Inferring DAGs and Phylogenetic Networks from Least Common Ancestors

Anna Lindeberg[1], Anton Alfonsson[1], Vincent Moulton[2], Guillaume E. Scholz[3], and Marc Hellmuth[1]

[1]*Department of Mathematics, Faculty of Science, Stockholm University, SE-10691 Stockholm, Sweden*
[2]*School of Computing Sciences, University of East Anglia, Norwich, NR4 7TJ, United Kingdom*
[3]*Independent Researcher, Leipzig, DE-04229, Germany*

Abstract

A least common ancestor (LCA) of two leaves in a directed acyclic graph (DAG) is a vertex that is an ancestor of both of the leaves, and for which no proper descendant of this vertex is also an ancestor of the two leaves. LCAs play a central role in representing hierarchical relationships in rooted trees and, more generally, in DAGs. In 1981, Aho et al. introduced the problem of determining whether a collection of pairwise LCA constraints on a set X, of the form $(i,j) < (k,l)$ with $i,j,k,l \in X$, can be realized by a rooted tree with leaf set X such that, whenever $(i,j) < (k,l)$ holds, the LCA of i and j in the tree is a descendant of the LCA of k and l. In particular, they presented a polynomial-time algorithm, called BUILD, to solve this problem. In many cases, however, such constraints cannot be realized by *any* tree. In these situations, it is natural to ask whether they can still be realized by a more general directed acyclic graph (DAG).

In this paper, we extend Aho. et al.'s problem from trees to DAGs, providing a theoretical and algorithmic framework for reasoning about LCA constraints in this more general setting. More specifically, given a collection R of LCA constraints, we introduce the notion of the *+-closure* R^+, which captures additional LCA relations implied by R. Using this closure, we then associate a *canonical DAG* G_R to R and show that a collection of constraints R can be realized by a DAG in terms of LCAs if and only if R is realized in this way by G_R. In addition, we adapt this construction to obtain *phylogenetic networks*, a special class of DAGs widely used in phylogenetics. In particular, we define a canonical phylogenetic network N_R for realizing R, and prove that N_R is *regular*, that is, it coincides with the Hasse diagram of the set system underlying N_R. Finally, we show that, for any DAG-realizable collection R, the *classical closure* of R, that is the collection of *all* LCA constraints that must hold in *every* DAG realizing R, coincides with its $+$-closure. All constructions introduced in this paper can be computed in polynomial time, and we provide explicit algorithms for each.

Keywords: DAG, phylogenetic network, lowest common ancestor, closure operations, incomparability, BUILD algorithm, triplets

1 Introduction

In a rooted tree, the least common ancestor of two vertices x and y, denoted $\mathrm{lca}(x,y)$, is the vertex that is an ancestor of both x and y, such that no proper descendant of this vertex is also an ancestor of both x and y. In the seminal paper [2], Aho et al. studied the following problem that is related to phylogenetic trees and relational databases. Suppose we are given a collection of constraints of the form $(i,j) < (k,l)$, with $i \neq j$, $k \neq l$, and $i,j,k,l \in X$. Can we construct a rooted tree T with leaf set X such that, for every constraint $(i,j) < (k,l)$, $\mathrm{lca}(i,j)$ is a descendant of $\mathrm{lca}(k,l)$ in T, or determine that no such tree exists? For example, the tree T as shown in Figure 1 realizes the constraints $(i,j) < (i,k)$ and $(j,k) < (j,l)$ since the vertex $\mathrm{lca}(i,j)$ is a proper descendant of $\mathrm{lca}(i,k)$, and $\mathrm{lca}(j,k)$ is a proper descendant of $\mathrm{lca}(j,l)$.

In [2] Aho et al. presented an efficient algorithm for solving this problem, called BUILD. Since the BUILD algorithm was introduced it has become an indispensable tool within phylogenetic studies, where BUILD and its variants have been used to construct trees representing the evolutionary history of genes, species, or other taxa. Several practical implementations and improvements have been developed over the years [9, 21, 23, 29]. In particular, constraints of the form $(i,j) < (k,l)$ can often be inferred directly from genomic sequence data, making such LCA-based constraints a useful approach to reconstructing phylogenetic trees [16, 17, 43].

In many cases, however, such constraints cannot be realized by any tree. For example, consider the constraints $(i,j) < (i,k)$, $(j,k) < (j,l)$, and $(j,l) < (i,k)$. It can be shown (e.g., using BUILD) that no tree can satisfy all three constraints simultaneously. Nevertheless, these constraints can be realized by the directed acyclic graph (DAG) N

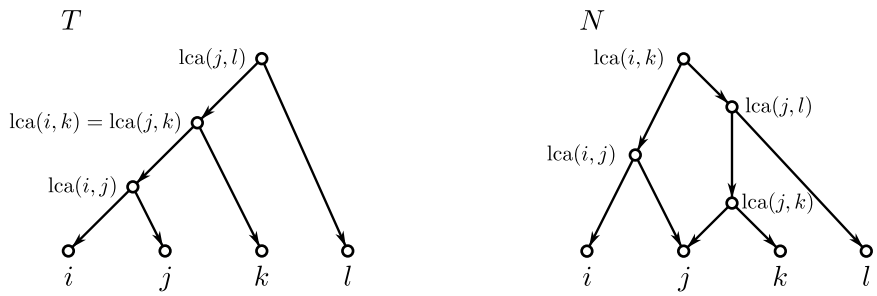

Figure 1: A tree T and a network N, with least common ancestors indicated next to each vertex. The LCA constraints $(i,j) < (i,k)$ and $(j,k) < (j,l)$ are realized by both T and N. The constraints $(i,j) < (i,k)$, $(j,k) < (j,l)$ and $(j,l) < (i,k)$ cannot be realized by any tree, but are realized by N.

illustrated in Figure 1. Surprisingly, and to the best of our knowledge, no prior work has studied LCA constraints in the broader setting of DAGs. A major challenge is that, in DAGs, the LCA of two leaves is not necessarily unique, and in some cases it may not exist at all (see Figure 2 below for an illustrative example). Nonetheless, in circumstances where these issues do not arise, it becomes an interesting problem to determine whether a DAG can realize a given collection of LCA constraints, for instance, in situations where they cannot be realized by a tree.

In this paper, we characterize LCA constraints that can be realized by DAGs and provide polynomial-time algorithms to determine their realizability and, when this is possible, to construct DAGs that realize them. In doing so, we also address several interesting combinatorial problems that arise when defining the closure of a collection of LCA constraints, that is, the collection obtained by repeatedly generating new constraints implied by the existing ones.

We now briefly outline the contents of the rest of this paper. After presenting the main definitions in the next section, Section 3 formalizes what it means for a DAG to realize a set of LCA constraints, where we introduce two main variants of realizability. The first, *strict realization*, generalizes the approach of Aho et al. [2]. Specifically, for $i,j,k,l \in X$, it assumes that the vertices $\mathrm{lca}(i,j)$ and $\mathrm{lca}(k,l)$ are both well-defined in the DAG G realizing the given collection of constraints, and that if $(i,j) < (k,l)$, then $\mathrm{lca}(i,j)$ is a descendant of $\mathrm{lca}(k,l)$ in G. The second, more general notion, *realization*, allows LCA pairs to coincide when constraints are "partially symmetric", while still respecting the partial order implied by asymmetric constraints.

We then address the problem of characterizing when a collection R of LCA constraints is realizable by some DAG. To this end, in Section 4 we introduce the *+-closure* operator for a collection R, an operator that produces a new collection R^+ that satisfies three simple rules, the most important being that R^+ is cross-consistent. Intuitively, cross-consistency captures the idea that certain "local" LCA relationships imply others. Specifically, it states that if in a DAG G we have that $\mathrm{lca}(i,k)$ and $\mathrm{lca}(j,l)$ are descendants of $\mathrm{lca}(x,y)$, then, whenever $\mathrm{lca}(i,j)$ is well-defined in G, it must also be a descendant of $\mathrm{lca}(x,y)$. The $+$-closure R^+ therefore captures additional information about any DAG G that could potentially realize R. In Section 5, using the $+$-closure, we define an equivalence relation $\sim_{R^+}$ on pairs of elements in X and define the *canonical DAG* G_R associated to R, whose vertices are the equivalence classes of $\sim_{R^+}$. In Theorem 5.10, we then show that R is realizable by a DAG if and only if it is realized by G_R, and also show that this is equivalent to R satisfying two simple axioms that are expressed in terms of R^+ (see Definition 5.1).

In Section 6, we study the problem of realizing collections of LCA constraints using *phylogenetic networks*, or simply *networks*. These are a special class of DAGs that generalize phylogenetic trees and are commonly used to represent reticulate evolution (see, e.g., [44, Chapter 10]). Several important classes of networks have the property that every internal vertex with at least two children is the LCA of at least two leaves. This includes, for example, normal networks [47] (a leading class of networks [12]), as well as regular and level-1 networks (see, e.g., [19, 20, 33] for more details on LCAs in networks). By modifying the canonical DAG G_R of a realizable collection of constraints R, we define in Section 6 a canonical *network* N_R that realizes R. We show that N_R is *regular*, meaning that is closely related to the Hasse diagram of the sets formed by taking the descendants in X of each vertex in N_R. Exploiting the fact that R^+ can be computed in polynomial time by repeatedly applying a set of implication rules (Theorem 4.5), we also present a polynomial-time algorithm for computing N_R (see Algorithm 1).

A natural way to define a closure for realizable relations is to follow the approach used for triplets or quartets in phylogenetic trees [5, 6, 32, 41]. In particular, given a realizable relation R, we define its "classical" closure, denoted by $\mathrm{cl}(R)$, to be the collection of LCA constraints that must hold in *every* DAG realizing R. In Section 7, we show the somewhat surprising result that, for any realizable R, the classical closure coincides with the $+$-closure; that is, $\mathrm{cl}(R) = R^+$. In Section 8, we extend the realizability problem by seeking a DAG that satisfies a collection of constraints R while simultaneously satisfying another set S of incomparability constraints. Finally, in Section 9, we discuss several interesting possible directions for future research.

All algorithms developed in this paper are implemented in the freely available Python package `RealLCA` [34].

2 Basics

Sets and Relations. In what follows, X will always denote a finite non-empty set. We denote with $\mathscr{P}(X)$ the powerset of X. A *set system* $\mathfrak{C}$ on X is a subset of $\mathscr{P}(X)$. A set system $\mathfrak{C}$ on X is *grounded* if $\{x\} \in \mathfrak{C}$ for all $x \in X$ and $\emptyset \notin \mathfrak{C}$, while $\mathfrak{C}$ is a *clustering system* if it is grounded and satisfies $X \in \mathfrak{C}$. We let $\mathscr{P}_2(X) := \{\{a,b\} \mid a,b \in X\}$ (with $a = b$ allowed) denote the set system consisting of all 1- and 2-element subsets of X. We will often write ab, respectively, aa for elements $\{a,b\}$, respectively, $\{a\}$ in $\mathscr{P}_2(X)$. Thus, $ab = ba$ always holds.

Given a set A, a subset $R \subseteq A \times A$ is a *binary relation (on A).* We shall often write $a\ R\ b$ instead of $(a,b) \in R$ and $a\ \not R\ b$ instead of $(a,b) \notin R$, for $a,b \in A$. In addition, we sometimes write $p\ R\ \ldots\ R\ q$ if there is a (p,q)-*chain in* R, i.e., some $a_0,\ldots,a_k$, $k \geq 1$ such that $a_0 = p\ R\ a_1\ R\ \ldots\ R\ a_{k-1} R\ a_k = q$.

Remark. *As all relations considered in this work are binary, we shall simply refer to them as* relations.

Furthermore, we define the *support* supp_R of a relation R on A as

$$\mathrm{supp}_R := \{p \in A \mid \text{ there is some } q \in A \text{ with } p\ R\ q \text{ or } q\ R\ p\},$$

that is, the subset of A that contains precisely those $p \in A$ that are in R-relation with some $q \in A$. We often consider relations R on $A = \mathscr{P}_2(X)$ in which case we extend supp_R to obtain

$$\mathrm{supp}_R^+ := \mathrm{supp}_R \cup \{xx \mid x \in X\}.$$

Example 2.1. *The elements in the relation* $R = \{(\{x\},\{a,b\}),(\{y\},\{a,b\}),(\{u,v\},\{a,b\})\} \subseteq \mathscr{P}_2(X) \times \mathscr{P}_2(X)$ *with* $X = \{x,y,a,b,u,v,w\}$ *can alternatively be written as* $xx\ R\ ab$, $yy\ R\ ab$ *and* $uv\ R\ ab$. *Moreover,* $\mathrm{supp}_R = \{xx,yy,uv,ab\}$ *and* $\mathrm{supp}_R^+ = \{xx,yy,aa,bb,uu,vv,ww,uv,ab\}$.

Let R be a relation on A. Then, R is *asymmetric* if $p\ R\ q$ implies $q\ \not R\ p$ for all $p,q \in A$, and it is *anti-symmetric* if $p\ R\ q$ and $q\ R\ p$ implies $p = q$ for all $p,q \in A$. Moreover, R is *transitive*, if $p\ R\ q$ and $q\ R\ r$ implies $p\ R\ r$ for all $p,q,r \in A$. If $B \subseteq A$, we say that R is B-*reflexive* if $(b,b) \in R$ for every $b \in B$. A relation R on A is *reflexive* if it is A-reflexive.

A key concept in this paper is that of closure operators, see e.g. [6, 7, 41]. A *closure operator* on a set S is a map $\phi\colon \mathscr{P}(S) \to \mathscr{P}(S)$ that satisfies the following three properties for all $R,R' \in \mathscr{P}(S)$:

Extensivity: $R \subseteq \phi(R)$.
Monotonicity: $R \subseteq R'$ implies $\phi(R) \subseteq \phi(R')$,
Idempotency: $\phi(\phi(R)) = \phi(R)$.

We let $\mathrm{tc}(R)$ denote the *transitive closure* of a relation R, that is, the relation where $(p,q) \in \mathrm{tc}(R)$ if and only if there is a (p,q)-chain in R, or equivalently, the inclusion-minimal relation that is transitive and that contains R (see e.g. [36, p.39]). It is straightforward to verify that tc is indeed a closure operator on $S = A \times A$ for all relations R on A.

Graphs. A *directed graph* $G = (V,E)$ is a tuple with non-empty vertex set $V(G) := V$ and arc set $E(G) \subseteq V \times V$. A *subgraph* of G is a directed graph $H = (V',E')$ such that $V' \subseteq V$ and $E' \subseteq E$. We put $\mathrm{outdeg}_G(v) := |\{u \in V\colon (v,u) \in E\}|$ and $\mathrm{indeg}_G(v) := |\{u \in V\colon (u,v) \in E\}|$ to denote the *out-degree* and *in-degree* of a vertex v, respectively. A vertex v with $\mathrm{outdeg}_G(v) = 0$ is a *leaf* of G and a vertex v with $\mathrm{indeg}_G(v) = 0$ is a *root* of G. A *hybrid* is a vertex v with $\mathrm{indeg}_G(v) \geq 2$. Note that leaves might also be hybrids. A directed graph G is *phylogenetic* if it does not contain a vertex v such that $\mathrm{outdeg}_G(v) = 1$ and $\mathrm{indeg}_G(v) \leq 1$, i.e., a vertex with only one outgoing arc and at most one incoming arc. We sometimes use $u \to v$ to denote the arc $(u,v) \in E(G)$ and $u \rightsquigarrow v$ to denote a directed uv-path in G.

Directed graphs G without directed cycles are called *directed acyclic graphs (DAGs)* [39]. In particular, DAGs do not contain arcs of the form (v,v). Let G be a DAG. We write $v \preceq_G w$ if and only if there is a directed wv-path in G. If $v \preceq_G w$ and $v \neq w$, we write $v \prec_G w$. If $u \to v$ is an arc in G, then $v \prec_G u$ and we call v a *child* of u and u a *parent* of v. In a DAG G where $u \preceq_G v$, we call u a *descendant* of v and v an *ancestor* of u. If $u \preceq_G v$ or $v \preceq_G u$, then u and v are $\preceq_G$-*comparable* and, otherwise, $\preceq_G$-*incomparable*.

For directed graphs $G = (V_G,E_G)$ and $H = (V_H,E_H)$, an *isomorphism between G and H* is a bijective map $\varphi\colon V_G \to V_H$ such that $(u,v) \in E_G$ if and only if $(\varphi(u),\varphi(v)) \in E_H$. If such a map exist, then G and H are *isomorphic*, in symbols $G \simeq H$.

An arc $e = (u,w)$ in a DAG G is a *shortcut* if there is a directed uw-path that does not contain the arc e [10, 35]. A DAG without shortcuts is *shortcut-free*. Throughout the paper, we will often use the "shortcut-free" version of a DAG G as defined next, which is also known as transitive reduction of G (see e.g. [1, p.133]).

Definition 2.2. *Let G be a DAG on X. Then, G^- denotes the DAG obtained from G by removal of all of its shortcuts.*

If G is a DAG whose set of leaves is X, then G is a *DAG on* X. Moreover, if a DAG G has a single root, then it is called a *network*. A *(rooted) tree* is a network that does not contain vertices with $\mathrm{indeg}_G(v) > 1$. A *star-tree* on X is a tree with a single root ρ and arcs $\rho \to x$ for all $x \in X$. In case G is a DAG on X, we have the following useful result concerning G^- from [33, L. 2.5].

Lemma 2.3. *Let G be a DAG on X. Then, G^- is a DAG on X that is uniquely determined and shortcut-free. Moreover, $V(G) = V(G^-)$ and, for all $u,v \in V(G)$, we have $u \preceq_G v$ if and only if $u \preceq_{G^-} v$.*

Least common ancestors. For a given DAG G on X and a non-empty subset $A \subseteq X$, a vertex $v \in V(G)$ is a *common ancestor of* A if v is an ancestor of every vertex in A. Moreover, v is a *least common ancestor* (LCA) of A if v is a $\preceq_G$-minimal vertex that is an ancestor of all vertices in A. The set $\mathrm{LCA}_G(A)$ comprises all LCAs of A in G.

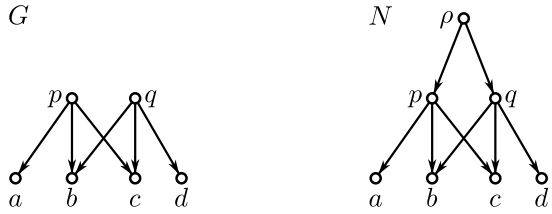

Figure 2: A DAG G and a network N, both having leaf-set $X = \{a,b,c,d\}$. Note that removing the vertex ρ and its incident arcs from N yields the DAG G. In N, we have $\mathrm{LCA}_N(\{x,y\}) \neq \emptyset$ for all $x,y \in X$. Moreover, $\mathrm{LCA}_G(\{a,d\}) = \emptyset$ while $\mathrm{LCA}_N(\{a,d\}) = \{\rho\}$ and so $\mathrm{lca}_N(ad) := \mathrm{lca}_N(\{a,d\})$ is well-defined whereas $\mathrm{lca}_G(\{a,d\})$ is not. In addition, in both DAGs G and N we have $\mathrm{LCA}_G(\{b,c\}) = \mathrm{LCA}_N(\{b,c\}) = \{p,q\}$, i.e., the LCA of b and c is not well-defined in either N or G. Both G and N are 2-lca-relevant, since e.g. $p = \mathrm{lca}_G(ab) = \mathrm{lca}_N(ab)$, $q = \mathrm{lca}_G(cd) = \mathrm{lca}_N(cd)$ and $\rho = \mathrm{lca}_N(ad)$.

In general, not every set $A \subseteq X$ has a (unique) least common ancestor in a DAG on X; for example, consider the DAG G in Figure 2 where $|LCA_G(\{b,c\})| > 1$ and $LCA_G(\{a,d\}) = \emptyset$. In a network N, the unique root is a common ancestor for all $A \subseteq L(N)$ and, therefore, $\mathrm{LCA}_N(A) \neq \emptyset$. Moreover, we are interested in DAGs where $|\mathrm{LCA}_G(A)| = 1$ holds for certain subsets $A \subseteq X$. For simplicity, we will write $\mathrm{lca}_G(A) = v$ in case that $\mathrm{LCA}_G(A) = \{v\}$ and say that $\mathrm{lca}_G(A)$ *is well-defined*; otherwise, we leave $\mathrm{lca}_G(A)$ *undefined*. To recall, we often write xy for sets $\{x,y\}$, which allows us to put $\mathrm{lca}_G(xy) := \mathrm{lca}_G(\{x,y\})$, if $\mathrm{lca}_G(\{x,y\})$ is well-defined. A DAG G on X is *2-lca-relevant* if, for all $v \in V(G)$ there are $x,y \in X$ such that $v = \mathrm{lca}_G(xy)$. Put simply, in a 2-lca-relevant DAG each vertex is the unique LCA for one or two leaves. In addition, a DAG G is *lca-relevant*, if for all $v \in V(G)$ there is a subset $A \subseteq X$ such that $v = \mathrm{lca}_G(A)$. Hence, 2-lca-relevant DAGs are lca-relevant. We note in passing that the property of being 2-lca-relevant can be tested in polynomial time; see [33, Obs. 7.6]. We illustrate these definitions in Figure 2.

Note that Lemma 2.3 implies that in case G is a DAG on X then, for all $u,v \in V(G)$, we have $u \preceq_G v$ if and only if $u \preceq_{G^-} v$. As a direct consequence of the latter we obtain the following relationship between least common ancestors of sets $A \subseteq X$ in G and G^-.

Observation 2.4 ([19, L. 2.13])**.** *Let G be a DAG on X. If $\mathrm{lca}_G(A)$ is well-defined for some $A \subseteq X$, then $\mathrm{lca}_{G^-}(A)$ is well-defined and we have $\mathrm{lca}_G(A) = \mathrm{lca}_{G^-}(A)$.*

We conclude this section with an important definition that relates LCAs in DAGs. As discussed in the introduction one of our main aims is to find a DAG that realizes a given set of LCA constraints. With this in mind we define two key relations on LCAs which naturally arise from any DAG G.

Definition 2.5 (The relations $\blacktriangleleft_G$ and $\trianglelefteq_G$)**.** *For a given DAG G on X, we define the relation $\trianglelefteq_G$ and $\blacktriangleleft_G$ on $\mathscr{P}_2(X)$ as*

$$\blacktriangleleft_G := \{(ab,xy) \mid \mathrm{lca}_G(ab), \mathrm{lca}_G(xy) \text{ are well-defined and } \mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)\}$$

and

$$\trianglelefteq_G := \{(ab,xy) \mid \mathrm{lca}_G(ab), \mathrm{lca}_G(xy) \text{ are well-defined and } \mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)\}.$$

Note that, by definition, $\blacktriangleleft_G \subseteq \trianglelefteq_G$. Moreover, if G is a DAG on X, then, $\mathrm{lca}_G(xy) \preceq_G \mathrm{lca}_G(xy)$ for all $x,y \in X$ if and only if $\mathrm{lca}_G(xy)$ is well-defined in G, which in turn is equivalent to $(xy,xy) \in \trianglelefteq_G$. Hence, $\mathrm{supp}_{\trianglelefteq_G}$ comprises precisely those pairs $xy \in \mathscr{P}_2(X)$ for which $\mathrm{lca}_G(xy)$ is well-defined. In contrast, $\mathrm{supp}_{\blacktriangleleft_G}$ could leave out pairs $xy \in \mathscr{P}_2(X)$ for which $\mathrm{lca}_G(xy)$ is well-defined. For example, if a is a leaf in G but not incident to any arc, then $\mathrm{lca}_G(aa)$ is well-defined but $\mathrm{lca}_G(aa) \not\prec_G \mathrm{lca}_G(xy)$ for any $x,y \in X$, i.e., $aa \notin \mathrm{supp}_{\blacktriangleleft_G}$. Nevertheless, as we shall see later in Proposition 7.7, $\mathrm{supp}^+_{\blacktriangleleft_G} = \mathrm{supp}^+_{\trianglelefteq_G}$ always holds.

3 Realizability of LCA Constraints

In this section, we define what it means for a DAG to realize a given collection of LCA constraints. We consider two variants of realizability: one motivated by the approach taken in [2] and one that naturally generalizes it.

First, recall that, as discussed in the introduction, Aho et al. [2] considered the following problem. Given a collection of constraints of the form $(a,b) < (c,d)$, $a \neq b$, $c \neq d$, with $a,b,c,d \in X$, can we construct a phylogenetic tree T on X so that if $(a,b) < (c,d)$, then $\mathrm{lca}_T(ab) \prec_T \mathrm{lca}_T(cd)$ or determine that no such tree exists? This motivates our first definition for realizability.

Definition 3.1 (Strict realization)**.** *A relation $R \subseteq \mathscr{P}_2(X) \times \mathscr{P}_2(X)$ is* strictly realizable *if there is a DAG G on X such that, for all $ab,cd \in \mathrm{supp}^+_R$, the vertices $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(cd)$ are well-defined and the following implication holds:*

(I0) *ab R cd implies that $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$.*

When **(I0)** *holds, we say that R is* strictly realized *by G.*

We could relax this definition by insisting that R is "realizable" if there is a DAG G such that, for all $ab\ R\ cd$, we have $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$. However, with this definition, the star tree (with appropriate leaf set) would *always* realize R, provided that there are no pairs $ab\ R\ xx$ with $ab \neq xx$. Hence, this type of "realizability" would be too permissive and becomes essentially trivial to satisfy. In order to generalize "strict realization", we provide the following definition.

Definition 3.2 (Realization)**.** *A relation $R \subseteq \mathscr{P}_2(X) \times \mathscr{P}_2(X)$ is* realizable *if there is a DAG G on X such that, for all $ab, cd \in \mathrm{supp}_R^+$ the vertices $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(cd)$ are well-defined and the following two implications hold:*

(I1) $(ab,cd) \in \mathrm{tc}(R)$ *and* $(cd,ab) \notin \mathrm{tc}(R)$ *implies that* $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$

(I2) $(ab,cd) \in \mathrm{tc}(R)$ *and* $(cd,ab) \in \mathrm{tc}(R)$ *implies that* $\mathrm{lca}_G(ab) = \mathrm{lca}_G(cd)$

When **(I1)** *and* **(I2)** *hold, we say that R is* realized *by G.*

As an example for Definition 3.2 consider the relation R pictured in Figure 3. In this example, $(xy,yz),(yz,xy) \in \mathrm{tc}(R)$ which, according to **(I2)** implies that $\mathrm{lca}_G(xy) = \mathrm{lca}_G(yz)$ must hold for any DAG G realizing R. Moreover, $(bc,ab) \in \mathrm{tc}(R)$ and $(ab,bc) \notin \mathrm{tc}(R)$ together with **(I1)** implies that $\mathrm{lca}_G(bc) \prec_G \mathrm{lca}_G(ab)$ must hold for any DAG G realizing R. In this way, it is straightforward to verify that the phylogenetic tree T in Figure 3 realizes R.

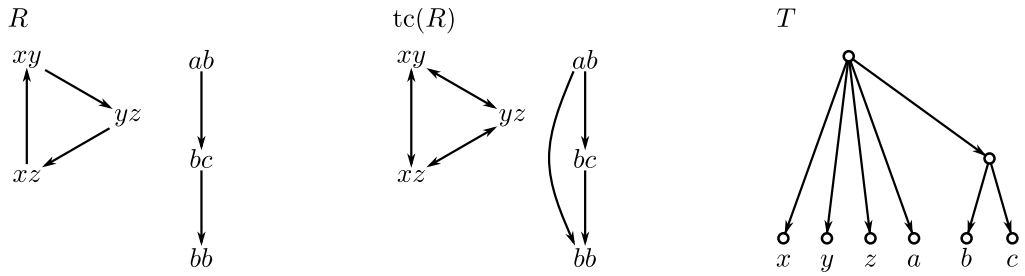

Figure 3: On the left we give a graphical representation of a relation relation R whose vertex set is $\mathrm{supp}_R = \{ab, bb, bc, xy, xz, yz\}$. Here we draw an arc $p \to q$ precisely if $q\ R\ p$. Similarly, in the middle we give the graphical representation of the transitive closure $\mathrm{tc}(R)$. The phylogenetic tree T on the right realizes R.

Although the notion of realizability as in Definition 3.2 looks, at a first glance, more restrictive than Definition 3.1 it is in fact a generalization of strict realization, as shown next.

Lemma 3.3. *Suppose that R is a relation on $\mathscr{P}_2(X)$. Then R is strictly realized by G if and only if R is realized by G and $\mathrm{tc}(R)$ is asymmetric.*

Proof. Suppose that R is a strictly realizable relation on $\mathscr{P}_2(X)$. Hence, there is a DAG G on X such that, for all $ab, cd \in \mathrm{supp}_R^+$, the vertices $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(cd)$ are well-defined and $ab\ R\ cd$ implies that $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. Let $ab, cd \in \mathrm{supp}_R^+$ be such that $(ab,cd) \in \mathrm{tc}(R)$. Thus, there is a (ab,cd)-chain in R that can be written as $p_0 = ab\ R\ p_1\ R\ \cdots\ R\ p_{k-1} R\ p_k = cd$, where $p_i \in \mathscr{P}_2(X)$. Since R is strictly realizable, we have $\mathrm{lca}_G(ab) = \mathrm{lca}_G(p_0) \prec_G \mathrm{lca}_G(p_1) \prec_G \cdots \prec_G \mathrm{lca}(p_k) = \mathrm{lca}_G(cd)$ and, therefore, $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. Note that, if $(cd,ab) \in \mathrm{tc}(R)$ similar arguments would yield $\mathrm{lca}_G(cd) \prec_G \mathrm{lca}_G(ab)$; contradicting $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. As G realizes R, the case $(cd,ab) \in \mathrm{tc}(R)$ can, therefore, not occur and $(cd,ab) \notin \mathrm{tc}(R)$ must hold, i.e., $\mathrm{tc}(R)$ is asymmetric. Hence, the conditions in **(I1)** are satisfied for any $(ab,cd) \in \mathrm{tc}(R)$ which together with $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$ for all such $(ab,cd) \in \mathrm{tc}(R)$ implies that G realizes R.

Suppose now that R is realized by G and that $\mathrm{tc}(R)$ is asymmetric. Since R is realized by G, it satisfies **(I1)** and **(I2)**. In particular, $\mathrm{tc}(R)$ satisfies **(I1)** and **(I2)** since $\mathrm{tc}(\mathrm{tc}(R)) = \mathrm{tc}(R)$. Hence, $\mathrm{tc}(R)$ is realized by G. Let $ab, cd \in \mathrm{supp}_R^+$ such that $ab\ R\ cd$ and thus, $(ab,cd) \in \mathrm{tc}(R)$. Since $\mathrm{tc}(R)$ is asymmetric it follows that $(cd,ab) \notin \mathrm{tc}(R)$. This and **(I1)** implies that $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. Hence, R is strictly realized by G. □

In the rest of this section we shall make some further remarks concerning our two notions of realizability as given in Definition 3.1 respectively 3.2 and their relationship. The first is straight-forward to check using the fact that $\mathrm{tc}(\mathrm{tc}(R)) = \mathrm{tc}(R)$ always holds, and so we leave the proof to the reader.

Observation 3.4. *Suppose that R is a relation on $\mathscr{P}_2(X)$. Then R is realized by a DAG G if and only if $\mathrm{tc}(R)$ is realized by G.*

In our next result we show that, as expected, for any DAG G the relations $\trianglelefteq_G$ and $\blacktriangleleft_G$ are both realized by G.

Lemma 3.5. *Every DAG G realizes $\trianglelefteq_G$ and strictly realizes $\blacktriangleleft_G$.*

Proof. Let G be a DAG on X and $R := \trianglelefteq_G = \{(ab,cd) \mid \mathrm{lca}_G(ab), \mathrm{lca}_G(cd)$ are well-defined and $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)\}$. It is easy to verify that $\preceq_G$ is transitive and that, therefore, R is transitive. Consequently, $\mathrm{tc}(R) = R$. We show now that **(I1)** and **(I2)** are satisfied. For **(I1)**, let $(ab,cd) \in \mathrm{tc}(R) = R$ and $(cd,ab) \notin \mathrm{tc}(R) = R$. By definition of R, we have $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$ and $\mathrm{lca}_G(cd) \not\preceq_G \mathrm{lca}_G(ab)$. Hence, $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$ must hold, i.e., **(I1)** is satisfied. For **(I2)**, suppose $(ab,cd),(cd,ab) \in \mathrm{tc}(R) = R$. By definition of R, we have $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$ and $\mathrm{lca}_G(cd) \preceq_G \mathrm{lca}_G(ab)$. That is, $\mathrm{lca}_G(ab) = \mathrm{lca}_G(cd)$ and **(I2)** holds.

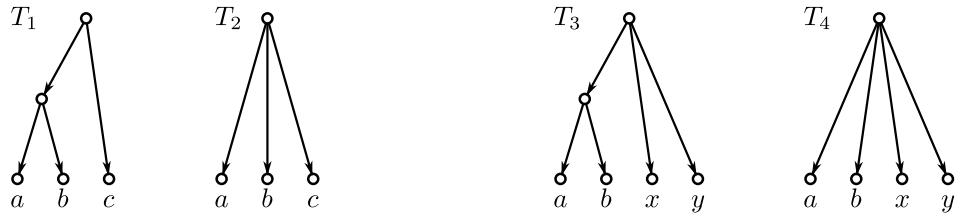

Figure 4: Four phylogenetic trees T_1, T_2, T_3 and T_4 used in Example 3.7, 3.8, 4.6 and 8.2.

To see that G strictly realizes $\blacktriangleleft_G$ we must show that **(I0)** holds. But, by definition of $\blacktriangleleft_G$, $ab \blacktriangleleft_G cd$ if and only if $\mathrm{lca}_G(ab)$ and $lca_G(cd)$ are well-defined and satisfy $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. Thus, **(I0)** is trivially satisfied and, therefore, G strictly realizes $\blacktriangleleft_G$. □

Many of the examples considered in this paper are, for simplicity, realizable by trees. However, we emphasize that this is not always the case; there exist relations that can be realized by some network but not by any tree, see the example in Figure 1.

We now show that if R is realized by a DAG G on X, then $R \subseteq \trianglelefteq_G$.

Lemma 3.6. *If R is a relation $\mathscr{P}_2(X)$ that is realized by the DAG G on X, then $\mathrm{lca}_G(ab)$ is well-defined for all $ab \in \mathrm{supp}_R^+$ and $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$ must hold for all $ab\ R\ cd$. Hence, $R \subseteq \trianglelefteq_G$.*

Proof. Note that, for all relations R, $\mathrm{supp}_R^+ \setminus \mathrm{supp}_R \subseteq \{xx \mid x \in X\}$. Hence, for all distinct $a,b \in X$ with $ab \in \mathrm{supp}_R^+$ we must have $ab \in \mathrm{supp}_R$. Thus, if G is a DAG on X that realizes R, then $\mathrm{lca}_G(ab)$ must be well-defined for such $ab \in \mathrm{supp}_R^+$ and, if $aa \in \mathrm{supp}_R^+$, then $\mathrm{lca}_G(aa) = a$ trivially holds. In particular, it is an easy task to verify that, if G realizes R, then $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$ must hold for all $ab\ R\ cd$, irrespective of which implication in **(I1)** or **(I2)** of Definition 3.2 applies. Hence, $R \subseteq \trianglelefteq_G$. □

Note that the converse of Lemma 3.6 is not necessarily true, i.e., we may have $R \subseteq \trianglelefteq_G$ for some DAG G even though G does not realize R. Hence, the notion of realizability given in Definition 3.2 is stronger than just assuming $R \subseteq \trianglelefteq_G$. To see this, consider the following example.

Example 3.7 (Subsets of $\trianglelefteq_G$ that are not realized by G)**.** *Consider the star-tree T_2 on $X = \{a,b,c\}$ in Figure 4. We have $\mathrm{supp}_{\trianglelefteq_{T_2}} = \{aa,bb,cc,ab,ac,bc\} = \mathscr{P}_2(X)$, as for all pairs in $\mathscr{P}_2(X)$, the respective LCA exists and is unique in T_2. Consider now the relation $R = \{(ab,bc)\} \subseteq \trianglelefteq_{T_2}$. Clearly, $(ab,bc) \in \mathrm{tc}(R)$ and $(bc,ab) \notin \mathrm{tc}(R)$. By **(I1)**, it must hold that $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(bc)$ for any DAG G realizing R. Hence, T_2 does not realize R since $\mathrm{lca}_{T_2}(ab) = \mathrm{lca}_{T_2}(bc)$. Note that R is realized by the tree T_1 in Figure 4.*

Interestingly, we can even have subsets $R \subseteq \trianglelefteq_G$ for some DAG G which are not realizable by *any* DAG as the following example shows.

Example 3.8 (Subsets of realizable relations that are not realizable)**.** *Consider the star-tree T_4 on $X = \{x,y,a,b\}$, illustrated in Figure 4 which, by Lemma 3.5, realizes $\trianglelefteq_{T_4}$. Clearly, $\trianglelefteq_{T_4}$ contains the subset $R = \{(xx,ab),(yy,ab),(ab,xy)\}$. Hence, the relation R requires, by* **(I1)**, *$\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)$ for any DAG G realizing R. Since $\mathrm{lca}_{T_4}(ab) = \mathrm{lca}_{T_4}(xy)$, the tree T_4 does not realize R. Even more, R cannot be realized by any DAG G. To see this, assume for contradiction, that G is a DAG that realizes R. By the latter arguments, $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)$ holds. At the same time $(xx,ab),(yy,ab) \in R$ together with Lemma 3.6 implies that $x = \mathrm{lca}_G(xx) \preceq_G \mathrm{lca}_G(ab)$ and $y = \mathrm{lca}_G(yy) \preceq_G \mathrm{lca}_G(ab)$ holds. As $\mathrm{lca}_G(xy)$ exists in G and since $\mathrm{lca}_G(ab)$ is a common ancestor of x and y it follows that $\mathrm{lca}_G(xy) \preceq_G \mathrm{lca}_G(ab)$ must hold (see also Lemma 4.1). This, however, violates the requirement that $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)$. Hence, $R \subseteq \trianglelefteq_{T_4}$ is not realizable by any DAG G.*

As Examples 3.7 and 3.8 show, the property that $R \subseteq \trianglelefteq_G$ for some DAG G neither implies that G realizes R nor that R is realizable at all. In contrast, we have the following result for the relation $\blacktriangleleft_G$.

Lemma 3.9. *Let R be a relation on $\mathscr{P}_2(X)$ and G a DAG on X. Then, R is strictly realized by G if and only if $R \subseteq \blacktriangleleft_G$.*

Proof. Let G be a DAG on X and R a relation on $\mathscr{P}_2(X)$. If R is strictly realized by G then, by definition, Condition **(I0)** is satisfied i.e. $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(cd)$ are well-defined and satisfy $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$ for all $(ab,cd) \in R$. By definition of $\blacktriangleleft_G$, we thus have $R \subseteq \blacktriangleleft_G$. Conversely, suppose $R \subseteq \blacktriangleleft_G$. By definition of $\blacktriangleleft_G$ and since $R \subseteq \blacktriangleleft_G$, for all $(ab,cd) \in R$, we have $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. Hence, **(I0)** is satisfied. By definition, R is therefore strictly realized by G. □

4 The +-Closure

Our next goal is to characterize when a relation R on $\mathscr{P}_2(X)$ can be realized by some DAG (or network) on X. To this end, we define and study in this section a closure relation R^+ associated with R, which contains additional information about any DAG G that could potentially realize R. A related approach for realizing LCA constraints using phylogenetic trees is outlined in [37, Section 2].

The key observation underlying the definition of R^+ is based on the following result, which shows how ancestor relations among LCAs can be inferred from other ones.

Lemma 4.1. *Let G be a DAG on X and suppose that $a,b,c,d,x,y \in X$ (not necessarily distinct) are such that $\mathrm{lca}_G(ab), \mathrm{lca}_G(ac), \mathrm{lca}_G(xy)$ and $\mathrm{lca}_G(bd)$ are all well-defined. Then, the following statement holds:*

$$\text{if } \mathrm{lca}_G(ac) \preceq_G \mathrm{lca}_G(xy) \text{ and } \mathrm{lca}_G(bd) \preceq_G \mathrm{lca}_G(xy), \text{ then } \mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy). \tag{1}$$

Proof. Let G be as defined in the statement of the lemma. Suppose that $\mathrm{lca}_G(ac) \preceq_G \mathrm{lca}_G(xy)$ and $\mathrm{lca}_G(bd) \preceq_G \mathrm{lca}_G(xy)$. Hence, $\mathrm{lca}_G(xy)$ is a common ancestor of a and b. This and the defining property of the least common ancestor $\mathrm{lca}_G(ab)$ implies that $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$. □

Although the implication in Equation (1) from Lemma 4.1 is only one of many that can be derived from "LCA-$\prec_G$ relationships", it turns out that this particular implication is sufficient to characterize realizable relations and motivates the following definition.

Definition 4.2 (Cross-Consistency)**.** *A relation R on $\mathscr{P}_2(X)$ is* cross-consistent *if for all $a,b,c,d,x,y \in X$ (not necessarily distinct) the following statement holds:*

$$\text{if } ac\ R\ xy,\ bd\ R\ xy \text{ and } ab \in \mathrm{supp}_R, \text{ then } ab\ R\ xy.$$

A given relation R on $\mathscr{P}_2(X)$ typically contains only "partial information" on pairs of LCAs in supp_R. We now define the aforementioned closure R^+ of R that allows us to extend R to encompass information about additional pairs.

Definition 4.3 (+-Closure)**.** *If R is a relation on $\mathscr{P}_2(X)$, then the* reflexive transitive cross-consistent closure (for short, +-closure) *is defined as*

$$R^+ := \bigcap_{R' \in \mathfrak{R}} R',$$

where $\mathfrak{R}$ denotes the set of all relations R' on $\mathscr{P}_2(X)$ that are supp_R^+-reflexive, transitive, cross-consistent, and satisfy $R \subseteq R'$.

It is an easy task to verify that the +-closure R^+ of a relation R is supp_R^+-reflexive, transitive and cross-consistent since $R^+ \subseteq R'$ and, therefore, $\mathrm{supp}_R^+ \subseteq \mathrm{supp}_{R^+} \subseteq \mathrm{supp}_{R'}$ for all $R' \in \mathfrak{R}$. We now show that R^+ also satisfies that classical closure axioms.

Proposition 4.4. *Let R be a relation on $\mathscr{P}_2(X)$. The operator $^+ \colon R \mapsto R^+$ is a closure operator on $\mathscr{P}_2(X) \times \mathscr{P}_2(X)$, i.e., it satisfies* extensivity $R \subseteq R^+$*;* monotonicity $R_1 \subseteq R_2 \implies R_1^+ \subseteq R_2^+$*; and* idempotency $(R^+)^+ = R^+$.

Proof. Let R be a relation on $\mathscr{P}_2(X)$. Let $\mathfrak{R}$ be the set of all relations on $\mathscr{P}_2(X)$ that are supp_R^+-reflexive, transitive, cross-consistent and contain R, so that $R^+ = \bigcap_{R' \in \mathfrak{R}} R'$. By definition, every $R' \in \mathfrak{R}$ satisfies $R \subseteq R'$. Hence $R \subseteq \bigcap_{R' \in \mathfrak{R}} R' = R^+$, i.e., extensivity holds.

Assume that $R_1 \subseteq R_2$ for two relations R_1 and R_2 on $\mathscr{P}_2(X)$. Let $\mathfrak{R}_i$ be the set of all relations on $\mathscr{P}_2(X)$ that are $\mathrm{supp}_{R_i}^+$-reflexive, transitive, cross-consistent and contain R_i for $i \in \{1,2\}$. Note that $R_1 \subseteq R_2$ implies $\mathrm{supp}_{R_1} \subseteq \mathrm{supp}_{R_2}$ and since both relations are on $\mathscr{P}_2(X)$ we thus have $\mathrm{supp}_{R_1}^+ \subseteq \mathrm{supp}_{R_2}^+$. Consequently, every $\mathrm{supp}_{R_2}^+$-reflexive relation is, in particular, also $\mathrm{supp}_{R_1}^+$-reflexive. Now, by definition and the fact that $R_1 \subseteq R_2$, it follows that any relation $S \in \mathfrak{R}_2$ is $\mathrm{supp}_{R_1}^+$-reflexive, transitive, cross-consistent and contains R_1, so $S \in \mathfrak{R}_1$. In other words, $\mathfrak{R}_2 \subseteq \mathfrak{R}_1$. Consequently,

$$R_1^+ = \bigcap_{S \in \mathfrak{R}_1} S \subseteq \bigcap_{S \in \mathfrak{R}_2} S = R_2^+$$

follows, i.e. monotonicity holds.

Now let $\mathfrak{R}_{R^+}$ be the set of $\mathrm{supp}_{R^+}^+$-reflexive, transitive and cross-consistent relations on $\mathscr{P}_2(X)$ that contain R^+. By construction R^+ itself belongs to $\mathfrak{R}_{R^+}$, hence

$$(R^+)^+ = \bigcap_{R' \in \mathfrak{R}_{R^+}} R' \subseteq R^+.$$

Moreover, by extensivity, we have $R^+ \subseteq (R^+)^+$. Thus, $(R^+)^+ = R^+$ which proves idempotency. □

We note in passing that for all $ab \in \mathrm{supp}_R^+$ (where $a = b$ might be possible) we have $ab\ R^+\ ab$ since R^+ is supp_R^+-reflexive. This together with cross-consistency of R^+ implies $aa\ R^+\ ab$ and $bb\ R^+ ab$ always holds.

We show now that the relation R^+ can be obtained by application of three simple rules on R and can therefore be computed in polynomial time.

Theorem 4.5. *Let R be a relation on $\mathscr{P}_2(X)$. Assume that S is a relation obtained from R by starting with $S = R$ and first applying the rule*

(R1) Reflexivity: *for all $p \in \mathrm{supp}_R^+$, add $p\ S\ p$.*

Afterwards, repeatedly apply one of the following two rules in any order, until none of the rules can be applied:

(R2) Transitivity: *if $p\ S\ q$ and $q\ S\ r$, add $p\ S\ r$.*

(R3) Cross-Consistency: *if* $ab \in \mathrm{supp}_S$ *and* $ac\ S\ xy$ *and* $bd\ S\ xy$ *for some* $c,d \in X$*, then add* $ab\ S\ xy$*.*

Then, $S = R^+$ *and* $\mathrm{supp}_R^+ = \mathrm{supp}_{R^+} = \mathrm{supp}_S$*. Furthermore,* R^+ *can be constructed in polynomial time in* $|X|$*.*

Proof. Suppose that S is obtained as outlined in the statement of the lemma and let $\mathfrak{R}$ be the set of all relations R' on $\mathscr{P}_2(X)$ that contain R and are supp_R^+-reflexive, transitive and cross-consistent. Since $R^+ = \cap_{R' \in \mathfrak{R}} R'$ have

$$R^+ \subseteq R' \text{ and, thus, } \mathrm{supp}_{R^+} \subseteq \mathrm{supp}_{R'} \text{ for all } R' \in \mathfrak{R}. \tag{2}$$

Since the rules **(R1)**, **(R2)** and **(R3)** are exhaustively applied, the final relation S must be supp_R^+-reflexive, transitive, cross-consistent and satisfies, by definition, $R \subseteq S$. Therefore, $S \in \mathfrak{R}$ holds, and consequently $R^+ \subseteq S$. Moreover, $\mathrm{supp}_S = \mathrm{supp}_R^+$ holds by construction.

We now show that $S \subseteq R^+$ must hold by considering the intermediate applications of the rules **(R1)**–**(R3)**. Specifically, let $S_0, S_1, \ldots, S_n$ denote the relations on $\mathscr{P}_2(X)$ for which $S_0 = R$, $S_n = S$ and where, for each $0 \leq i \leq n-1$, S_{i+1} is constructed from S_i by applying exactly one of **(R1)**, **(R2)** and **(R3)**. Observe that Proposition 4.4 ensures that $S_0 = R$ must be a subset of R^+. Moreover, S_1 is obtained from $S_0 = R$ by applying **(R1)** and thus $S_1 = R \cup \{(p,p) \mid p \in \mathrm{supp}_R^+\}$. The latter equality taken together with $R \subseteq R^+$ and the fact that R^+ is supp_R^+-reflexive, imply that $S_1 \subseteq R^+$ and, thus, $\mathrm{supp}_{S_1} \subseteq \mathrm{supp}_{R^+}$.

Now, assume that S_i is contained in R^+ for some fixed i with $1 \leq i < n$. Thus S_{i+1} is constructed from S_i by applying exactly one of **(R2)** and **(R3)**. We now consider what happens in each of these two cases.

If S_{i+1} is obtained from S_i by applying **(R2)**, then $S_{i+1} = S_i \cup \{(p,r)\}$ for some pair (p,r) for which $(p,q),(q,r) \in S_i \subseteq R^+$. By Eq. (2), $(p,q),(q,r) \in R'$ for all $R' \in \mathfrak{R}$. Moreover, since each relation in $\mathfrak{R}$ is transitive, the pair (p,r) must be contained in R' for all $R' \in \mathfrak{R}$. By definition of R^+, we have $p\ R^+\ r$. Consequently, $S_{i+1} \subseteq R^+$.

If S_{i+1} is obtained from S_i by applying **(R3)**, then $S_{i+1} = S_i \cup \{(ab,xy)\}$ for some pair (ab,xy) for which $ab \in \mathrm{supp}_{S_i}$. In particular, **(R3)** requires the existence of some $(ac,xy),(bd,xy) \in S_i$. Since, by assumption, $S_i \subseteq R^+$, we have $(ac,xy),(bd,xy) \in R^+$. By Eq. (2), $(ac,xy),(bd,xy) \in R'$ and $ab \in \mathrm{supp}_{R'}$ for all $R' \in \mathfrak{R}$. Moreover, since each relation in $\mathfrak{R}$ is cross-consistent, the pair (ab,xy) must also be contained in R' for all $R' \in \mathfrak{R}$. By definition of R^+, we have $ab\ R^+\ xy$. Consequently, $S_{i+1} \subseteq R^+$.

In both of these cases, $S_{i+1} \subseteq R^+$ holds and so, by induction, it follows that $S = S_n \subseteq R^+$. This together with $R^+ \subseteq S$ implies that $R^+ = S$. In particular, $\mathrm{supp}_R^+ = \mathrm{supp}_S = \mathrm{supp}_{R^+}$.

To complete the proof we now show that R^+ can be constructed in polynomial time in $|X|$. By the arguments above, R^+ can be obtained by repeatedly applying Rules **(R1)**, **(R2)**, and **(R3)**, starting with $S = R$ and extending it step by step. Since $S \subseteq \mathscr{P}_2(X) \times \mathscr{P}_2(X)$ and $|\mathscr{P}_2(X)| = \binom{|X|}{2} + |X| \in O(|X|^2)$, it follows that $|S| \in O(|X|^4)$ at each step. Thus, checking whether one of the Rules **(R1)**, **(R2)**, or **(R3)** can be applied to S can be done in polynomial time, and adding a pair (p,q) to S requires only constant time. Clearly, the process of constructing R^+ by applying Rules **(R1)**, **(R2)**, and **(R3)** must terminate, since at most $O(|X|^4)$ pairs can be added to S. In summary, R^+ can be constructed in polynomial time in $|X|$. □

In the following example, we demonstrate why R^+ is useful.

Example 4.6. *Consider the relation R with* $xx\ R\ ab$*,* $yy\ R\ ab$*,* $aa\ R\ xy$*, and* $bb\ R\ xy$*, which is realized by the star-tree* T_4 *on* $X = \{a,b,x,y\}$*, as illustrated in Figure 4. Observe first the accordance with Lemma 4.1: we have* $x = \mathrm{lca}_{T_4}(xx) \preceq_{T_4} \mathrm{lca}_{T_4}(ab)$ *and* $y = \mathrm{lca}_{T_4}(yy) \preceq_{T_4} \mathrm{lca}_{T_4}(ab)$*, and moreover* $\mathrm{lca}_{T_4}(xy) \preceq_{T_4} \mathrm{lca}_{T_4}(ab)$*. In fact,* $\mathrm{lca}_{T_4}(xy) = \mathrm{lca}_{T_4}(ab)$ *holds.*

However, the relation R is not cross-consistent, since $xx\ R\ ab$ *and* $yy\ R\ ab$ *hold, but* $xy\ R\ ab$ *does not. In contrast, we have both* $xy\ R^+\ ab$ *and* $ab\ R^+\ xy$*. As we shall see later in Lemma 4.9, this implies that* $\mathrm{lca}_G(xy) \preceq_G \mathrm{lca}_G(ab)$ *and* $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$*, and thus* $\mathrm{lca}_G(xy) = \mathrm{lca}_G(ab)$ *must hold for any DAG G realizing R. For a full representation of a relation R and its* $+$*-closure* R^+*, we refer to Figure 5.*

Example 4.6 shows that it is helpful to think of R^+ as the *minimal* extension of R for which $\preceq_G$-comparability requirements between unique LCAs are enforced across all DAGs G that realize R.

We conclude this section with some observations and results concerning the closure R^+ that will be useful later. First note that, since R^+ is by definition supp_R^+-reflexive, the equality $\mathrm{supp}_R^+ = \mathrm{supp}_{R^+}$ in Theorem 4.5 implies:

Observation 4.7. *The* $+$*-closure* R^+ *of a relation R is reflexive.*

Moreover, since R^+ is transitive, we have $\mathrm{tc}(R^+) = R^+$. This together with $R \subseteq R^+$ and extensivity of the transitive closure implies:

Observation 4.8. *For all relations R we have* $\mathrm{tc}(R) \subseteq R^+$*.*

The following result shows that the closure R^+ contains valuable additional information about any DAG G that can potentially realize R. Indeed, we now show that if a DAG G realizes R, then we not only have $R \subseteq \trianglelefteq_G$ but we must have $R^+ \subseteq \trianglelefteq_G$.

Lemma 4.9. *Assume a relation R on* $\mathscr{P}_2(X)$ *is realized by a DAG G on X. Then for all* $a,b,x,y \in X$

$$ab\ R^+\ xy \quad \Longrightarrow \quad \mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy).$$

In particular, $R^+ \subseteq \trianglelefteq_G$ *for every DAG G that realizes R.*

Proof. Suppose that R on $\mathscr{P}_2(X)$ is a relation that is realized by the DAG G on X. Let $S_0, S_1, \ldots, S_n$ denote relations on supp_R^+ for which $S_0 = R$, $S_n = R^+$ and where, for each S_{i+1} is constructed from S_i, $0 \leq i \leq n-1$, by applying exactly one of **(R1)**, **(R2)** and **(R3)** to S_i; the existence of at least one such a sequence is ensured by Theorem 4.5. We will prove that, for each $0 \leq i \leq n$, we have

$$ab\ S_i\ xy \implies \mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy). \tag{3}$$

By Lemma 3.6, $ab\ R\ xy$ always implies that $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$. Thus, for $i = 0$ where $S_i = R$, Eq. 3 trivially holds. Moreover, by definition, S_1 is obtained from R by applying **(R1)** and hence $S_1 = R \cup \{(p,p) \mid p \in \mathrm{supp}_R^+\}$. Clearly the elements $(p,p) \in S_1 \setminus R$ satisfy Eq. 3, since $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(ab)$ holds for all $p = ab \in \mathrm{supp}_R^+$. Therefore, Eq. 3 is satisfied for all elements in S_1 and thus, for $i \in \{0,1\}$.

Assume next, that Eq. 3 holds for some fixed $i \geq 1$ and consider S_{i+1}. By definition, $S_{i+1} = S_i \cup \{(p,q)\}$ for some $p,q \in \mathrm{supp}_R^+$. Moreover, by the induction hypothesis, Eq. 3 holds for all pairs in $S_i = S_{i+1} \setminus \{(p,q)\}$. Hence, it suffices to show that Eq. 3 hold for $p\ S_{i+1}\ q$. To this end, we consider now the two possible cases for how $p\ S_{i+1}\ q$ was derived.

If $p\ S_{i+1}\ q$ was added via **(R2)**, then there must exist $ab, xy, cd \in \mathrm{supp}_R^+$ for which $p = ab\ S_i\ xy$ and $xy\ S_i\ cd = q$. By Eq. 3, $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$ and $\mathrm{lca}_G(xy) \preceq_G \mathrm{lca}_G(cd)$. Thus, by transitivity of $\preceq_G$ (concatenation of directed paths in G), $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$ holds as well.

If $p\ S_{i+1}\ q$ was added via **(R3)**, then there must exist $ab, ac, bd, xy \in \mathrm{supp}_R^+$ for which $ac\ S_i\ xy$, $bd\ S_i\ xy$ and where $p = ab$, respectively $q = xy$. Since G realizes R and $ab, ac, bd, xy \in \mathrm{supp}_R^+$ it follows that $\mathrm{lca}_G(ab), \mathrm{lca}_G(ac), \mathrm{lca}_G(xy)$ and $\mathrm{lca}_G(bd)$ must be well-defined in G. By Eq. 3, $\mathrm{lca}_G(ac) \preceq_G \mathrm{lca}_G(xy)$ and $\mathrm{lca}_G(bd) \preceq_G \mathrm{lca}_G(xy)$. By Lemma 4.1, $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$ holds.

In summary, for each of the relations S_i, and thus, in particular, for $S_n = R^+$, Eq. 3 is satisfied. Hence, $ab\ R^+\ xy \implies \mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$ for every DAG G that realizes R. This and Def. 2.5 implies that $R^+ \subseteq \trianglelefteq_G$ for every DAG G that realizes R. □

The final result of this section shows that for every DAG G, the underlying relation $\trianglelefteq_G$ and its $+$-closure agree.

Proposition 4.10. *For all DAGs G we have $\trianglelefteq_G = \trianglelefteq_G^+$.*

Proof. Let G be a DAG on X. To show that $\trianglelefteq_G = \trianglelefteq_G^+$ by Theorem 4.5 it suffices, to show that $\trianglelefteq_G$ is $\mathrm{supp}^+_{\trianglelefteq_G}$-reflexive, transitive and cross-consistent. Since $x = \mathrm{lca}_G(xx)$ is well-defined for all $x \in X$ and $x \preceq_G x$, we have $xx \trianglelefteq_G xx$ for all $x \in X$. In particular, it follows that $\mathrm{supp}_{\trianglelefteq_G} = \mathrm{supp}^+_{\trianglelefteq_G}$. Since $\preceq_G$ is reflexive, i.e. $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(ab)$ for all $ab \in \mathscr{P}_2(X)$ where $\mathrm{lca}_G(ab)$ is well-defined, it follows that $\trianglelefteq_G$ is reflexive and, thus, in particular $\mathrm{supp}^+_{\trianglelefteq_G}$-reflexive. Similarly, transitivity of $\preceq_G$ implies that $\trianglelefteq_G$ is transitive. We conclude the proof of the proposition by showing that $\trianglelefteq_G$ is cross-consistent. To this end, suppose there are $a,b,c,d,x,y \in X$ (not necessarily distinct) such that $ac \trianglelefteq_G xy$ and $bd \trianglelefteq xy$. By definition, $\mathrm{lca}_G(ac) \preceq_G \mathrm{lca}_G(xy)$ and $\mathrm{lca}_G(bd) \preceq_G \mathrm{lca}_G(xy)$ holds. Suppose now that $ab \in \mathrm{supp}_{\trianglelefteq_G}$ and thus, that $\mathrm{lca}_G(ab)$ is well-defined. Then we can apply Lemma 4.1, to conclude that $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$ holds. Hence, $ab \trianglelefteq_G xy$ holds and $\trianglelefteq_G$ is cross-consistent. In summary, $\trianglelefteq_G = \trianglelefteq_G^+$ holds for all DAGs G. □

As we shall see later in Proposition 7.7, $\blacktriangleleft_G \neq \blacktriangleleft_G^+$ for all DAGs G.

5 The Canonical DAG and Characterization of Realizability

In this section, we shall give a characterization of when a relation R on $\mathscr{P}_2(X)$ is realizable by a DAG. Our approach proceeds as follows. We employ the closure R^+ to define an equivalence relation $\sim_{R^+}$ comprising pairs (p,q) that satisfy $p\ R^+\ q$ and $q\ R^+\ p$. Based on $\sim_{R^+}$, we will define the *canonical DAG* G_R whose vertices are the equivalence classes of $\sim_{R^+}$ and whose arcs are induced by the partial order on these classes. The classes $[aa]$ correspond to leaves of G_R and are relabeled by a to obtain a DAG on X. We then show in Theorem 5.10 that R can be realized by a DAG if and only if it is realized by G_R, and that this is equivalent to R satisfying the following two simple conditions.

Definition 5.1. *For a relation R on $\mathscr{P}_2(X)$, we define the following two conditions:*

Condition **(X1):** *For all $a,b,x \in X$: $ab \neq xx$ implies $(ab,xx) \notin R^+$.*

Condition **(X2):** *For all $a,b,x,y \in X$: $(ab,xy) \in \mathrm{tc}(R)$ and $(xy,ab) \notin \mathrm{tc}(R)$ implies $(xy,ab) \notin R^+$.*

We start by showing that **(X1)** and **(X2)** are necessary conditions for a relation R to be realizable.

Lemma 5.2. *If R on $\mathscr{P}_2(X)$ is realizable, then R satisfies* **(X1)** *and* **(X2)**.

Proof. Suppose that R on $\mathscr{P}_2(X)$ is realized by the DAG G on X. Assume, for contradiction, that **(X1)** does not hold. Hence, there are $a,b,x \in X$ with $ab \neq xx$ and $ab\ R^+\ xx$. By Lemma 4.9, $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xx) = x$. Since x is a leaf, it does not have any children and thus, $\mathrm{lca}_G(ab) = \mathrm{lca}_G(xx) = x$ which immediately implies that $a = b = x$. Thus, $ab = xx$; a contradiction.

For Condition **(X2)**, suppose that $(ab,xy) \in \mathrm{tc}(R)$ and $(xy,ab) \notin \mathrm{tc}(R)$. Since $\mathrm{tc}(R) \subseteq R^+$ by Observation 4.8, we have $ab\ R^+\ xy$. This together with Lemma 4.9 implies that $\mathrm{lca}_G(ab) \preceq \mathrm{lca}_G(xy)$. Assume, for contradiction, that $xy\ R^+\ ab$. In this case, Lemma 4.9 implies $\mathrm{lca}_G(xy) \preceq \mathrm{lca}_G(ab)$ and thus, $\mathrm{lca}_G(ab) = \mathrm{lca}_G(xy)$. But this contradicts the fact that G realizes R, since $(ab,xy) \in \mathrm{tc}(R)$ and $(xy,ab) \notin \mathrm{tc}(R)$ together require **(I1)**: $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)$. Therefore, $(xy,ab) \notin R^+$ and we can conclude that **(X2)** holds. □

Given a relation R on $\mathscr{P}_2(X)$, we now define the aforementioned equivalence relation $\sim_{R^+}$.

Lemma 5.3. *Let R be a relation on $\mathscr{P}_2(X)$. Then, the relation $\sim_{R^+}$ defined on supp_R^+ by putting for all $p,q \in \mathrm{supp}_R^+$,*

$$p \sim_{R^+} q \quad \Longleftrightarrow \quad p\, R^+\, q \text{ and } q\, R^+\, p$$

is an equivalence relation.

Proof. Put $\sim := \sim_{R^+}$ and recall that, by Observation 4.7, R^+ is reflexive and we have $p \sim p$ for all $p \in \mathrm{supp}_R^+$, i.e., $\sim$ is reflexive. Moreover, one easily observes that $p \sim q$ if and only $q \sim p$, i.e., $\sim$ is symmetric. Suppose that $p \sim q$ and $q \sim r$. Then, $p \sim q$ implies $p\, R^+\, q$ and $q\, R^+\, p$ while $q \sim r$ implies $q\, R^+\, r$ and $r\, R^+\, q$. Since R^+ is transitive, $p\, R^+\, q$ and $q\, R^+\, r$ implies that $p\, R^+\, r$, while $r\, R^+\, q$ and $q\, R^+\, p$ implies that $r\, R^+\, p$ holds. Consequently, $p \sim r$, i.e., $\sim$ is transitive. In summary, $\sim$ is an equivalence relation on supp_R^+. □

We now define a partial order on the equivalence classes of $\sim_{R^+}$.

Lemma 5.4. *Let R be a relation on $\mathscr{P}_2(X)$ and let $\sim_{R^+}$ be the equivalence relation on supp_R^+ as specified in Lemma 5.3. Let $[p]$ be the $\sim_{R^+}$-class that contains p, and let Q be the set of $\sim_{R^+}$-classes. Then the order $\leq_{R^+}$ on Q defined by*

$$[p] \leq_{R^+} [q] \quad \Longleftrightarrow \quad p\, R^+\, q.$$

is a partial order, i.e., it is reflexive, transitive, and anti-symmetric.

Proof. Put $\leq := \leq_{R^+}$. We first argue that this order $\leq$ is well-defined by showing that $[p] \leq [q]$ if and only if $p'\, R^+\, q'$ for all $p' \in [p]$ and all $q' \in [q]$, that is, the choice of elements in $[p]$ and $[q]$ does not matter. For the *if*-direction observe that $p'\, R^+\, q'$ for all $p' \in [p]$ and all $q' \in [q]$ implies that $p\, R^+\, q$ since $p \in [p]$ and $q \in [q]$. By definition, $[p] \leq [q]$ holds. For the *only if*-direction suppose that $[p] \leq [q]$ and thus, $p\, R^+\, q$ holds. Let $p' \in [p]$ and $q' \in [q]$. Since $q \sim q'$, we have $q\, R^+\, q'$. Since R^+ is transitive and $p\, R^+\, q$ we obtain $p\, R^+\, q'$. Moreover $p \sim p'$ implies $p'\, R^+\, p$. Since R^+ is transitive and $p\, R^+\, q'$, we obtain $p'\, R^+\, q'$. Since $p, p' \in [p]$ and $q, q' \in [q]$ can be chosen arbitrarily the statements follows. In summary, the order $\leq$ is well-defined.

We continue now with showing that $\leq$ is a partial order on Q. As noted in Observation 4.7, R^+ is reflexive, and thus $[p] \leq [p]$. Moreover, we have $[p] \leq [q]$ and $[q] \leq [p]$ if and only if $p\, R^+\, q$ and $q\, R^+\, p$ and thus, if and only if $p \sim q$ in which case $[p] = [q]$. Hence, $\leq$ is anti-symmetric. Furthermore, if $[p] \leq [q]$ and $[q] \leq [r]$, then $p\, R^+\, q$ and $q\, R^+\, r$. By transitivity of R^+, we have $p\, R^+\, r$. Hence, $[p] \leq [r]$, i.e., $\leq$ is transitive. In summary, $(Q, \leq)$ is a partially ordered set. □

We now define the canonical directed graph of a relation R on $\mathscr{P}_2(X)$.

Definition 5.5 (Canonical Directed Graph)**.** *Let R be a relation on $\mathscr{P}_2(X)$ and let $\sim_{R^+}$ be the equivalence relation on supp_R^+ as specified in Lemma 5.3. Let Q be the set of $\sim_{R^+}$-classes and $\leq_{R^+}$ be the partial order on Q as specified in Lemma 5.4. The* canonical directed graph *G_R is defined as as follows:*

1. *Add one vertex for each class $[ab] \in Q$ to G_R.*
2. *For all distinct classes $[p], [q] \in Q$ with $[p] \leq_{R^+} [q]$, add an arc $[q] \to [p]$ to G_R.*
3. *Finally, relabel all vertices $[aa]$ in G_R by a.*

An example of the canonical directed graph G_R of a realizable relation R is provided in Figure 5. We emphasize that the construction of G_R is solely based on properties of the relation R^+. Similarly, the construction of G_{R^+} depends only on the relation $(R^+)^+$. Since, by Proposition 4.4, $(R^+)^+ = R^+$, it follows that the construction of G_{R^+} is determined by the relation R^+, just as for G_R. Hence, we obtain

Observation 5.6. *For all relations R we have $G_R = G_{R^+}$.*

As we shall see in Theorem 5.10, if R can be realized by some DAG, then it is realized by G_R. To this end, we first give two useful properties of G_R in the case that R satisfies Condition **(X1)**.

Lemma 5.7. *Suppose that R satisfies* **(X1)**. *Then for each vertex $[xy]$ with $x \neq y$ in G_R, the arcs $[xy] \to x$ and $[xy] \to y$ exist.*

Proof. Suppose that R satisfies **(X1)**. This, in particular, implies that the $\sim_{R^+}$-class $[xx]$ is distinct from any $\sim_{R^+}$-class $[ab]$ for which $a, b \in X$ with $ab \neq xx$. Since R^+ is, by Observation 4.7, reflexive and $xx \in \mathrm{supp}_R^+$ (which corresponds to the $\sim_{R^+}$-class $[xx]$ that is relabeled by x in G_R), we always have a vertex x in G_R for all $x \in X$. If $xy \in \mathrm{supp}_R^+$ with $x \neq y$, then reflexivity of R^+ implies $xy\, R^+\, xy$ and cross-consistency implies that $xx\, R^+\, xy$ and $yy\, R^+\, xy$. Hence, $[xx] \leq_{R^+} [xy]$ and $[yy] \leq_{R^+} [xy]$. By construction, the arcs $[xy] \to [xx]$ and $[xy] \to [yy]$ have been added to G_R. Since $[xx]$ is finally relabeled by x in G_R, for each vertex $[xy]$ with $x \neq y$ in G_R, the arcs $[xy] \to x$ and $[xy] \to y$ exist. □

Proposition 5.8. *Let R be a relation on $\mathscr{P}_2(X)$ that satisfies Condition* **(X1)**. *Then, the canonical directed graph G_R is a DAG on X such that $[ab] = \mathrm{lca}_{G_R}(ab)$ for all vertices $[ab]$ of G_R with $a \neq b$ and $x = \mathrm{lca}_{G_R}(xx)$ for all $x \in X$. In particular, G_R is 2-lca-relevant, phylogenetic and realizes R^+.*

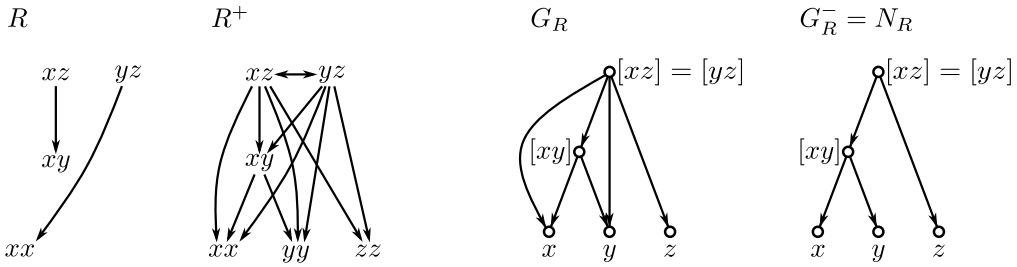

Figure 5: On the left we give a graphical representation of the relation relation $R = \{(xy,xz),(xx,yz)\}$ on $X = \{x,y,z\}$. Here we draw an arc $p \to q$ precisely if $q\ R\ p$. In addition, the graphical representation of R^+ is provided where arcs (p,p) are omitted for all $p \in \mathrm{supp}_R^+$. Furthermore, the canonical DAG G_R and the DAG $G_R^- = N_R$ that is obtained from G_R by removal of all shortcuts is shown. Both G_R and G_R^- realize R.

Proof. We start by showing that the leaf set of $G := G_R$ is X. Let Q be the set of all $\sim_{R^+}$-classes. By **(X1)**, no class $[xx] \in Q$ can coincide with a class $[ab] \in Q$ for which $ab \neq xx$. By Theorem 4.5**(R1)**, $[xx] \in Q$ for all $x \in X$. Hence, the two sets $Q_1 = \{[xx] \colon x \in X\}$ and $Q_2 = \{[ab] \in Q \colon a \neq b\}$ form a bipartition of Q, i.e., $Q = Q_1 \cup Q_2$ and $Q_1 \cap Q_2 = \emptyset$. By the latter arguments, we have $x,y \in X$ and $x \neq y$ if and only if $[xx] \neq [yy]$ and $[xx],[yy] \in Q_1 \subseteq Q$. Thus, there is a 1-to-1 correspondence between the vertices in X and the classes in Q_1. Moreover, by **(X1)**, $(ab,xx) \notin R^+$ for all $a,b,x \in X$ with $ab \neq xx$. Hence, $[ab] \not\leq_{R^+} [xx]$ for all $a,b,x \in X$ with $ab \neq xx$. Therefore, one never adds an arc of the form $x \to v$ for any $x \in X$ to G. Moreover, by construction, G does not contain arcs (v,v) for any $v \in V(G)$. In summary, the set X forms a subset of leaves in G and, in addition, each vertex not in X must be a class-vertex $[ab] \in Q_2$ that has, by Lemma 5.7, children a and b. Hence, X is precisely the leaf set of G.

We now show that G is a DAG. Recall that G cannot contain arcs (v,v) for any $v \in V(G)$. Moreover, since X is the leaf set of G and since all $x \in X$ have no children, a leaf $x \in X$ cannot be contained in any cycle in G. Now, assume for contradiction, that G contains a cycle $v_0 \to v_1 \to \cdots \to v_k = v_0$. Since no arcs (v,v) exist in G, we have $k \geq 1$. Since, $v_i \notin X$ for any i, we have $v_i \in Q_2$, i.e, v_i is a class-vertex $[q_i] \in Q_2$. Since $k \geq 1$, the arc $v_0 \to v_1$ exists and thus, $v_0 \neq v_1$. Therefore, $[q_0] \neq [q_1]$. Moreover, the existence of the arc $v_0 \to v_1$ implies $[q_1] \leq_{R^+} [q_0]$. Since $\leq_{R^+}$ is transitive (cf. Lemma 5.4) and the path $v_1 \rightsquigarrow v_k = v_0$ exists, we must have $[q_0] \leq_{R^+} [q_1]$. However, by Lemma 5.4, $\leq_{R^+}$ is anti-symmetric and so $[q_0] = [q_1]$ must hold; a contradiction. Hence, G is a DAG.

Now let $v \in V(G)$. Suppose that $v = [ab]$ for some class $[ab] \in Q$ with $a \neq b$. We show that v is the unique least common ancestor of a and b in G. By Lemma 5.7, the arcs $[ab] \to a$ and $[ab] \to b$ exist. Hence, $[ab]$ is a common ancestor of a and b. Now suppose that $w \in V(G)$ is a common ancestor of a and b with $w \neq [ab]$. Then w must be in Q_2, i.e., $w = [xy] \in Q$ with $x \neq y$. Since w is a common ancestor of a and b, there are directed paths $[xy] \rightsquigarrow a$ and $[xy] \rightsquigarrow b$ in G. By construction of G and since $\leq_{R^+}$ is transitive (cf. Lemma 5.4), we must have $[aa] \leq_{R^+} [xy]$ and $[bb] \leq_{R^+} [xy]$. By definition of $\leq_{R^+}$, we have $aa\ R^+\ xy$ and $bb\ R^+\ xy$. Since, $[ab] \in Q$ we have, by construction of $\sim_{R^+}$, $ab \in \mathrm{supp}_R^+$. By Theorem 4.5, $\mathrm{supp}_R^+ = \mathrm{supp}_{R^+}$ and, therefore, $ab \in \mathrm{supp}_{R^+}$. This together with that fact that R^+ is cross-consistent implies that $ab\ R^+\ xy$. Again, by definition of $\leq_{R^+}$, we have $[ab] \leq_{R^+} [xy] = w$. Since $w \neq [ab]$, the arc $w \to [ab]$ exists in G and, therefore, $[ab] \preceq_G w$. Since $w = [xy]$ was an arbitrary common ancestor of a and b, $[ab] \preceq_G w$ holds for *all* common ancestors of a and b. Hence $[ab]$ is the *unique* least common ancestor of a and b. Moreover, if $v = a$ for some $a \in X$, the unique least common ancestor $\mathrm{lca}_G(aa)$ is a since $a \in X$ is a leaf. In summary, G is a DAG in which $\mathrm{lca}_G(aa) = a$ for all $a \in X$ and $\mathrm{lca}_G(ab) = [ab]$ for all $[ab] \in Q$ with $a \neq b$.

By the latter arguments and since each vertex in G is either of the form $[ab] \in Q_2$ or x for some $[xx] \in Q_1$, it follows that, for each vertex v in G, there are $x,y \in X$ such the $v = \mathrm{lca}_G(xy)$. Therefore, G is 2-lca-relevant and, in particular, lca-relevant. This observation allows us to apply [33, L. 3.10] to conclude that G is phylogenetic.

We show now that G realizes R^+. By definition, R^+ is transitive and thus, $\mathrm{tc}(R^+) = R^+$. Let $(ab,cd) \in \mathrm{tc}(R^+) = R^+$. Suppose that $(cd,ab) \notin \mathrm{tc}(R^+) = R^+$. Hence, $[ab] \neq [cd]$ and $[ab] \leq_{R^+} [cd]$ which implies that the arc $[cd] \to [ab]$ exists. By the latter arguments, we therefore have $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$, which implies that **(I1)** holds for G and R^+. If $(cd,ab) \in \mathrm{tc}(R^+) = R^+$, then $ab \sim_{R^+} cd$ and thus, $[ab] = [cd]$. This together with the latter arguments implies that $\mathrm{lca}_G(ab) = \mathrm{lca}_G(cd)$ holds. Thus, **(I2)** holds for G and R^+. In summary, G realizes R^+. □

Since Proposition 5.8 ensures that the canonical directed graph G_R is a DAG for all relations satisfying **(X1)**, we refer to it as the canonical *DAG* G_R. Before proceeding, we also note that Proposition 5.8 ensures that the canonical DAG G_R realizes R^+ if R satisfies **(X1)**. This, however, does not imply that R is realizable, as shown in the following example.

Example 5.9 (Relations R that are not realizable, although R^+ is realizable)**.** *Let $R = \{(xx,ab),(yy,ab),(ab,xy)\}$ be the relation on $X = \{x,y,a,b\}$ as provided in Example 3.8. As argued in Example 3.8, R is not realizable by any DAG G. However, it is straight-forward to check that R satisfies* **(X1)***, which together with Proposition 5.8 implies that G_R realizes R^+. Hence, R^+ is realizable, although R is not.*

Example 5.9 shows that **(X1)** is not sufficient on its own for a relation R to be realizable. In particular, contraposition of Lemma 5.2 shows that also **(X2)** alone is insufficient to characterize realizable relations. We now prove the main result of this section.

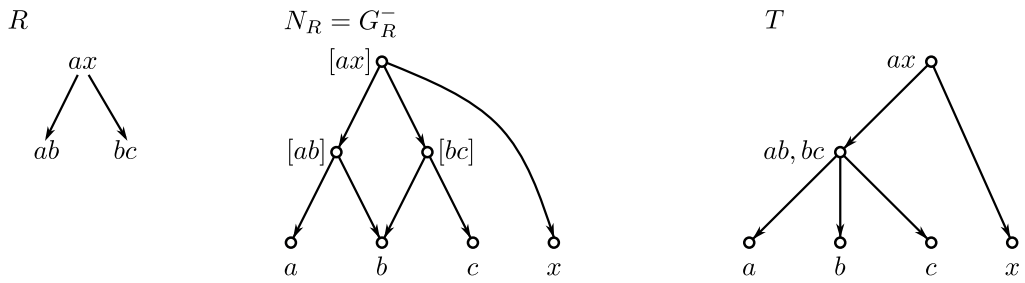

Figure 6: On the left we give a graphical representation of the relation relation $R = \{(ab,ax),(bc,ax)\}$. Here we draw an arc $p \to q$ precisely if $q\ R\ p$. In addition, the canonical network $N_R = G_R^-$ and a tree T that both realize R is shown. Both N_R and T are minimal, however, only T is minimum. Each pair $p \in \{ab, bc, ax\}$ in the drawings marks the corresponding vertex $\mathrm{lca}_T(p)$.

Theorem 5.10. *For a relation R on $\mathscr{P}_2(X)$ the following statements are equivalent:*

1. *R is realizable.*
2. *R satisfies* **(X1)** *and* **(X2)**.
3. *R is realized by its canonical DAG G_R.*

Proof. If Condition (1) is satisfied, then Lemma 5.2 implies that R satisfies **(X1)** and **(X2)**, i.e., Condition (2) holds.

Assume now that Condition (2) is satisfied. Consider the canonical DAG $G := G_R$ of R. Proposition 5.8 implies that G is DAG on X in which $\mathrm{lca}_G(aa) = a$ for all $a \in X$ and $\mathrm{lca}_G(ab) = [ab]$ for all $[ab] \in Q$ with $a \neq b$. We show that R is realized by G. Let $ab, xy \in \mathrm{supp}_R^+$ and suppose that $(ab,xy) \in \mathrm{tc}(R)$. By Observation 4.8, $\mathrm{tc}(R) \subseteq R^+$ and it follows that $(ab,xy) \in R^+$.

Assume, first that $(xy,ab) \notin \mathrm{tc}(R)$. By **(X2)**, $(xy,ab) \notin R^+$. Therefore, $[ab] \neq [xy]$ and $[ab] \leq_{R^+} [xy]$. If $a \neq b$, the vertex $v = [ab]$ exists in G and if $a = b$, the vertex $v = a \in X$ exists in G. In either case, the arc $[xy] \to v$ exists in G and, by Proposition 5.8, $\mathrm{lca}_G(ab) = v$ holds. Moreover, Proposition 5.8 implies $[xy] = \mathrm{lca}_G(xy)$. In summary, $\mathrm{lca}_G(ab) = v \prec_G [xy] = \mathrm{lca}_G(xy)$. Hence, Condition **(I1)** holds.

Assume, now that $(xy,ab) \in \mathrm{tc}(R)$. Recall that $\mathrm{tc}(R) \subseteq R^+$, so we have $(ab,xy),(xy,ab) \in R^+$. Hence ab and xy are in the same $\sim_{R^+}$-class, i.e., $[ab] = [xy]$ refer to the same vertex in G. By Proposition 5.8, we have $[xy] = \mathrm{lca}_G(xy)$ and $[ab] = \mathrm{lca}_G(ab)$ which together with $[ab] = [xy]$ implies that $\mathrm{lca}_G(xy) = \mathrm{lca}_G(ab)$. Thus, Condition **(I2)** holds. In summary, G realizes R. Hence, Condition (2) implies Condition (3).

Clearly, Condition (3) implies (1). In summary, the three statement (1), (2) and (3) are equivalent. □

Corollary 5.11. *Let R be a relation such that $R = R^+$. Then R satisfies* **(X2)**. *In particular, R is realizable if and only if it satisfies* **(X1)**.

Proof. Let R be a relation such that $R = R^+$. Since R^+ is transitive and $R = R^+$, we have $\mathrm{tc}(R) = \mathrm{tc}(R^+) = R^+$. Therefore, Condition **(X2)** is trivially satisfied. Thus, if R satisfies **(X1)**, Theorem 5.10 implies that R is realizable. Conversely, if R is realizable, then R satisfies **(X1)** by Theorem 5.10. □

The latter results also allow us to characterize strictly realizable as follows.

Theorem 5.12. *For a relation R the following statements are equivalent:*

1. *R is strictly realizable.*
2. *R satisfies* **(X1)** *and* **(X2)** *and* $\mathrm{tc}(R)$ *is asymmetric.*
3. *R is realized by its canonical DAG G_R and* $\mathrm{tc}(R)$ *is asymmetric.*
4. *$R \subseteq \blacktriangleleft_{G_R}$ for the canonical DAG G_R of R.*
5. *$R \subseteq \blacktriangleleft_G$ for some DAG G.*

Proof. By Theorem 5.10 together with Lemma 3.3, Condition (1), (2) and (3) are equivalent. Suppose that Condition (3) holds. In this case, Lemma 3.3 implies that R is strictly realized by G_R. By Lemma 3.9, $R \subseteq \blacktriangleleft_{G_R}$ and thus, Condition (4) is satisfied. Clearly, Condition (4) implies (5). By Lemma 3.9, Condition (5) implies Condition (1). □

We conclude this section with a brief discussion of "minimal" and "minimum" DAGs realizing a relation R. Observe first that the canonical DAG G_R of a realizable relation R is, in general, not the only DAG that realizes R. Moreover, the Examples in Figure 6 and Figure 9 show that G_R is not necessarily *minimum*, i.e., it does not always have the minimum number of vertices among all DAGs G that realize R.

Although the canonical DAG is not necessarily minimum with respect to the number of vertices, it may still be minimal in a structural sense. To formalize this, we call a DAG G that realizes a relation R *minimal* if every DAG H obtained from G by a sequence of arc contractions[1] fails to realize R. However, as the following example illustrates, G_R is not always minimal.

[1] In an arc contraction, an arc (u,v) is removed from G and u and v are identified, and then any resulting multi-arcs are suppressed to give a single arc.

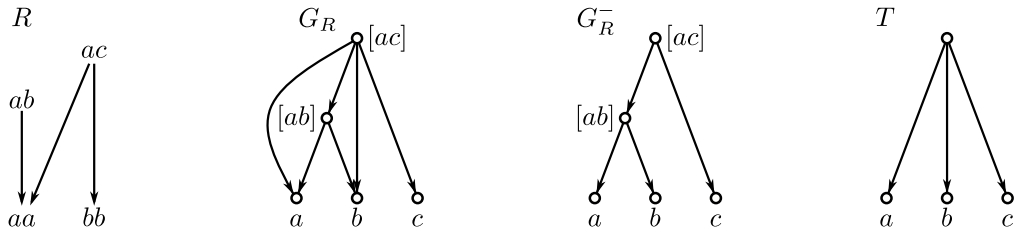

Figure 7: On the left we give a graphical representation of the relation relation $R = \{(aa,ac),(bb,ac),(aa,ab)\}$ on $X = \{a,b,c\}$ as in Example 5.13. Here we draw an arc $p \to q$ precisely if $q\ R\ p$. In addition, the canonical DAG G_R, the DAG G_R^- that is obtained from G_R by removal of all shortcuts and a DAG T is shown. Both, T and G_R^- are trees and realize R. Moreover, contracting the arc $ac \to ab$ in G_R, respectively, G_R^- yields T. Hence, neither G_R nor G_R^- is a "minimal" DAG realizing R.

Example 5.13 (The canonical DAG is not minimal)**.** *Let $R = \{(aa,ac),(bb,ac),(aa,ab)\}$ be a relation on $X = \{a,b,c\}$. One easily verifies that the star-tree T on X realizes R. By cross-consistency and since $(aa,ac),(bb,ac) \in R$ and $ab \in \mathrm{supp}_{R^+}$ it follows that $(ab,ac) \in R^+$. It is also easy to see that $(ac,ab) \notin R^+$. The canonical DAG G_R is shown in Figure 7. One easily verifies that T can be obtained from G_R by contracting the arc $ac \to ab$ (and removal of resulting multiple arcs). Equivalently, T is obtained from G_R^- by contracting the arc $ac \to ab$. Hence, neither G_R nor G_R^- is minimal.*

6 The Canonical Network: Properties and Algorithmic Construction

In the previous section, we saw how to realize a relation R by its canonical DAG G_R. In this section, we focus on realizing a relation via a canonical *network* N_R. This network is obtained by a fairly simple modification of G_R. Interestingly, we will show that N_R is regular, meaning that it is closely related to the Hasse diagram of its underlying set system, which arises by taking the descendants in X of vertices in N_R. In addition, we present a polynomial-time algorithm for computing both G_R and N_R.

We start by relating DAGs to set systems. For a DAG G on X and a vertex $v \in V(G)$ we put $\mathfrak{C}_G(v) = \{x \in X \mid x \preceq_G v\}$, i.e. the set of leaves that are descendants of v. This set is called the *cluster* of v, and we let $\mathfrak{C}_G = \{\mathfrak{C}_G(v) \mid v \in V(G)\}$ denote the set system on X comprising all clusters of G. Clearly $\mathfrak{C}_G$ is grounded for all DAGs G. In the opposite direction, we can also associate a DAG to a given set system $\mathfrak{C}$ as follows. The *Hasse diagram* $\mathcal{H}(\mathfrak{C})$ of a set system $\mathfrak{C} \subseteq \mathscr{P}(X)$ is the DAG with vertex set $\mathfrak{C}$ and arc from $A \in \mathfrak{C}$ to $B \in \mathfrak{C}$ if (i) $B \subsetneq A$ and (ii) there is no $C \in \mathfrak{C}$ with $B \subsetneq C \subsetneq A$. We note that $\mathcal{H}(\mathfrak{C})$ is also known as the *cover digraph* of $\mathfrak{C}$ [4]. We now give a key definition.

Definition 6.1 ([4])**.** *A DAG $G = (V,E)$ is* regular *if the map $\varphi\colon V \to V(\mathcal{H}(\mathfrak{C}_G))$ defined by $v \mapsto \mathfrak{C}_G(v)$ is an isomorphism between G and $\mathcal{H}(\mathfrak{C}_G)$.*

We emphasize that not every Hasse diagram $\mathcal{H}(\mathfrak{C})$ is regular. For example, consider the set system $\mathfrak{C} = \{\{x\},\{x,y\}\}$ where $\mathcal{H}(\mathfrak{C})$ consists of a single arc and where each vertex v satisfies $\mathfrak{C}(v) = \{\{x\}\}$. Then the map $\varphi\colon V(\mathcal{H}(\mathfrak{C}_G)) \to V(\mathcal{H}(\mathfrak{C}_G))$ defined by $v \mapsto \mathfrak{C}(v)$ maps both of the vertices $\{x,y\}$ and $\{x\}$ to $\{x\}$ and thus, does not yield an isomorphism. However, as shown in [33, L. 4.7], $\mathcal{H}(\mathfrak{C})$ is regular and phylogenetic, whenever $\mathfrak{C}$ is grounded.

We now show that if R is a realizable relation, then the DAG G_R^- obtained by removing short cuts from G_R enjoys some useful properties.

Lemma 6.2. *If R is realizable, then G_R^- is a phylogenetic, 2-lca-relevant and regular DAG on X that realizes R.*

Proof. Suppose that R is a realizable relation on $\mathscr{P}_2(X)$. By Theorem 5.10, the canonical DAG $G := G_R$ is a DAG on X realizes R. By Lemma 2.3, G^- is a DAG on X that is uniquely determined and shortcut-free. Observation 2.4 implies that $v = \mathrm{lca}_G(xy)$ implies $v = \mathrm{lca}_{G^-}(xy)$ for all $v \in V(G) = V(G^-)$. Since, by Proposition 5.8, G is 2-lca-relevant, it follows that G^- is 2-lca-relevant and, hence, lca-relevant. Even more, Lemma 2.3 implies that $u \preceq_G v$ if and only if $u \preceq_{G^-} v$. Taking the latter arguments together, G^- realizes R. Since G^- is shortcut-free and lca-relevant, we can apply [33, Thm. 4.10] to conclude that G^- is regular. Since G^- is a DAG on X, we have $\mathfrak{C}_G(x) = \{x\}$ and, thus, the clustering $\mathfrak{C}_G$ is grounded. By [33, L. 4.7], G^- is phylogenetic. □

We now define the canonical network for a relation.

Definition 6.3 (Canonical Network)**.** *Suppose that R is a realizable relation on $\mathscr{P}_2(X)$. Let G_R be the canonical DAG which, by Theorem 5.10 realizes R. We construct the* canonical network *N_R from G_R as follows.*

- *Remove all shortcuts from G_R resulting in G_R^-.*
- *If G_R^- is a network, then put $N_R := G_R^-$.*
  Else, G_R^- has roots $\rho_1,\ldots,\rho_k$, $k \geq 2$, and we obtain N_R by first adding a new vertex ρ and then adding the arcs $\rho \to \rho_i$, for all $1 \leq i \leq k$.

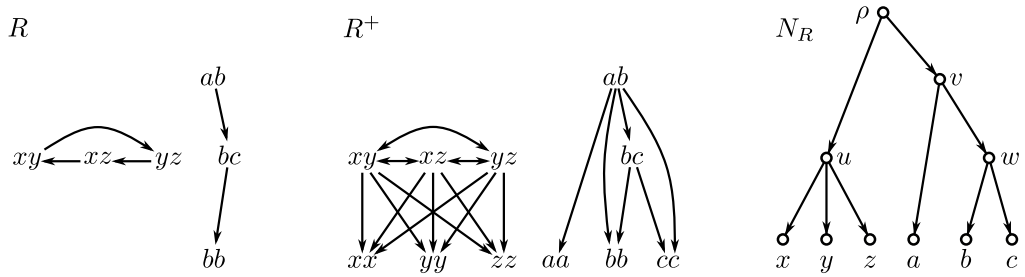

Figure 8: On the left we give a graphical representation of the relation relation R. Here we draw an arc $p \to q$ precisely if $q\ R\ p$. In addition, the graphical representation of R^+ is provided where arcs (p,p) are omitted for all $p \in \mathrm{supp}_R^+$. Furthermore, the canonical network N_R is shown. In this example G_R, which is distinct from N_R, is obtained from N_R by removal of the root ρ and its incident arcs. In N_R, we have $u = [xy] = [xz] = [yz]$, $v = [ab]$, and $w = [bc]$.

An example of a canonical network N_R that is distinct from G_R^- is shown in Figure 8. We now show that, as well as some enjoying other properties, the canonical network is always regular. Note that, even so, regular DAGs or networks that realize R are not necessarily uniquely determined – see e.g. Figure 9.

Proposition 6.4. *If R is realizable, then the canonical network N_R realizes R. Moreover, N_R is regular and phylogenetic and satisfies $[ab] = \mathrm{lca}_{N_R}(ab)$ for all vertices $ab \in \mathrm{supp}_R^+$ with $a \neq b$ and $x = \mathrm{lca}_{N_R}(xx)$ for all $x \in X$.*

Proof. Suppose that R is a realizable relation on $\mathscr{P}_2(X)$. In the following, we put $G := G_R$ and $N := N_R$. By Lemma 6.2, G^- is phylogenetic, 2-lca-relevant, regular and realizes R. Since G^- realizes R, $\mathrm{lca}_{G^-}(ab)$ is well-defined for all $ab \in \mathrm{supp}_R^+$.

Suppose first that G^- is already a network. By construction, $N = G^-$. By Theorem 5.10, R satisfies Condition **(X1)**. This together with Lemma 2.3, Observation 2.4 and Proposition 5.8 implies that $[ab] = \mathrm{lca}_N(ab)$ for all vertices $[ab] \in V(G)$ with $a \neq b$ and $x = \mathrm{lca}_N(xx)$ for all $x \in X$. Note that, by construction of G, for every $a, b \in X$ such that $a \neq b$, we have $[ab] \in V(G)$ if and only if $ab \in \mathrm{supp}_R^+$. Hence, if G^- is a network, then we are done.

Suppose now that G^- is not a network. In this case, G^- has roots $\rho_1, \ldots, \rho_k$ with $k \geq 2$. We first argue that $\mathtt{C}_{G^-}(\rho_i) \neq X$, $1 \leq i \leq k$. Assume, for contradiction, that $\mathtt{C}_{G^-}(\rho_i) = X$ for some i. In this case, $\mathtt{C}_{G^-}(v) \subseteq \mathtt{C}_{G^-}(\rho_i)$ for all $v \in V(G^-)$. Since G^- is regular, [33, L. 4.7] implies that G^- satisfies the so-called path-cluster-comparability (PCC) property which, in particular, implies that ρ_i and ρ_j must be $\preceq_{G^-}$-comparable for all j; a contradiction to ρ_i and ρ_j being roots for at least two distinct i and j. Consequently, $\mathtt{C}_{G^-}(\rho_i) \neq X$ for each $1 \leq i \leq k$.

Now, recall that we construct N by taking G^- and adding a new vertex ρ and arcs $\rho \to \rho_i$ for all $1 \leq i \leq k$. It is easy to verify that N is a phylogenetic network. To show that N is regular, we show first that N is lca-relevant. By Lemma 6.2, G^- is 2-lca-relevant, i.e., for all $v \in V(G^-)$ there are leaves $x, y \in X$ (not necessarily distinct) such that $v = \mathrm{lca}_{G^-}(xy)$. It is straightforward to verify that $v = \mathrm{lca}_N(xy)$ whenever $v \in V(G^-) = V(N) \setminus \{\rho\}$ and $v = \mathrm{lca}_{G^-}(xy)$. We show now that $\rho = \mathrm{lca}_N(X)$. As argued above, all children of ρ_i of ρ in N, satisfy $\mathtt{C}_N(\rho_i) \neq X$. Moreover, any descendant v of ρ_i must satisfy $\mathtt{C}_N(v) \subseteq \mathtt{C}_N(\rho_i)$ (e.g. cf. [20, L.17]). Hence, $\mathtt{C}_N(v) \neq X$ for all $v \in V(N) \setminus \{\rho\}$. Since ρ is an ancestor of all leaves in X and since there is no vertex $v \prec_N \rho$ with $\mathtt{C}_N(v) = X$, it follows that $\mathrm{lca}_N(X) = \rho$. In summary, for all vertices $v \in V(N)$ there is a subset $A \subseteq X$ such that $v = \mathrm{lca}_N(A)$. Therefore, N is lca-relevant.

We show now that N is shortcut-free. Observe first that the subgraph G^- of N is shortcut-free by Lemma 2.3. Hence any shortcut in N must be of the form $\rho \to v$ for some $v \in V(N)$. Since ρ is only adjacent to $\{\rho_1, \ldots, \rho_k\}$, a shortcut must be of the form $\rho \to \rho_i$ for some $1 \leq i \leq k$. However, $\rho \to \rho_i$ being a shortcut requires an alternative $\rho\rho_i$-path not containing the arc $\rho \to \rho_i$. Since, by construction, no such path exists in N, N is shortcut-free. Since N is, in addition, lca-relevant, we can apply [33, Thm. 4.10] to conclude that N is regular.

Finally, recall that $\mathrm{lca}_{G^-}(ab)$ is well-defined for all $ab \in \mathrm{supp}_R^+$. Note that, if $ab \in \mathrm{supp}_R^+$, then $\mathrm{lca}_{G^-}(ab)$ is contained in $V(G^-) = V(N) \setminus \{\rho\}$. Moreover, G^- is an induced subgraph of N. It is now straightforward to verify that N realizes R and that $[ab] = \mathrm{lca}_N(ab)$ for all vertices $ab \in \mathrm{supp}_R^+$ with $a \neq b$ and $x = \mathrm{lca}_N(xx)$ for all $x \in X$. □

We conclude this section by presenting the polynomial-time algorithm which checks whether or not a relation R is realizable and, if so, returns the canonical DAG and network realizing R. The algorithm is presented in Algorithm 1 and its correctness and runtime analysis is given in the following result.

Theorem 6.5. *For a relation R on $\mathscr{P}_2(X)$, verifying whether R is realizable and, if so, constructing a DAG on X that realizes R can be done in polynomial time in $|X|$ using Algorithm 1. In particular, each one of G_R,G_R^- and N_R can be constructed in polynomial time, where G_R is the canonical DAG and N_R the canonical network of R.*

Proof. Let R be a relation on $\mathscr{P}_2(X)$. By Theorem 5.10, R is realizable if and only if it satisfies **(X1)** and **(X2)**. It is straightforward to verify that `false` is correctly returned by Algorithm 1 whenever R is not realizable by any DAG G. Lines 3 to Line 7 mimic the steps used to construct the canonical DAG G_R (see also the corresponding lemmas and definitions referenced in Algorithm 1). Since G_R is computed only when R satisfies **(X1)** and **(X2)**, Theorem 5.10 ensures that G_R is indeed a DAG that realizes R, whenever R is realizable.

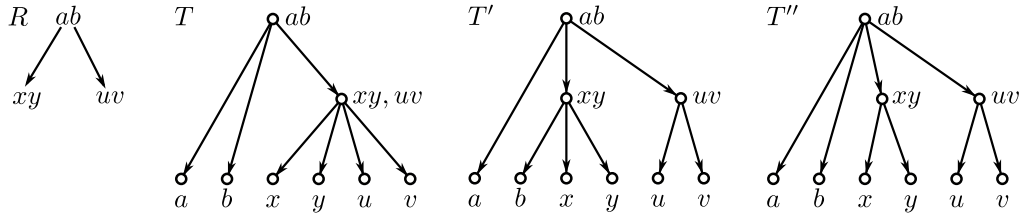

Figure 9: Three non-isomorphic phylogenetic trees T, T' and T'' on $X = \{a,b,x,y,u,v\}$ that all realize the relation R with $xy\ R\ ab$ and $uv\ R\ ab$. Each pair $p \in \{ab,xy,uv\}$ in the drawings marks the corresponding vertex $\mathrm{lca}_{T^*}(p)$ in the tree $T^* \in \{T,T',T''\}$. Here, $T'' \simeq G_R^- = N_R$. Moreover, all three trees are regular and neither of them can be obtained from the other by contraction of arcs, i.e., they are minimal. However, only T has a minimum number of vertices among all DAGs realizing R.

Algorithm 1 REALIZABILITY OF A RELATION R

Input: A binary relation R on $\mathscr{P}_2(X)$
Output: Returns the canonical DAG G_R and network N_R realizing R if R is realizable and, otherwise, `false` is returned.

1: Compute supp_R^+
2: Compute R^+ according to Theorem 4.5
3: **if** R satisfies Conditions **(X1)** and **(X2)** **then**
4: Compute the equivalence relation $\sim_{R^+}$ on supp_R^+ as defined in Lemma 5.3
5: Compute the set Q comprising all $\sim_{R^+}$-classes
6: Compute the partial order $\leq_{R^+}$ on the classes in Q as defined in Lemma 5.4
7: Compute the canonical DAG G_R according to Definition 5.5
8: Compute the canonical network N_R according to Definition 6.3
9: **return** G_R and N_R
10: **else return** `false`

For the runtime, observe first that supp_R^+ can clearly be determined in polynomial time in $|X|$ (Line 1). Moreover, Theorem 4.5 implies that R^+ can also be determined in polynomial time in $|X|$ (Line 2). As outlined at the end the proof of Theorem 4.5, we have $|R| \in O(|X|^4)$. By similar arguments, $|R^+| \in O(|X|^4)$. We can thus assume that R is provided as an $|X|^2 \times |X|^2$ matrix A with entries 0 and 1 such that $A_{uv,xy} = 1$ if and only of $uv\ R\ xy$. Hence, A is the adjacency matrix of a directed graph, here called $G(A)$. To verify **(X1)** in Line 3, we have to check for all $x \in X$ and the $O(|X|^2)$ sets $ab \in \mathscr{P}_2(X)$ with $a \neq b$ if $(ab,xx) \in R^+$ or not. Since we can check in constant time as whether or not $(ab,xx) \in R^+$ based on the matrix A, **(X1)** can be verified in $O(|X|^3)$ time. To check **(X2)** in Line 3 we must verify if, for all $(ab,xy) \in \mathrm{tc}(R)$ and $(xy,ab) \notin \mathrm{tc}(R)$, we have $(xy,ab) \notin R^+$. We can compute the transitive closure of R in the graph $G(A)$ with the Floyd-Warshall Algorithm in polynomial time in $|X|$. Hence, **(X2)** can be verified in polynomial time. Computing $\sim_{R^+}$, Q, and $\leq_{R^+}$ as in Line 4 - 6 can all be achieved in polynomial time in $|X|$, since their computation requires only a finite number of comparison of elements in R^+. Moreover, it is easy to verify that the graph G_R (Line 7) as specified in Def. 5.5 can be computed in polynomial time in $|X|$. Finding shortcuts in G_R can be done naively by removing the arc (u,v) from G_R and checking whether a uv-path remains in the resulting graph using a depth-first search. Since G_R, by Proposition 5.8, is 2-lca-relevant, it follows that each vertex in G_R is the unique least common ancestor of at least two vertices out of $|\mathscr{P}_2(X)|$ possible leaf combinations. We can, thus, conclude that the number of vertices and arcs in G_R is bounded by $|\mathscr{P}_2(X)| \in O(|X|^2)$ and $O(|X|^4)$, respectively. Hence, finding all shortcuts in G_R and removing them to obtained G_R^- can be achieved in in polynomial time in $|X|$. It is now an easy task to verify that N_R can be constructed in in polynomial time in $|X|$. Hence, Algorithm 1 runs in polynomial time in $|X|$. □

It is easy to check that verifying if a relation is asymmetric can be performed in polynomial time. This together with Theorem 5.12 and Theorem 6.5 implies the following result.

Theorem 6.6. *Verifying whether a relation R is strictly realizable and, if so, constructing a DAG that strictly realizes R can be done in polynomial time in $|X|$.*

7 The Classical Closure and Closed Relations

In previous sections, we defined the closure R^+ of a relation R and then showed how to use this to construct the canonical DAG for a realizable relation. However, a more natural way to define the closure for a realizable relation is to follow the approach used for so-called triplets or quartets in trees [5, 6, 32, 41]. More specifically, we define the closure $\mathrm{cl}(R)$ as the intersection of all relations $\preceq_G$ over DAGs G that realize R. In this section we will show that, somewhat surprisingly, these two ways of taking closures are actually equivalent (see Theorem 7.5).

We begin by formally defining the alternative notion of closure.

Definition 7.1 (Classical closure)**.** *Let R be a relation on $\mathscr{P}_2(X)$ that is realizable and let $\mathfrak{G}$ denote the set of all DAGs G on X that realize R. Then, we define the closure of R as*

$$\mathrm{cl}(R) := \bigcap_{G \in \mathfrak{G}} \preceq_G .$$

In particular, the last definition implies that $\mathrm{cl}(R)$ contains all pairs (ab,xy) for which $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$ must hold in any DAG G that realizes R.

Now recall that in order to show that $\mathrm{cl}(R)$ is a "valid" closure for realizable relations, we must ensure that $\mathrm{cl}(R)$ satisfies the following three properties:

1. *Extensivity:* $R \subseteq \mathrm{cl}(R)$.
2. *Monotonicity:* If $R_1 \subseteq R_2$ for two realizable relations on $\mathscr{P}_2(X)$, then $\mathrm{cl}(R_1) \subseteq \mathrm{cl}(R_2)$.
3. *Idempotency:* $\mathrm{cl}(\mathrm{cl}(R)) = \mathrm{cl}(R)$.

A particular difficulty in proving these axioms directly lies in the fact that $\mathrm{cl}(R)$ is defined only on *realizable* relations. Hence, to show that $\mathrm{cl}(\mathrm{cl}(R))$ can be applied, we must verify that $\mathrm{cl}(R)$ is realizable by some DAG. This, however, is not trivial. Clearly, $\mathrm{cl}(R) \subseteq \trianglelefteq_G$ for all DAGs $G \in \mathfrak{G}$ that realize R. However, as Example 3.7 shows, this does not necessarily imply that any of the DAGs $G \in \mathfrak{G}$ realize $\mathrm{cl}(R)$. In fact, as shown in Example 3.8, there are subsets of realizable relations that are not realizable. Thus, we cannot assume a priori that $\mathrm{cl}(R)$ is realizable by any DAG. To get around this issue, we could have defined $\mathrm{cl}(R)$ for all relations, realizable or not. In this case, $\mathrm{cl}(R) = \emptyset$ for all non-realizable relations R. However, we would be faced with a similar problem. If R is non-empty and realizable but $\mathrm{cl}(R)$ is not realizable, then $R \not\subseteq \mathrm{cl}(R) = \emptyset$; violating extensivity.

We now proceed with proving that $\mathrm{cl}(R) = R^+$ and thus, that $\mathrm{cl}(R)$ is realizable for any realizable relation R. We first establish some further results. The first (in particular Condition (2)) strengthens Lemma 4.9.

Lemma 7.2. *Let R be a realizable relation on $\mathscr{P}_2(X)$ and $H \in \{G_R, G_R^-, N_R\}$ with G_R being the canonical DAG and N_R being the canonical network of R. Then, the following statements hold:*

1. *H realizes R.*
2. *For all $ab,xy \in \mathrm{supp}_R^+$, $\mathrm{lca}_H(ab)$ and $\mathrm{lca}_H(xy)$ are well-defined, and the following two conditions hold*
   (a) $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy) \iff ab\ R^+\ xy$.
   (b) $\mathrm{lca}_H(ab) \prec_H \mathrm{lca}_H(xy) \iff ab\ R^+\ xy$ *and* $(xy,ab) \notin R^+$.
3. *H realizes R^+.*

Proof. Let R be a realizable relation on $\mathscr{P}_2(X)$ and $H \in \{G_R, G_R^-, N_R\}$ with G_R being the canonical DAG and N_R being the canonical network of R. For all choices of $H \in \{G_R, G_R^-, N_R\}$, Theorem 5.10, Lemma 6.2 respectively Proposition 6.4 imply that H realizes R. Hence, Condition (1) holds.

We continue with proving Condition (2). Since R is realizable, we can assume that Condition (1) holds, i.e., that H realizes R. By Observation 3.6, $\mathrm{lca}_H(uv)$ is well-defined for all $uv \in \mathrm{supp}_R^+$. We continue by proving the equivalence in Condition (2.a). Let $ab,xy \in \mathrm{supp}_R^+$. The *if*-direction is ensured by Lemma 4.9. For the *only if*-direction, assume that $\mathrm{lca}_H(ab) \preceq_N \mathrm{lca}_H(xy)$. Recall that the vertices of H consist of the leaves in X, the equivalence classes $[cd]$ with $cd \in \mathscr{P}_2(X)$ and $c \neq d$ and, in case that $H = N_R$, the unique root ρ that may or may not correspond to one of the equivalence classes (artificially added to G_R^- in the latter case). Since, by assumption, $ab,xy \in \mathrm{supp}_R^+$, the equivalence classes $[ab]$ and $[xy]$ will by construction "correspond" to some vertex of H, but are possibly relabeled by a leaf in X. To circumvent this technicality, we let $v = [ab]$ if $a \neq b$ and, otherwise, $v = a$. Moreover, let $w = [xy]$ if $x \neq y$ and, otherwise, $w = x$. If $H = G_R$ or $H = N_R$, then Proposition 5.8 respectively Proposition 6.4 implies that $v = \mathrm{lca}_H(ab)$ and $w = \mathrm{lca}_H(xy)$. If $H = G_R^-$, the latter together with Observation 2.4 implies that $v = \mathrm{lca}_H(ab)$ and $w = \mathrm{lca}_H(xy)$. Since we have assumed that $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy)$ and for all possibilities of H we have shown that $v = \mathrm{lca}_H(ab)$ and $\mathrm{lca}_H(xy) = w$, we conclude that $v \preceq_H w$. Clearly, if $H = G_R$, we have $v \preceq_{G_R} w$. If $H = G_R^-$, then Lemma 2.3 implies that $v \preceq_{G_R} w$. If $H = N_R$, then the latter together with the fact that G_R^- differs from N_R only by possibly a newly added root implies $v \preceq_{G_R} w$. In summary, $v \preceq_H w$ implies that $v \preceq_{G_R} w$, for any choice $H \in \{G_R, G_R^-, N_R\}$. By construction of G_R, the latter ensures that $[ab] \leq_{R^+} [xy]$ and by definition of $\leq_{R^+}$, $ab\ R^+\ xy$ must hold. Therefore, Condition (2.a) holds. Condition (2.b) immediately follows from (2.a), since $\mathrm{lca}_H(ab) \prec_H \mathrm{lca}_N(xy)$ if and only if $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy)$ and $\mathrm{lca}_H(xy) \not\preceq_H \mathrm{lca}_G(ab)$. In summary, Condition (2) holds.

We continue with proving Condition (3), which we have already proven for the case $H = G_R$ in Proposition 5.8. Since R is realizable, we can assume that Condition (2) holds. Let $H \in \{G_R, G_R^-, N_R\}$. Since R^+ is transitive, it holds that $R^+ = \mathrm{tc}(R^+)$. Let $(ab,xy) \in \mathrm{tc}(R^+) = R^+$. If $(xy,ab) \notin \mathrm{tc}(R^+) = R^+$, then Condition (2.b) implies that $\mathrm{lca}_H(ab) \prec_H \mathrm{lca}_H(xy)$, i.e., **(I1)** holds for R^+ and H. If $(xy,ab) \in \mathrm{tc}(R^+) = R^+$, then Condition (2.a) implies that $\mathrm{lca}_H(ab) = \mathrm{lca}_H(xy)$, i.e., **(I2)** holds for H and R^+. Hence, H realizes R^+. □

Lemma 7.2 shows that the realizability of R implies realizability of R^+. However, R might be realized by some DAG G that does not realize R^+, i.e., Conditions (2) and (3) in Lemma 7.2 are not necessarily satisfied for arbitrary DAGs. To see this, we provide an example.

Example 7.3 (G realizes R but not R^+)**.** *Consider the relation $R = \{(aa,ac),(bb,ac),(aa,ab)\}$ on $X = \{a,b,c\}$ as shown in Example 5.13. As argued in Example 5.13, the star-tree T on X realizes R and we have $(ab,ac) \in R^+$ and*

$(ac,ab) \notin R^+$. *Since* R^+ *is transitive, we have* $\mathrm{tc}(R^+) = R^+$. *Hence,* $(ab,ac) \in R^+$ *and* $(ac,ab) \notin R^+$ *implies together with* **(I1)** *that any DAG* G *realizing* R^+ *must satisfy* $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(ac)$. *Since* $\mathrm{lca}_T(ab) = \mathrm{lca}_T(ac)$ *it follows that* T *does not realize* R^+. *Hence, Condition (3) in Lemma 7.2 does not hold for* $H = T$. *In particular,* $\mathrm{lca}_T(ac) \preceq_T \mathrm{lca}_T(ab)$ *holds, but* $(ac,ab) \notin R^+$, *i.e., Condition (2) in Lemma 7.2 does not hold.*

As stated in Section 4, R^+ contains additional information about any DAG G that could potentially realize R. Example 7.3 may seem somewhat in contrast to this statement, since G may realize R but does not necessarily realize R^+. However, in light of Lemma 4.9, the statement is consistent: for any DAG G realizing R, if $ab\ R^+\ xy$ then $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$. In particular, for Example 7.3, we have $(ab,ac) \in R^+$ and $\mathrm{lca}_T(ab) \preceq_T \mathrm{lca}_T(ac)$ for the star-tree realizing $R = \{(aa,ac),(bb,ac),(aa,ab)\}$. We can go even further: we now show that for every realizable relation R, there exists a network G that realizes R and satisfies $R^+ = \trianglelefteq_G$, which strengthens the fact that $R^+ \subseteq \trianglelefteq_G$ (cf. Lemma 4.9).

Lemma 7.4. *For every realizable relation* R, *there exists a network* G *that realizes* R *and satisfies* $\trianglelefteq_G = R^+$.

Proof. Let R be a realizable relation on $\mathscr{P}_2(X)$. By Lemma 7.2, R^+ is realized by the canonical network $N := N_R$. Recall from Theorem 4.5 that $\mathrm{supp}_{R^+} = \mathrm{supp}_R^+$. Let ρ_N denote the unique root of N. In the following, we say that a DAG G with $V(N) \subseteq V(G)$ is $\preceq_N$-preserving if the following property is satisfied: $v \preceq_N u$ if and only if $v \preceq_G u$ for all $u,v \in V(N)$. Note that, by definition, $xx \in \mathrm{supp}_R^+$ for all $x \in X$. Therefore, $ab \notin \mathrm{supp}_{R^+} = \mathrm{supp}_R^+$ implies $a \neq b$.

Now, let G be the directed graph obtained from N by adding, for all $ab \in \mathscr{P}_2(X)$ such that $ab \notin \mathrm{supp}_{R^+}$, two new vertices v_{ab} and u_{ab}, and the six arcs $v_{ab} \to a$, $v_{ab} \to b$, $\rho_N \to v_{ab}$, $u_{ab} \to a$, $u_{ab} \to b$ and $\rho_N \to u_{ab}$. As N is a network on X, one easily observes that the leaf set of G remains X. In G, both the vertex v_{ab} and the vertex u_{ab} have ρ_N as its unique parent and precisely two children, and these two children are leaves in X. It is, therefore, straightforward to verify that G remains a network. By construction, $V(N) \subseteq V(G)$ and G is $\preceq_N$-preserving.

We now show that $\trianglelefteq_G = R^+$. Suppose first that $ab,xy \in \mathscr{P}_2(X)$ are such that $ab\ R^+\ xy$. Since N realizes R^+, the LCAs $\mathrm{lca}_N(ab)$ and $\mathrm{lca}_N(xy)$ must be well-defined, and satisfy $\mathrm{lca}_N(ab) \preceq_N \mathrm{lca}_N(xy)$. Moreover, $ab\ R^+\ xy$ implies that $ab,xy \in \mathrm{supp}_{R^+}$. Note that if $v \in V(G) \setminus V(N)$, then $v = v_{cd}$ or $v = u_{cd}$ for some $cd \notin \mathrm{supp}_{R^+}$. In particular, the descendants of v_{cd} are precisely v_{cd}, c and d and the descendants of u_{cd} are precisely u_{cd}, c and d. Taking the latter arguments together, there is no vertex $v \in V(G) \setminus V(N)$ such that v is an ancestor of both a and b in G, and there is no vertex $v \in V(G) \setminus V(N)$ such that v is an ancestor of both x and y in G. In other words, all common ancestors of a and b in G as well as of x and y in G are located in $V(N) \subseteq V(G)$. This, together with the fact that G is $\preceq_N$-preserving, implies that $\mathrm{lca}_G(ab) = \mathrm{lca}_N(ab)$ and $\mathrm{lca}_G(xy) = \mathrm{lca}_N(xy)$ are well-defined and, moreover, that $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$. To summarize up to this point, we have shown that $ab\ R^+\ xy$ implies that $ab \trianglelefteq_G xy$ i.e. that $R^+ \subseteq \trianglelefteq_G$.

To show that $\trianglelefteq_G \subseteq R^+$, suppose that $ab,xy \in \mathscr{P}_2(X)$ are such that $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(xy)$ are well-defined, and satisfy $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$. We argue first that $ab \in \mathrm{supp}_{R^+}$. To this end, assume for contradiction, that $ab \notin \mathrm{supp}_{R^+}$. In this case, the new vertices v_{ab} and u_{ab} were added to G. It is easy to verify that, by construction, v_{ab} and u_{ab} are both contained in $\mathrm{LCA}_G(ab)$, which contradicts the assumption that $\mathrm{lca}_G(ab)$ is well-defined. Hence, $ab \in \mathrm{supp}_{R^+}$ must hold. Using similar arguments, we can conclude that $xy \in \mathrm{supp}_{R^+}$. Since $\mathrm{lca}_G(ab)$ is, by assumption, well-defined and by the arguments in the preceding paragraph, we have $\mathrm{lca}_G(ab) \in V(N)$. In particular, $\mathrm{lca}_G(ab) = \mathrm{lca}_N(ab)$ holds as G is $\preceq_N$-preserving. Similarly, $\mathrm{lca}_G(xy) = \mathrm{lca}_N(xy)$. Since $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$ and since G is $\preceq_N$-preserving it follows that $\mathrm{lca}_N(ab) \preceq_N \mathrm{lca}_N(xy)$. This together with Lemma 7.2 implies that $ab\ R^+\ xy$. We have thus shown that $\trianglelefteq_G \subseteq R^+$, concluding the equality $\trianglelefteq_G = R^+$.

Finally, we show that G realizes R. To this end, let $(ab,cd) \in \mathrm{tc}(R)$. By Observation 4.8, $\mathrm{tc}(R) \subseteq R^+$ and, therefore, $(ab,cd) \in R^+$. Since $\trianglelefteq_G = R^+$ it follows that $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(cd)$ are well-defined and satisfy $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$.

There are now two cases to consider, either $(cd,ab) \notin \mathrm{tc}(R)$ or $(cd,ab) \in \mathrm{tc}(R)$. Suppose first that $(cd,ab) \notin \mathrm{tc}(R)$. Since R is realizable, Theorem 5.10 ensures that R satisfies **(X2)**. Hence, $(cd,ab) \notin R^+$. This and $\trianglelefteq_G = R^+$ implies that $\mathrm{lca}_G(cd) \not\preceq_G \mathrm{lca}_G(ab)$. This together with $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$ implies that $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. Thus, Condition **(I1)** is satisfied. Suppose now that $(cd,ab) \in \mathrm{tc}(R)$. Since, by Observation 4.8, $\mathrm{tc}(R) \subseteq R^+$ and, by similar arguments as before, $\mathrm{lca}_G(cd) \preceq_G \mathrm{lca}_G(ab)$. This and $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$ implies that $\mathrm{lca}_G(ab) = \mathrm{lca}_G(cd)$. Thus, Condition **(I2)** is satisfied. In summary, G realizes R. □

We now prove that $\mathrm{cl}(R) = R^+$ and derive some further properties of $\mathrm{cl}(R)$.

Theorem 7.5. *Let* R *be a realizable relation on* $\mathscr{P}_2(X)$. *Then,* $\mathrm{cl}(R) = R^+$ *and* $\mathrm{cl}(R)$ *is realizable. Furthermore,* cl *is a closure operator, i.e., it satisfies the classical closure axioms* Extensivity, Monotonicity, *and* Idempotency. *In addition,* $\mathrm{cl}(R)$ *can be computed in polynomial time in* $|X|$ *by exhaustively applying the rules* **(R1)**, **(R2)** *and* **(R3)** *as specified in Theorem 4.5.*

Proof. Let R be as in the statement of the theorem. Furthermore, let $\mathfrak{G}$ denote the set of all DAGs G on X that realize R and $\mathfrak{R}$ the set of all relations S on $\mathscr{P}_2(X)$ that are supp_R^+-reflexive, transitive, cross-consistent, and satisfy $R \subseteq S$.

Let $G \in \mathfrak{G}$. By Proposition 4.10, $\trianglelefteq_G^+ = \trianglelefteq_G$. Hence, $\trianglelefteq_G$ is transitive, $\mathrm{supp}^+_{\trianglelefteq_G}$-reflexive and cross-consistent. Lemma 3.6 implies that $R \subseteq \trianglelefteq_G$ which in particular ensures that $\mathrm{supp}_R \subseteq \mathrm{supp}_{\trianglelefteq_G}$. As G is a DAG on X, $\trianglelefteq_G$ is a relation on $\mathscr{P}_2(X)$ and, thus, $\mathrm{supp}_R^+ \subseteq \mathrm{supp}^+_{\trianglelefteq_G}$. The latter two facts together with $\trianglelefteq_G$ being $\mathrm{supp}^+_{\trianglelefteq_G}$-reflexive ensure that $\trianglelefteq_G$ is supp_R^+-reflexive. Together with transitivity and cross-consistency of $\trianglelefteq_G$, we conclude that $\trianglelefteq_G \in \mathfrak{R}$. Hence, for all $G \in \mathfrak{G}$ we have $\trianglelefteq_G \in \mathfrak{R}$ which implies that $R^+ = \cap_{R' \in \mathfrak{R}} R' \subseteq \cap_{G \in \mathfrak{G}} \trianglelefteq_G = \mathrm{cl}(R)$. Moreover, since R is realizable, Lemma 7.4 ensures the existence of a DAG $G \in \mathfrak{G}$ for which $\trianglelefteq_G = R^+$. By definition, $\mathrm{cl}(R) \subseteq \trianglelefteq_G = R^+$. Thus, $\mathrm{cl}(R) = R^+$.

Since R is realizable, it satisfies **(X1)** (cf. Theorem 5.10). By Proposition 5.8, the canonical DAG G_R realizes $R^+ = \mathrm{cl}(R)$. Hence, $\mathrm{cl}(R)$ is realizable.

Furthermore, since $\mathrm{cl}(R) = R^+$ and since R^+ satisfies the closure axioms (cf. Proposition 4.4) if follows that cl satisfies the closure axioms. Moreover, $\mathrm{cl}(R) = R^+$ together with Theorem 4.5 imply that $\mathrm{cl}(R)$ can be computed in polynomial time in $|X|$ by exhaustively applying the rules **(R1)**, **(R2)** and **(R3)**. □

Observation 5.6 together with Theorem 7.5 immediately implies:

Corollary 7.6. *For all realizable relations R we have $G_R = G_{\mathrm{cl}(R)}$.*

Intriguingly, using the classical closure, we now show that for all DAGs G, the information contained in $\trianglelefteq_G$ is entirely determined by that contained in $\blacktriangleleft_G$ and its $+$-closure $\blacktriangleleft_G^+$.

Proposition 7.7. *For all DAGs G it holds that $\trianglelefteq_G = \trianglelefteq_G^+ = \mathrm{cl}(\trianglelefteq_G) = \blacktriangleleft_G^+ = \mathrm{cl}(\blacktriangleleft_G)$ and $\blacktriangleleft_G \neq \blacktriangleleft_G^+$. In particular, $\mathrm{supp}^+_{\trianglelefteq_G} = \mathrm{supp}^+_{\blacktriangleleft_G}$. Moreover, for all DAGs G and H, it holds that*

$$\blacktriangleleft_G = \blacktriangleleft_H \iff \trianglelefteq_G = \trianglelefteq_H .$$

Proof. Let G be a DAG. By Proposition 4.10 and Theorem 7.5, we have $\trianglelefteq_G = \trianglelefteq_G^+ = \mathrm{cl}(\trianglelefteq_G)$ and $\blacktriangleleft_G^+ = \mathrm{cl}(\blacktriangleleft_G)$. Hence, it suffices to show that $\trianglelefteq_G = \blacktriangleleft_G^+$. By definition, $\blacktriangleleft_G \subseteq \trianglelefteq_G$. By Proposition 4.4, $\blacktriangleleft_G^+ \subseteq \trianglelefteq_G^+$. As $\trianglelefteq_G^+ = \trianglelefteq_G$, we have $\blacktriangleleft_G^+ \subseteq \trianglelefteq_G$. To show that $\trianglelefteq_G \subseteq \blacktriangleleft_G^+$, let $(ab,xy) \in \trianglelefteq_G$ and thus, $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(xy)$ are well-defined and satisfy $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$. If $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)$, then $(ab,xy) \in \blacktriangleleft_G \subseteq \blacktriangleleft_G^+$ and we are done. Suppose that $\mathrm{lca}_G(ab) = \mathrm{lca}_G(xy)$. If $a = b$, then $a = \mathrm{lca}_G(ab) = \mathrm{lca}_G(xy)$ implies that $x = y = a$. Since $\blacktriangleleft_G^+$ is $\mathrm{supp}^+_{\blacktriangleleft_G}$-reflexive, we have $(aa,aa) = (ab,xy) \in \blacktriangleleft_G^+$. Suppose that $a \neq b$. Note that, in this case, $\mathrm{lca}_G(aa) \prec_G \mathrm{lca}_G(ab)$ and, therefore, $ab \in \mathrm{supp}_{\blacktriangleleft_G} \subseteq \mathrm{supp}_{\blacktriangleleft_G^+}$. Since $\mathrm{lca}_G(ab) = \mathrm{lca}_G(xy)$, we have $aa \ \blacktriangleleft_G \ xy$ and $bb \ \blacktriangleleft_G \ xy$. This together with $ab \in \mathrm{supp}_{\blacktriangleleft_G^+}$ and cross-consistency of $\blacktriangleleft_G^+$ implies that $(ab,xy) \in \blacktriangleleft_G^+$. Hence, $\trianglelefteq_G \subseteq \blacktriangleleft_G^+$ and, therefore, $\trianglelefteq_G = \blacktriangleleft_G^+$ holds. In summary, $\trianglelefteq_G = \trianglelefteq_G^+ = \mathrm{cl}(\trianglelefteq_G) = \blacktriangleleft_G^+ = \mathrm{cl}(\blacktriangleleft_G)$. This together with Theorem 4.5 implies that $\mathrm{supp}^+_{\blacktriangleleft_G} = \mathrm{supp}_{\blacktriangleleft_G^+} = \mathrm{supp}_{\trianglelefteq_G^+} = \mathrm{supp}^+_{\trianglelefteq_G}$.

We show now that $\blacktriangleleft_G \neq \blacktriangleleft_G^+$. Observe first that $\blacktriangleleft_G \neq \trianglelefteq_G$, since for any leaf $x \in X$, we have $(xx,xx) \in \trianglelefteq_G$ but $(xx,xx) \notin \blacktriangleleft_G$. As shown above, $\blacktriangleleft_G^+ = \trianglelefteq_G$. Taking the latter two arguments together, we obtain $\blacktriangleleft_G \neq \blacktriangleleft_G^+$.

Finally, let G and H be two DAGs. We show that $\blacktriangleleft_G = \blacktriangleleft_H \iff \trianglelefteq_G = \trianglelefteq_H$. If $\blacktriangleleft_G = \blacktriangleleft_H$, then $\blacktriangleleft_G^+ = \blacktriangleleft_H^+$ and, by the latter arguments, it holds that $\trianglelefteq_G = \trianglelefteq_H$. Suppose now that $\trianglelefteq_G = \trianglelefteq_H$. Assume, for contradiction that $\blacktriangleleft_G \neq \blacktriangleleft_H$. Thus, we can, without loss of generality, assume that there is some $(ab,xy) \in \blacktriangleleft_G$ such that $(ab,xy) \notin \blacktriangleleft_H$. Since $(ab,xy) \in \blacktriangleleft_G$, we have $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)$. Since $\blacktriangleleft_G \subseteq \trianglelefteq_G$ and $\trianglelefteq_G = \trianglelefteq_H$, if follows that $(ab,xy) \in \trianglelefteq_H$ and, hence, that $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy)$. This together with $(ab,xy) \notin \blacktriangleleft_H$ implies $\mathrm{lca}_H(ab) = \mathrm{lca}_H(xy)$. Hence, $(xy,ab) \in \trianglelefteq_H$. Since $\trianglelefteq_H = \trianglelefteq_G$ it follows that $(xy,ab) \in \trianglelefteq_G$ and, therefore, $\mathrm{lca}_G(xy) \preceq_G \mathrm{lca}_G(ab)$; a contradiction to $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(xy)$. Hence, $\blacktriangleleft_G = \blacktriangleleft_H$. □

Interestingly, using this last result, we can also show that it is sufficient to know $\trianglelefteq_H$ and supp^+_R to determine $\mathrm{cl}(R)$ for any realizable relation R, where H is its canonical DAG or network.

Theorem 7.8. *Let R be a realizable relation and $H \in \{G_R, G_R^-, N_R\}$ with G_R being the canonical DAG and N_R being the canonical network of R. Then it holds that*

$$\mathrm{cl}(R) = \trianglelefteq_H \cap (\mathrm{supp}^+_R \times \mathrm{supp}^+_R).$$

Proof. Let R be a realizable relation on $\mathscr{P}_2(X)$ and $H \in \{G_R, G_R^-, N_R\}$ be one of the DAGs on X. For all choices of H, Lemma 7.2 implies that H realizes R. This and Lemma 4.9 implies that $R^+ \subseteq \trianglelefteq_H$. By Theorem 7.5, $\mathrm{cl}(R) = R^+$ and, therefore, $\mathrm{cl}(R) \subseteq \trianglelefteq_H$. Moreover, $\mathrm{cl}(R) = R^+$ together with Theorem 4.5 implies that $\mathrm{supp}^+_R = \mathrm{supp}_{R^+} = \mathrm{supp}_{\mathrm{cl}(R)}$. Taking the latter two arguments together with the fact that $\mathrm{cl}(R) = \mathrm{cl}(R) \cap (\mathrm{supp}_{\mathrm{cl}(R)} \times \mathrm{supp}_{\mathrm{cl}(R)})$, we obtain

$$\mathrm{cl}(R) = \mathrm{cl}(R) \cap (\mathrm{supp}^+_R \times \mathrm{supp}^+_R) \subseteq \trianglelefteq_H \cap (\mathrm{supp}^+_R \times \mathrm{supp}^+_R).$$

Now, let $(ab,xy) \in \trianglelefteq_H \cap (\mathrm{supp}^+_R \times \mathrm{supp}^+_R)$. By definition of $\trianglelefteq_H$ we have $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy)$, which taken together with $ab,xy \in \mathrm{supp}^+_R$ and Lemma 7.2(2.a) ensures that $(ab,xy) \in R^+$. We therefore have that $\trianglelefteq_H \cap (\mathrm{supp}^+_R \times \mathrm{supp}^+_R) \subseteq R^+ = \mathrm{cl}(R)$, where the last equality is ensured by Theorem 7.5. This concludes that $\mathrm{cl}(R) = \trianglelefteq_H \cap (\mathrm{supp}^+_R \times \mathrm{supp}^+_R)$. □

We conclude this section by considering the relations that remain invariant under the classical closure operator. In particular, we want to characterize which realizable relations are closed, i.e., those R that satisfy $R = \mathrm{cl}(R)$. By Theorem 7.5 one characterization for such relations is that $R = R^+$ holds. In the following theorem, we characterize closed relations in terms of a "stronger" form of realizability.

Theorem 7.9. *Let R be a realizable relation. Then $R = \mathrm{cl}(R)$ if and only if there exists a DAG H such that for all $ab, cd \in \mathrm{supp}^+_R$, $\mathrm{lca}_H(ab)$ and $\mathrm{lca}_H(cd)$ are well-defined and satisfy*

$$ab \ R \ cd \iff \mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(cd). \tag{4}$$

In particular, $R = \mathrm{cl}(R)$ if and only if the Equivalence (4) holds for R and $H \in \{G_R, G_R^-, N_R\}$ with G_R being the canonical DAG and N_R being the canonical network of R.

Proof. Let R be a realizable relation on $\mathscr{P}_2(X)$. Recall from Theorem 7.5 that $\mathrm{cl}(R) = R^+$, a fact that will be used repeatedly throughout this proof. The *only if*-direction is an immediate consequence of Lemma 7.2 since, if $R = \mathrm{cl}(R) = R^+$, then $ab\ R^+\ xy$ holds if and only if $ab\ R\ xy$, for all $ab, xy \in \mathrm{supp}_R^+$.

For the *if*-direction, assume that there is a DAG H such that, for all $ab, cd \in \mathrm{supp}_R^+$, $\mathrm{lca}_H(ab)$ and $\mathrm{lca}_H(cd)$ are well-defined and the Equivalence in (4) is satisfied. We proceed by showing that R is supp_R^+-reflexive, transitive and cross-consistent. Since $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(ab)$ for all $ab \in \mathscr{P}_2(X)$ for which $\mathrm{lca}_H(ab)$ is well-defined, $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(ab)$ in particular holds for all $ab \in \mathrm{supp}_R^+$. By Eq. (4), it follows that R is supp_R^+-reflexive. If $ab\ R\ xy$ and $xy\ R\ cd$ holds, Eq. (4) ensures that $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy)$ and $\mathrm{lca}_H(xy) \preceq_H \mathrm{lca}_H(cd)$ which, by transitivity of $\preceq_H$, shows that $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(cd)$. Hence, Eq. (4) implies that $ab\ R\ cd$ i.e. R is transitive. Finally, suppose $ab\ R\ xy$, $cd\ R\ xy$ and $ac \in \mathrm{supp}_R$. Thus, the least common ancestors $\mathrm{lca}_H(ac)$, $\mathrm{lca}_H(ab)$, $\mathrm{lca}_H(cd)$ and $\mathrm{lca}_H(xy)$ are well-defined and satisfy, by Eq. (4), $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy)$ and respectively $\mathrm{lca}_H(cd) \preceq_H \mathrm{lca}_H(xy)$. Lemma 4.1 implies $\mathrm{lca}_H(ac) \preceq_H \mathrm{lca}_H(xy)$ which together with Eq. (4) and the fact that $ac \in \mathrm{supp}_R \subseteq \mathrm{supp}_R^+$ implies that $ac\ R\ xy$. In other words, R is cross-consistent. Since we have shown that R is supp_R^+-reflexive, transitive and cross-consistent, Theorem 4.5 ensures that $R = R^+ = \mathrm{cl}(R)$.

We prove now the last statement in the theorem. Suppose that $R = \mathrm{cl}(R) = R^+$. Since R is realizable, Lemma 7.2 implies that $H \in \{G_R, G_R^-, N_R\}$ realizes R. In particular, Lemma 7.2(2) implies that, for all $ab, cd \in \mathrm{supp}_R^+$, it holds that $ab\ R^+\ cd \iff \mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(cd)$. Since $R = R^+$, the Equivalence (4) holds for R and $H \in \{G_R, G_R^-, N_R\}$. Suppose now that $ab\ R\ cd \iff \mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(cd)$ holds for R and $H \in \{G_R, G_R^-, N_R\}$. By Lemma 7.2(2), we have $ab\ R^+\ cd \iff \mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(cd)$. It now immediately follows that $R = R^+ = \mathrm{cl}(R)$. □

8 Involving Incomparability Constraints

Given a collection of taxa X (such as species or genes) and three taxa $a, b, c \in X$, in a phylogenetic tree T on X, we say that a is evolutionary more closely related to b than to c if $\mathrm{lca}_T(ab) \prec_T \mathrm{lca}_T(ac)$. This holds because, for any vertex v in T, the vertex $\mathrm{lca}_T(av)$ always lies on the (unique) path between the root of T and v. Hence, if $\mathrm{lca}_T(ab) \prec_T \mathrm{lca}_T(ac)$, the "branching point" corresponding to $\mathrm{lca}_T(ab)$ must have occurred more recently (i.e., closer to the present) than the branching point corresponding to $\mathrm{lca}_T(ac)$. This notion of evolutionary relatedness has proven useful in the characterization and inference of so-called best matches [16, 17, 43].

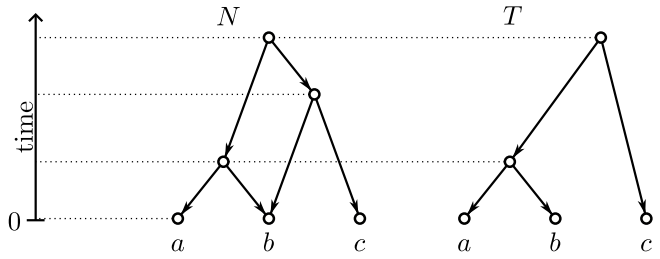

Figure 10: A phylogenetic tree T and network N on the set $X = \{a, b, c\}$ with time from the present indicated. In both DAGs, $\mathrm{lca}(ab)$, $\mathrm{lca}(ac)$ and $\mathrm{lca}(bc)$ are well-defined, and a is evolutionary closer to b than to c. The latter holds because the time elapsed since $\mathrm{lca}(ab)$ existed is less than that since $\mathrm{lca}(ac)$ existed. However, $\mathrm{lca}_N(ab)$ and $\mathrm{lca}_N(ac)$ are $\preceq_N$-incomparable.

This definition of evolutionary relatedness, however, does not directly translate to phylogenetic networks. In particular, it may occur that the vertices $\mathrm{lca}_N(ab)$ and $\mathrm{lca}_N(ac)$ are both well-defined but $\preceq_N$-incomparable, even though a is still evolutionary closer to b than to c; see Figure 10 for an illustrative example. Such situations naturally arise in reticulate evolutionary processes such as hybridization or horizontal gene transfer, where different parts of the genome may follow distinct ancestral histories. It is therefore natural to introduce *incomparability constraints*, capturing pairs of LCA vertices that are not ordered by $\preceq_N$. These constraints provide both a combinatorial means to represent structural incomparability in the ancestral relation and a biologically meaningful way to describe lineages that coexist without a clear hierarchical relationship.

To capture these ideas more formally we make the following definition.

Definition 8.1. *Let R and S be two relations on $\mathscr{P}_2(X)$. We say that the ordered pair (R, S) is* realized *by a DAG G on X if the following two conditions hold.*

1. *G realizes R.*
2. *For all $(ab, xy) \in S$ both $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(xy)$ are well-defined and $\preceq_G$-incomparable.*

In this case, we also say that (R, S) is realizable.

In the rest of this section we shall derive a characterization for when an ordered pair of relations is realizable (see Theorem 8.5). To this end, we first consider when the union of two realizable relations is realizable. Note that, as the following example shows, the union may not be realizable.

Example 8.2 (Union of realizable relations that is not realizable)**.** *Consider the two trees T_3 and T_4 as shown in Figure 4 and the two relations $R_1 = \{(xx,ab),(yy,ab)\}$ and $R_2 = \{(ab,xy)\}$ on $\mathscr{P}_2(X)$ with $X = \{a,b,x,y\}$. One can easily check that $R_1 = \mathrm{tc}(R_1)$ and $R_2 = \mathrm{tc}(R_2)$ and that R_1 is realized by the star-tree T_4 while R_2 is realized by the tree T_3. However, $R = R_1 \cup R_2$ is exactly the relation as in Example 3.8, where we have argued that R is not realizable. Note that here $\mathrm{tc}(R_1) \cup \mathrm{tc}(R_2) \neq \mathrm{tc}(R_1 \cup R_2)$ since $(xx,xy) \in \mathrm{tc}(R)$ but $(xx,xy) \notin \mathrm{tc}(R_1) \cup \mathrm{tc}(R_2)$.*

Even though we have seen that the union of two realizable relations R and R' may not be realizable, we now show that this is the case when R and R' can be realized by the same DAG.

Lemma 8.3. *Suppose that R and R' are relations on $\mathscr{P}_2(X)$ such that there exists a DAG G on X realizing both R and R'. Then, G realizes $R \cup R'$.*

Proof. Let R and R' be relations on $\mathscr{P}_2(X)$ that are both realized by the DAG G on X. We show that $\tilde{R} \coloneqq R \cup R'$ satisfies **(I1)** and **(I2)**.

Suppose first that $(ab,cd) \in \mathrm{tc}(\tilde{R})$ and $(cd,ab) \notin \mathrm{tc}(\tilde{R})$. Since $(ab,cd) \in \mathrm{tc}(\tilde{R})$, there is a (ab,cd)-chain in $\mathrm{tc}(\tilde{R})$ and thus some $p_0,\dots,p_k \in \mathrm{supp}_{\tilde{R}}$, $k \geq 1$ such that $ab = p_0 \ \tilde{R} \ p_1 \ \tilde{R} \ \dots \ \tilde{R} \ p_{k-1} \ \tilde{R} \ p_k = cd$. In particular, $p_j \ R^j \ p_{j+1}$ for all $j \in \{0,\dots,k-1\}$ and some $R^j \in \{R,R'\}$. This together with Lemma 3.6 and the fact that G realizes both R and R' implies that $\mathrm{lca}_G(p_j) \preceq_G \mathrm{lca}_G(p_{j+1})$ all $j \in \{0,\dots,k-1\}$. By transitivity of the $\preceq_G$-relation, it follows that $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$. Assume, for contradiction, that for the elements in the (ab,cd)-chain as above, we also had that $(p_{j+1},p_j) \in \ \mathrm{tc}(R^j)$ for all $j \in \{0,\dots,k-1\}$. In this case, there is a (cd,ab)-chain $p_k = cd \ \mathrm{tc}(R^{k_1}) \ p_{k-1} \ \mathrm{tc}(R^{k_2}) \ \dots \ \mathrm{tc}(R^1) \ p_1 \ \mathrm{tc}(R^{k_0}) \ p_0 = ab$ in $\mathrm{tc}(\tilde{R})$ (just replace $p_{j+1} \ \mathrm{tc}(R^j) \ p_j$ by a respective (p_{j+1},p_j)-chain in R^j, $0 \leq j \leq k-1$). Hence, $(cd,ab) \in \mathrm{tc}(\tilde{R})$, a contradiction. Thus, there is at least one $j \in \{0,\dots,k-1\}$, such that $(p_j,p_{j+1}) \in R^j \subseteq \mathrm{tc}(R^j)$ and $(p_{j+1},p_j) \notin \mathrm{tc}(R^j)$. This and the fact that G realizes R and R' with **(I2)** implies $\mathrm{lca}_G(p_j) \prec_G \mathrm{lca}_G(p_{j+1})$. Since $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(p_j)$ and $\mathrm{lca}_G(p_{j+1}) \preceq_G \mathrm{lca}_G(cd)$ it follows that $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(cd)$. Therefore, **(I1)** holds.

Suppose that $(ab,cd) \in \mathrm{tc}(\tilde{R})$ and $(cd,ab) \in \mathrm{tc}(\tilde{R})$ holds. Using similar arguments as in the proof for Condition **(I1)** one can show that $(ab,cd) \in \mathrm{tc}(\tilde{R})$ implies $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(cd)$, and $(cd,ab) \in \mathrm{tc}(\tilde{R})$ implies $\mathrm{lca}_G(cd) \preceq_G \mathrm{lca}_G(ab)$. Therefore, $\mathrm{lca}_G(ab) = \mathrm{lca}_G(cd)$ and **(I2)** holds. □

Now, for two relations R and S on $\mathscr{P}_2(X)$ we let

$$R_S \coloneqq R \cup \{(aa,ab),(bb,ab) \mid ab \in \mathrm{supp}_S \setminus \mathrm{supp}_R^+ \text{ and } a \neq b\}.$$

Lemma 8.4. *Let R and S be two relations on $\mathscr{P}_2(X)$. The pair (R,S) is realized by the DAG G if and only if the pair (R_S,S) is realized by the DAG G.*

Proof. Let R and S be two relations on $\mathscr{P}_2(X)$. Since $R \subseteq R_S$ it follows that $\mathrm{tc}(R) \subseteq \mathrm{tc}(R_S)$ and $\mathrm{supp}_R^+ \subseteq \mathrm{supp}_{R_S}^+$.

For the *if*-direction suppose that G realizes (R_S,S). We need to show that (R,S) is also realized by G, that is, both **(I1)** and **(I2)** hold. We first consider **(I2)**. To this end, suppose that $(ab,cd) \in \mathrm{tc}(R)$ and $(cd,ab) \in \mathrm{tc}(R)$. Since $\mathrm{tc}(R) \subseteq \mathrm{tc}(R_S)$ it follows that $(ab,cd) \in \mathrm{tc}(R_S)$ and $(cd,ab) \in \mathrm{tc}(R_S)$. Since G realizes R_S, **(I2)** implies $\mathrm{lca}_G(cd) = \mathrm{lca}_G(ab)$. Hence, **(I2)** holds also for R with respect to G. We now consider **(I1)**. Assume that $(ab,cd) \in \mathrm{tc}(R)$ and $(cd,ab) \notin \mathrm{tc}(R)$. Therefore, $ab \neq cd$. Assume for contradiction that $(cd,ab) \in \mathrm{tc}(R_S)$. Then, there exists a sequence $p_1,\dots,p_k$, $k \geq 2$ of pairwise distinct elements of $\mathscr{P}_2(X)$ such that $p_1 = cd$, $p_k = ab$, and $(p_j,p_{j+1}) \in R_S$ for all $j \in \{1,\dots,k-1\}$. Since $(cd,ab) \notin \mathrm{tc}(R)$, there must exist one such j such that $(p_j,p_{j+1}) \in R_S \setminus R$. By definition of the set $R_S \setminus R$, this implies $p_j = xx$ and $p_{j+1} = xy$ for some distinct $x,y \in X$. Since R is realizable, Theorem 5.10 ensures that R satisfies **(X1)**, so there is no element $q \neq aa$ in $\mathscr{P}_2(X)$ such that $(q,aa) \in R^+$ and, thus, in particular, no element $q \neq aa$ in $\mathscr{P}_2(X)$ such that $(q,aa) \in R \subseteq R^+$. Therefore, $p_j = p_1 = cd$ must hold, which implies $c = d$. In particular, we have $(ab,cc) \in \mathrm{tc}(R)$. By Observation 4.8, $\mathrm{tc}(R) \subseteq R^+$ and, therefore, $(ab,cc) \in R^+$. However, we have $ab \neq cc$, and so we obtain a contradiction to **(X1)**. Therefore, $(cd,ab) \notin \mathrm{tc}(R_S)$ must hold and we can conclude, by similar arguments as for **(I2)**, that **(I1)** holds for R with respect to G.

For the *only if*-direction assume that (R,S) is realized by the DAG G on X. We first show that $R_S \setminus R$ is also realized by G. Note that $R_S \setminus R$ does not contain pairs (p,q) and (q,r) with p,q,r being pairwise distinct. Thus, $R_S \setminus R$ is already transitive, i.e., $\mathrm{tc}(R_S \setminus R) = R_S \setminus R$. Let $(p,q) \in \mathrm{tc}(R_S \setminus R)$. By definition of $R_S \setminus R$, there exists $a,b \in X$ distinct such that $p = aa$ and $q = ab$. It also follows directly from the definition that $(ab,aa) \notin \mathrm{tc}(R_S \setminus R)$. Moreover, $ab \in \mathrm{supp}_S$, so since G realizes (R,S), $\mathrm{lca}_G(ab)$ is well defined. Clearly, $a = \mathrm{lca}_G(aa) \prec_G \mathrm{lca}_G(ab)$ always holds. This concludes the proof that G realizes $R_S \setminus R$. Since, in addition, G realizes R by assumption and since $R_S = R \cup (R_S \setminus R)$, Lemma 8.3 implies that G realizes R_S. Hence, G realizes (R_S,S). □

We now prove the main result of this section.

Theorem 8.5. *Let R and S be two relations on $\mathscr{P}_2(X)$. The pair (R,S) is realizable if and only if*

(a) R_S is realizable, and

(b) $S \subseteq I \coloneqq \{(p,q) \in \mathscr{P}_2(X) \times \mathscr{P}_2(X) \mid (p,q),(q,p) \notin R_S^+\}$.

Moreover, if (R,S) is realizable, then (R,S) is realized by any $H \in \{G_{R_S}, G_{R_S}^-, N_{R_S}\}$ with G_{R_S} being the canonical DAG and N_{R_S} being the canonical network of R_S. In particular, verifying whether (R,S) is realizable and, if so, constructing a DAG on X that realizes (R,S) can be done in polynomial time in $|X|$.

Proof. Suppose the pair (R,S) is realized by the DAG G on X. By Lemma 8.4, (R_S,S) is realized by G. In particular, R_S is realizable, i.e., Condition (a) holds. Now assume, for contradiction, that (b) does not hold, i.e. $S \not\subseteq I$. Hence, there is some $(ab,xy) \in S$ such that $ab\ R_S^+\ xy$ or $xy\ R_S^+\ ab$. Since G realizes (R_S,S) and $(ab,xy) \in S$, we must have that $\mathrm{lca}_G(ab)$ and $\mathrm{lca}_G(xy)$ are $\preceq_G$-incomparable. However, Lemma 4.9 together with $ab\ R_S^+\ xy$ or $xy\ R_S^+\ ab$ implies that $\mathrm{lca}_G(ab) \preceq_G \mathrm{lca}_G(xy)$ or $\mathrm{lca}_G(xy) \preceq_G \mathrm{lca}_G(ab)$; a contradiction. Hence, Condition (b) holds.

Conversely, suppose that R_S is realizable and $S \subseteq I$. By Lemma 7.2, R_S is realized by any $H \in \{G_{R_S}, G^-_{R_S}, N_{R_S}\}$. Now, take $ab,xy \in \mathscr{P}_2(X)$ such that $ab\ S\ xy$. By definition of R_S, we have $ab \in \mathrm{supp}_{R_S} \subseteq \mathrm{supp}^+_{R_S}$ and therefore $\mathrm{lca}_H(ab)$ is well-defined in H (c.f. Observation 3.6). Analogously, $\mathrm{lca}_H(xy)$ is well-defined in H. Suppose, for contradiction, that $\mathrm{lca}_H(ab)$ and $\mathrm{lca}_H(xy)$ are $\preceq_H$-comparable. We may, without loss of generality, assume that $\mathrm{lca}_H(ab) \preceq_H \mathrm{lca}_H(xy)$. In this case, Lemma 7.2 implies that $ab\ R_S^+\ xy$ holds; a contradiction to $(ab,xy) \in S \subseteq I$. Hence, $\mathrm{lca}_H(ab)$ and $\mathrm{lca}_H(xy)$ are $\preceq_N$-incomparable for all $(ab,xy) \in S$. Therefore, H realizes the pair (R_S,S). By Lemma 8.4, (R,S) is realized by H.

Finally, it is straightforward to see that checking if $S \subseteq I$ holds can be done in polynomial time in $|X|$. By Theorem 6.5, testing whether R_S is realizable and, if so, constructing H can also be accomplished in polynomial time in $|X|$. Together with the fact that (R,S) is realizable if and only if Conditions (a) and (b) are satisfied (in which case (R,S) is realized by H as argued above), this implies that one can decide in polynomial time in $|X|$ whether (R,S) is realizable and, if so, construct a DAG on X that realizes (R,S). □

9 Discussion and Outlook

In this paper, we have characterized when a set of LCA constraints is realizable by a DAG or a phylogenetic network. Moreover, by relating realizability to generalizations of classical closure operators for phylogenetic trees, we were able to develop a polynomial-time algorithm for deciding whether or not a set of LCA constraints is realizable, and for constructing a DAG or network realizing this set if this is the case. The algorithms introduced in this work have been implemented in Python and are freely available at [34]. There remain a number of interesting future directions to explore.

Triplets and Networks. Approaches to reconstructing phylogenetic trees that employ BUILD usually rely on *triplets*, i.e., binary phylogenetic trees on three leaves (see the tree T in Figure 11 for an illustrative example). Similarly, the literature on phylogenetic network reconstruction contains numerous results on how, and under which conditions, a network can be reconstructed from triplets or suitable generalizations thereof [8, 14, 18, 24–26, 28, 30, 38, 42, 46]. However, the notion of a triplet is more subtle in the context of networks. To be more precise, for a DAG G, there are (at least) two ways to express that a triplet $ab|c$ is displayed by G:

(T1) $a,b,c \in X$ and $\mathrm{lca}_G(ab) \prec_G \mathrm{lca}_G(ac) = \mathrm{lca}_G(bc)$;

(T2) $a,b,c \in X$ and G contains distinct vertices u and v, as well as pairwise internally vertex-disjoint directed paths $u \rightsquigarrow a$, $u \rightsquigarrow b$, $v \rightsquigarrow u$, and $v \rightsquigarrow c$.

Note that the triplet $t = ab|c$ under **(T1)** naturally translates to the LCA constraint $R_t = \{(ab,ac),(ab,bc),(ac,bc),(bc,ac)\}$, in which case the canonical network N_{R_t} is just the triplet t. While in rooted phylogenetic trees the conditions **(T1)** and **(T2)** are equivalent, this is not the case for general DAGs or networks (see Figure 11). There are ways to infer triplets according to **(T1)** from genomic data (see e.g. [40, 43]). In this way one could, in principle, infer networks from genomic data by first inferring triplets according to **(T1)**, then creating the corresponding LCA constraints, and finally computing the canonical network in case the constraints are realizable. It is not hard to see that triplets satisfying **(T1)** also satisfy **(T2)**. The converse, however, is in general not satisfied, see Figure 11 for several examples. Inferring **(T2)**-triplets that do not satisfy **(T1)** from genomic data will probably be more challenging. This is because such triplets require evidence of distinct, internally disjoint ancestral paths and therefore depend on explicitly reconstructed network structures.

Limitation and Extension of Condition (X1) and (X2). The implication in Equation (1) of Lemma 4.1 is only one of many that can be derived from the "LCA-$\prec_G$ relationships". Nevertheless, Equation (1) laid the foundation for the notion of cross-consistency, leading to the $+$-closure R^+ of a relation R. This, in turn, allowed the definition of conditions **(X1)** and **(X2)**, which characterize relations that are realizable by some DAG. However, for practical purposes it would be interesting to understand when a relation can be realized by a DAG that lies within a fixed class. There are examples of subsets $R \subseteq \trianglelefteq_H$, with H belonging to a class $\mathscr{N}$ of DAGs, for which G_R does not yield a network in $\mathscr{N}$. This occurs mainly because R contains only partial information about $\trianglelefteq_H$, see Figure 6. In this example, the vertices $[ab]$ and $[bc]$ in the canonical network N_R are never identified by our approach, since $ab\,R^+\,bc$ and $ab\,R^+\,bc$ does not hold. Moreover, since neither $ab\,R^+\,bc$ nor $bc\,R^+\,ab$ holds, the vertices $[ab]$ and $[bc]$ are $\preceq_{N_R}$-incomparable. However, R is realized by the tree T shown in Figure 6. Consequently, to infer DAGs or networks belonging to a particular class $\mathscr{N}$, such as phylogenetic trees or other types of networks, **(X1)** and **(X2)** must be extended by adding properties that will ensure that a DAG within $\mathscr{N}$ is returned whenever one exists that realizes R. If $\mathscr{N}$ is the class of phylogenetic trees, the BUILD algorithm could be applied; however, our focus is on a non-recursive approach, with particular interest in the structural properties of relations R that are realizable by a tree. Similar questions can be asked

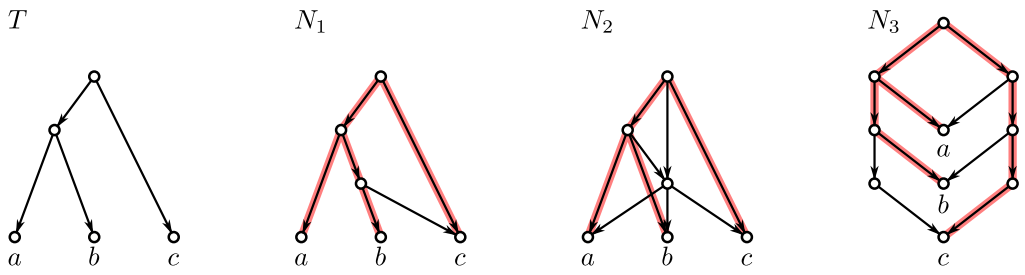

Figure 11: Four networks T, N_1, N_2 and N_3 on the set $X = \{a, b, c\}$. The phylogenetic tree T displays the triplet $ab|c$ according to **(T1)** and **(T2)**. All three networks N_1, N_2 and N_3 display the triplet $ab|c$ according to **(T2)** (highlighted by the shaded red arcs) but not according to **(T1)**. In N_1 we have $\mathrm{lca}_{N_1}(bc) \prec_{N_1} \mathrm{lca}_{N_1}(ab) = \mathrm{lca}_{N_1}(ac)$. Hence, instead of $ab|c$, N_1 displays the triplet $bc|a$ according to **(T1)**. In N_2, we have $\mathrm{lca}_{N_2}(bc) = \mathrm{lca}_{N_2}(ab) = \mathrm{lca}_{N_2}(ac)$. Hence, none of the triplets $ab|c$, $bc|a$ and $ac|b$ is displayed in N_2 according to **(T1)**. The network N_3 displays all of the triplets $ab|c$, $bc|a$ and $ac|b$ according to **(T2)**. However, none of the LCAs $\mathrm{lca}_{N_3}(ab)$, $\mathrm{lca}_{N_3}(bc)$ and $\mathrm{lca}_{N_3}(ac)$ is well-defined in N_3, i.e., N_3 does not display any of triplets $ab|c$, $bc|a$ and $ac|b$ according to **(T1)**.

for the tuple (R, S), where S contains additional information about incomparable LCA constraints. A further question arising in this context is that of "encoding" of networks, i.e., for which network classes $\mathscr{N}$ does it hold that $\trianglelefteq_N = \trianglelefteq_{N'}$ if and only if $N \sim N'$ for all $N, N' \in \mathscr{N}$, where $\sim$ denotes an isomorphism between N and N' that is the identity on the leaf-set?

Underlying Optimization Problems. The are various interesting optimization problems that arise in the context of our results. One problem concerns finding the "simplest" DAG or network that realizes a given relation R, a notion that can be formalized in various ways. For instance, if one restricts to trees, a natural objective could be to find a tree with the minimum number of vertices among all trees realizing R, a problem that is known to be NP-hard [31]. In our setting, the canonical DAG G_R is not necessarily minimum-sized, i.e., it does not always have the fewest vertices among all DAGs realizing R (see for example Figures 6 and 9). Determining the computational complexity of finding a DAG or network realizing R with the minimum number of vertices thus remains an open problem. However, in DAGs, "simplicity" need not be necessarily measured by vertex count. An alternative criterion is proximity to a tree structure, for instance, minimizing the total number of hybrid vertices (those of in-degree greater than one) or the maximum number of hybrids within any biconnected component[2]. We therefore also pose the problem of determining the computational complexity of finding a DAG or network that realizes R with as few hybrids as possible, either globally or per biconnected component. Similar questions can be asked for tuples (R, S).

A third optimization problem concerns non-realizable relations. Recall that only relations satisfying **(X1)** and **(X2)** are realizable (cf. Theorem 5.10). Given a non-realizable relation R, one may ask whether it is possible to efficiently determine the largest subset $R' \subseteq R$ that is realizable, or whether this problem is NP-hard. If R is a set of LCA constraints or represents a set of triplets, it has been shown that finding a maximum-sized subset that can be realized by a tree T is NP-hard [15, 27, 48]. However, our framework allows greater flexibility: we can consider sets R containing LCA constraints (resp., LCA constraints of triplets according to **(T1)**), and ask when they can be realized by some DAG or network without restricting ourselves to trees. As shown in Proposition 5.8 and Example 5.9, there exist non-realizable relations R for which R^+ is realizable, in particular, those satisfying **(X1)** but not **(X2)**. This raises the question of whether the canonical DAG G_R that realizes R^+ could serve as a scaffold for determining a largest realizable subset, or for identifying minimal conflicts within relations satisfying **(X1)**. In other words, can G_R be used to define a measure of how far a relation is from being realizable which can be exploited in practice? Or is G_R possibly already a DAG that realizes as many constraints in R as possible? Similar questions can be asked for tuples (R, S).

Matroids. The study of matroids in phylogenetics has become an active research area in recent years [3, 11, 13, 22, 32]. A classical question in this context is to find "minimum closure-representatives", i.e., minimum-sized subsets $R' \subseteq R$ where R encodes partial information about phylogenetic networks or trees and R' satisfies $\mathrm{CL}(R') = \mathrm{CL}(R)$ for a well-defined closure operator CL [32, 41]. In certain cases, such as triplets in trees, these minimum-sized sets R' form the bases of a matroid, so they all have the same size and can be determined using a simple greedy algorithm [41]. In our setting, this corresponds to finding minimum-sized subsets $R' \subseteq R$ such that $\mathrm{cl}(R') = \mathrm{cl}(R)$. However, as illustrated in Figure 12, it can happen that $|R'| \neq |R''|$ for different minimum-sized subsets, showing that, in general, no matroid structure underlies LCA constraints and their closure. The latter raises the question of how difficult it is to find a minimum-sized subset $R' \subseteq R$ for a realizable relation R such that $\mathrm{cl}(R') = \mathrm{cl}(R)$. Moreover, one may restrict attention to relations R that can be realized by particular classes of networks, for example, phylogenetic trees. This leads to the question of whether there exist network classes $\mathscr{N}$ for which the tuple

$$(R, \mathscr{F}_R), \quad \text{where } \mathscr{F}_R = \{R'' \subseteq R' \mid R' \subseteq R \text{ is inclusion-minimal with } \mathrm{cl}(R') = \mathrm{cl}(R)\},$$

[2] A *biconnected component* of G is an inclusion-maximal subgraph $H = (V', E')$ of G such that H remains weakly connected after removal of any one of its vertices and its incident arcs [45].

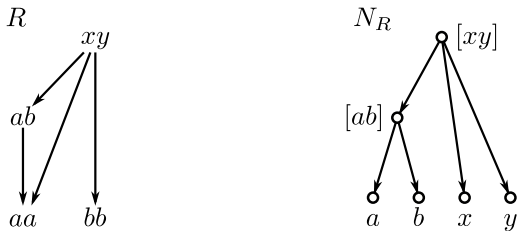

Figure 12: On the left we give a graphical representation of a relation relation $R = \{(aa,xy),(bb,xy),(ab,xy),(aa,ab)\}$ on $\mathscr{P}_2(X)$ with $X = \{a,b,x,y\}$. Here we draw an arc $p \to q$ precisely if $q\,R\,p$. Right the canonical network N_R is shown. Both subsets $R' = \{(ab,xy)\}$ and $R'' = \{(aa,xy),(bb,xy),(aa,ab)\}$ are inclusion-minimal w.r.t. the property that $\mathrm{cl}(R') = \mathrm{cl}(R'') = \mathrm{cl}(R)$. However, $|R'| \neq |R''|$.

forms a matroid, with R a relation realized by some network $N \in \mathscr{N}$. Moreover, if we restrict R to relations that contain "full information" about the underlying network, i.e., $R = \trianglelefteq_N$ for some network $N \in \mathscr{N}$, does this make it easier to determine minimum-sized subsets $R' \subseteq R$ such that $\mathrm{cl}(R') = \mathrm{cl}(\trianglelefteq_N) = \trianglelefteq_N$? Similar questions can be asked for tuples (R,S).

Acknowledgment

We thank the organizers of the Bielefeld meeting "New Directions in Experimental Mathematics" (June 30 – July 1, 2025), where MH proposed the open problem on LCA constraints for DAGs and networks, and the BIRS Workshop "Novel Mathematical Paradigm for Phylogenomics" (August 24–29, 2025), where further discussions helped refine the ideas presented in this work.

We also thank Nadia Tahiri, Leo van Iersel, Peter F. Stadler and Lars Berling for fruitful discussions on the topic.

Author Statement

All authors declare that they have no conflicts of interest. MH and AL conceptualized the main ideas. MH, AL, VM, and GS developed and proved the theoretical concepts and wrote the main manuscript. AA implemented Algorithm 1. All authors reviewed, edited, and approved the final manuscript.

References

[1] Aho AV, Garey MR, Ullman JD (1972) The transitive reduction of a directed graph. SIAM Journal on Computing 1(2):131–137, DOI 10.1137/0201008

[2] Aho AV, Sagiv Y, Szymanski TG, Ullman JD (1981) Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. SIAM Journal on Computing 10(3):405–421, DOI 10.1137/0210030

[3] Ardila F, Klivans CJ (2006) The bergman complex of a matroid and phylogenetic trees. Journal of Combinatorial Theory, Series B 96(1):38–49, DOI 10.1016/j.jctb.2005.06.004

[4] Baroni M, Semple C, Steel M (2005) A framework for representing reticulate evolution. Ann Comb 8:391–408, DOI 10.1007/s00026-004-0228-0

[5] Bryant D (1997) Building trees, hunting for trees, and comparing trees: theory and methods in phylogenetic analysis. PhD thesis, University of Canterbury

[6] Bryant D, Steel M (1995) Extension operations on sets of leaf-labelled trees. Adv Appl Math 16(4):425–453

[7] Caspard N, Monjardet B (2003) The lattices of closure systems, closure operators, and implicational systems on a finite set: a survey. Discrete Applied Mathematics 127(2):241–269, DOI 10.1016/S0166-218X(02)00209-3

[8] Daniel H Huson CS Regula Rupp (2011) Phylogenetic Networks: Concepts, Algorithms and Applications. Cambridge University Press

[9] Deng Y, Fernández-Baca D (2018) Fast compatibility testing for rooted phylogenetic trees. Algorithmica 80(8):2453–2477, DOI 10.1007/s00453-017-0330-4

[10] Döcker J, Linz S, Semple C (2019) Displaying trees across two phylogenetic networks. Theoretical Computer Science 796:129–146, DOI 10.1016/j.tcs.2019.09.003

[11] Dress A, Huber K, Steel M (2014) A matroid associated with a phylogenetic tree. Discrete Mathematics & Theoretical Computer Science Vol. 16 no. 2, DOI 10.46298/dmtcs.2078

[12] Francis A (2025) "Normal" phylogenetic networks may be emerging as the leading class. Journal of Theoretical Biology 614:112236, DOI 10.1016/j.jtbi.2025.112236

[13] Francisco AP, Bugalho M, Ramirez M, Carriço JA (2009) Global optimal eburst analysis of multilocus typing data using a graphic matroid approach. BMC Bioinformatics 10(1):152, DOI 10.1186/1471-2105-10-152

[14] Gambette P, Huber KT, Kelk S (2017) On the challenge of reconstructing level-1 phylogenetic networks from triplets and clusters. Journal of Mathematical Biology 74(7):1729–1751, DOI 10.1007/s00285-016-1068-3

[15] Gasieniec L, Jansson J, Lingas A, Östlin A (1997) On the complexity of computing evolutionary trees. In: Jiang T, Lee DT (eds) Computing and Combinatorics, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 134–145, DOI 10.1007/BFb0045080

[16] Geiß M, Chávez E, González Laffitte M, López Sánchez A, Stadler BMR, Valdivia DI, Hellmuth M, Hernández Rosales M, Stadler PF (2019) Best match graphs. Journal of Mathematical Biology 78(7):2015–2057, DOI 10.1007/s00285-019-01332-9

[17] Geiß M, Stadler PF, Hellmuth M (2020) Reciprocal best match graphs. Journal of Mathematical Biology 80(3):865–953, DOI 10.1007/s00285-019-01444-2

[18] Habib M, To TH (2012) Constructing a minimum phylogenetic network from a dense triplet set. Journal of Bioinformatics and Computational Biology 10(05):1250013, DOI 10.1142/S0219720012500138

[19] Hellmuth M, Lindeberg A (2026) Characterizing and transforming dags within the I-lca framework. Discrete Applied Mathematics 378:584–593, DOI 10.1016/j.dam.2025.08.037

[20] Hellmuth M, Schaller D, Stadler PF (2023) Clustering systems of phylogenetic networks. Theory in Biosciences 142(4):301–358, DOI 10.1007/s12064-023-00398-w

[21] Henzinger MR, King V, Warnow T (1999) Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology. Algorithmica 24(1):1–13, DOI 10.1007/PL00009268

[22] Hollering B, Sullivant S (2021) Identifiability in phylogenetics using algebraic matroids. Journal of Symbolic Computation 104:142–158, DOI 10.1016/j.jsc.2020.04.012

[23] Holm J, de Lichtenberg K, Thorup M (2001) Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. J ACM 48(4):723–760, DOI 10.1145/502090.502095

[24] Huber KT, van Iersel L, Kelk S, Suchecki R (2011) A practical algorithm for reconstructing level-1 phylogenetic networks. IEEE/ACM Transactions on Computational Biology and Bioinformatics 8(3):635–649, DOI 10.1109/TCBB.2010.17

[25] van Iersel L, Kelk S (2011) Constructing the simplest possible phylogenetic network from triplets. Algorithmica 60(2):207–235, DOI 10.1007/s00453-009-9333-0

[26] van Iersel L, Moulton V (2014) Trinets encode tree-child and level-2 phylogenetic networks. Journal of Mathematical Biology 68(7):1707–1729, DOI 10.1007/s00285-013-0683-5

[27] Jansson J (2001) On the complexity of inferring rooted evolutionary trees. Electronic Notes in Discrete Mathematics 7:50–53, DOI 10.1016/S1571-0653(04)00222-7, brazilian Symposium on Graphs, Algorithms and Combinatorics

[28] Jansson J, Sung WK (2006) Inferring a level-1 phylogenetic network from a dense set of rooted triplets. Theoretical Computer Science 363(1):60–68, DOI 10.1016/j.tcs.2006.06.022, computing and Combinatorics

[29] Jansson J, Ng JHK, Sadakane K, Sung WK (2005) Rooted maximum agreement supertrees. Algorithmica 43(4):293–307, DOI 10.5555/3118737.3118847

[30] Jansson J, Nguyen NB, Sung WK (2006) Algorithms for combining rooted triplets into a galled phylogenetic network. SIAM Journal on Computing 35(5):1098–1121, DOI 10.1137/S0097539704446529

[31] Jansson J, Lemence RS, Lingas A (2012) The complexity of inferring a minimally resolved phylogenetic supertree. SIAM Journal on Computing 41(1):272–291, DOI 10.1137/100811489

[32] Levy M (2024) Triplet matroids and closure in phylogenetics. Master's thesis, University of Canterbury, DOI 10.26021/15536

[33] Lindeberg A, Hellmuth M (2025) Simplifying and characterizing DAGs and phylogenetic networks via least common ancestor constraints. Bull Math Biol 87(3):44, DOI 10.1007/s11538-025-01419-z

[34] Lindeberg A, Alfonsson A, Hellmuth M (2025) `RealLCA`. `https://github.com/AnnaLindeberg/RealLCA` (accessed Nov 7, 2025)

[35] Linz S, Semple C (2020) Caterpillars on three and four leaves are sufficient to reconstruct binary normal networks. Journal of Mathematical Biology 81(4):961–980, DOI 10.1007/s00285-020-01533-7

[36] Matoušek J, Nešetřil J (2009) Invitation to discrete mathematics. Oxford University Press

[37] Ng MP, Wormald NC (1996) Reconstruction of rooted trees from subtrees. Discrete Applied Mathematics 69(1):19–31, DOI 10.1016/0166-218X(95)00074-2

[38] Poormohammadi H, Eslahchi C, Tusserkani R (2014) Tripnet: A method for constructing rooted phylogenetic networks from rooted triplets. PLOS ONE 9(9):1–1, DOI 10.1371/journal.pone.0106531

[39] Prof Jørgen Bang-Jensen PGZGa (2009) Digraphs: Theory, Algorithms and Applications, 2nd edn. Springer Monographs in Mathematics, Springer

[40] Schaller D, Stadler PF, Hellmuth M (2021) Complexity of modification problems for best match graphs. Theoretical Computer Science 865:63–84, DOI 10.1016/j.tcs.2021.02.037

[41] Seemann CR, Hellmuth M (2018) The matroid structure of representative triple sets and triple-closure computation. European Journal of Combinatorics 70:384–407, DOI 10.1016/j.ejc.2018.02.013

[42] Semple C, Toft G (2021) Trinets encode orchard phylogenetic networks. Journal of Mathematical Biology 83(3):28, DOI 10.1007/s00285-021-01654-7

[43] Stadler PF, Geiß M, Schaller D, López Sánchez A, González Laffitte M, Valdivia DI, Hellmuth M, Hernández Rosales M (2020) From pairs of most similar sequences to phylogenetic best matches. Algorithms for Molecular Biology 15(1):5, DOI 10.1186/s13015-020-00165-2

[44] Steel M (2016) Phylogeny: discrete and random processes in evolution. SIAM

[45] Steven S Skiena; Pemmaraju SV (2003) Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica. Cambridge University Press

[46] van Iersel L, Kole S, Moulton V, Nipius L (2022) An algorithm for reconstructing level-2 phylogenetic networks from trinets. Information Processing Letters 178:106300, DOI 10.1016/j.ipl.2022.106300

[47] Willson SJ (2010) Properties of normal phylogenetic networks. Bulletin of Mathematical Biology 72(2):340–358, DOI 10.1007/s11538-009-9449-z

[48] Wu BY (2004) Constructing the maximum consensus tree from rooted triples. Journal of Combinatorial Optimization 8(1):29–39, DOI 10.1023/B:JOCO.0000021936.04215.68