode of NonbinaryCycleKiller is again much faster than TerminusEst and produces solutions that are at most a factor 1.75 from the optimum (less than 1.011 on average).

Endnotes

[a]In [15], it has been shown that this number can go up to 40 when running Dendroscope on a similar processor but allocating all cores for one instance, i.e. exploiting the possibilities of parallel computation of this implementation.

[b]We note that NonbinaryCycleKiller uses a row-generation ILP formulation - based on [33] - to solve DFVS, rather than the polynomial-size formulation used by CycleKiller. ILP is in neither case a bottleneck for the running time.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

All the authors conceived the ideas, designed and conducted the experiments, and wrote and approved the paper.

Authors' information

Leo van Iersel was supported by a Veni grant of The Netherlands Organisation for Scientific Research (NWO). Nela Lekić was supported by a Vrije Competitie grant of of The Netherlands Organisation for Scientific Research (NWO).

Acknowledgements

A preliminary version of this paper (restricted to the binary case) appeared in the proceedings of the 12th Workshop on Algorithms in Bioinformatics (WABI 2012) [28]. We thank Simone Linz and Leen Stougie for fruitful discussions. This publication is the contribution no. 2014-040 of the Institut des Sciences de l'Evolution de Montpellier (ISE-M, UMR 5554). This work has been partially funded by the French *Agence Nationale de la Recherche, Investissements d'avenir/Bioinformatique* (ANR-10-BINF-01-02, *Ancestrome*), and it has benefited from the ISE-M computing facilities.

Author details

[1]Centrum Wiskunde & Informatica (CWI), P.O. Box 94079, 1090 GB, Amsterdam, The Netherlands. [2]Department of Knowledge Engineering (DKE), Maastricht University, P.O. Box 616, 6200 MD, Maastricht, The Netherlands. [3]ISEM, CNRS – Université Montpellier II, Place Eugène Bataillon, 34095 Montpellier, France.

Received: 02 December 2013 Accepted: 24 April 2014
Published: 05 May 2014

References

1. Gascuel O, (ed.): *Mathematics of Evolution and Phylogeny*. UK: Oxford University Press Inc.; 2005.
2. Gascuel O, Steel M, (eds.): *Reconstructing Evolution: New Mathematical and Computational Advances*. UK: Oxford University Press; 2007.
3. Bapteste E, van Iersel LJJ, Janke A, Kelchner S, Kelk SM, McInerney JO, Morrison DA, Nakhleh L, Steel M, Stougie L, Whitfield J: **Networks: expanding evolutionary thinking.** *Trends Genet* 2013, **29**(8):439–441.
4. Huson DH, Rupp R, Scornavacca C: *Phylogenetic Networks: Concepts, Algorithms and Applications*. UK: Cambridge University Press; 2011.
5. Huson DH, Scornavacca C: **A survey of combinatorial methods for phylogenetic networks.** *Genome Biol Evol* 2011, **3:**23–35.
6. Nakhleh L: **Evolutionary phylogenetic networks: models and issues.** In *The Problem Solving Handbook for Computational Biology and Bioinformatics*. Edited by Heath L, Ramakrishnan N. Berlin: Springer; 2009.
7. Bordewich M, Semple C: **Computing the minimum number of hybridization events for a consistent evolutionary history.** *Discrete Appl Math* 2007, **155**(8):914–928.
8. Flum J, Grohe M: *Parameterized Complexity Theory*. Berlin: Springer; 2006.
9. Downey RG, Fellows MR: *Parameterized Complexity (Monographs in Computer Science)*. Berlin: Springer; 1999.
10. Bordewich M, Linz S, John KS, Semple C: **A reduction algorithm for computing the hybridization number of two trees.** *Evol Bioinform* 2007, **3:**86–98.
11. Chen Z-Z, Wang L: **Hybridnet: a tool for constructing hybridization networks.** *Bioinformatics* 2010, **26**(22):2912–2913.
12. Collins J, Linz S, Semple C: **Quantifying hybridization in realistic time.** *J Comp Biol* 2011, **18:**1305–1318.
13. Whidden C, Beiko RG, Zeh N: **Fixed-parameter algorithms for maximum agreement forests.** *SIAM J Comput*, **42**(4):1431–1466.
14. Huson DH, Scornavacca C: **Dendroscope 3: An interactive tool for rooted phylogenetic trees and networks.** *Syst Biol* 2012, **61**(6):1061–1067.
15. Albrecht B, Scornavacca C, Cenci A, Huson DH: **Fast computation of minimum hybridization networks.** *Bioinformatics* 2012, **28**(2):191–197.
16. Chen Z-Z, Wang L: **Algorithms for reticulate networks of multiple phylogenetic trees.** *IEEE/ACM Trans Comput Biol Bioinf* 2012, **9**(2):372–384.
17. Chen Z-Z, Wang L: **An ultrafast tool for minimum reticulate networks.** *J Comput Biol* 2013, **20**(1):38–41.
18. Piovesan T, Kelk S: **A simple fixed parameter tractable algorithm for computing the hybridization number of two (not necessarily binary) trees.** *IEEE/ACM Trans Comput Biol Bioinf* 2013, **10**(1):18–25.
19. Linz S, Semple C: **Hybridization in non-binary trees.** *IEEE/ACM Trans Comput Biol Bioinf* 2009, **6**(1):30–45.
20. Kelk SM, van Iersel LJJ, Lekic N, Linz S, Scornavacca C, Stougie L: **Cycle killer...qu'est-ce que c'est? on the comparative approximability of hybridization number and directed feedback vertex set.** *SIAM J Discr Math* 2012, **26**(4):1635–1656.
21. van Iersel LJJ, Kelk SM, Lekić N, Stougie L: **Approximation algorithms for nonbinary agreement forests.** *SIAM J Discrete Math* 2014, **28**(1):49–66.

22. Whidden C: **rSPR.** http://kiwi.cs.dal.ca/Software/RSPR.
23. Whidden C, Beiko RG, Zeh N: **Fast FPT algorithms for computing rooted agreement forests: Theory and experiments.** In *Proceedings of the 9th International Symposium on Experimental Algorithms (SEA)*. Lect Notes Comput Sc, vol. 6049, pp. 141–153 SpringerL: Berlin; 2010.
24. Whidden C, Beiko RG, Zeh N: **Fixed-Parameter and Approximation Algorithms for Maximum Agreement Forests of Multifurcating Trees.** ArXiv preprint: http://arxiv.org/abs/1305.0512 (2013).
25. Kelk SM: **CycleKiller.** http://skelk.sdf-eu.org/cyclekiller.
26. van Iersel LJJ: **NonbinaryCycleKiller.** http://homepages.cwi.nl/~iersel/cyclekiller.
27. Kelk SM: **TerminusEst.** http://skelk.sdf-eu.org/terminusest.
28. van Iersel LJJ, Kelk SM, Lekic N, Scornavacca C: **A practical approximation algorithm for solving massive instances of hybridization number.** In *Algorithms in Bioinformatics*. Lect Notes Comput Sc, vol. 7534, pp. 430–440. Edited by Raphael B, Tang J. Berlin: Springer; 2012.
29. Baroni M, Grünewald S, Moulton V, Semple C: **Bounding the number of hybridisation events for a consistent evolutionary history.** *J Math Biol* 2005, **51:**171–182.
30. Rouard M, Guignon V, Aluome C, Laporte M-A, Droc G, Walde C, Zmasek CM, Périn C, Conte MG: **Greenphyldb v2.0: comparative and functional genomics in plants.** *Nucleic Acids Res* 2010. doi:10.1093/nar/gkq811. Epub 2010 Sep 22.
31. Scornavacca C, Berry V, Ranwez V: **Building species trees from larger parts of phylogenomic databases.** *Inform Comput* 2011, **209**(3):590–605.
32. Scornavacca C: **SSIMUL.** http://www.atgc-montpellier.fr/ssimul/.
33. Even G, Naor J, Schieber B, Sudan M: **Approximating minimum feedback sets and multicuts in directed graphs.** *Algorithmica* 1998, **20**(2):151–174.

doi:10.1186/1471-2105-15-127
Cite this article as: van Iersel *et al.*: A practical approximation algorithm for solving massive instances of hybridization number for binary and nonbinary trees. *BMC Bioinformatics* 2014 **15**:127.