**Noname manuscript No.**
(will be inserted by the editor)

# Trinets encode tree-child and level-2 phylogenetic networks

**Leo van Iersel · Vincent Moulton**

the date of receipt and acceptance should be inserted later

**Abstract** Phylogenetic networks generalize evolutionary trees, and are commonly used to represent evolutionary histories of species that undergo reticulate evolutionary processes such as hybridization, recombination and lateral gene transfer. Recently, there has been great interest in trying to develop methods to construct rooted phylogenetic networks from *triplets*, that is rooted trees on three species. However, although triplets determine or *encode* rooted phylogenetic trees, they do not in general encode rooted phylogenetic networks, which is a potential issue for any such method. Motivated by this fact, Huber and Moulton recently introduced *trinets* as a natural extension of rooted triplets to networks. In particular, they showed that level-1 phylogenetic networks *are* encoded by their trinets, and also conjectured that all "recoverable" rooted phylogenetic networks are encoded by their trinets. Here we prove that recoverable binary level-2 networks and binary tree-child networks are also encoded by their trinets. To do this we prove two decomposition theorems based on trinets which hold for *all* recoverable binary rooted phylogenetic networks. Our results provide some additional evidence in support of the conjecture that trinets encode all recoverable rooted phylogenetic networks, and could also lead to new approaches to construct phylogenetic networks from trinets.

**Keywords** Phylogenetic network · directed graph · reticulate evolution · uniqueness · encoding · trinet

**Mathematics Subject Classification (2000)** 68R05 · 05C20 · 92D15

## 1 Introduction

Phylogenetic trees are routinely used in biology to represent the evolutionary relationships between a given set of species. More formally, for a set $X$ of species, a *rooted phylogenetic tree* is

Leo van Iersel
Centrum Wiskunde & Informatica (CWI), P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
E-mail: l.j.j.v.iersel@gmail.com

Vincent Moulton
School of Computing Sciences, University of East Anglia, Norwich, NR4 7TJ, United Kingdom
E-mail: vincent.moulton@cmp.uea.ac.uk

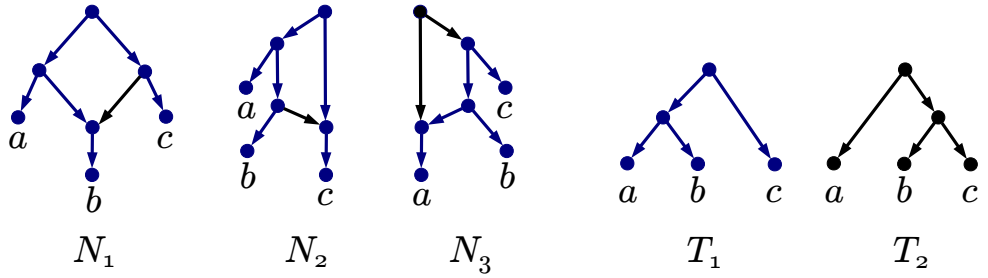


**Fig. 1** Three non-isomorphic tree-child, level-1 networks that all have the same set of rooted triplets, that is, $Tr(N_1) = Tr(N_2) = Tr(N_3) = \{T_1, T_2\}$. Blue is used to illustrate how $T_1$ is contained in $N_1, N_2$ and $N_3$.

a rooted (graph theoretical) tree that has no indegree-1 outdegree-1 vertices, and in which the leaves are bijectively labelled by the elements in $X$ (Semple and Steel, 2003); a *(rooted) triplet* is a phylogenetic tree with three leaves. Given a rooted phylogenetic tree $T$ and three of its leaves, there is unique triplet spanned by those leaves that is contained in $T$. A fundamental result in phylogenetics states that $T$ is in fact *encoded* by its triplets, that is, $T$ is the unique phylogenetic tree containing the set of triplets that arises from taking all combinations of three leaves in $T$ (Dress et al., 2012). This result is important since it has led to various approaches to constructing phylogenetic trees from set of triplets cf. e.g. Aho et al. (1981); Ranwez et al. (2007); Scornavacca et al. (2008).

Recently, there has been some interest in using networks rather than trees to represent evolutionary relationships between species that have undergone reticulate evolution (Huson et al., 2011; Morrison, 2011). This is motivated by the fact that processes such as hybridization, recombination and lateral gene transfer can lead to evolutionary histories which are not best represented by a tree. Formally, a *(rooted phylogenetic) network* for a set $X$ of species is a directed acyclic graph that has a single root, has no indegree-1 outdegree-1 vertices, and has its leaves bijectively labelled by $X$ (see Section 2 for full definitions concerning networks). Such a network is called *binary* if all vertices have indegree and outdegree at most two and all vertices with indegree two have outdegree one. In addition, a binary network is called *level-k* (Gambette et al., 2009, 2012; Gambette and Huber, 2012; Habib and To, 2012; van Iersel et al., 2009a; van Iersel and Kelk, 2011a) if each biconnected component has at most $k$ indegree-2 vertices, and it is called *tree-child* (Cardona et al., 2010, 2009c; van Iersel et al., 2010; Willson, 2012) if each non-leaf vertex has at least one child which has indegree 1. Note that a rooted phylogenetic tree is a network, but that networks are more general since they can represent evolutionary events where species combine rather than speciate.

As with phylogenetic trees, efficient algorithms have been developed which, given a set of triplets, aim to build a network that contains this set (see e.g. Byrka et al. (2010); Habib and To (2012); Huber et al. (2011); Jansson et al. (2006)). However, these algorithms share a common weakness in that, even if *all* of the triplets within a given network are taken as input, there is no guarantee that the original network will be reconstructed. This is because, in contrast to trees, the triplets in a network do not necessarily encode the network (Gambette and Huber, 2012). For example, Figure 1 presents three different networks that all contain the same set of triplets. Note that a similar observation has been made concerning the set of trees and set of clusters displayed by a network (see e.g. Huson et al. (2011); van Iersel and Kelk (2011b)).

Motivated by this problem, Huber and Moulton (2012) recently proposed a possible alternative way to encode rooted phylogenetic networks by introducing a natural extension of rooted triplets

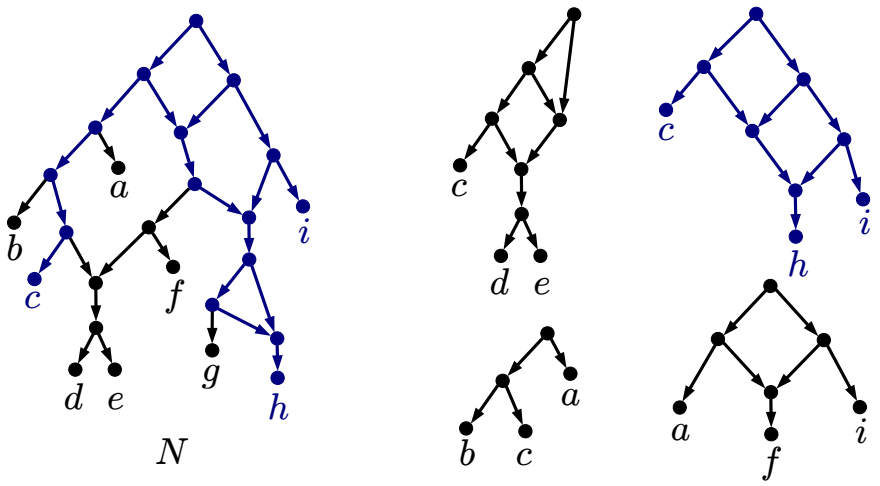


**Fig. 2** Example of a rooted phylogenetic network $N$ (left) and four of the trinets exhibited by $N$ (right). The network $N$ is binary, recoverable, has level 3 and has the tree-child property. Blue is used to illustrate how $N$ exhibits the pictured trinet on $\{c, h, i\}$.

to networks. More specifically, a *trinet* is a rooted phylogenetic network on three leaves. As with the triplets in a tree, a network contains or "exhibits" a trinet on every three leaves (see Section 2). For example, Figure 2 presents a phylogenetic network and four of the trinets that it exhibits. The main result of Huber and Moulton (2012) implies that level-1 networks are encoded by their trinets. Moreover, it is conjectured that any "recoverable" network (a network that satisfies some relatively mild condition which we recall below) is also encoded by its trinets. Here, we provide some evidence in support of this conjecture by showing that recoverable level-2 and tree-child networks are also encoded by their trinets.

We now give an overview of the rest of this paper, in which all networks are assumed to be binary. After presenting some preliminaries in Section 2, we begin by studying the relationship between the structure of a network and the trinets that it exhibits. In particular, in Section 3 we present two decomposition theorems for general networks. Essentially, these two theorems state that the cut-arcs of a network (that is, arcs whose removal disconnect the network) can be directly deduced from its set of trinets (Theorem 1), and that a network is encoded by its trinets if and only if each of its biconnected components is encoded by its trinets (Theorem 2). In tandem, these theorems essentially restrict the problem of deciding whether or not trinets encode networks to the class of networks that do not have any cut-arcs apart from pendant arcs (so-called "simple" networks).

By restricting our attention to simple networks, in Section 4 we show that a recoverable level-2 network is always encoded by its trinets (Corollary 1). To do this, we use the concept of "generators" for level-$k$ networks, introduced by van Iersel et al. (2009a) and van Iersel et al. (2009b). In Section 5, we then use alternative techniques to prove that tree-child networks are also encoded by their trinets (Theorem 4). Note that this class of networks includes the class of regular networks (Baroni et al., 2004). Thus it is interesting to note that a regular network is encoded by the set of trees[1] that it contains (Willson, 2010), but that this is not the case for tree-child networks (e.g. all of the networks in Figure 1 contain the same set of trees). In Section 6, we con-

[1] Note that all of these trees have the same leaf-set as the network.

clude the paper with two corollaries and a discussion of our results and possible future research directions.

Ultimately, it is hoped that the results presented in this paper will lead to new methods for constructing phylogenetic networks. In principle it should be straight-forward to infer low-level trinets for biological datasets consisting of molecular sequences using existing methods to construct phylogenetic networks. For example, given a multiple sequence alignment, the most parsimonious or most likely level-1 or level-2 trinet for every sub-alignment of three sequences could be computed using, e.g., methods described by Jin et al. (2006, 2009), which becomes computationally tractible since there are a bounded number of such trinets (under certain natural restrictions, see Sections 2 and 6). The structural results in this paper, such as the decomposition theorems presented in Section 3, could then be used to help design algorithms to construct networks from the trinets inferred in this way. Note that this has the potential advantage that 'breakpoints' need not be computed for the multiple alignment, a first (and sometimes quite difficult) step that is commonly required for constructing phylogenetic networks from phylogenetic trees or clusters (cf. e.g. Nakhleh (2011, Section 2)).

## 2 Preliminaries

Throughout the paper, $X$ is a finite set. As mentioned in the introduction, a *rooted phylogenetic network* on $X$ is a directed acyclic graph with a single indegree-0 vertex (the *root*) and a bijective labelling of its outdegree-0 vertices (*leaves*) by the elements of $X$. We identify each leaf with its label. A phylogenetic network is *binary* if all vertices have indegree and outdegree at most 2 and all vertices with indegree 2 have outdegree 1. We will often refer to a rooted phylogenetic network simply as a *phylogenetic network* or a *network* for short. See Figures 1, 2 and 3 for examples. Let $u$ and $v$ be two vertices of a phylogenetic network $N$. If $(u, v)$ is an arc of $N$, then we say that $u$ is a *parent* of $v$ and that $v$ is a *child* of $u$. Furthermore, we write $u \leq_N v$ and say that $v$ is *below* $u$, if there is a directed path from $u$ to $v$ in $N$, or $u = v$. For two leaves $x$ and $y$, we say that $x$ is *below* $y$ if the parent of $x$ is below the parent of $y$. For an arc $a = (u, v)$ and a vertex $w$, we say that $w$ is *below* $a$ if $w$ is below $v$.

Let $D$ be a directed graph with a single root $\rho$. The indegree of a vertex $v$ of $D$ is denoted $\delta^-(v)$ and $v$ is said to be a *reticulation vertex* or a *reticulation* if $\delta^-(v) \geq 2$. The *reticulation number* of $D$ is defined as

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

Hence, the reticulation number of a binary network is simply the number of its reticulation vertices.

We say that a vertex $v$ of $D$ is a *cut-vertex* if its removal disconnects the underlying undirected graph of $D$. Similarly, an arc $a$ of $D$ is a *cut-arc* if its removal disconnects the underlying undirected graph of $D$. A directed graph is called *biconnected* if it has no cut-vertices. A *biconnected component* is a maximal biconnected subgraph (i.e. a biconnected subgraph that is not contained in any other biconnected subgraph). Note that, by this definition, each cut-arc is a biconnected component. We call these the *trivial biconnected components*. Thus, rephrasing the definitions given in the introduction, a phylogenetic network is *level-k* if each biconnected component has reticulation number at most $k$, it is *tree-child* if every non-leaf vertex of the network has at least one child that is not a reticulation, and it is *simple* if the head of each cut-arc is a leaf.


**Fig. 3** The phylogenetic network on the left is not recoverable because it has a strongly redundant biconnected component. The phylogenetic network on the right is recoverable, because its only nontrivial biconnected component, although redundant, is not strongly redundant.

As explained in the introdution, in this paper we are mainly concerned with extending a result by Huber and Moulton (2012) from level-1 networks to level-2 and tree-child networks. Level-1 networks have a relatively simple and well-understood structure (Gambette and Huber, 2009; Gambette et al., 2009; Jansson et al., 2006). Level-2 networks are more general and the underlying topologies of their biconnected components can be more complicated (van Iersel et al., 2009a; van Iersel and Kelk, 2011a). The class of tree-child networks is a different generalization of the class of level-1 networks that is not directly comparable to the class of level-2 networks. On the one hand, tree-child networks can have complicated biconnected components with many reticulations. On the other hand, tree-child networks can not have any "invisible vertices", i.e. vertices from which all paths lead to reticulations.

Given a nontrivial biconnected component $B$, we say that $B$ is *redundant* if it has only one outgoing arc and we say that $B$ is *strongly redundant* if it has only one outgoing arc $(u, v)$ and all leaves of the network are below $v$. We say that a phylogenetic network $N$ is *recoverable* if it has no strongly redundant biconnected components (see e.g. Figure 3). We remark that all level-1 networks are recoverable (Huber and Moulton, 2012). Moreover, neither level-1 nor tree-child networks can have any redundant biconnected components. On the other hand, there are level-2 networks that *do* have redundant (and strongly redundant) biconnected components (see Figure 3).

A *trinet* is a phylogenetic network with three leaves. Ignoring leaf-labels, there are 14 distinct level-1 trinets, 8 of which are binary (Huber and Moulton, 2012). Note that there is an infinite number of level-2 trinets, and even of recoverable level-2 trinets. On the other hand, it is not too difficult to see that the number of level-2 trinets without redundant biconnected components is finite. In the appendix, we show that there are 220 such trinets (restricting to binary trinets). Moreover, we show that the number of level-$k$ trinets without redundant biconnected components is bounded by a function of $k$. We shall also return to this point in Section 6.

Given a network $N$ on $X$ and $X' \subseteq X$, a *lowest stable ancestor* $LSA(X')$ is defined as a vertex $w \notin X'$ of $N$ for which all paths from the root to any $x \in X'$ pass through $w$, and such that no vertex below $w$ has this property. A *lowest common ancestor* of $X'$ in $N$ is a vertex $w$ such that $w \leq_N x$ for all $x \in X'$ and no vertex below $w$ has this property. Note that the lowest stable ancestor is unique but that this not necessarily the case for a lowest common ancestor (Fischer and Huson, 2010). If a lowest common ancestor of $X'$ *is* unique, then we denote it by $LCA(X')$. Keep in mind that, even if the lowest common ancestor is unique, it is not necessarily

equal to the lowest stable ancestor. For two vertices $u, v$, we write $LSA(u, v)$ as shorthand for $LSA(\{u, v\})$ and $LCA(u, v)$ as shorthand for $LCA(\{u, v\})$. The following easily proven fact will be useful later on.

**Observation 1** *If $N$ is a phylogenetic network on $X$ and $X' \subseteq X$ with $X' \geq 2$, then there exist $x, y \in X'$ such that $LSA(x, y) = LSA(X')$.*

Given a phylogenetic network $N$ on $X$ and $\{x, y, z\} \subseteq X$, the trinet on $\{x, y, z\}$ *exhibited* by $N$ is defined as the trinet obtained from $N$ by deleting all vertices that are not on any path from $LSA(\{x, y, z\})$ to $x$, $y$ or $z$ and subsequently suppressing all indegree-1 outdegree-1 vertices and parallel arcs. See Figure 2 for some examples. We note that this definition is equivalent to the definition of "display" of Huber and Moulton (2012) but we call it "exhibit" to clearly distinguish it from other usages of "display" (in particular, the definition of when a network displays a tree or triplet). We will often (implicitly) use the following observation.

**Observation 2** *Given a phylogenetic network $N$ on $X$ and $\{x, y, z\} \subseteq X$, the trinet on $\{x, y, z\}$ exhibited by $N$ can be obtained from $N$ by removing all leaves except $x, y$ and $z$ and repeatedly applying the following operations until none is applicable:*

- *deleting all unlabelled outdegree-0 vertices;*
- *deleting all indegree-0 outdegree-1 vertices;*
- *suppressing all indegree-1 outdegree-1 vertices;*
- *suppressing all parallel arcs; and*
- *suppressing all strongly redundant biconnected components.*

By suppressing parallel arcs, we mean replacing each set of parallel arcs by a single arc. By suppressing strongly redundant biconnected components, we mean replacing each such component by a single vertex.

The following observation, linking lowest common ancestors in networks and their exhibited trinets, will be used in the proof of Theorem 4.

**Observation 3** *Suppose that $u$ is the unique lowest common ancestor of two leaves $x$ and $y$ in a network $N$ and that $P$ is a trinet exhibited by $N$ that contains $x$ and $y$. Then, $P$ contains $u$ (where we consider $P$ as being obtained from $N$ as described in Observation 2) and $u$ is the unique lowest common ancestor of $x$ and $y$ in $P$.*

We now make a definition that will be crucial for the decomposition theorems in Section 3 (note that a somewhat related definition was used by Habib and To (2012)).

**Definition 1** Let $N$ be a phylogenetic network on $X$ and $A \subseteq X$. Then, $A$ is a *CA-set (Cut-Arc set)* of $N$ if there exists a cut-arc $(u, v)$ of $N$ such that $A = \{x \in X \mid v \leq_N x\}$.

For example, the CA-sets of network $N$ in Figure 2 are $\{d, e\}, \{g, h\}$ and all singletons $\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}$. In the next section, we will make use of the following easily proven fact that relates the CA-sets of a network to the CA-sets of its exhibited trinets.

**Observation 4** *Let $N$ be a phylogenetic network on $X$ and $P$ the trinet on $\{x, y, z\} \subseteq X$ exhibited by $N$. If $A \subseteq X$ is a CA-set of $N$, then $A \cap \{x, y, z\}$ is a CA-set of $P$.*

The lowest stable ancestor of a CA-set has the following interesting property.

**Observation 5** *If $N$ is a phylogenetic network on $X$ and $A \subseteq X$ a CA-set of $N$, then the arc entering $LSA(A)$ is a cut-arc. Moreover, $A = \{x \in X \mid LSA(A) \leq_N x\}$.*

Given two phylogenetic networks $N$ and $N'$ on $X$, we write $N = N'$ if there is a graph isomorphism between $N$ and $N'$ that preserves leaf labels, i.e. if there exists a bijective function $f : V(N) \to V(N')$ such that $f(x) = x$ for each leaf $x$ of $N$ and such that for every $u, v \in V(N)$ holds that $(u, v)$ is an arc of $N$ if and only if $(f(u), f(v))$ is an arc of $N'$.

We use $Tn(N)$ to denote the set of all trinets exhibited by a phylogenetic network $N$. A phylogenetic network $N$ is *encoded* by its set of trinets $Tn(N)$ if there is no recoverable phylogenetic network $N' \neq N$ with $Tn(N) = Tn(N')$.

## 3 Decomposition theorems for trinets

It is well known that any graph can be decomposed into its biconnected components. We begin by showing that trinets can be used to recover this decomposition of binary phylogenetic networks. Note that similar results have been proven for triplets by van Iersel and Kelk (2011a) and for quartets in unrooted phylogenetics networks by Gambette et al. (2009).

**Theorem 1** *Let $N$ be a recoverable binary phylogenetic network on $X$, and $A \subset X$. Then, $A$ is a CA-set of $N$ if and only if $|A| = 1$ or, for all $z \in X \setminus A$ and $x, y \in A$ with $x \neq y$, $\{x, y\}$ is a CA-set of the trinet on $\{x, y, z\}$ exhibited by $N$.*

*Proof* Let $A \subset X$. If $|A| = 1$ then the theorem clearly holds since each leaf of a binary network has a cut-arc entering it and thus forms a singleton CA-set. Hence, we assume $|A| \geq 2$.

To prove the "only if" direction, assume that $A$ is a CA-set of $N$. Let $z \in X \setminus A$ and $x, y \in A$ with $x \neq y$. There exists a unique trinet $P$ on $\{x, y, z\}$ in $Tn(N)$. Since $A$ is a CA-set of $N$, it follows from Observation 4 that $\{x, y\}$ is a CA-set of $P$ and we are done.

It remains to prove the "if" direction. Assume that for all $z \in X \setminus A$ and $x, y \in A$ with $x \neq y$, $\{x, y\}$ is a CA-set of the trinet on $\{x, y, z\}$ exhibited by $N$. Assume that $A$ is not a CA-set of $N$.

First assume that the arc entering $LSA(A)$ is a cut-arc $a$. By Observation 1, there exist $x, y \in A$ such that $LSA(x, y) = LSA(A)$. Moreover, there exist $z \in X \setminus A$ below $a$ because $A$ is not a CA-set. Consider the trinet $P$ on $\{x, y, z\}$ exhibited by $N$. By the definition of "exhibit", all paths from $LSA(\{x, y, z\})$ and hence from $LSA(x, y)$ to $x$, $y$ and $z$ are retained in $P$, although vertices on the paths might be suppressed and several paths might be collapsed into single paths. It follows that $z$ is also below $LSA(x, y)$ in $P$. Then it follows from Observation 5 that $\{x, y\}$ is not a CA-set of $P$, which is a contradiction. Hence, there is no cut-arc entering $LSA(A)$, which implies that $LSA(A)$ is in a nontrivial biconnected component.

Now, let $B$ be the nontrivial biconnected component of $N$ containing $LSA(A)$. It is not too difficult to show that $B$ has a single vertex with no ancestors in $B$. We call this vertex $r_B$ the *root* of $B$. Choose $x, y \in A$ that are below different cut-arcs leaving $B$ such that $LSA(x, y) = LSA(A)$. First, we observe that there is no leaf $z \in X \setminus A$ below $LSA(x, y)$, because otherwise we could argue as before that the trinet on $\{x, y, z\}$ exhibited by $N$ does not have $\{x, y\}$ as a CA-set. Pick $z \in X \setminus A$ arbitrarily below a cut-arc leaving $B$. Note that neither $x$ nor $y$ is below this cut-arc because otherwise $z$ would be below $LSA(x, y)$.

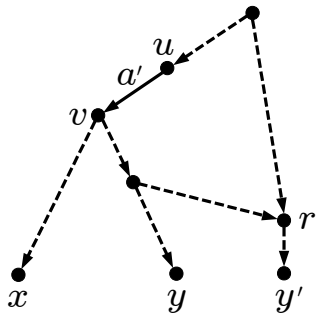


**Fig. 4** Illustration of network $N$ in the proof of Theorem 1. Dashed arcs denote directed paths. Arc $a'$ corresponds to cut-arc $a$ of trinet $P$ on $\{x, y, z\}$, while $a'$ is not a cut-arc of $N$.

By assumption, the trinet $P$ on $\{x, y, z\}$ exhibited by $N$ has $\{x, y\}$ as a CA-set. This means that $P$ has a cut-arc $a$ such that $x$ and $y$ are below $a$ but $z$ is not. Consider the arc $a' = (u, v)$ of $N$ corresponding to cut-arc $a$ of $P$. Observe that $v$ is the lowest stable ancestor of $x$ and $y$ in $N$, because all paths from $LSA(x, y)$, to $x$ and $y$ are retained in $P$ (although vertices on the paths might be suppressed and several paths might be collapsed into single paths). Also observe that $a'$ is not a cut-arc in $N$ because $x, y$ and $z$ are below three different cut-arcs leaving a biconnected component $B$. Thus, $a'$ is some arc of $B$ and the operations from Observation 2 that turn $N$ into $P$ destroy the biconnectivity of $B$. Observe that the only one of these operations that does not preserve biconnectivity is the deletion of unlabelled outdegree-0 vertices in case they have indegree greater than 1. We claim that there then exists a reticulation $r$ in $N$ with directed paths from $v$ to $r$ and from some ancestor of $u$ to $r$ not passing through $a'$ (see Figure 4).

To prove this claim, first note that, since $a'$ is not a cut-arc of $N$, there is some undirected path $U$ in $N$ from $v$ to $u$ not passing through $a'$. Consider the last vertex $r$ of $U$ that is below $v$. Clearly, $r$ is a reticulation. Let $p$ be the next vertex on path $U$. Then $p$ is not below $v$. Clearly, $p$ is below the root and, since $p$ is not below $v$, the path from the root to $p$ does not pass through $a'$. It follows that $r$ is a reticulation with directed paths from $v$ to $r$ and from some ancestor of $u$ to $r$ not passing through $a'$.

Now, consider any leaf $y'$ below $r$. First observe that $y' \in A$ because no leaves $z \in X \setminus A$ are below $LSA(x, y)$. However, then we obtain a contradiction because $LSA(x, y')$ is closer to the root than $LSA(x, y) = v$. □

Note that we could have used Lemma 3 of van Iersel and Kelk (2011a) in the proof of the last result. However, we presented the above proof since it is shorter, self-contained and provides some insight into how to make arguments using trinets. We also note that for an arbitrary binary recoverable network $N$ there is not necessarily a bijection between the cut-arcs of $N$ and the CA-sets of $N$ because different cut-arcs might correspond to the same CA-set. However, it is easy to see that the following related observation does hold.

**Observation 6** *If $N$ is a binary phylogenetic network without redundant biconnected components, then there is a bijection between the cut-arcs of $N$ and the CA-sets of $N$.*

We now turn to showing that, roughly speaking, a binary network is encoded by its trinets if and only if each of its biconnected components is encoded by its trinets. To this end, let $N$ be a phylogenetic network and $B$ a nontrivial biconnected component with $b$ outgoing cut-arcs $a_1 = (u_1, v_1), \ldots, a_b = (u_b, v_b)$. Consider the phylogenetic network $N_B$ obtained from $N$

by deleting all biconnected components except for $B, a_1, \ldots, a_b$ and labelling $v_1, \ldots, v_b$ by new labels $y_1, \ldots, y_b$ that are not in $X$. We call $N_B$ a *restriction* of $N$ to $B$. Note that $N_B$ is unique up to the choice of the new labels $y_1, \ldots, y_b$. Furthermore, $N_B$ is a simple network.

**Theorem 2** *A recoverable binary phylogenetic network $N$ on $X$, with $|X| \geq 3$, is encoded by its trinets $Tn(N)$ if and only if, for each nontrivial biconnected component $B$ of $N$ with at least four outgoing cut-arcs, $N_B$ is encoded by $Tn(N_B)$.*

*Proof* To prove the "only if" direction of the theorem, suppose that $N$ is a recoverable binary phylogenetic network on $X$ that is encoded by its trinets $Tn(N)$. Consider any nontrivial biconnected component $B$ of $N$ with at least four outgoing cut-arcs. For contradiction, suppose that $N_B$ is not encoded by $Tn(N_B)$, i.e. there exists a recoverable network $N'_B \neq N_B$ such that $Tn(N_B) = Tn(N'_B)$. By Theorem 1, $N'_B$ has the same CA-sets as $N_B$. Hence, all CA-sets of $N'_B$ are singletons. In addition, we claim that $N'_B$ has no redundant biconnected components. If it had one, then there would be only one leaf, say $x$, below it. However, then all trinets containing $x$ would have a redundant biconnected component with $x$ directly below it. This is not possible because $Tn(N_B) = Tn(N'_B)$ and it is easily checked that for each leaf $x$ there exists a trinet in $Tn(N_B)$ with no such redundant biconnected component. Hence, $N'_B$ has no redundant biconnected components. Combining this with the previous observation that all CA-sets of $N'_B$ are singletons, it now follows that $N'_B$ consists of one nontrivial biconnected component with leaves attached to it by cut-arcs, i.e. it is a simple network. Let $B'$ be the nontrivial biconnected component of $N'_B$. Let $N'$ be the result of replacing $B$ by $B'$ in $N$. We will show that $Tn(N) = Tn(N')$, which will contradict the fact that $N$ is encoded by $Tn(N)$, since $N'$ is clearly recoverable.

To show that $Tn(N) = Tn(N')$, let $P \in Tn(N)$ and let $x, y$ and $z$ be the leaves of $P$. If $x, y$ and $z$ are all below different cut-arcs, or all below the same cut-arc leaving $B$, then clearly $P \in Tn(N')$ since the only difference between $N$ and $N'$ is that $B$ is replaced by $B'$, and $Tn(N_B) = Tn(N'_B)$. Now suppose that $P$ contains leaves $x, y$ that are below the same cut-arc leaving $B$ and a leaf $z$ below a different cut-arc leaving $B$. Then consider a fourth leaf $q$ that is below a third cut-arc leaving $B$. Since $Tn(N_B) = Tn(N'_B)$, the trinets on $\{x, z, q\}$ exhibited by $N$ and $N'$ are the same. Hence, the binet (phylogenetic network on two leaves) on $\{x, z\}$ exhibited (defined in the same way as for trinets) by $N$ and by $N'$ is the same. Hence, the trinet on $\{x, y, z\}$ exhibited by $N$ and by $N'$ is the same, and so $P \in Tn(N')$. The case that $P$ contains one or more leaves that are not below $B$ can be handled similarly. It therefore easily follows that $Tn(N) = Tn(N')$, as required.

To prove the "if" direction, let $N$ be a recoverable phylogenetic network on $X$ such that for each nontrivial biconnected component $B$ with at least four outgoing cut-arcs the network $N_B$ is encoded by $Tn(N_B)$. Let $N'$ be a recoverable network on $X$ with $Tn(N) = Tn(N')$. We will show that $N = N'$.

First observe that, for a biconnected component $B$ with precisely 3 outgoing cut-arcs, $N_B$ is trivially encoded by $Tn(N_B)$, since in that case $N_B$ is isomorphic to the single trinet in $Tn(N_B)$.

The rest of the proof is by induction on $|X|$. If $|X| = 3$, then, since $N$ and $N'$ are recoverable, they are both equal to the single trinet in $Tn(N)$ and we are done. Assume $|X| \geq 4$. Consider the root $\rho$ of $N$. We shall assume that $\rho$ is in some nontrivial biconnected component $B_\rho$ and that $a_1 = (u_1, v_1), \ldots, a_b = (u_b, v_b)$ are the cut-arcs leaving $B_\rho$. The case that $\rho$ is not in a nontrivial biconnected component can be handled in a similar way, with arcs $a_1, \ldots, a_b$ being the arcs leaving $\rho$ (and $b = 2$ since $N$ is binary).

Let $N_1, \ldots, N_b$ be the networks rooted at $v_1, \ldots, v_b$. More precisely, for $1 \leq i \leq b$, let $N_i$ be the network obtained from $N$ by deleting all vertices that are not below $v_i$. Suppose that $X_i$ is the leaf-set of $N_i$. Then, since $b \geq 2$, we have $|X_i| < |X|$. Note that $N_i$ is not necessarily recoverable.

Now, by Theorem 1, $N'$ has the same CA-sets as $N$. Thus, $X_i$ is a CA-set of $N'$ for $i = 1, \ldots, b$. Since the root $\rho$ of $N$ is in some nontrivial biconnected component $B_\rho$, it follows quite easily that also the root $\rho'$ of $N'$ is in some nontrivial biconnected component $B'_\rho$. Let $a'_1 = (u'_1, v'_1), \ldots, a'_b = (u'_b, v'_b)$ be the cut-arcs leaving $B'_\rho$. Let $N'_1, \ldots, N'_b$ be the networks rooted at $v'_1, \ldots, v'_b$. Assume without loss of generality that $N'_i$ is a network on $X_i$ for $i = 1, \ldots, b$. To show that $N = N'$, it remains to show that $N_{B_\rho} = N_{B'_\rho}$ and that $N_i = N'_i$ for $i = 1, \ldots, b$.

First, we show that $N_{B_\rho} = N_{B'_\rho}$. Observe that $Tn(N_{B_\rho}) = Tn(N_{B'_\rho})$ (if for any three leaves $y_j, y_k, y_\ell$ the trinet in $Tn(N_{B_\rho})$ and the trinet in $Tn(N_{B'_\rho})$ would be different then, for any three leaves $x_j, x_k, x_\ell$ below $a_j, a_k, a_\ell$ respectively, the trinet in $Tn(N)$ and the trinet in $Tn(N')$ would be different). If $b \geq 4$, then it is clear that $N_{B_\rho} = N_{B'_\rho}$ because $Tn(N_{B_\rho}) = Tn(N_{B'_\rho})$ and by assumption $N_{B_\rho}$ is encoded by $Tn(N_{B_\rho})$. Moreover, $b \geq 2$ since $N$ is recoverable. For $b = 3$ the statement $N_{B_\rho} = N_{B'_\rho}$ is trivially true. Hence, the only case left is $b = 2$. Consider two leaves $x, y$ of $N$ that are below the same cut-arc leaving $B_\rho$ and a leaf $z$ that is below the other cut-arc leaving $B_\rho$. These leaves exist since $|X| \geq 3$. Consider the trinet $P$ in $Tn(N)$ on $\{x, y, z\}$. Let $B_\rho(P)$ be the biconnected component of $P$ containing the root of $P$. Then, $N_{B_\rho(P)} = N_{B_\rho}$. Moreover, since $N'$ also exhibits $P$, $N_{B_\rho(P)} = N_{B'_\rho}$. It follows that $N_{B_\rho} = N_{B'_\rho}$.

Now, let $i \in \{1, \ldots, b\}$. We will show that $N_i = N'_i$. Observe that $Tn(N_i) = Tn(N'_i)$ for similar reasons as we used in the previous paragraph. Since $|X_i| < |X|$, the statement $N_i = N'_i$ follows by induction if (a) $N_i$ and $N'_i$ are recoverable and (b) $|X_i| \geq 3$. To show the general case, consider the networks $R_i$ and $R'_i$ obtained from $N_i$ and $N'_i$ respectively by suppressing all strongly redundant biconnected components. Then, $R_i$ and $R'_i$ are recoverable. Hence, if $|X_i| \geq 3$, $R_i = R'_i$ by induction. If $|X_i| = 1$, then clearly $R_i = R'_i$ because both consist of a single leaf. The only case left is $|X_i| = 2$. Consider any leaf $z \in X \setminus X_i$ and the trinet $P$ on $X_i \cup \{z\}$. By Observation 4, $X_i$ is a CA-set of $P$. Let $P^*$ be the result of deleting all vertices that are not below $LSA(X_i)$. Then, $P^* = R_i$ since $R_i$ is recoverable. Moreover, since $P$ is exhibited by $N'$ and $R'_i$ is recoverable, we also have $P^* = R'_i$. Hence, in all cases, $R_i = R'_i$. So, to complete the proof that $N_i = N'_i$, it remains to show that $N_i$ and $N'_i$ have the same strongly redundant biconnected components, in the same order. To prove this, we distinguish the cases $|X_i| = 1$ and $|X_i| \geq 2$.

First, suppose $|X_i| = 1$, say $X_i = \{x\}$. Let $y, z \in X \setminus X_i$ such that $LSA(u_i) \leq_N z$. (Note that there must be at least one leaf $z$ below $LSA(u_i)$ in $N$ since otherwise the arc entering $LSA(u_i)$ would be a cut-arc and there are no cut-arcs above $u_i$.) Consider the trinet $P$ on $\{x, y, z\}$. Let $a$ be the cut-arc in $P$ such that $x$ is below $a$, such that $y$ and $z$ are not below $a$ and such that there is no cut-arc $a'$ with this property with $a$ below $a'$. Arc $a$ corresponds to arc $(u_i, v_i)$ of $N$. To see this, note that $a$ cannot be above $LSA(u_i)$ because $z$ is below $LSA(u_i)$. Moreover, $a$ cannot be between $LSA(u_i)$ and $u_i$ because then it would not be a cut-arc. Consider the network $P_x$ obtained from $P$ by deleting all vertices that are not below $a$. Then, $P_x = N_i = N'_i$.

Now suppose that $|X_i| \geq 2$. Let $z \in X \setminus X_i$ such that $LSA(u_i) \leq_N z$ and let $x, y \in X_i$ such that $LSA(x, y) = LSA(X_i)$ (such $x, y$ exist by Observation 1). Consider the trinet $P$ on $\{x, y, z\}$. Consider the cut-arc $a$ of $P$ such that $x$ and $y$ are below $a$, $z$ is not below $a$ and such that there is no cut-arc $a'$ with this property with $a$ below $a'$. Then, for similar reasons as in the previous paragraph, $a$ corresponds to arc $(u_i, v_i)$ of $N$. Let $D$ be the directed graph obtained from $P$ by deleting all vertices that are not below $a$ and deleting all vertices that are below $LSA(x, y)$. Then, $D$ is isomorphic to the strongly redundant biconnected components of $N_i$ and of $N'_i$. Now,

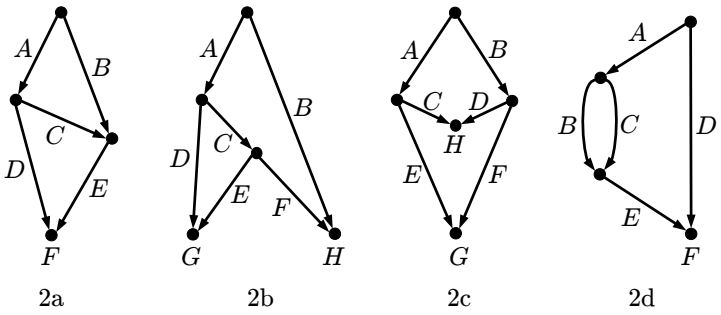


**Fig. 5** The four level-2 generators. Each side is labelled by a capital letter.

since $R_i = R'_i$ and $N_i$ and $N'_i$ have the same strongly redundant biconnected components, in the same order, as required. □

## 4 Trinets encode level-2 networks

In this section we show that binary recoverable level-2 networks are encoded by their trinets. To do this, we will consider each biconnected component of such a network separately, and will apply some structural results concerning these components that were presented by van Iersel et al. (2009a). Throughout the section, we restrict to binary networks.

We begin by recalling some relevant definitions. A level-$k$ phylogenetic network is called a *simple level-$k$* network if it contains one nontrivial biconnected component $B$ containing exactly $k$ reticulations and no cut-arcs other than the ones leaving $B$ (see the left of Figure 6 for an example of a simple level-2 network). A (binary) level-$k$ *generator* is a directed acyclic biconnected multigraph with exactly $k$ reticulations with indegree 2 and outdegree at most 1, a single vertex with indegree 0 and outdegree 2, and apart from that only vertices with indegree 1 and outdegree 2. The arcs and outdegree-0 vertices of a generator are called its *sides*. For example, all level-2 generators are depicted in Figure 5.

Note that deleting all leaves of a simple level-$k$ network $N$ gives a level-$k$ generator $G_N$. We call $G_N$ the *underlying generator* of $N$. Conversely, $N$ can be reconstructed from $G_N$ by "hanging leaves" from the sides of $G_N$ as follows (van Iersel et al., 2009a):

- for each arc $a$ of $G_N$, replace $a$ by a directed path with $\ell \geq 0$ internal vertices $v_1, \ldots, v_\ell$ and, for each such internal vertex $v_i$, add a leaf $x_i \in X$ and an arc $(v_i, x_i)$; and
- for each indegree-2 outdegree-0 vertex $v$, add a leaf $x \in X$ and an arc $(v, x)$.

We say that a leaf $x$ "is on side" $s$ if it is hung on side $s$ in this construction of $N$ from $G_N$. More precisely, for a leaf $x \in X$ of a simple level-$k$ network $N$ with underlying generator $G_N$ and a side $s$ of $G_N$, we say that $x$ *is on side* $s$ if $s$ is an indegree-2 outdegree-0 vertex of $G_N$ and $(s, x)$ is an edge of $N$ or if $s$ is an edge $(u, v)$ of $G_N$ and the parent of $x$ in $N$ lies on the directed path from $u$ to $v$ in $N$.

Now, given a level-$k$ generator $G$, we call a set of sides of $G$ a *set of crucial sides* if it contains all vertices with indegree 2 and outdegree 0 together with at least one arc of each pair of parallel arcs. Consider any simple level-$k$ network $N$ on $X$ with underlying generator $G$ and a trinet $P$ on $X' \subseteq X$. We say that $P$ is a *crucial trinet* of $N$ if $X'$ contains at least one leaf on each side in

some set of crucial sides of $G$. For example, Figure 6 depicts a simple level-2 network, one crucial trinet and two non-crucial trinets. The following observation can be verified by inspecting all level-2 generators in Figure 5.

**Observation 7** *If $G$ is a level-2 generator, then it has a set of crucial sides of size at most 2. Hence, every simple level-2 network $N$ has at least one crucial trinet. Moreover, for every leaf $x$ of $N$, there exists a crucial trinet of $N$ containing $x$.*

Note that for $k \geq 4$ there exist level-$k$ networks that have no crucial trinets, e.g. networks that have at least four leaves whose parents are reticulations.

Before proving the main result of this section, we prove one other useful fact.

**Lemma 1** *Let $N$ be a simple level-$k$ network, $G$ its underlying generator and $P \in Tn(N)$. Then, $P$ is a crucial trinet of $N$ if and only if $P$ is a simple level-$k$ network. Moreover, if $P$ is a crucial trinet of $N$ then $G$ is its underlying generator.*

*Proof* First assume that $P$ is a crucial trinet. Then, if $P$ is obtained from $N$ as in the construction in Observation 2, no reticulations are deleted because $P$ contains a leaf below each reticulation. Moreover, no parallel arcs are suppressed because $P$ contains a leaf for each set of parallel arcs of $G$. Hence, $P$ is, like $N$, a simple level-$k$ network with underlying generator $G$.

Now assume that $P$ is not a crucial trinet. Then there are two cases. The first case is that there is some leaf that is the child of a reticulation in $N$ but that is not contained in $P$. Then, if $P$ is obtained from $N$ as in the construction in Observation 2, at least one reticulation is deleted. The second case is that there is some pair of parallel arcs in $G$ for which $P$ does not contain a leaf that is on any of the corresponding sides of $N$. Then parallel arcs are suppressed in obtaining $P$ from $N$. In both cases, the reticulation number of $P$ is strictly smaller than the reticulation number of $N$. It follows that $P$ is not a simple level-$k$ network because such networks have exactly $k$ reticulations. □

**Theorem 3** *Every binary, simple level-2 network on $X$, with $|X| \geq 3$, is encoded by its trinets.*

*Proof* Let $N$ be any binary, simple level-2 network on $X$, with $|X| \geq 3$. Assume that $Tn(N') = Tn(N)$ for some recoverable network $N'$. We will show that $N' = N$.

We begin by showing that $N'$ is a binary, simple, level-2 network. First, $N'$ is simple network because its set of CA-sets equals the set of CA-sets of $N$ by Theorem 1 and it has no redundant biconnected components for similar reasons as in the proof of Theorem 2. Second, observe that any simple level-$k$ network with $k > 2$ has a level-$k'$ trinet with $k' > 2$. (If there are at least three leaves whose parent is a reticulation, take three such leaves. Otherwise, take all leaves whose parent is a reticulation and take the remaining leaves on sides that form parallel arcs in the underlying generator of $N$, choosing at most one leaf per pair of parallel arcs. The trinet on the chosen three leaves has at least 3 reticulations.) It follows that $N'$ is a level-2 network since $Tn(N') = Tn(N)$ contains only level-2 trinets. Third, $N'$ is binary. Indeed, assume that $N'$ has a vertex with outdegree greater than 2 and let $c_1, c_2, c_3$ be three of its children. Then, consider three (not necessarily different) leaves $x_1, x_2$ and $x_3$ below $c_1, c_2$ and $c_3$ respectively. Then, any trinet containing $x_1, x_2$ and $x_3$ exhibited by $N'$ is not binary, while all trinets in $Tn(N') = Tn(N)$ are binary.

Now, let $G$ be the underlying generator of $N$. First, we show that $G$ is also the underlying generator of $N'$. By Observation 7, $N$ has at least one crucial trinet $P_c$. By Lemma 1, $P_c$ is a

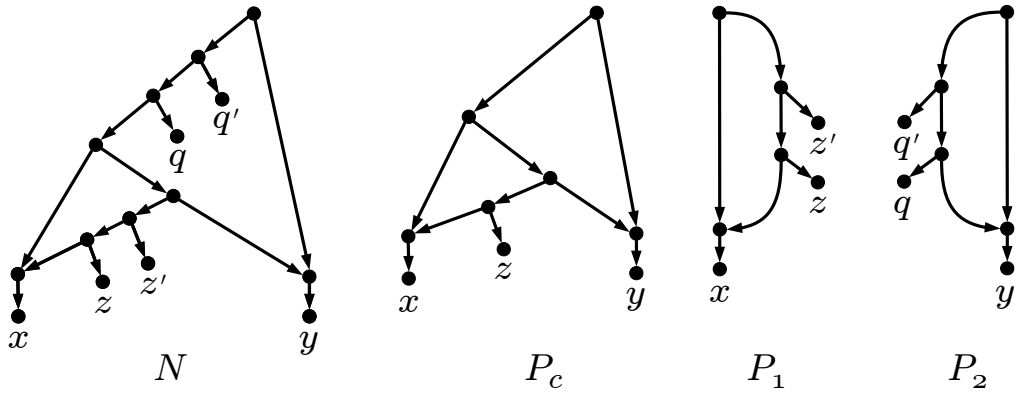


**Fig. 6** The underlying generator of simple level-2 network $N$ is $2b$ (see Figure 5). Leaf $x$ is on side $G$, $y$ is on side $H$, $z$ and $z'$ are on side $E$ and $q$ and $q'$ are on side $A$. Trinet $P_c$, the trinet on $\{x, y, z\}$, is one of the four crucial trinets, which determine the side each leaf is on. Trinet $P_1$ on $\{x, z, z'\}$ and trinet $P_2$ on $\{y, q, q'\}$, which are non-crucial trinets, determine the order of the leaves on each side. This is Case "$G = 2b$" in the proof of Lemma 3.

simple level-2 network and its underlying generator is $G$. Since $Tn(N) = Tn(N')$, $P_c$ is also a trinet of $N'$. Moreover, $P_c$ is a crucial trinet of $N'$ by Lemma 1 because $N'$ is a simple level-2 network and $P_c$ is a simple level-2 network. Hence, $G$ is the underlying generator of $N'$, again by Lemma 1.

The remainder of the proof is divided into four cases, based on the four level-2 generators $2a$, $2b$, $2c$ and $2d$ (see Figure 5 for these generators and the labels of their sides).

Case $G = 2a$. First, observe that there are no symmetries, i.e. no relabelling of the sides of $2a$ gives an isomorphic generator. Let $x$ be the leaf on side $F$ in $N$. Since $x$ is then the leaf on side $F$ in every crucial trinet of $N$, and since these crucial trinets are exhibited by $N'$, and since there are no symmetries, it follows that $x$ is also the leaf on side $F$ in $N'$. Now consider any side $s \neq F$ of $N$ and any leaf $y$ on that side. Consider any crucial trinet $P_c$ of $N$ containing $y$. Then $y$ is on side $s$ in $P_c$ and, since $P_c$ is exhibited by $N'$ and there are no symmetries, $y$ is on side $s$ in $N'$. Hence, each leaf is on the same side in $N'$ as it is in $N$. It remains to show that the leaves on each side are in the same order in $N$ and $N'$. Consider a side $s$ with at least two leaves and two leaves $y, z$ on that side such that $z$ is below $y$. It follows that $z$ is below $y$ in the crucial trinet on $\{x, y, z\}$ and from that it follows that $z$ is below $y$ in $N'$. We conclude that $N' = N$ since both networks have the same underlying generator, the same leaves on each side, and the same order of the leaves on each side.

Case $G = 2b$. Again, there are no symmetries. Let $x$ be the leaf on side $G$, $y$ the leaf on side $H$ and $z$ a leaf on some other side $s$ (see Figure 6). Then, the trinet $P_c$ on $\{x, y, z\}$ is crucial and, since there are no symmetries, it follows that leaves $x, y, z$ are, respectively, on sides $G, H, s$ in $P_c$ and hence in $N'$. Consequently, all leaves are on the same side in $N'$ as in $N$. To see that they are in the same order, first consider two leaves $z, z'$ that are both on side $C, D$ or $E$ and consider the (non-crucial) trinet $P_1$ on $\{x, z, z'\}$. Observe that $P_1$ is a simple level-1 network and that $z$ and $z'$ are on the same side of $P_1$. Moreover, if $z$ is below $z'$ in $N$, then $z$ is below $z'$ in $P_1$, and hence $z$ is below $z'$ in $N'$. Now consider leaves $q, q'$ both on side $A, B$ or $F$. Then the trinet $P_2$ on $\{y, q, q'\}$ is a simple level-1 network and, as before, if $q$ is below $q'$ in $N$, then $q$ is below $q'$ in $P_2$ and hence in $N'$. It follows that $N = N'$ as required.

Case $G = 2c$. In this case there is some symmetry since sides $A, C$ and $E$ can be interchanged with $B, D$ and $F$, respectively, to obtain an isomorphic generator. Similarly, sides $C, H, D$ can be interchanged with $E, G, F$, respectively, again yielding an isomorphic generator. Let $x$ be on side $G$, $y$ on side $H$ and $z$ on some other side $s$ in $N$. Then, the crucial trinet $P_c$ on $\{x, y, z\}$ implies that $x$ and $y$ are on side $G$ and $H$ in $N'$. Assume without loss of generality that $x$ is on side $G$ and $y$ on side $H$ in $N'$. Then, again using trinet $P_c$, it follows that $z$ is on side $A$ or $B$ in $N'$ if it is on side $A$ or $B$ in $N$. Similarly, $z$ is on side $C$ or $D$ in $N'$ if it is on side $C$ or $D$ in $N$ and $z$ is on side $E$ or $F$ in $N'$ if it is on side $E$ or $F$ in $N$.

Now, consider two leaves $z, z'$ that are both on side $A, B, C$ or $D$ of $N$. In view of the trinet on $\{y, z, z'\}$, $z$ and $z'$ are also on the same side of $N'$ and in the same order as in $N$. Similarly, for two leaves $z, z'$ that are both on side $E$ or $F$. Also, the trinet on $\{x, z, z'\}$ implies that $z$ and $z'$ are on the same side of $N'$ and in the same order. Thus, leaves that are on the same side in $N$ are on the same side in $N'$ and in the same order. First assume that there is at least one leaf on side $A$ in $N$ and that the leaves that are on side $A$ in $N$ are on side $A$ in $N'$. Let $a$ be one such leaf on side $A$. Then, any leaf $c$ that is on side $C$ in $N$ is on side $C$ in $N'$ by the trinet on $\{a, c, y\}$ (because $a$ and $c$ are on the same side of this trinet, which is a simple level-1 network). Similarly, for leaf $z$ on side $s \in \{B, D\}$ in $N$ holds that $z$ is on side $s$ in $N'$ by the trinet on $\{a, z, y\}$ and for leaf $z$ on side $s \in \{E, F\}$ in $N$ holds that $z$ is on side $s$ in $N'$ by the trinet on $\{a, z, x\}$. It follows that $N = N'$ because all leaves are on the same side, in the same order. Now assume that the leaves that are on side $A$ in $N$ are not on side $A$ in $N'$. Then these leaves are on side $B$ in $N'$. Then we can argue in exactly the same way that the leaves that are on sides $B, C, D, E, F$ in $N$ are on sides $A, D, C, F, E$ in $N'$. Hence, again $N = N'$ by relabelling the sides appropriately. Finally, if there is no leaf on side $A$, then there is a leaf on one of the sides $B, C, D, E, F$ (since $|X| \geq 3$) and we can apply similar arguments based on that leaf.

Case $G = 2d$. In this case, the only symmetry is that sides $B$ and $C$ can be interchanged with $C$ and $B$, respectively. Let $x$ be the leaf on side $F$, $y$ a leaf on side $B$ or $C$ and $z$ a leaf on some side $s \in \{A, B, C, D, E\}$ in $N$. Note that there exists at least one such leaf $z$ since $|X| \geq 3$. Then, by the crucial trinet on $\{x, y, z\}$, $x$ is on side $F$ and $y$ is on side $B$ or $C$. Without loss of generality, $y$ is on the same side in $N'$ as in $N$. So it follows that $z$ is on side $s$ in $N'$. Hence, without loss of generality (i.e. by relabelling sides $B$ and $C$ if necessary), each leaf is on the same side in $N'$ as in $N$. Now consider two leaves $z, z'$ that are on the same side of $N$. Then the trinet on $\{x, z, z'\}$ implies that the order of $z$ and $z'$ is the same in $N'$ as in $N$. We can conclude that $N' = N$, since (after possibly relabelling sides $B$ and $C$) both networks have the same leaves on the same sides in the same order. □

**Corollary 1** *Every binary recoverable level-2 network $N$ on $X$, with $|X| \geq 3$, is encoded by its set of trinets $Tn(N)$.*

*Proof* Follows from Theorem 2, Lemma 3 and the fact that level-1 networks are encoded by their trinets (Huber and Moulton, 2012). □

## 5 Trinets encode tree-child networks

In this section we show that tree-child networks are encoded by their trinets. We begin by presenting a definition, a lemma, and an observation. A directed path in a network is called a *tree path* if it does not contain any reticulations apart from possibly its first vertex. It is easily

seen that from every vertex of a tree-child network there is a directed tree path that ends at some leaf.

**Lemma 2** *Suppose that a network $N$ has an arc $(u, v)$ such that $v$ is a reticulation and such that there is no directed path from $u$ to the other parent of $v$. Suppose that there are tree paths from $u$ to a leaf $x$ and from $v$ to a leaf $y$. Then, $x$ and $y$ are distinct and $u$ is their unique lowest common ancestor in $N$.*

*Proof* Let $P_u^x$ be the tree path from $u$ to $x$ and $P_v^y$ the tree-path from $v$ to $y$. Then $P_u^x$ and $P_v^y$ cannot intersect since they are tree paths. Hence, $x$ and $y$ are distinct. Moreover, $u$ is clearly a common ancestor of $x$ and $y$. Let $w \neq u$ be any common ancestor of $x$ and $y$. Then it remains to prove that $u$ is below $w$. Since $w$ is an ancestor of $x$, there exists a directed path $P_w^x$ from $w$ to $x$. Since $P_u^x$ is a tree path, $P_w^x$ is either completely contained in $P_u^x$ or $P_w^x$ joins $P_u^x$ in $u$. In the latter case, we are done. Hence, assume that $P_w^x$ is completely contained in $P_u^x$. Then $w$ lies on $P_u^x$. However, there is also a directed path $P_w^y$ from $w$ to $y$. This path can join $P_v^y$ only in $v$ because $P_v^y$ is a tree path. Moreover, since $P_w^y$ cannot pass through $u$ (it would lead to a directed cycle), it passes through the other parent of $v$. However, that means that there is a directed path from $u$ to the other parent of $v$. □

**Observation 8** *Suppose that a network $N$ contains a tree path from a reticulation $r$ to a leaf $x$. Then, $r$ is the only reticulation with a tree path to $x$. Moreover, suppose that $p_1$ and $p_2$ are the parents of $r$ and that there is a directed path from $p_1$ to $p_2$ and a tree-path from $p_1$ to a leaf $y$. In addition, suppose that $P$ is a trinet exhibited by $N$ that contains $x$ and $y$. Then, $P$ contains $r$ and a tree path from $r$ to $x$.*

Notice that, in Observation 8, the presence of leaf $y$ in trinet $P$ ensures that, in the process of obtaining $P$ from $N$, the incoming arcs of $r$ do not become parallel arcs, which would have to be suppressed. We are now ready to prove the main result of this section.

**Theorem 4** *Every binary tree-child network $N$ on $X$, with $|X| \geq 3$, is encoded by its set of trinets $Tn(N)$.*

*Proof* The proof is by induction on the level $k$ of the network. The induction basis for $k = 1$ has been shown by Huber and Moulton (2012).

Let $k \geq 2$ and assume that every binary, tree-child, level-$(k-1)$ network with at least three leaves is encoded by its trinets. Let $N$ be a binary tree-child level-$k$ network on $X$, with $|X| \geq 3$, and let $\mathcal{T} = Tn(N)$. If $|X| = 3$, the theorem is obviously true, hence we can assume $|X| \geq 4$. By Theorem 2, we may assume that $N$ is a simple level-$k$ network, i.e. it has a single nontrivial biconnected component $B$ and no cut-arcs except for the ones leaving $B$. Consequently, for each cut-arc $(u, v)$ of $N$, the vertex $v$ is a leaf.

Now, let $N'$ be any recoverable network on $X$ exhibiting $\mathcal{T}$. We will show that $N' = N$. Suppose that $x$ is a leaf of $N$ at maximum distance from the root.

We first claim that the parent of $x$ is a reticulation $r$ such that there is no arc between the parents of $r$. To see this, first assume that $r$ is not a reticulation. Then it has some other child $s$, which must be a leaf because otherwise there would be a tree-path from $s$ to some leaf at greater distance from the root than $x$. However, in that case, the arc entering $r$ would be a cut-arc, which is not possible because $N$ is a simple level-$k$ network. Hence, $r$ is a reticulation. Now let $p_1$ and $p_2$ be the parents of $r$ and assume that there is an arc $(p_1, p_2)$. Since $p_2$ is not a reticulation by the tree-child property, it has a second child $s$. Observe that, by the tree-child property, $s$

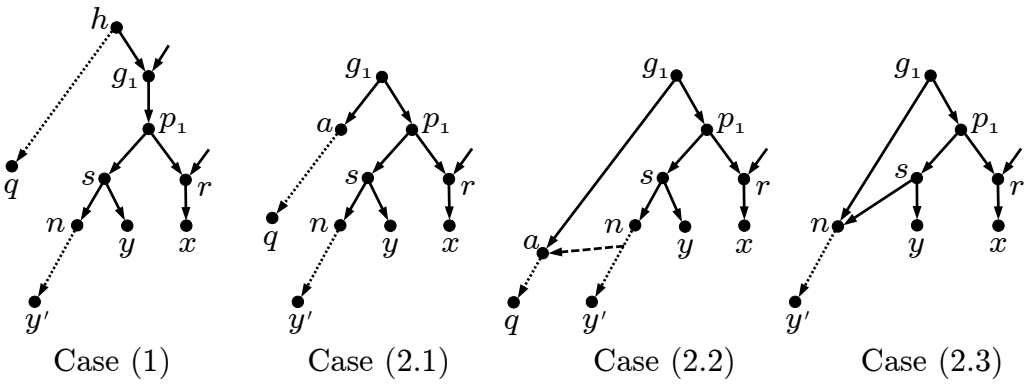


**Fig. 7** Illustration of the main cases in the proof of Theorem 4. Dotted arcs denote tree paths (directed paths not passing through reticulations). The dashed arc from below $n$ to $a$ in Case (2.2) denotes a directed path that might contain reticulations. Case (3) is very similar to Case (1).

cannot be a reticulation. Moreover, if $s$ is not a leaf then it has two children, which must be leaves because $x$ has maximum distance from the root. But, this is not possible because then the arc entering $s$ would be a cut-arc. Hence, $s$ is a leaf. However, then there are only two leaves below $p_1$. Since $|X| \geq 4$, there must be some arc entering $p_1$ and this is a cut-arc. This is again a contradiction to the fact that $N$ is a simple level-$k$ network. Thus, we conclude that the parent $r$ of $x$ is a reticulation and that there is no arc between the parents of $r$, as claimed.

Now, let $\mathcal{T}^*$ be the result of removing all trinets containing $x$ from $\mathcal{T}$ and let $N^*$ be the result of removing $x$ from $N$ and "cleaning up" the network by repeatedly deleting unlabelled outdegree-0 vertices and indegree-0 outdegree-1 vertices and suppressing indegree-1 outdegree-1 vertices, until a valid network is obtained. Note that it is not necessary to suppress parallel arcs and redundant biconnected components because these cannot arise by the described modifications since $N$ is a tree-child network. Moreover, it can easily be seen that $N^*$ is again a tree-child network and has level $k-1$. Since $N$ has at least four leaves, $N^*$ has at least three leaves. Hence, by induction, $N^*$ is encoded by its trinets. It follows that removing $x$ from $N'$, and cleaning up in the same way as we did in $N$, also gives $N^*$. Hence it only remains to show that the location of $x$ in $N$ and $N'$ is the same.

To this end, consider again the reticulation $r$ in $N$ of which $x$ is the child and the parents $p_1$ and $p_2$ of $r$ in $N$ and in $N'$. We need to show that the location of $p_1$ and $p_2$ is the same in $N$ and $N'$. We consider $p_1$ and note that the same arguments can be applied to $p_2$. By the tree-child property, there is a tree path in $N$ from $p_1$ to a leaf $y$. Thus, $p_1$ has outdegree 2 and hence it is not a reticulation. We distinguish three cases: (1) the parent $g_1$ of $p_1$ in $N$ is a reticulation, (2) the parent $g_1$ of $p_1$ in $N$ has outdegree 2, and (3) $p_1$ is the root $\rho$ of $N$. See Figure 7 for some illustrations of these cases.

Case (1): The parent $g_1$ of $p_1$ in $N$ is a reticulation. Let $h$ be a parent of $g_1$ such that there exists no directed path from $h$ to the other parent of $g_1$. Then there is a tree path from $h$ to some leaf $q$. Observe that $q$ and $y$ must be distinct and that $h$ must be their unique lowest common ancestor in $N$ by Lemma 2. Moreover, the same holds in $N^*$ and consequently in $N'$ because removing leaf $x$ and cleaning up as specified does not affect lowest common ancestors of other leaves. Consider the trinet $P_{xyq} \in \mathcal{T}$ on $\{x, y, q\}$. In $P_{xyq}$, $h$ is also the unique lowest common ancestor of $q$ and $y$ by Observation 3. Moreover, in $P_{xyq}$ also, $x$ is below the reticulation-child

(which we can also call $g_1$) of $h$. This means that, because $N'$ exhibits $P_{xyq}$, $p_1$ is below $g_1$ in $N'$. We will show that $p_1$ is in fact the child of $g_1$ in $N'$ (just as it is in $N$).

Let $s$ be the child of $p_1$ other than $r$, in $N$. Note that $s$ cannot be a reticulation by the tree-child property. If $s$ is a leaf, then $s = y$ is the child of $g_1$ in $N^*$ and, since we know that $p_1$ is below $g_1$ in $N'$, $p_1$ can only be the child of $g_1$ in $N'$. Now suppose that $s$ is not a leaf in $N$. Then it has two children, and one of these children is $y$ because otherwise $y$ would be at greater distance from the root than $x$. Let $n$ be the other child of $x$. Note that $n$ may or may not be a reticulation but that $n$ cannot be equal to $r$ because there is no arc between the parents of $r$ (by the choice of $x$). In either case, there is a tree path from $n$ to some other leaf $y'$. If $n$ is not a reticulation, then it is clear that $s$ is the unique lowest common ancestor of $y$ and $y'$ in $N$, in $N'$ and in $N^*$. If $n$ is a reticulation, then this follows from Lemma 2. In view of the trinet on $\{y, y', x\}$, it follows that $LCA(x, y) \leq_{N'} LCA(y, y')$, i.e. $p_1 \leq_{N'} s$. Since $s$ is below $p_1$ and we already know that $p_1$ is below $g_1$, we conclude that $p_1$ is the child of $g_1$ in $N'$, as required.

Case (2): The parent $g_1$ of $p_1$ has outdegree 2. Then $g_1$ has some child $a$ other than $p_1$. Note that $a$ cannot be equal to $r$ because there is no arc between the parents of $r$. From $a$ there is a tree path to some leaf $q$. As before, let $s$ be the child of $p_1$ other than $r$, in $N$. There is again a tree path from $s$ to some leaf $y$. If $s$ is a leaf, then again $s = y$ and, in view of the trinet on $\{x, y, q\}$, $p_1$ is the parent of $y$ in $N'$. Now we distinguish three subcases: (2.1) $a \neq n$ and there is no directed path from $s$ to $a$, (2.2) $a \neq n$ and there *is* a directed path from $s$ to $a$ and (2.3) $a = n$ (and consequently there is no directed path from $s$ to $a$).

In Case (2.1), $q$ and $y$ are distinct. If $a$ is not a reticulation, then $g_1$ is clearly the unique lowest common ancestor of $q$ and $y$. If $a$ is a reticulation, then $g_1$ is the unique lowest common ancestor of $q$ and $y$ by Lemma 2. We can now use similar arguments as in Case (1) to show that, in $N'$, $p_1$ is the child of $g_1$ on the directed path to $y$.

In Case (2.2), $a$ is the unique reticulation from which there is a tree path to $q$ and $g_1$ is its parent from which there is a directed path to the other parent, in $N, N^*, N'$ and in the trinet on $\{q, y, x\}$, by Observation 8 applied to reticulation $a$. In view of this trinet, $p_1$ is below $g_1$, on the directed path to $y$. We can now use similar arguments as in Case (1) to show that, in $N'$, $p_1$ is the child of $g_1$ on the directed path to $y$.

In Case (2.3), $s$ is the unique lowest common ancestor of $y$ and $y'$ in $N, N^*, N'$ and in any trinet containing $y, y'$ by Observation 3 and Lemma 2. Also, $g_1$ is the parent of $s$ in $N^*$. In view of the trinet on $\{y, y', x\}$, $p_1$ has to be the parent of $s = LCA(y, y')$ in $N'$. Thus, the location of $p_1$ is the same in $N$ and $N'$.

Case (3): $p_1$ is the root $\rho$ of $N$. We define $s, n, y, y'$ as before. In this case, $s$ is the root of $N^*$. Hence, $s$ cannot be a leaf since $|X| \geq 3$. We can argue as in Case (1), concluding that $p_1$ is the root of $N'$.

After applying exactly the same arguments to $p_2$ as we did to $p_1$, it follows that, in all cases, the location of $p_1$ and $p_2$ is the same in $N$ and $N'$. Hence, the location of $x$ is the same in $N$ and $N'$. It follows that $N = N'$. □

## 6 Discussion

We have proven that binary, recoverable level-2 and binary tree-child networks are encoded by their trinets, using two distinct methods of proof. We expect that our results could also hold for non-binary networks, and it would be of interest to verify this.

For settling the question if all recoverable phylogenetic networks are encoded by their trinets, the decomposition theorems in Section 3 will be useful since they essentially show that it is sufficient to answer this question for simple networks (i.e. networks having no cut-arcs apart from pendant arcs). It would be interesting to investigate whether there are constructive results for recovering these decompositions and, if so, to try to develop algorithms for their computation.

The proof for level-2 networks might be extended to show that higher level networks are encoded by their trinets (or be used to provide a counter-example). However, a new technique would have to be developed for $k \geq 4$ since, for such $k$, there exist level-$k$ networks that have no crucial trinets. Another difficulty is that the number of generators for level-$k$ networks grows very rapidly (the number of level-$k$ generators is at least $2^{k-1}$ (Gambette et al., 2009)) making a similar case analysis impossible in general. To prove that tree-child networks are encoded by trinets, we heavily depended on special properties of such networks, and we have not been able to find a way to extend our proof to even slightly more general networks (e.g. reticulation-visible networks (Huson et al., 2011)).

We note that a natural extension to the definition of "exhibit" is to define it as in Observation 2 but to suppress not only *strongly* redundant biconnected components, but *all* redundant biconnected components. If one then changes the definition of "recoverable" accordingly (i.e. to not having any redundant biconnected components), then it can be checked that all proofs in this paper still hold. This could be relevant when reconstructing phylogenetic networks via trinets, because the number of recoverable trinets then becomes bounded by a function of $k$ (see the appendix).

It is also worth noting that Theorem 2, Theorem 3 and Theorem 4 can be combined to provide the following more general result.

**Corollary 2** *If $X$ is a finite set with $|X| \geq 3$ and $\mathcal{N}$ is the set of binary recoverable phylogenetic networks $N$ on $X$ for which each biconnected component of $N$ either*

- *has at most two reticulations; or*
- *is tree-child; or*
- *has at most three outgoing cut-arcs,*

*then each $N \in \mathcal{N}$ is encoded by $Tn(N)$.*

In addition, we note that our results also yield some new metrics on level-2 and tree-child networks. These are of potential intesest since several metrics have been recently developed for special classes of networks (see e.g. Cardona et al. (2008, 2009a,b, 2011); Huber and Moulton (2012)). More specifically, Corollary 2 immediately implies the following result (where $\Delta$ denotes the symmetric difference of two sets).

**Corollary 3** *If $X$ is a finite set with $|X| \geq 3$ and $\mathcal{N}$ is as in Corollary 2, then the map $d : \mathcal{N} \times \mathcal{N} \to \mathbb{R}$ defined by*

$$d(N, N') := |Tn(N) \Delta Tn(N')|,$$

*for all $N, N' \in \mathcal{N}$, is a metric on $\mathcal{N}$.*

Finally, it could be of some interest to study some algorithmic issues related to the results that we have presented. For example, it would be interesting to know whether or not it is possible to reconstruct a recoverable (level-2 or tree-child) network from a set of trinets in polynomial time. Hopefully shedding light on this and related complexity problems could help provide new algorithms for constructing phylogenetic networks.

Finally, it could be of some interest to study some algorithmic issues related to the results that we have presented. For example, it would be interesting to know whether or not it is possible to reconstruct a recoverable (level-2 or tree-child) network from a set of trinets in polynomial time. Hopefully shedding light on this and related complexity problems could help provide new algorithms for constructing phylogenetic networks. In addition, although trinets encode level-2 and tree-child networks, it will be important in practice to develop ways to deal with noisy data where some trinets might not be correct. This might involve combining trinet approaches with triplet approaches, or devising ways to identify or possibly filter out seemingly incorrect trinets. In pratice, the strategy of computing and combining trinets could become very complicated for large $k$, but it could still be of some interest to develop systematic approaches to at least construct low level networks from trinet data.

**Acknowledgements** Leo van Iersel was supported by a Veni grant of The Netherlands Organisation for Scientific Research (NWO). We are grateful to the anonymous reviewers for their useful comments.

## References

A. V. Aho, Y. Sagiv, T. G. Szymanski, and J. D. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM J. Comput.*, 10(3):405–421, 1981. ISSN 0097-5397.

M. Baroni, C. Semple, and M. Steel. A framework for representing reticulate evolution. *Ann. Comb.*, 8:391–408, 2004.

J. Byrka, S. Guillemot, and J. Jansson. New results on optimizing rooted triplets consistency. *Discrete Appl. Math.*, 158:1136–1147, 2010.

G. Cardona, M. Llabrés, F. Rosselló, and G. Valiente. A distance metric for a class of tree-sibling phylogenetic networks. *Bioinformatics*, 24:1481–1488, 2008.

G. Cardona, M. Llabrés, F. Rosselló, and G. Valiente. Metrics for phylogenetic networks I: Generalization of the robinson-foulds metric. *IEEE ACM T. Comput. Bi.*, 6:46–61, 2009a.

G. Cardona, M. Llabrés, F. Rosselló, and G. Valiente. Metrics for phylogenetic networks II: Nodal and triplets metrics. *IEEE ACM T. Comput. Bi.*, 6:454–469, 2009b.

G. Cardona, F. Rosselló, and G. Valiente. Comparison of tree-child phylogenetic networks. *IEEE ACM T. Comput. Bi.*, 6(4):552–569, 2009c.

G. Cardona, M. Llabrés, F. Rosselló, and G. Valiente. Path lengths in tree-child time consistent hybridization networks. *Inform. Sciences*, 180(3):366–383, 2010.

G. Cardona, M. Llabrés, F. Rosselló, and G. Valiente. Comparison of galled trees. *IEEE ACM T. Comput. Bi.*, 8:410–427, 2011.

A. Dress, K. T. Huber, J. Koolen, V. Moulton, and A. Spillner. *Basic Phylogenetic Combinatorics*. Cambridge University Press, 2012.

J. Fischer and D. Huson. New common ancestor problems in trees and directed acyclic graphs. *Inform. Process. Lett.*, 110:331–335, 2010.

P. Gambette and K. T. Huber. A note on encodings of phylogenetic networks of bounded level. Technical Report arXiv:0906.4324, June 2009.

P. Gambette and K. T. Huber. On encodings of phylogenetic networks of bounded level. *J. Mol. Biol.*, 65(1):157–180, 2012.

P. Gambette, V. Berry, and C. Paul. The structure of level-k phylogenetic networks. In *Proc. of Combinatorial Pattern Matching*, number 5577, pages 289–300, 2009.

P. Gambette, V. Berry, and C. Paul. Quartets and unrooted phylogenetic networks. *J. Bioinformatics Comput. Biol.*, 10(4):1250004, 2012.

M. Habib and T.-H. To. Constructing a minimum phylogenetic network from a dense triplet set. *J. Bioinformatics Comput. Biol.*, 10(5), 2012.

K. T. Huber and V. Moulton. Encoding and constructing 1-nested phylogenetic networks with trinets. *Algorithmica*, 2012. To appear.

K. T. Huber, L. J. J. van Iersel, S. M. Kelk, and R. Suchecki. A practical algorithm for reconstructing level-1 phylogenetic networks. *IEEE ACM T. Comput. Bi.*, 8(3):635–649, 2011.

D. H. Huson, R. Rupp, and C. Scornavacca. *Phylogenetic Networks: Concepts, Algorithms and Applications*. Cambridge University Press, 2011.

J. Jansson, N. B. Nguyen, and W.-K. Sung. Algorithms for combining rooted triplets into a galled phylogenetic network. *SIAM J. Comput.*, 35(5):1098–1121, 2006.

G. Jin, L. Nakhleh, S. Snir, and T. Tuller. Maximum likelihood of phylogenetic networks. *Bioinformatics*, 22:2604–2611, 2006.

G. Jin, L. Nakhleh, S. Snir, and T. Tuller. Parsimony score of phylogenetic networks: Hardness results and a linear-time heuristic. *IEEE ACM T. Comput. Bi.*, 6:495–505, 2009.

D. Morrison. *Introduction to phylogenetic networks*. RJR Productions, Uppsala, 2011.

L. Nakhleh. Evolutionary phylogenetic networks: Models and issues. In L. S. Heath and N. Ramakrishnan, editors, *Problem Solving Handbook in Computational Biology and Bioinformatics*. Springer Berlin / Heidelberg, 2011.

V. Ranwez, V. Berry, A. Criscuolo, P.-H. Fabre, S. Guillemot, C. Scornavacca, and E. J. P. Douzery. PhySIC: A veto supertree method with desirable properties. *Syst. Biol.*, 56(5): 798–817, 2007.

C. Scornavacca, V. Berry, V. Lefort, E. Douzery, and V. Ranwez. PhySIC_IST: cleaning source trees to infer more informative supertrees. *BMC Bioinformatics*, 9(1):413, 2008.

C. Semple and M. Steel. *Phylogenetics*. Oxford University Press, 2003. ISBN 0-19-850942-1.

L. J. J. van Iersel and S. M. Kelk. Constructing the simplest possible phylogenetic network from triplets. *Algorithmica*, 60(2):207–235, 2011a.

L. J. J. van Iersel and S. M. Kelk. When two trees go to war. *J. Theor. Biol.*, 269(1):245–255, 2011b.

L. J. J. van Iersel, J. C. M. Keijsper, S. M. Kelk, L. Stougie, F. Hagen, and T. Boekhout. Constructing level-2 phylogenetic networks from triplets. *IEEE ACM T. Comput. Bi.*, 6(4): 667–681, 2009a.

L. J. J. van Iersel, S. M. Kelk, and M. Mnich. Uniqueness, intractability and exact algorithms: Reflections on level-$k$ phylogenetic networks. *J. Bioinformatics Comput. Bi.*, 7(2):597–623, 2009b.

L. J. J. van Iersel, C. Semple, and M. Steel. Locating a tree in a phylogenetic network. *Inform. Process. Lett.*, 110(23):1037–1043, 2010.

S. J. Willson. Regular networks can be uniquely constructed from their trees. *IEEE ACM T. Comput. Bi.*, 8(3):785–796, 2010.

S. J. Willson. Tree-average distances on certain phylogenetic networks have their weights uniquely determined. *Algorithm Mol. Biol.*, 7(13), 2012.

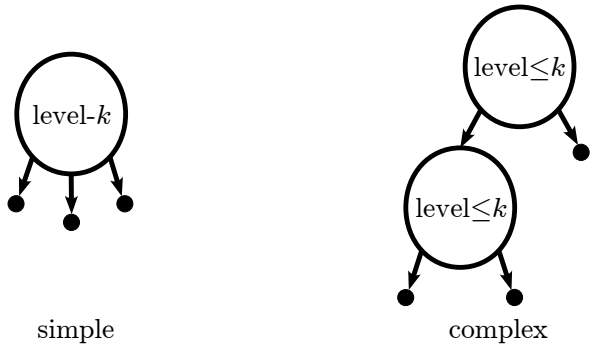


**Fig. 8** The two types of level-$k$ trinets without redundant biconnected components: simple and complex.

## Appendix

Throughout this appendix, we only consider trinets that are binary and have no redundant biconnected components, and we will not mention this restriction again. We show that the number of nonisomorphic unlabelled level-2 trinets is 220. Subsequently, we will bound the number of nonisomorphic unlabelled level-$k$ trinets.

First note that a trinet can have at most two nontrivial biconnected components. We say that a trinet is *complex* if it is not simple, i.e. if it contains some cut-arc whose head is not a leaf. Hence, there are two types of trinets: simple and complex, see Figure 8.

The number of unlabelled simple level-2 trinets is easily calculated using the four level-2 generators from Figure 5, by considering all possible ways of adding three leaves to the generators. There are 25 simple level-2 trinets of type 2a, 6 of type 2b, 2 of type 2c and 5 of type 2d and there are 2 simple level-1 trinets. Hence, there are 40 unlabelled, simple level-2 trinets.

We now turn to computing the number of unlabelled, complex level-2 trinets. We say that a network is a *binet* if it has two leaves. A complex level-2 trinet can be build from two level-2 binets, see Figure 8. For the bottom binet the leaf-labelling is irrelevant, but for the top binet the leaf labelling is important. The number of unlabelled level-2 binets is 10 (5 of type 2a, 1 of type 2b, 1 of type 2c, 1 of type 2d, 1 level-1 binet with a single reticulation and 1 level-0 binet). Furthermore, the number of leaf-labelled level-2 binets is 18 (for the binet of type 2c and for the level-0 binet the two possible leaf-labellings give isomorphic networks). Hence, the number of unlabelled complex level-2 trinets is $18 \times 10 = 180$.

We conclude that the total number of nonisomorphic unlabelled level-2 trinets is $40 + 180 = 220$.

We now extend the above calculations to level-$k$ trinets. That this number is a function of $k$ can be seen from the fact that Gambette et al. (2009) proved that the number of level-$k$ generators is a function of $k$, say $f(k)$, and the number of arcs of each level-$k$ generator is bounded by $4k$. To compute the number of simple level-$k$ trinets with exactly $k$ reticulations, for a fixed generator, first observe that the each trinet needs to have a leaf below each outdegree-0 vertex of the generator and each generator has at least one such vertex. The other (at most) two leaves can each be on one of the $4k$ arcs of the generator. Hence, for a fixed generator, the number of simple level-$k$ trinets with exactly $k$ reticulations is bounded by $(4k)^2$. The total number of simple level-$k$ trinets is then at most $16(1^2 + 2^2 + \ldots + k^2)f(k) \leq 16k^3 f(k)$.

To compute the number of complex level-$k$ trinets, we first compute the number of level-$k$ binets. All level-$k$ binets are simple, and we can use similar arguments as in the previous paragraph to show that there are at most $4k^2 f(k)$ unlabelled level-$k$ binets. The number of leaf-labelled level-$k$ binets is at most twice as large and hence at most $8k^2 f(k)$. Now we can compute the number of complex level-$k$ trinets in the same way as we computed the number of complex level-2 trinets, and their number is at most $32k^4 f(k)^2$.

It was shown by Gambette et al. (2009) that $2^{k-1} \leq f(k) \leq k!^2 50^k$. Hence, we can conclude that the total number of nonisomorphic unlabelled level-$k$ trinets is at most $16k^3 f(k) + 32k^4 f(k)^2 \leq 16k^3 k!^2 50^k + 32k^4 k!^4 50^{2k}$. Note that this upper bound might be quite crude.

Finally, note that using the systematic way of constructing level-$k$ generators given by Gambette et al. (2009), level-$k$ trinets can be constructed systematically using the above strategy.