Bulletin of Mathematical Biology (2026) 88:209
https://doi.org/10.1007/s11538-026-01783-4

Society for Mathematical Biology

ORIGINAL RESEARCH



Regularizing and Normalizing DAGs and Phylogenetic Networks

Marc Hellmuth[1] · **Anna Lindeberg**[2] · **Vincent Moulton**[3]

Received: 20 May 2026 / Accepted: 8 October 2026
© The Author(s) 2026

Abstract

Phylogenetic networks and, more generally, directed acyclic graphs (DAGs) represent hierarchical structure beyond trees, for instance in the presence of reticulate evolutionary events such as hybridization or horizontal gene transfer. A central question is which parts of such graphs are essential with respect to leaf-observable information, and which parts can be removed without changing this information. Resolving this question can lead to principled simplification methods for phylogenetic networks, such as the recent normalization approach of Francis et al. In this paper, we study this question from three related perspectives: clusters displayed by a DAG G, least common ancestors (LCAs) of subsets of its leaf set, and visibility, a path-based property of vertices. We first introduce an LCA-based simplification procedure called *i-regularization*. For a DAG G and $i \geq 1$, the DAG $\mathrm{REG}_i(G)$ retains precisely those vertices that occur as unique LCAs of leaf subsets of size at most i, removes the remaining non-leaf vertices by a graph-editing operation $\ominus$, and then deletes shortcuts. We show that $\mathrm{REG}_i(G)$ admits a Hasse-diagram characterization in terms of the corresponding lca-clusters. We then compare LCA-based regularization with normalization. Using the same $\ominus$-operator, we describe the cover construction underlying normalization, identify visible vertices that are nevertheless removed, and characterize when regularization and normalization coincide. Together, these results provide a unified framework for cluster-based, LCA-based, and visibility-based simplifications of DAGs and phylogenetic networks.

Keywords Phylogenetic network · Regular network · Normal network · Cluster · Least common ancestors

✉ Anna Lindeberg
anna.lindeberg@math.su.se

[1] Faculty of Mathematics and Computer Science, Leipzig University, 04109 Leipzig, Germany

[2] Department of Mathematics, Faculty of Science, Stockholm University, SE-10691 Stockholm, Sweden

[3] School of Computing Sciences, University of East Anglia, Norwich NR4 7TJ, UK

Published online: 21 October 2026

1 Introduction

Directed acyclic graphs (DAGs) and rooted phylogenetic networks are widely used to model hierarchical structure in situations where purely tree-like representations are too restrictive. A classical example arises in evolutionary biology, where reticulation events such as hybridization or horizontal gene transfer cannot be captured by trees alone (Huson and Scornavacca 2011). Beyond phylogenetics, DAG-based models also appear in areas such as database theory (Aho et al. 1981), workflow analysis and process modeling (van der Aalst 1998, 2004), or in the theory of Bayesian and causal networks (Pearl 2009; Koller and Friedman 2009).

A recurring theme in the study of such graphs is to determine which aspects of the underlying topology can be recovered from "coarse" information derived from the leaves, and how one can systematically simplify a given DAG or network without destroying the leaf-implied structure of interest. In phylogenetics, the leaves represent present-day species (or other taxa), while internal vertices encode ancestral ones. It is therefore natural to ask which internal vertices are structurally essential and which are redundant with respect to a specified type of leaf-based information.

Three classical viewpoints on this question are central to this paper. The first is *cluster information*: every vertex v in a DAG G with leaf set X induces a cluster $\mathcal{C}_G(v) = \{x \in X \mid \text{there is a directed path from } v \text{ to } x \text{ in } G\}$, and the resulting set system $\mathfrak{C}(G)$ records how leaves are grouped by ancestral vertices. The second is *least common ancestors (LCAs)*: for a subset $A \subseteq X$, its least common ancestors (when they exist) identify the "lowest" vertices that jointly explain the ancestry of A. The third is *visibility*, a path-based notion that captures whether a vertex lies on all root-to-leaf paths leading to at least one leaf. For example, in Fig. 1, taking G to be the DAG N_1, the vertex u induces the cluster $\{a, b\}$, the LCA of the vertices a and c is ρ, and the vertex u is visible, since all paths from the root ρ to the leaf a pass through u.

This leads to a fundamental question: given a DAG or network together with its induced leaf-based information – such as clusters, LCAs, or visibility – which structural components are essential for representing this information, and which are superfluous? In other words, can certain internal vertices be removed or suitably transformed to yield a simpler directed graph that still faithfully captures the prescribed leaf-based structure? From this perspective, simplifying a DAG or network means defining a principled transformation that eliminates "irrelevant" substructures – those that do not contribute to the chosen leaf-based information – while preserving the structural relationships among the leaves. The main challenge is to formalize this notion of

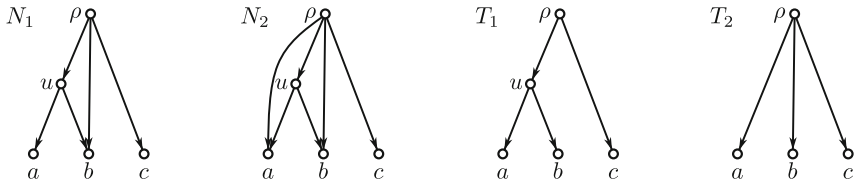


Fig. 1 Two networks N_1, N_2 and two trees T_1, T_2, for which $T_1 = \text{REG}(N_1) = \text{NORM}(N_1)$ and $T_1 = \text{REG}(N_2) \neq \text{NORM}(N_2) = T_2$, see Sect. 3

irrelevance and to design simplification procedures that are canonical, well-behaved, and compatible with the underlying leaf-based representation.

In Francis et al. (2021), Francis, Huson and Steel introduced a *normalization* procedure that transforms a network N into a normal network $\textsc{Norm}(N)$ by retaining only visible vertices, removing shortcuts, and suppressing vertices of in-degree and out-degree equal to one. More recently, Lindeberg and Hellmuth (2025, 2026) proposed an alternative simplification approach based on retaining only those vertices that serve as least common ancestors for specified subsets of leaves. A third, conceptually related, construction is to consider the Hasse diagram of the clustering system $\mathfrak{C}(G)$, which yields a canonical DAG that is minimal in the sense of representing all leaf-induced clusters. Note that there are other approaches for simplifying DAGs and networks. Some focus on producing a tree from a given DAG or network, see for example (Gusfield 2014; Huber et al. 2019; Dress et al. 2010; Heiss et al. 2025; Willson 2011; Baroni et al. 2006), and further simplifications that do not necessarily produce a tree are presented in Moret et al. (2004); Willson (2012, 2022).

Although the normalization, the LCA-driven and the Hasse diagram approaches produce a simplified version of the underlying DAG or network – essentially by removing vertices deemed "irrelevant" and rewiring certain arcs – each of them relies on different criteria. Normalization is based on visibility and thus on path-based properties, whereas the LCA-driven simplification retains vertices according to their role in explaining leaf subsets. The relationship between a DAG G and the Hasse diagram $\mathcal{H}(\mathfrak{C}(G))$ of its cluster system has also been partially explored in Lindeberg and Hellmuth (2025); Hellmuth and Lindeberg (2026); Baroni et al. (2005); in essence, $\mathcal{H}(\mathfrak{C}(G))$ retains single vertices v for which $\mathsf{C}_G(v)$ is contained in $\mathfrak{C}(G)$ and its arcs reflect how such clusters are related with respect to set inclusion. Our main contribution is to highlight the similarities and differences between these three approaches to simplifying DAGs and networks, and to place them within a unified conceptual framework.

In the following, we begin with the LCA-driven perspective and specialize the simplification framework originally introduced in Lindeberg and Hellmuth (2025) to a systematic "regularization" approach based on the notion of *lca-relevance*. More specifically, for each integer $i \geq 1$, we consider the set of vertices that arise as the unique LCA, denoted $\mathrm{lca}_G(A)$, for some subset $A \subseteq X$ with $|A| \leq i$, and we delete all other non-leaf vertices using the graph-editing operator $\ominus$ (which generalizes suppression and certain arc contractions). After subsequently removing shortcuts (denoted with a superscripted "$-$"), this yields the *i-regularization* $\textsc{Reg}_i(G) = (G \ominus W)^-$ of G. Intuitively, $\textsc{Reg}_i(G)$ is a canonical simplification of G that preserves the LCA structure of all leaf subsets of size at most i. Moreover, $\textsc{Reg}_i(G)$ is *regular*, which means that it is completely determined by its set of clusters.

Although regularity of $\textsc{Reg}_i(G)$ is an immediate consequence of results in Lindeberg and Hellmuth (2025), that paper does not describe the set of clusters in $\mathfrak{C}(\textsc{Reg}_i(G))$. To complete the picture, we give a cluster-level characterization of i-regularization in Theorem 3.5. Specifically, we show that $\textsc{Reg}_i(G)$ is isomorphic to the Hasse diagram of certain *lca-clusters*: clusters C for which $\mathrm{lca}_G(C)$ is well-defined and forced by some subset A with $|A| \leq i$. Thus, Theorem 3.5 provides a direct link between LCA-based simplification and the classical cluster/Hasse-diagram viewpoint.

This is complemented by Theorem 3.8, which characterizes when the Hasse diagram $\mathcal{H}(\mathfrak{C}(G))$ coincides with the fully regularized graph $\mathrm{REG}(G) := \mathrm{REG}_{|X|}(G)$.

Next, we consider normal networks and their behavior under regularization. In Proposition 3.11, we extend the connection between normal and regular networks established in Theorem 3.9 of Willson (2010) by clarifying the relationships between normality, strong normality, lca-relevance, and regularity. We then show in Theorem 3.14 precisely how regularization acts on a normal network. Besides providing further structural insight into normal networks, these results provide key ingredients to characterize when normalization and regularization yield the same network, cf. Theorem 4.6.

We then turn to the principal goal of this paper: comparing regularization with the normalization $\mathrm{NORM}(N)$ of N. These two approaches are distinct. For example, for the networks N_1 and N_2 in Fig. 1, the networks $\mathrm{REG}(N_1)$ and $\mathrm{NORM}(N_1)$ coincide, whereas the networks $\mathrm{REG}(N_2)$ and $\mathrm{NORM}(N_2)$ do not. Although both approaches aim to remove "irrelevant" vertices, they differ precisely in the criterion used to determine which vertices are retained: normalization is based on visibility, a path-based notion that captures whether a vertex lies on all root-to-leaf paths to some leaf, whereas i-regularization is driven by lca-relevance.

To better understand the interplay between these two notions of relevance, in Sect. 4 we express both constructions within the common framework of transformations of the form $(N \ominus W)^-$. The key difference lies in the choice of the vertex set W. For i-regularization, W consists, by definition, of the vertices that are not unique LCAs of leaf subsets of size at most i. For normalization, we first show that the cover construction $\mathrm{COV}(N)$ can be expressed as $\mathrm{COV}(N) = (N \ominus \overline{\mathfrak{vis}}(N))^-$, where $\overline{\mathfrak{vis}}(N)$ is the set of non-visible vertices of N. We then identify a subclass of visible vertices that are suppressed in the second step of normalization, the so-called *dispensably visible* vertices, and provide criteria for detecting them directly in N. This yields an $\ominus$-based characterization of normalization itself and, ultimately, allows us to characterize precisely when $\mathrm{REG}_i(N)$ and $\mathrm{NORM}(N)$ coincide; see Theorem 4.6, which is the main comparison result of the paper.

The rest of this paper is organized as follows. In Sect. 2 we recall the required background on DAGs, shortcuts, normal networks, the $\ominus$-operator, clusters, and LCAs. Section 3 introduces i-regularization, discusses its basic properties, and characterizes $\mathrm{REG}_i(G)$ via lca-clusters and Hasse diagrams. It also discusses normal and strongly normal networks and relates their structure to lca-relevance. In particular, we describe how normal networks behave under i-regularization, that is, under the transformation $N \mapsto \mathrm{REG}_i(N)$. In Sect. 4 we then develop the framework for the comparison of network regularization and normalization, derive $\ominus$-based formulas for $\mathrm{COV}(N)$ and $\mathrm{NORM}(N)$, and identify conditions under which regularization and normalization agree. Finally, in Sect. 5 we conclude with a discussion of some future directions.

2 Preliminaries

In this section, we recall the main definitions and results concerning DAGs, LCAs, clusters, and the $\ominus$-operator that are needed throughout the paper. Most of this material

is taken from, or follows directly from, Lindeberg and Hellmuth (2025) and related work; it is included here for reference and to fix notation. Those readers who are more familiar with these concepts can move on to the next section and refer back to the relevant statements if necessary.

Basics. In what follows, X will always be a finite non-empty set and $\mathscr{P}_{\leq i}(X)$ the set of non-empty subsets of X of size at most i. Moreover, $A \cup B$ denotes the *disjoint union* of two sets A and B. A collection $\mathfrak{T}$ of non-empty subsets of X is a *hierarchy* if $X \in \mathfrak{T}$, $\{x\} \in \mathfrak{T}$ for all $x \in X$, and for all $A, B \in \mathfrak{T}$, we have $A \cap B \in \{\emptyset, A, B\}$.

A *directed graph* $G = (V, E)$ is an ordered tuple with non-empty vertex set $V(G) := V$ and arc set $E(G) := E \subseteq V \times V$. We sometimes use $u \to v$ to refer to the arc $(u, v) \in E(G)$ and $u \rightsquigarrow v$ to denote a directed uv-path in G. We put $\mathrm{outdeg}_G(v) := |\{u \in V : (v, u) \in E\}|$ and $\mathrm{indeg}_G(v) := |\{u \in V : (u, v) \in E\}|$ 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 $\mathrm{indeg}_G(v) \leq 1$.

Directed graphs G without directed cycles are called *directed acyclic graphs (DAGs)*. If G is a DAG with leaf set X, then G is a *DAG on* X. A *network* is a DAG with a unique root. A *(rooted) tree* is a network that does not contain any hybrid vertices. A DAG G is *separated* if all hybrids v in G have $\mathrm{outdeg}_G(v) = 1$ (Hellmuth et al. 2023).

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*.

Two DAGs G and H are *isomorphic* if there is an isomorphism between G and H, i.e., a bijective map $\varphi : V(G) \to V(H)$ such that $(u, v) \in E(G)$ if and only if $(\varphi(u), \varphi(v)) \in E(H)$. If this is the case, we write $G \simeq H$.

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 (Linz and Semple 2020; Döcker et al. 2019). A DAG without shortcuts is *shortcut-free*. If $F \subseteq E$ is the set of all shortcuts of $G = (V, E)$, then we put $G^- := (V, E \setminus F)$.

Lemma 2.1 ((Lindeberg and Hellmuth, 2025, L. 2.5) and (Willson, 2016, L. 7.2)) *Let G be a DAG on X. Then, G^- is a shortcut-free DAG on X such that, for all $u, v \in V(G)$, we have $u \preceq_G v$ if and only if $u \preceq_{G^-} v$. Moreover, if G and H are shortcut-free DAGs such that $V := V(G) = V(H)$ and, for all $u, v \in V$, it holds that $u \preceq_G v$ if and only if $u \preceq_H v$, then $G = H$.*

Normal Networks. Normal networks form a prominent class of phylogenetic networks (Francis et al. 2021; Willson 2010); see Francis (2025) for a comprehensive overview. Although our primary focus lies on normal *networks*, several technical results are more naturally formulated in the broader setting of DAGs. For this reason, we develop the theory at the level of DAGs and specialize to networks where appropriate.

Definition 2.2 A child u of v in a DAG G is a *tree-child* if $\mathrm{indeg}_G(u) = 1$. A DAG G on X is *tree-child* if every vertex $v \in V(G) \setminus X$ has a tree-child (Cardona et al. 2009). A DAG is *normal* if it is tree-child and shortcut-free (Willson 2010).

A common restriction in the phylogenetic literature, in particular for normal networks, is that hybrid vertices are required to have a unique child. We explicitly call such networks *separated*. To avoid some technical issues caused by separated vertices, we introduce the following class of DAGs:

Definition 2.3 A DAG is *strongly normal* if it is normal and does not contain any vertices that have only one child.

Let N be a network on X with unique root ρ. A vertex v of N is *visible (in N)* if there exists a leaf $x \in X$ such that every ρx-path of N passes through v. Let $\mathfrak{vis}(N)$ denote the set of visible vertices of N and put $\overline{\mathfrak{vis}}(N) := V(N) \setminus \mathfrak{vis}(N)$. Observe that, by definition, ρ and every $x \in X$ are visible vertices of N. The following result provides an interesting characterization of normal networks.

Theorem 2.4 (Cardona et al., 2009, Lem. 2) *A network is normal if and only if it is shortcut-free and all its vertices are visible.*

The $\ominus$-operator. We continue by summarizing important concepts from Lindeberg and Hellmuth (2025) where, in particular, the following $\ominus$-operator is studied.

Definition 2.5 *(Shanavas et al. (2024))* Let $G = (V, E)$ be a DAG and $v \in V$. Then $G \ominus v = (V', E')$ is the directed graph with vertex set $V' = V \setminus \{v\}$ and arcs $(p, q) \in E'$ precisely if $v \neq p$, $v \neq q$ and $(p, q) \in E$, or if $(p, v) \in E$ and $(v, q) \in E$. For a non-empty subset $W = \{w_1, \ldots, w_\ell\} \subsetneq V$, define $G \ominus W := (\ldots((G \ominus w_1) \ominus w_2) \ldots) \ominus w_\ell$. For notational reasons, put $G \ominus \emptyset = G$.

In simple terms, the directed graph $G \ominus v$ is obtained from $G = (V, E)$ by deleting the vertex v together with all incident arcs, and then adding an arc $p \to q$ for every parent p of v and every child q of v. If v is a leaf or a root, we simply remove v and its incident arcs. If v has in-degree one and out-degree one in G, then $G \ominus v$ is well-known as the directed graph obtained from G by *suppressing* v. More generally, if v has a unique child u, then $G \ominus v$ can be viewed as contracting the arc $v \to u$ to the single vertex u. Similarly, if v has a unique *parent* u, then $G \ominus v$ coincides with contracting the arc $u \to v$ to u. Hence, the $\ominus$-operator unifies and generalizes several common graph-editing operations on DAGs or networks.

It is also worth noting that the definition of $G \ominus W$ is motivated by the fact that the $\ominus$-operator is commutative. This fact is mentioned — without proof — in Shanavas et al. (2024); Lindeberg and Hellmuth (2025). For completeness, we shall prove this in the appendix of the paper.

Lemma 2.6 *For all DAG G and distinct vertices $u, v \in V(G)$ it holds that $(G \ominus v) \ominus u = (G \ominus u) \ominus v$.*

Proof The proof is straightforward but tedious and, therefore, outsourced to Appendix A. □

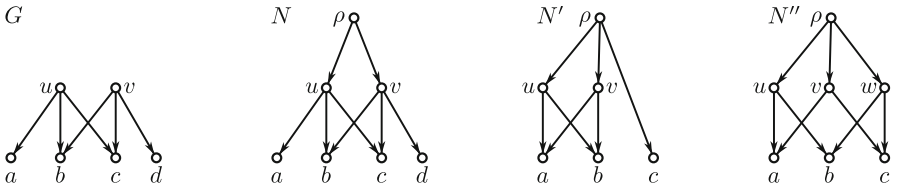


Fig. 2 Four DAGs that illustrate certain concepts related to least common ancestors; see the text for further details

Hence, the order of the vertices in W does not affect the result of $G \ominus W := (\dots((G \ominus w_1) \ominus w_2)\dots) \ominus w_\ell$. It is easily seen that $G \ominus W$ remains a DAG and that the existence of directed paths in G is preserved in $G \ominus W$. The following lemma summarizes these simple but important properties of the $\ominus$-operator.

Lemma 2.7 (Lindeberg and Hellmuth, 2025, Obs 5.3) *Let G be a DAG on X and $W \subseteq V(G) \setminus X$. Then, $G \ominus W$ is a DAG on X that satisfies $u \preceq_G v \iff u \preceq_{G\ominus W} v$ for all $u, v \in V(G \ominus W) = V(G)\setminus W$.*

Let G be a DAG on X. For every $v \in V(G)$ we define the *cluster* of v as the set $\mathcal{C}_G(v) := \{x \in X \mid x \preceq_G v\}$. The set $\mathfrak{C}(G) = \{\mathcal{C}_G(v) \mid v \in V(G)\}$ comprises all clusters of the vertices in G.

Just as every DAG G on X naturally gives rise to a set system on X, we can also associate to every set system on X a corresponding DAG on X, as we now explain. The *Hasse diagram* $\mathcal{H}(\mathfrak{C})$ of a set system $\mathfrak{C}$ on X is the DAG with vertex set $\mathfrak{C}$ and arc set containing an arc $A \to B$ for $A, B \in \mathfrak{C}$ if and only if (i) $B \subsetneq A$ and (ii) there is no $C \in \mathfrak{C}$ with $B \subsetneq C \subsetneq A$. The Hasse diagram $\mathcal{H}(\mathfrak{C})$ is also known as the *cover digraph* of $\mathfrak{C}$ (Baroni et al. 2005). Intuitively, DAGs that are indistinguishable from the Hasse diagram of their set of clusters have much of their structure determined by these clusters. Such DAGs are called *regular*, and are formally defined as follows.

Definition 2.8 *(Baroni et al. (2005))* A DAG $G = (V, E)$ is *regular* if the map $\varphi\colon V \to V(\mathcal{H}(\mathfrak{C}(G)))$ defined by $v \mapsto \mathcal{C}_G(v)$ is an isomorphism between G and $\mathcal{H}(\mathfrak{C}(G))$.

Note that regular DAGs have been shown to be the only DAGs that can arise from certain simple models of evolution, see Baroni et al. (2005); Baroni and Steel (2006) for details. As we shall see, regular DAGs are also closely related to properties involving so-called least common ancestors, which we define next. 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 $A \subseteq \mathcal{C}_G(v)$, i.e., if v is an ancestor of every vertex in A. Moreover, v is a *least common ancestor* (LCA) of A if v is $\preceq_G$-minimal among all common ancestors of A. The set $\mathrm{LCA}_G(A)$ comprises all LCAs of A in G.

In general, not every set $A \subseteq X$ has a (unique) least common ancestor in a DAG; for example, consider the DAG G in Fig. 2 where $|\mathrm{LCA}_G(\{b, c\})| > 1$ and $\mathrm{LCA}_G(\{a, d\}) = \emptyset$. In a network N on X, the unique root is a common ancestor for all $A \subseteq X$ and, therefore, $\mathrm{LCA}_N(A) \neq \emptyset$. For simplicity, we will write $\mathrm{lca}_G(A) = v$ in case that $\mathrm{LCA}_G(A) = \{v\}$, in which case we say that $\mathrm{lca}_G(A)$ *is well-defined*; otherwise, we leave $\mathrm{lca}_G(A)$ *undefined*. Moreover, when $\mathrm{lca}_G(\{x, y\})$ is well-defined

for some $x, y \in X$, we write $\mathrm{lca}_G(x, y)$ to refer to $\mathrm{lca}_G(\{x, y\})$. A vertex $v \in V(G)$ is a *k-lca vertex* if $v = \mathrm{lca}_G(A)$ for some $A \subseteq X$ with $|A| = k$. In addition, a DAG G satisfies the *cluster-lca (CL)* property if $\mathrm{lca}_G(\mathcal{C}_G(v))$ is well-defined for all $v \in V(G)$.

Not all DAGs satisfy the (CL) property. For example, in the network N' of Fig. 2, $\mathrm{LCA}_{N'}(\mathcal{C}_{N'}(v)) = \mathrm{LCA}_{N'}(\{a, b\}) = \{u, v\}$ and thus, $\mathrm{lca}_{N'}(\mathcal{C}_{N'}(v))$ is not well-defined. Nevertheless, as the following result shows, $\mathrm{lca}_G(\mathcal{C}_G(v))$ is well-defined whenever $v = \mathrm{lca}_G(A)$ for some subset $A \subseteq X$. This, and the other properties stated in the following lemma, are simple consequences of the definitions and the fact that $\mathcal{C}_G(u) \subseteq \mathcal{C}_G(v)$ for all $u, v \in V(G)$ such that $u \preceq_G v$.

Lemma 2.9 ((Shanavas et al., 2024, Obs. 2) & (Lindeberg and Hellmuth, 2025, Cor. 3.6)) *If G is a DAG on X and $A \subseteq X$ is a non-empty subset such that $\mathrm{lca}_G(A)$ is well-defined, then $\mathcal{C}_G(\mathrm{lca}_G(A))$ is the unique inclusion-minimal cluster in $\mathfrak{C}(G)$ containing A, and $\mathrm{lca}_G(\mathcal{C}_G(\mathrm{lca}_G(A))) = \mathrm{lca}_G(A)$. In particular, $v \in V(G)$ is a k-lca vertex in G for some k if and only if $v = \mathrm{lca}_G(\mathcal{C}_G(v))$. Moreover, if $u \in V(G)$ is a vertex such that $A \subseteq \mathcal{C}_G(u)$, then $\mathrm{lca}_G(A) \preceq_G u$.*

Note that, if w is the unique child of a vertex v in a DAG G, then $\mathcal{C}_G(v) = \mathcal{C}_G(w)$ holds. This together with Lemma 3.4 and 3.5 in Lindeberg and Hellmuth (2025) implies the following natural property of vertices with out-degree one.

Lemma 2.10 *If v is a vertex of a DAG G on X such that $\mathrm{outdeg}_G(v) = 1$, then $v \notin \mathrm{LCA}_G(A)$ for any $A \subseteq X$.*

The following definition from Lindeberg and Hellmuth (2025) captures the notion of a DAG where every vertex is a k-lca vertex for some $k \leq i$.

Definition 2.11 A DAG $G = (V, E)$ on X is i-lca-relevant if, for all $v \in V$, there is some $A \in \mathscr{P}_{\leq i}(X)$ such that $v = \mathrm{lca}_G(A)$.

In other words, G is i-lca-relevant precisely if every vertex of G is a k-lca vertex for some $k \leq i$. Hence, if G is i-lca-relevant for some integer i, then G is also j-lca-relevant for all $j \geq i$. As shown by the DAG N'' in Fig. 2, the converse of the latter statement is not always satisfied: N'' is 3-lca-relevant, but not 2-lca-relevant since its root ρ is not the unique LCA for any subset consisting of one or two leaves only. Note also that the DAG G in Fig. 2 is 2-lca-relevant, as $\mathrm{lca}_G(x) = x$ for all $x \in \{a, b, c, d\}$, and $u = \mathrm{lca}_G(a, b)$, $v = \mathrm{lca}_G(c, d)$ are all well-defined. This holds even though $\mathrm{lca}_G(b, c)$ is undefined. We call $|X|$-lca-relevant DAGs on X simply *lca-relevant* since, in this case, each vertex v is the unique LCA of just *some* subset of its leaves, with no restriction on the size of that set. By way of example, the DAGs G, N and N'' of Fig. 2 are lca-relevant, while the network N' is not. Clearly, all i-lca-relevant DAGs are, in particular, lca-relevant.

We also note that there is a surprisingly close connection between regular and lca-relevant DAGs:

Theorem 2.12 (Lindeberg and Hellmuth, 2025, Thm. 4.10) *A DAG G is regular if and only if G is lca-relevant and shortcut-free.*

We close this section by collecting together a couple of results that connect the $\ominus$-operator with clusters and LCAs and that we shall need later on. The first observation is a direct consequence of Lemma 2.1 and Lemma 2.7.

Observation 2.13 *Let $G = (V, E)$ be a DAG on X and $W \subseteq V \setminus X$. Then, $H = (G \ominus W)^-$ is a shortcut-free DAG on X that satisfies $u \preceq_G v \iff u \preceq_H v$ for all $u, v \in V(H) = V \setminus W$. Consequently, $\mathsf{C}_H(v) = \mathsf{C}_G(v)$ for all $v \in V \setminus W$ and, thus, $\mathfrak{C}(H) = \{\mathsf{C}_G(v) \mid v \in V \setminus W\} \subseteq \mathfrak{C}(G)$.*

In case W solely contains vertices of out-degree one, we have further control over properties of $G \ominus W$.

Lemma 2.14 *Let $G = (V, E)$ be a DAG on X and let $W \subseteq \{v \in V \mid \mathrm{outdeg}_G(v) = 1\}$. Then $G \ominus W$ is a DAG on X such that*

(1) $\mathrm{LCA}_{G \ominus W}(A) = \mathrm{LCA}_G(A)$ *for all non-empty $A \subseteq X$ and*
(2) $1 \leq \mathrm{outdeg}_{G \ominus W}(v) \leq \mathrm{outdeg}_G(v)$ *for all $v \in V(G \ominus W) \setminus X$.*

In particular, if $W' \subseteq \{w \in W \mid$ the unique child c of w satisfies $\mathrm{indeg}_G(c) = 1\}$, then $\mathrm{outdeg}_{G \ominus W'}(v) = \mathrm{outdeg}_G(v)$ for all $v \in V(G \ominus W')$.

Proof Let G and W be as stated. Since every vertex in W has out-degree one, W contains no leaf of G. Hence, Lemma 2.7 implies that $G \ominus W$ is again a DAG on X.

For Statement (1), note that if $w \in W$, then $w \notin \mathrm{LCA}_G(Y)$ for any $Y \subseteq X$, see Lemma 2.10. This together with (Lindeberg and Hellmuth, 2025, Thm 5.5) ensures that $\mathrm{LCA}_{G \ominus W}(A) = \mathrm{LCA}_G(A)$ for all non-empty $A \subseteq X$.

We next prove Statement (2). Consider first a single vertex $w \in W$, and let c be its unique child. By Definition 2.5, passing from the current DAG G to the graph $G \ominus w$ can only affect the out-degree of a parent p of w. By construction of $G \ominus w$, w is removed from $\mathrm{child}_G(p)$ and c is added as a child of p unless c is already a child of p. Hence the out-degree of p either remains unchanged or decreases by one; in either case c is a child of p in $G \ominus w$. This observation can be applied successively to the vertices of W. Indeed, every vertex $w \in W$ that has not yet been removed still has out-degree one: if its unique child is removed at an earlier step, that child is replaced by its unique child. Consequently, $1 \leq \mathrm{outdeg}_{G \ominus W}(v) \leq \mathrm{outdeg}_G(v)$ for every $v \in V(G \ominus W) \setminus X$.

Finally, let $W' \subseteq \{w \in W \mid$ the unique child c of w satisfies $\mathrm{indeg}_G(c) = 1\}$. We prove the claim by induction on $|W'|$. The assertion is trivial for $W' = \emptyset$. Suppose now that $W' \neq \emptyset$ and that the claim holds for every proper subset of W'. Since G is acyclic and every vertex of W' has out-degree one, there exists some $w \in W'$ whose unique child c does not belong to W'. Put $S := W' \setminus \{w\}$ and $H := G \ominus S$. By the induction hypothesis, $\mathrm{outdeg}_H(v) = \mathrm{outdeg}_G(v)$ for every $v \in V(H)$. In particular, $\mathrm{outdeg}_H(w) = 1$. Since $w, c \notin S$, the vertex c and the arc $w \to c$ is still present in H. Hence c is the unique child of w in H. Moreover, $\mathrm{indeg}_G(c) = 1$, with w being the unique parent of c in G. Any arc $p \to c$ in $H = G \ominus S$ corresponds to a directed pc-path in G whose internal vertices belong to S. Since the last arc of every such path must be $w \to c$ and $w \notin S$, this is only possible if $p = w$. Consequently, $\mathrm{indeg}_H(c) = 1$. Now let p be any parent of w in H. Since w is the unique parent of c, we have $(p, c) \notin E(H)$. Thus, by Definition 2.5, passing from H to $H \ominus w$ replaces the child w of p by the child c and therefore preserves the out-degree of p. The out-degrees of all other remaining vertices are unaffected. Hence $\mathrm{outdeg}_{H \ominus w}(v) = \mathrm{outdeg}_H(v)$ for every $v \in V(H \ominus w)$. Since $H \ominus w = (G \ominus S) \ominus w = G \ominus W'$, we obtain $\mathrm{outdeg}_{G \ominus W'}(v) = \mathrm{outdeg}_G(v)$ for every $v \in V(G \ominus W')$. □

The condition on W' in Lemma 2.14 is sufficient to ensure that out-degrees are preserved, but cannot easily be omitted. Indeed, suppose that a vertex v has a child c that is also a child of a parent p of v. Then removing v via the $\ominus$-operator reduces the out-degree of p by one, so $\mathrm{outdeg}_{G\ominus v}(p) = \mathrm{outdeg}_G(p) - 1$. Even in a shortcut-free DAG, two vertices of out-degree one may have the same parent and the same child. Removing both vertices then decreases the out-degree of their common parent by one. Thus, shortcut-freeness alone does not guarantee preservation of out-degrees under $G \ominus W$.

In the final definition for this section, we provide a slight generalization of regular DAGs.

Definition 2.15 *(Hellmuth et al. (2023))* A DAG is *semi-regular* if it is shortcut-free and satisfies the following condition

(PCC): for all $u, v \in V(G)$ it holds that u and v are $\preceq_G$-comparable if and only if $\mathcal{C}_G(u) \subseteq \mathcal{C}_G(v)$ or $\mathcal{C}_G(v) \subseteq \mathcal{C}_G(u)$.

Theorem 4.6 of Lindeberg and Hellmuth (2025) shows that semi-regular DAGs generalize regular DAGs, i.e., that every regular DAG is also semi-regular. As noted earlier, the DAGs G, N, and N'' in Fig. 2 are regular, and thus also semi-regular. In contrast, the network N' is not semi-regular: it contains the $\preceq_{N'}$-incomparable vertices u and v that satisfy $\mathcal{C}_{N'}(u) = \{a, b\} = \mathcal{C}_{N'}(v)$. Therefore, N' violates the (PCC) property.

3 Regularizations

In this section, we consider a systematic way to transform a DAG into a regular DAG while preserving its least common ancestor (LCA) structure up to some prescribed level. The regularization procedure considered here is a specialization of the more general LCA-based simplification framework introduced in Lindeberg and Hellmuth (2025). We formulate this construction explicitly in terms of subsets of bounded cardinality, which leads to the notion of i-regularization. Intuitively, this amounts to removing vertices that do not play any role as LCAs of "small" subsets of leaves.

In Sect. 3.1, we recall the basic properties of this construction and derive new connections between i-regularization, lca-clusters, and Hasse diagrams. In particular, we characterize the set of clusters of $\mathrm{REG}_i(G)$ in terms of lca-clusters and show that $\mathrm{REG}_i(G)$ is isomorphic to the Hasse diagram of this cluster system (see Theorem 3.5). We further characterize when the fully regularized DAG $\mathrm{REG}(G)$ coincides with the Hasse diagram of the complete cluster system $\mathfrak{C}(G)$ (see Theorem 3.8).

In Sect. 3.2, we investigate how LCA-based regularization acts on normal networks and, in particular, clarify the relationship between normality, strong normality, and lca-relevance. We show that if N is a normal network, then $\mathrm{REG}(N)$ is again a normal network, and characterize when $\mathrm{REG}(N) = N$ (see Theorem 3.14). These structural results also provide important ingredients for Sect. 4, where we compare regularization with normalization within the common $\ominus$-framework.

3.1 Regularizing DAGs

Recall that $\mathscr{P}_{\leq i}(X)$ denotes the non-empty subsets of X of size at most i. For an integer $i \geq 1$ and a DAG G, define the set

$$\mathfrak{lca}_i(G) := \left\{v \in V(G) \mid v = \mathrm{lca}_G(A) \text{ for some } A \in \mathscr{P}_{\leq i}(X)\right\} \text{ and}$$
$$\overline{\mathfrak{lca}_i}(G) := V(G) \backslash \mathfrak{lca}_i(G).$$

In other words, $\mathfrak{lca}_i(G)$ contains each vertex of G that is a k-lca vertex for some $k \leq i$ while $\overline{\mathfrak{lca}_i}(G)$ is the complement of that set. Note that $X \subseteq \mathfrak{lca}_i(G)$ for all i, since every $x \in X$ satisfies $x = \mathrm{lca}_G(\{x\})$ for any DAG G on X.

Another simple observation stems from Lemma 2.10, namely, if a vertex v has out-degree one, then it is not an LCA of *any* set of leaves. Consequently, vertices of out-degree one are in particular always elements of $\overline{\mathfrak{lca}_i}(G)$ for any integer i and any DAG G. The next lemma gives a slight extension of this result which we will use later.

Lemma 3.1 *Suppose G is a DAG on X and let $W = \{w \in V(G) \mid \mathrm{outdeg}_G(w) = 1\}$. Then,*

$$X = \mathfrak{lca}_1(G) \subseteq \mathfrak{lca}_2(G) \subseteq \ldots \subseteq \mathfrak{lca}_{|X|}(G) \subseteq V(G)\backslash W, \quad \textit{and}$$
$$W \subseteq \overline{\mathfrak{lca}_{|X|}}(G) \subseteq \ldots \subseteq \overline{\mathfrak{lca}_2}(G) \subseteq \overline{\mathfrak{lca}_1}(G) = V(G)\backslash X$$

Proof Let G be a DAG on X, $|X| = \ell$ and $W = \{w \in V(G) \mid \mathrm{outdeg}_G(w) = 1\}$. Note that $\mathfrak{lca}_1(G) = X$ follows from the fact that $v = \mathrm{lca}_G(\{x\})$ for some $x \in X$ if and only if $v = x$. Moreover, the inclusions $\mathfrak{lca}_i(G) \subseteq \mathfrak{lca}_{i+1}(G)$ for each $1 \leq i < \ell$ follow from the definition: if $v \in \mathfrak{lca}_i(G)$, then v is a j-lca vertex for some j with $1 \leq j \leq i$ and thus clearly also for some j with $1 \leq j \leq i+1$, i.e. $v \in \mathfrak{lca}_{i+1}(G)$. Moreover, Lemma 2.10 implies that $w \notin \mathrm{LCA}_G(A)$ for all $w \in W$ and non-empty $A \subseteq X$. Consequently, $\mathfrak{lca}_\ell(G) \cap W = \emptyset$. Therefore,

$$X = \mathfrak{lca}_1(G) \subseteq \mathfrak{lca}_2(G) \subseteq \ldots \subseteq \mathfrak{lca}_\ell(G) \subseteq V(G)\backslash W.$$

The second chain of inclusions is now an immediate consequence of the latter statement and the fact that $\overline{\mathfrak{lca}_i}(G) = V(G) \setminus \mathfrak{lca}_i(G)$ for all i. □

We now introduce the main definition of this section. It defines a transformation that maps a DAG G to a DAG G' by first removing, via the $\ominus$-operator, all vertices that are not k-lca vertices for any $k \leq i$, and then removing all shortcuts. As we shall see, this transformation always yields an i-lca-relevant and regular DAG; see also Theorem 3.3.

Definition 3.2 For an integer $i \geq 1$, the *i-regularization* of a DAG G is $\mathrm{REG}_i(G) := (G \ominus \overline{\mathfrak{lca}_i}(G))^-$. In particular, if G is a DAG on X, then we define the *regularization* of G to be $\mathrm{REG}(G) := \mathrm{REG}_{|X|}(G)$.

Definition 3.2 is illustrated in Fig. 3. The figure also shows that, for a DAG G on X and integers $j < i$, the DAG $\mathrm{REG}_i(G)$ can be viewed as a refinement of $\mathrm{REG}_j(G)$, with the sequence starting at the arc-less DAG $\mathrm{REG}_1(G) = (X, \emptyset)$.

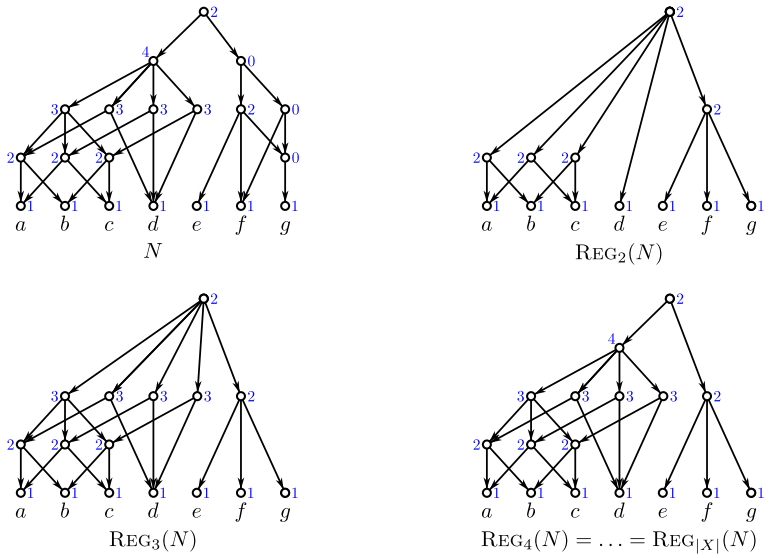


Fig. 3 A network N on $X = \{a, b, c, d, e, f, g\}$. Numbers i next to a vertex v denote the smallest integers such that $v \in \mathfrak{lca}_i(N)$. If $i = 0$, then the respective vertex is not a k-lca vertex for any k. In addition several networks $\text{REG}_i(N)$ are shown. Note that $\text{REG}_1(N) = (X, \emptyset)$ just consists of X and no arcs

As outlined in the introduction, various approaches to simplify networks have been proposed in recent years. In Lindeberg and Hellmuth (2025), Lindeberg and Hellmuth studied simplifications $G \ominus W$ of DAGs G in which each vertex $v \notin W$ is an i-lca vertex for some $i \in \mathscr{I}$, where $\mathscr{I}$ is a set of positive integers. The i-regularization of Definition 3.2 is therefore a natural specialization of this construction to $\mathscr{I} = \{1, \ldots, i\}$, followed by the removal of all shortcuts. In particular, the simplification $\varphi_{\text{lca}}(G)$ introduced in (Lindeberg and Hellmuth, 2025, Sec. 6) coincides with the "full" regularization of G as given in Definition 3.2, that is,

$$\text{REG}(G) = (G \ominus \overline{\mathfrak{lca}_{|X|}}(G))^{-} = \varphi_{\text{lca}}(G).$$

It follows that $\text{REG}(G)$ satisfies three additional "simplification" axioms that are desirable for such transformations, provided that G is a DAG satisfying the (CL) property; see (Lindeberg and Hellmuth, 2025, Sec. 6) and Heiss et al. (2025) for further details.

It is worth noting that, in general, there is no direct relationship between the number of roots of a DAG G and the number of roots of $\text{REG}_i(G)$ for $i \geq 1$. For example, the network N in Fig. 4 loses its unique root under regularization: $\text{REG}(N)$ has two distinct roots. On the other hand, the DAG G in Fig. 5 has three roots, while $\text{REG}(G)$ has only one. Thus, even when G is a network, $\text{REG}_i(G)$ need not be a network, and conversely, a DAG with several roots may have a regularization that is a network.

Our main motivation for considering $\text{REG}_i(G)$ stems from results established in Lindeberg and Hellmuth (2025), which we restate using our notation.



Fig. 4 The network N has a unique root, while the corresponding DAG $\mathrm{REG}(N)$ has two

Theorem 3.3 (Lindeberg and Hellmuth (2025)) *Let G be a DAG on X and $i \geq 1$ be an integer. Then, $\mathrm{REG}_i(G)$ is an i-lca-relevant and regular DAG on X. Moreover, G is regular if and only if $\mathrm{REG}(G) = G$.*

Proof Theorem 5.5 in Lindeberg and Hellmuth (2025) ensures that $G \ominus \overline{\mathfrak{lca}_i}(G)$ is i-lca-relevant for every DAG G and integer $i \geq 1$. Since $\mathrm{REG}_i(G) = (G \ominus \overline{\mathfrak{lca}_i}(G))^-$ is shortcut-free, Observation 2.13 implies that $\mathrm{REG}_i(G)$ is i-lca-relevant. Thus, regularity of $\mathrm{REG}_i(G)$ follows from Theorem 2.12. The second statement is an immediate consequence of the first statement and Theorem 2.12. □

The key point of Theorem 3.3 is that $\mathrm{REG}_i(G)$ is regular, which motivates its name. Consequently, its structure is determined solely by its set of clusters. Although (Lindeberg and Hellmuth 2025) implicitly considers $\mathrm{REG}_i(G)$, it does not provide a description of $\mathfrak{C}(\mathrm{REG}_i(G))$, which we now establish. To this end, we first introduce a new definition.

Definition 3.4 Let G be a DAG on X. The set of *lca-clusters* of G is defined to be

$$\mathfrak{C}_{\mathrm{lca}}(G) := \{C \in \mathfrak{C}(G) \mid \mathrm{lca}_G(C) \text{ is well-defined}\}.$$

Furthermore, for each integer $i \in \{1, 2, \ldots, |X|\}$, we define

$$\mathfrak{C}^i_{\mathrm{lca}}(G) := \{C \in \mathfrak{C}_{\mathrm{lca}}(G) \mid \exists A \in \mathscr{P}_{\leq i}(X) : \ C \text{ is the unique inclusion-minimal element in } \mathfrak{C}(G) \text{ containing } A\}.$$

Definition 3.4 is illustrated with the help of the following example.

Example In Fig. 2, two networks N' and N'' are shown that both have leaf set $X = \{a, b, c\}$. For N', we have $\mathfrak{C}(N') = \{\{a, b, c\}, \{a, b\}, \{a\}, \{b\}, \{c\}\}$ where $\mathrm{C}_{N'}(u) = \mathrm{C}_{N'}(v) = \{a, b\}$, i.e., u and v are not distinguishable from their associated clusters. Neither u nor v serve as a unique LCA for any subset of leaves. In particular, $\mathrm{C}_{N'}(v) = \mathrm{C}_{N'}(u) = \{a, b\} \notin \mathfrak{C}_{\mathrm{lca}}(N')$ and we obtain $\mathfrak{C}_{\mathrm{lca}}(N') = \{\{a, b, c\}, \{a\}, \{b\}, \{c\}\}$. It is easy to verify that $\mathfrak{C}^1_{\mathrm{lca}}(N') = \{\{a\}, \{b\}, \{c\}\}$ and $\mathfrak{C}^3_{\mathrm{lca}}(N') = \mathfrak{C}_{\mathrm{lca}}(N')$. In this example, we even have $\mathfrak{C}^2_{\mathrm{lca}}(N') = \mathfrak{C}_{\mathrm{lca}}(N')$ since $\{a, b, c\} \in \mathfrak{C}_{\mathrm{lca}}(N')$ is the unique inclusion-minimal element in $\mathfrak{C}(N')$ that contains for example $A = \{a, c\} \in \mathscr{P}_{\leq i}(X)$. For N'', we have $\mathfrak{C}(N'') = \{A \mid \emptyset \neq A \subseteq X\} = \mathscr{P}_{\leq 3}(X)$ and $\mathfrak{C}^i_{\mathrm{lca}}(N'') = \mathscr{P}_{\leq i}(X)$, $1 \leq i \leq 3$.

Note that, by definition, $\mathfrak{C}^i_{\mathrm{lca}}(G) \subseteq \mathfrak{C}^j_{\mathrm{lca}}(G)$ always holds whenever $i \leq j$. Moreover, there are DAGs G, for example phylogenetic trees, for which $\mathfrak{C}_{\mathrm{lca}}(G) = \mathfrak{C}(G)$ holds. We now describe the set of clusters of $\mathrm{REG}_i(G)$.

Theorem 3.5 *For each DAG G on X and every $i \in \{1, 2, \ldots, |X|\}$, $\mathfrak{C}(\mathrm{REG}_i(G)) = \mathfrak{C}^i_{\mathrm{lca}}(G)$. In particular, since $\mathrm{REG}_i(G)$ is regular, $\mathrm{REG}_i(G) \simeq \mathcal{H}(\mathfrak{C}^i_{\mathrm{lca}}(G))$.*

Proof Let G be a DAG on X, put $\ell := |X|$ and let $i \in \{1, \ldots, \ell\}$. We first note that the statements in Observation 2.13 are, in particular, valid for $H := \mathrm{REG}_i(G) = (G \ominus \overline{\mathfrak{lca}_i}(G))^-$ and that $V(H) = \mathfrak{lca}_i(G)$. Hence, to prove the theorem, we show that $\mathfrak{C}(H) = \mathfrak{C}^i_{\mathrm{lca}}(G)$.

First, suppose $C \in \mathfrak{C}(H)$. By definition, there exists some $v \in V(H)$ such that $\mathsf{C}_H(v) = C$. Recall that, from Observation 2.13, $\mathsf{C}_H(v) = \mathsf{C}_G(v)$. Moreover, since $v \in \mathfrak{lca}_i(G) = V(H)$ there is some $A \in \mathscr{P}_{\leq i}(X)$ such that $v = \mathrm{lca}_G(A)$. The latter implies that $\mathsf{C}_G(\mathrm{lca}_G(A)) = \mathsf{C}_G(v) = C$. Moreover, since $\mathrm{lca}_G(A)$ is well-defined, we can apply Lemma 2.9 to conclude that $\mathrm{lca}_G(\mathsf{C}_G(\mathrm{lca}_G(A))) = \mathrm{lca}_G(C)$ is well-defined (and equals v). In other words, $C \in \mathfrak{C}_{\mathrm{lca}}(G)$. Moreover, Lemma 2.9 also ensures that $C = \mathsf{C}_G(\mathrm{lca}_G(A))$ is the unique inclusion-minimal cluster in $\mathfrak{C}(G)$ that contains A. We have thus shown that $C \in \mathfrak{C}^i_{\mathrm{lca}}(G)$. In other words, $\mathfrak{C}(H) \subseteq \mathfrak{C}^i_{\mathrm{lca}}(G)$.

To see that $\mathfrak{C}^i_{\mathrm{lca}}(G) \subseteq \mathfrak{C}(H)$ holds, consider any cluster $C \in \mathfrak{C}^i_{\mathrm{lca}}(G)$. One easily verifies that there exists some $v \in V(G)$ such that $\mathsf{C}_G(v) = C$ and $v = \mathrm{lca}_G(C)$, see Lemma 2.9. We now claim that $v \in V(H)$. Since, by assumption, $C \in \mathfrak{C}^i_{\mathrm{lca}}(G)$, there exists, by definition, some $A \in \mathscr{P}_{\leq i}(X)$ such that C is the unique inclusion-minimal cluster in $\mathfrak{C}(G)$ containing A. Since $A \subseteq C$ and $v = \mathrm{lca}_G(C)$, v is a common ancestor of A and $\mathrm{LCA}_G(A) \neq \emptyset$. Consider any $v' \in \mathrm{LCA}_G(A)$. Since $A \subseteq \mathsf{C}_G(v')$ and C is the unique inclusion-minimal cluster in $\mathfrak{C}(G)$ containing A, we must have $C \subseteq \mathsf{C}_G(v')$. Hence, v' is a common ancestor C. The latter together with the fact that $v = \mathrm{lca}_G(C)$ is the unique least common ancestor of C implies that $v = \mathrm{lca}_G(C) \preceq_G v'$ must hold. As the latter holds for every $v' \in \mathrm{LCA}_G(A)$, we can conclude that $\mathrm{LCA}_G(A) = \{v\}$, i.e., that $v = \mathrm{lca}_G(A)$. Since $A \in \mathscr{P}_{\leq i}(X)$, it follows that $v \in \mathfrak{lca}_i(G) = V(H)$, as claimed.

Observation 2.13 then implies that $C = \mathsf{C}_G(v) = \mathsf{C}_H(v) \in \mathfrak{C}(H)$ i.e. $\mathfrak{C}^i_{\mathrm{lca}}(G) \subseteq \mathfrak{C}(H)$. Finally, to complete the proof, observe that Theorem 3.3 implies that $\mathrm{REG}_i(G)$ is regular. By definition of regular DAGs, $\mathrm{REG}_i(G) \simeq \mathcal{H}(\mathfrak{C}^i_{\mathrm{lca}}(G))$. □

Theorem 3.5 provides a characterization of the structure of $\mathrm{REG}_i(G)$ in terms of certain clusters of G. In the rest of this section, we shall pinpoint consequences of this result and explore further properties of the i-regularization operator.

A natural starting point is to consider the regularization of the archetypal phylogenetic structures, namely trees. One easily verifies that, in a tree, a vertex v is a k-lca vertex for some k if and only if v does not have out-degree one. Hence, $\overline{\mathfrak{lca}_i}(T)$ is precisely the set of vertices of out-degree one in any tree T and for all $i \geq 2$. Moreover, there is a well-known bijective correspondence between phylogenetic trees on X and hierarchies on X, see e.g. (Semple and Steel, 2003, Thm. 3.5.2), Thus, Theorem 3.5 immediately characterizes those graphs G for which $\mathrm{REG}_i(G)$ is a tree. We collect these observations in the following statement.

Observation 3.6 *For all $i \geq 2$, the i-regularization $\mathrm{REG}_i(T)$ of a tree T on X coincides with T', where T' is the tree obtained from T by suppressing all vertices of out-degree one. In particular, $\mathrm{REG}_i(T) = T$ for all phylogenetic trees T and all $i \geq 2$. Moreover,*

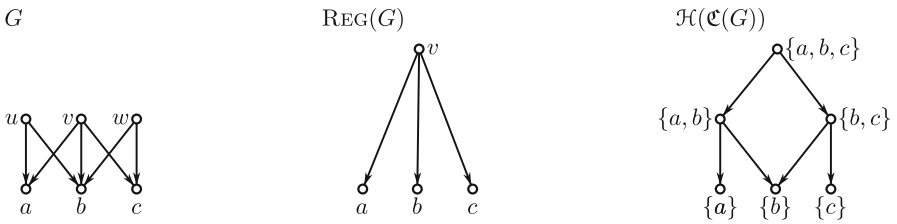


Fig. 5 A DAG G for which $\mathcal{H}(\mathfrak{C}(G)) \not\simeq \text{REG}(G)$ holds

for any DAG G and $i \geq 2$, the graph $\text{REG}_i(G)$ is a tree if and only if $\mathfrak{C}^i_{\text{lca}}(G)$ is a hierarchy.

Another straightforward observation is that LCAs are unaffected by the presence or absence of shortcuts. Consequently, adding or removing shortcuts in G does not affect $\text{REG}_i(G)$. This can be formalized as follows.

Lemma 3.7 *For every DAG G and integer $i \geq 1$, $\text{REG}_i(G) = \text{REG}_i(G^-)$.*

Proof Recall that, by definition, $V(G) = V(G^-)$ and that Lemma 2.1 states that $u \preceq_G v$ if and only if $u \preceq_{G^-} v$ for all $u, v \in V(G)$. It is now straightforward to verify that, for every non-empty $A \subseteq X$, $\text{LCA}_G(A) = \text{LCA}_{G^-}(A)$ and, therefore, $\mathfrak{lca}_i(G) = \mathfrak{lca}_i(G^-)$. This together with Lemma 2.7 implies that $V(\text{REG}_i(G)) = \mathfrak{lca}_i(G) = V(\text{REG}_i(G^-))$. Since $\text{REG}_i(G)$ and $\text{REG}_i(G^-)$ are shortcut-free the latter, taken together with the last statement of Lemma 2.1, implies that $\text{REG}_i(G) = \text{REG}_i(G^-)$. □

Note that $\mathcal{H}(\mathfrak{C}(G))$ and $\text{REG}(G)$ may differ; see Fig. 5. In this example, for the DAG G we have $v = \text{lca}_G(a, c)$, while neither u nor w is the unique LCA of any subset of the leaves. As a result, $\text{REG}(G)$ is just a "star tree". In contrast, $\mathcal{H}(\mathfrak{C}(G))$ not only differs substantially from G but also from $\text{REG}(G)$. As the next results shows, $\mathcal{H}(\mathfrak{C}(G))$ and $\text{REG}(G)$ are isomorphic precisely if $\mathfrak{C}(G) = \mathfrak{C}_{\text{lca}}(G)$.

Theorem 3.8 *For every DAG G, $\text{REG}(G) \simeq \mathcal{H}(\mathfrak{C}_{\text{lca}}(G))$. Moreover, $\text{REG}(G) \simeq \mathcal{H}(\mathfrak{C}(G))$ if and only if G satisfies property (CL).*

Proof Let G be a DAG on X and put $\ell = |X|$. Theorem 3.5 states that $\mathfrak{C}(\text{REG}(G)) = \mathfrak{C}(\text{REG}_\ell(G)) = \mathfrak{C}^\ell_{\text{lca}}(G)$. By definition and Lemma 2.9, $\mathfrak{C}^\ell_{\text{lca}}(G) = \mathfrak{C}_{\text{lca}}(G)$ holds. Since, by Theorem 3.3, $\text{REG}(G)$ is regular, the latter two arguments ensure that $\text{REG}(G) \simeq \mathcal{H}(\mathfrak{C}_{\text{lca}}(G))$ for all DAGs G. The second statement now follows immediately from the fact that, by definition, G satisfies property (CL) if and only if $\text{lca}_G(\mathbb{C}_G(v))$ is well-defined for all $v \in V(G)$, which is, if and only if $\mathfrak{C}(G) = \mathfrak{C}_{\text{lca}}(G)$. □

We close this section by discussing some computational aspects of i-regularization. As a starting point, note that the property of being 2-lca-relevant can be tested in polynomial time. More generally, for every fixed constant integer k, testing whether a DAG is k-lca-relevant can be done in polynomial time; see (Lindeberg and Hellmuth, 2025,

Obs. 7.6). Moreover, (Lindeberg and Hellmuth, 2025, Cor. 3.12) gives a polynomial-time algorithm for testing whether a given DAG is lca-relevant. In contrast, testing whether a DAG G is k-lca-relevant when k is part of the input is NP-complete (Lindeberg and Hellmuth, 2025, Thm. 7.5). We now extend these results to the computation of $\text{REG}_i(G)$ for certain values of i.

Proposition 3.9 *Given a DAG $G = (V, E)$ on X, the DAG $\text{REG}(G)$ can be constructed in polynomial time. Moreover, if i is a fixed constant, then $\text{REG}_i(G)$ can be constructed in polynomial time in $|V|$.*

Proof Let $G = (V, E)$ be a DAG on X. By definition, $\text{REG}(G) = (G \ominus \overline{\text{lca}_{|X|}}(G))^-$ and $\overline{\text{lca}_{|X|}}(G) = \{v \in V \mid v \text{ is not a } k\text{-lca vertex for any } k \leq |X|\}$. Together with Lemma 2.9, this implies that $\overline{\text{lca}_{|X|}}(G) = \{v \in V \mid v \neq \text{lca}_G(\text{C}_G(v))\}$. For each vertex $v \in V$, the set $\text{LCA}_G(\text{C}_G(v))$ can be computed in polynomial time using the algorithm from Lindeberg and Hellmuth (2025). Hence, we can determine in polynomial time whether $v = \text{lca}_G(\text{C}_G(v))$, and thus construct $\overline{\text{lca}_{|X|}}(G)$ in polynomial time. Finally, $\text{REG}(G) = (G \ominus \overline{\text{lca}_{|X|}}(G))^-$ can be computed in $O(|V|^2)$ time; see (Lindeberg and Hellmuth, 2025, Section 5) for further details.

Now let i be a fixed constant. We can determine $\overline{\text{lca}_i}(G)$ by computing $\text{LCA}_G(A)$ for all subsets $A \subseteq X$ with $|A| \leq i$. There are $O(|X|^i)$ such subsets, and each corresponding LCA set can be computed in polynomial time using the algorithm from Lindeberg and Hellmuth (2025). Since i is fixed, this yields a polynomial-time algorithm for constructing $\overline{\text{lca}_i}(G)$. As above, $\text{REG}_i(G) = (G \ominus \overline{\text{lca}_i}(G))^-$ can then be computed in polynomial time. □

3.2 Regularizing Normal Networks

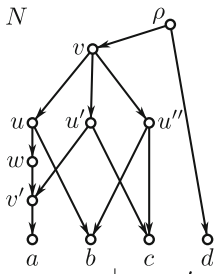
In this section, we investigate how LCA-based regularization acts on normal networks and, in particular, clarify the relationships between normality, strong normality, and lca-relevance. We show that regularization preserves normality and characterize precisely when $\text{REG}(N) = N$; see Theorem 3.14. These structural results also provide important ingredients for Sect. 4, where we compare regularization with normalization within the common $\ominus$-framework.

We note that the property of a vertex being visible and being a k-lca vertex are two unrelated properties; neither of them implies the other. In fact, there are networks N containing vertices of every possible combination of (non-)visible and (non-)k-lca vertices, see N and the table in Fig. 6. Nevertheless, as the next result shows, every strongly normal network is k-lca-relevant for $k \geq 2$. Thus, all vertices are k-lca vertices and, by Theorem 2.4, at the same time visible. This result has, in parts, been proven by Willson (Willson, 2010, Thm 3.9), albeit in somewhat different setting and using alternative notation.

Lemma 3.10 *Every strongly normal DAG on X is k-lca-relevant for all $k \in \{2, \ldots, |X|\}$.*

Proof Since the proof is straightforward, we have placed it in Appendix A. □

We now provide some characterizations of strongly normal networks.



	a, b, c, d	ρ	u, u', u''	v	v'	w
visible	Yes	Yes	No	Yes	Yes	No
k-lca	Yes ($k = 1$)	Yes ($k = 2, 3, 4$)	Yes ($k = 2$)	Yes ($k = 3$)	No	No

Fig. 6 A network N on $\{a, b, c, d\}$. The table specifies which of the vertices of N are (non-)visible or (non-)k-lca vertices

Proposition 3.11 *Let N be a network. Then, the following four statements are equivalent.*

(1) N is strongly normal.
(2) N is normal and 2-lca-relevant.
(3) N is normal and lca-relevant.
(4) N is normal, phylogenetic and does not contain hybrids having only one child.

Proof Lemma 3.10 shows that Condition (1) implies (2), and the implication from (2) to (3) is immediate by definition. If N is normal and lca-relevant, then the contraposition of Lemma 2.10 implies that no vertex of N has out-degree one. Hence, Condition (3) implies (1). The equivalence between (1) and (4) follows from the fact that, in a phylogenetic network, only hybrid vertices can have out-degree one. □

We now show that, for a normal DAG G, normality is preserved under the $\ominus$-operator when applied to vertices of out-degree one, and that these vertices are precisely those contained in $\overline{\mathrm{lca}_i}(G)$.

Proposition 3.12 *Let G be a normal DAG on X and $W \subseteq U := \{v \in V(G) \mid \mathrm{outdeg}_G(v) = 1\}$. Then, $G \ominus W$ is a normal DAG on X and $G \ominus U$ is strongly normal. In particular, $\overline{\mathrm{lca}_i}(G) = U$ for all integers $i \geq 2$.*

Proof Since the proof is straightforward but technical, we have placed it in Appendix A. □

Note that normal networks are not the only type of DAGs for which $\overline{\mathrm{lca}_i}(N) = \{v \in V(N) \mid \mathrm{outdeg}_N(v) = 1\}$. For example, consider the network N_1 in Fig. 7, for which $\overline{\mathrm{lca}_i}(N_1) = \{u\}$ for all $i \geq 2$, i.e., $\overline{\mathrm{lca}_i}(N_1)$ contains precisely the only vertex with out-degree one. Here, the vertex u has no tree-child and thus N_1 is not normal. Nevertheless, N_1 is phylogenetic and shortcut-free.

A straightforward consequence of Proposition 3.12 is the following. For a normal DAG G, the graph $G \ominus \overline{\mathrm{lca}_2}(G)$ is already shortcut-free. Moreover, $\overline{\mathrm{lca}_i}(G) = \overline{\mathrm{lca}_2}(G)$ for all $i \geq 2$, and hence $\mathrm{REG}_i(G)$ is obtained simply by removing the vertices of out-degree one. Hence, we obtain

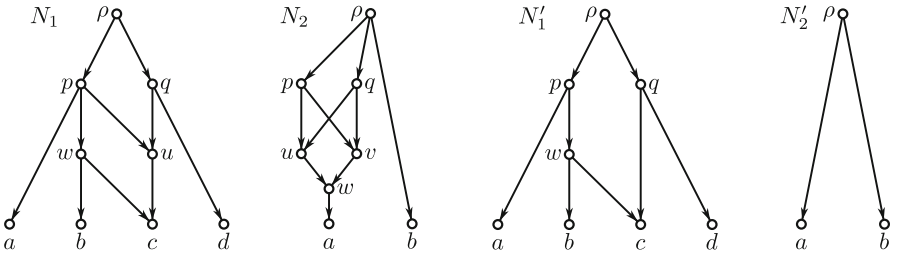


Fig. 7 Two networks N_1 and N_2 for which the regularization and normalization coincide, i.e., $\text{REG}(N_1) = N_1' = \text{NORM}(N_1)$ and $\text{REG}(N_2) = N_2' = \text{NORM}(N_2)$

Corollary 3.13 *For every normal DAG G it holds that* $\text{REG}(G) = \text{REG}_i(G) = G \ominus \overline{\text{lca}_2}(G)$ *for all integers $i \geq 2$, and, in particular,* $V(\text{REG}(G)) = \text{lca}_2(G)$.

Recall that $\text{REG}(N)$ may have multiple roots, even though N itself is a network (c.f. Fig. 4). Nevertheless, the latter cannot happen if N is a normal network as we now show in the main result of this section.

Theorem 3.14 *Let N be a normal network on X. Then,* $\text{REG}(N)$ *is a normal phylogenetic network on X. In particular, the following statements are equivalent.*

(1) $\text{REG}(N) = N$.
(2) N is regular.
(3) N is strongly normal.

Moreover, if $|X| \geq 2$, then Statements (1), (2), (3) and

(4) $\text{REG}_i(N) = N$ *for all* $i \in \{2, \ldots, |X|\}$

are equivalent.

Proof Let N be a normal network on X and let ρ be its unique root. By Corollary 3.13 and Proposition 3.12 we have that $\text{REG}(N) = N \ominus \overline{\text{lca}_2}(N)$, where $\overline{\text{lca}_2}(N) = \{v \in V(N) \mid \text{outdeg}_N(v) = 1\}$. This together with Proposition 3.12 implies that $\text{REG}(N)$ is a normal DAG on X. We show that $\text{REG}(N)$ is a network, i.e, that $\text{REG}(N)$ has a unique root.

If $\rho \in V(\text{REG}(N))$, then $v \preceq_N \rho$ for all $v \in V(N)$. Observation 2.13 therefore implies $v \preceq_{\text{REG}(N)} \rho$, for all $v \in V(\text{REG}(N))$ and hence, ρ remains the unique root of $\text{REG}(N)$. Now suppose that $\rho \notin V(\text{REG}(N))$, that is, $\text{outdeg}_N(\rho) = 1$. Starting at ρ, follow the unique outgoing arc until reaching the first vertex v with $\text{outdeg}_N(v) \neq 1$; such a vertex exists and may be a leaf. Then v is the unique $\preceq_N$-maximal vertex of N whose out-degree is distinct from one. By Observation 2.13, $u \preceq_{\text{REG}(N)} v$ for every $u \in V(\text{REG}(N))$. Thus, v is the unique root of $\text{REG}(N)$. In summary, $\text{REG}(N)$ is a normal *network* on X. Moreover, Theorem 3.3 implies that $\text{REG}(N)$ is regular and thus, by (Lindeberg and Hellmuth, 2025, Thm 4.6), also phylogenetic.

We now prove the remaining statements in the theorem. By Theorem 3.3, Statements (1) and (2) are equivalent. Suppose that (1) holds. Since N is shortcut-free, $N^- = N = \text{REG}(N)$. This together with Theorem 3.3 implies that N is lca-relevant.

Since N is normal, Proposition 3.11 implies that N is strongly normal. Hence, (1) implies (3). Suppose now that (3) holds.

Suppose first that $|X| = 1$ and let $X = \{x\}$. If N has more than one vertex, then its root ρ must have at least two distinct children u and v, since N is strongly normal. Since N is normal, Theorem 2.4 implies that both u and v are visible. Since $X = \{x\}$, visibility of u means that every ρx-path passes through u, and visibility of v means that every ρx-path passes through v. Hence, any ρx-path starting with the arc $\rho \to u$ must later pass through v, so $v \prec_N u$. Similarly, any ρx-path starting with $\rho \to v$ must later pass through u, so $u \prec_N v$, contradicting acyclicity. Hence, N consists of its single leaf and thus $\mathrm{REG}(N) = N$. Therefore, (3) implies (1) in case $|X| = 1$. In other words, Statements (1), (2) and (3) are equivalent in case $|X| = 1$.

Suppose now that $|X| \geq 2$. Lemma 3.10 implies that N is i-lca-relevant for all $i \in \{2, \ldots, |X|\}$. Thus, $\overline{\mathfrak{lca}_i}(N) = \emptyset$ and since N is shortcut-free, we obtain $N = N^- = (N \ominus \overline{\mathfrak{lca}_i}(N))^- = \mathrm{REG}_i(N)$ for all $i \in \{2, \ldots, |X|\}$. Hence, (3) implies (4). Finally, since $\mathrm{REG}(N) = \mathrm{REG}_{|X|}(N)$ and $|X| \geq 2$, Statement (4) immediately implies (1). Therefore, Statements (1), (2), (3) and (4) are equivalent in case $|X| \geq 2$. □

Note, in particular, that Statements (2) and (4) of Theorem 3.14 are not necessarily equivalent if N is not normal. For example, the network N'' in Fig. 2 is regular, but $\mathrm{REG}_2(N'') = N'' \ominus \rho \neq N''$.

4 Normalizing versus Regularizing

In Francis et al. (2021) the concept of *normalization* was introduced, that is, the transformation of a given network N into a normal version $\mathrm{NORM}(N)$ of the network that aims to retain the underlying evolution information whilst producing a network that has the desirable features of a normal network. Since clustering systems naturally arise in network inference, $\mathrm{REG}(N)$ – by virtue of being regular – also encodes useful phylogenetic signals through leaf-based information, since every vertex is the unique LCA of some subset of leaves (cf. Theorem 3.3). It is therefore natural to investigate how $\mathrm{REG}(N)$ and $\mathrm{NORM}(N)$ are related, as this could be useful information in practice when using these techniques.

One of the main goals of this section is to characterize those networks for which $\mathrm{REG}(N) = \mathrm{NORM}(N)$ holds. In particular, we show that $\mathrm{NORM}(N)$ can be fully expressed as $(N \ominus W)^-$ for a particular subset $W \subseteq V(N)$ and thus, conceptually, falls within the same paradigm as $\mathrm{REG}(N)$. Before providing the definition of $\mathrm{NORM}(N)$, we emphasize that (implicitly), all networks in Francis et al. (2021) are separated, i.e., all hybrids v in N have $\mathrm{outdeg}_N(v) = 1$. Moreover, no vertex v of the networks used in Francis et al. (2021) satisfies $\mathrm{indeg}(v) = \mathrm{outdeg}(v) = 1$. Hence, they are phylogenetic "up to" possibly the root having out-degree 1. We thus slightly generalize the definition of $\mathrm{NORM}(N)$ to arbitrary networks N.

Definition 4.1 *(Francis et al., 2021, Sec. 2)* Let N be a network. The directed graphs $\mathrm{COV}(N)$ and $\mathrm{NORM}(N)$ are constructed as follows.

(1) Let $G = (\mathfrak{vis}(N), A)$ be the DAG whose arc set A comprises all arcs $u \to v$ for which $u, v \in \mathfrak{vis}(N)$ and $v \prec_N u$ holds

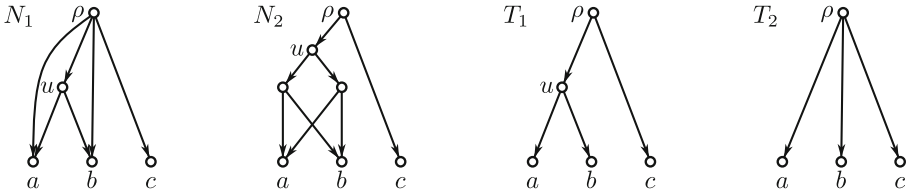


Fig. 8 Two networks N_1, N_2 and two trees T_1, T_2 on $\{a, b, c\}$, for which $T_1 = \text{REG}(N_1) = \text{NORM}(N_2)$ and $T_2 = \text{NORM}(N_1) = \text{REG}(N_2)$

(2) Put $\text{COV}(N) = G^-$.
(3) $\text{NORM}(N)$ is obtained from $\text{COV}(N)$ by suppressing all vertices v of $\text{COV}(N)$ for which $\text{indeg}_{\text{COV}(N)}(v) = \text{outdeg}_{\text{COV}(N)}(v) = 1$.

There are many networks N for which the normalization $\text{NORM}(N)$ and the regularization $\text{REG}(N)$ of N coincide. For example, considering the networks shown in Fig. 7, we have both $N_1' = \text{NORM}(N_1) = \text{REG}(N_1)$ and $N_2' = \text{NORM}(N_2) = \text{REG}(N_2)$. However, this is not always the case. For example, see Fig. 8 where $\text{NORM}(N_1) = \text{REG}(N_2) = T_2 \neq T_1 = \text{REG}(N_1) = \text{NORM}(N_2)$. In particular, there are lca-vertices that are not visible (e.g. vertex u in N_1) and there are visible vertices that are not lca-vertices (e.g. the vertex u in N_2). In other words, $\text{REG}(N)$ and $\text{NORM}(N)$ may identify different vertices as "omittable". A technical difficulty in comparing regularization and normalization is that the operator REG is invariant under the addition or removal of shortcuts. More precisely, $\text{REG}(N^-) = \text{REG}(N)$ always holds; cf. Lemma 3.7(2). In contrast, the normalization operator NORM does not share this invariance property.

To illustrate this fact, consider the networks shown in Fig. 8. The network N_1 differs from T_1 only by the presence of the two shortcuts $\rho \to a$ and $\rho \to b$. Although $\text{REG}(N_1^-) = \text{REG}(N_1) = T_1$, we have $\text{NORM}(N_1) = T_2 \neq T_1 = \text{NORM}(N_1^-)$. Thus, in general, $\text{NORM}(N) \neq \text{NORM}(N^-)$. Nevertheless, we show that normalization can be characterized using the $\ominus$-operator together with the removal of shortcuts. To this end, we start with establishing the connection between the graph $\text{COV}(N)$ in Definition 4.1 and the $\ominus$-operator combined with shortcut removal.

Lemma 4.2 *For all networks N on X, $\text{COV}(N) = (N \ominus \overline{\mathfrak{vis}}(N))^-$ is a normal network on X.*

Proof Since the proof is straightforward, we have placed it in Appendix A. □

To express $\text{NORM}(N)$ in terms of the $\ominus$-operator, we need to determine the set W for which $\text{NORM}(N) = (N \ominus W)^-$ holds. To this end, we define the following set

$$\mathfrak{dvis}(N) := \{v \in \mathfrak{vis}(N) \mid \text{indeg}_{\text{COV}(N)}(v) = \text{outdeg}_{\text{COV}(N)}(v) = 1\}.$$

The vertices in $\mathfrak{dvis}(N)$ are called *dispensably visible* (or d-vis) vertices. By definition, $\text{NORM}(N) = \text{COV}(N)$ if and only if $\text{COV}(N)$ contains no vertices u with $\text{indeg}_{\text{COV}(N)}(u) = \text{outdeg}_{\text{COV}(N)}(u) = 1$, that is, if and only if N contains no d-vis vertices. Importantly, the d-vis vertices in $\mathfrak{vis}(N)$ are determined in terms of their properties in $\text{COV}(N)$. Nevertheless, as the next result shows d-vis vertices of N can be identified solely based on properties they satisfy in N itself.

Lemma 4.3 *Let N be a network on X. We define the following two sets for all vertices $v \in \mathfrak{vis}(N)$:*

$$P_{\mathrm{vis}}(v) := \{u \in \mathfrak{vis}(N) \setminus \{v\} \mid \text{there is some } uv\text{-path } P \text{ in } N \text{ s.t. } V(P) \cap \mathfrak{vis}(N) = \{u, v\}\}$$
$$C_{\mathrm{vis}}(v) := \{u \in \mathfrak{vis}(N) \setminus \{v\} \mid \text{there is some } vu\text{-path } P \text{ in } N \text{ s.t. } V(P) \cap \mathfrak{vis}(N) = \{u, v\}\}$$

Then, for all $v \in \mathfrak{vis}(N)$, the following two statements are equivalent.

(1) $v \in \mathfrak{dvis}(N)$.
(2) (a) $P_{\mathrm{vis}}(v) \neq \emptyset$ *and* $P_{\mathrm{vis}}(v)$ *contains a unique* $\preceq_N$*-minimal vertex and*
(b) $C_{\mathrm{vis}}(v) \neq \emptyset$ *and* $C_{\mathrm{vis}}(v)$ *contains a unique* $\preceq_N$*-maximal vertex.*

Moreover, if $v \in \mathfrak{dvis}(N)$, then $v \notin \mathrm{LCA}_N(A)$ for all non-empty $A \subseteq X$. In particular, $\mathfrak{dvis}(N) \subseteq \overline{\mathfrak{lca}_i}(N)$ for all integers $i \geq 2$.

Proof Let N, $P_{\mathrm{vis}}(v)$ and $C_{\mathrm{vis}}(v)$ be defined as in the lemma's statement for some $v \in \mathfrak{vis}(N)$. We recall that Lemma 4.2 and Observation 2.13 together implies that

$$v \preceq_{\mathrm{Cov}(N)} u \iff v \preceq_N u, \quad \text{for all } u, v \in \mathfrak{vis}(N) = V(\mathrm{Cov}(N)). \tag{1}$$

First assume that v is d-vis and thus, that $\mathrm{indeg}_{\mathrm{Cov}(N)}(v) = \mathrm{outdeg}_{\mathrm{Cov}(N)}(v) = 1$. Let c denote the unique child of v in $\mathrm{Cov}(N)$. Hence, $v, c \in \mathfrak{vis}(N)$ and there is no vertex u of $\mathrm{Cov}(N)$ such that $c \prec_{\mathrm{Cov}(N)} u \prec_{\mathrm{Cov}(N)} v$. This together with the definition of $\mathrm{Cov}(N)$ implies that $c \prec_N v$ and that there is no $u \in \mathfrak{vis}(N)$ such that $c \prec_N u \prec_N v$. The latter two arguments imply that every path $P = v \rightsquigarrow c$ in N must satisfy $V(P) \cap \mathfrak{vis}(N) = \{v, c\}$ and, therefore, $c \in C_{\mathrm{vis}}(v)$. Let $w \in C_{\mathrm{vis}}(v)$ be a vertex such that $w \neq c$. By definition of $C_{\mathrm{vis}}(v)$, it holds that $v \neq w$ and there is a vw-path in N and thus, $w \prec_N v$ and, furthermore, $w \in \mathfrak{vis}(N) = V(\mathrm{Cov}(N))$ which together with Eq. (1) implies that $w \prec_{\mathrm{Cov}(N)} v$. Since c is the unique child of v in $\mathrm{Cov}(N)$ and $w \neq c$, we therefore have $w \prec_{\mathrm{Cov}(N)} c$ which, again, is equivalent to $w \prec_N c$. In other words, c must be the unique $\preceq_N$-maximal vertex in the set $C_{\mathrm{vis}}(v)$. With analogous arguments, one shows that the unique parent p of v in $\mathrm{Cov}(N)$ is the unique $\preceq_N$-minimal element of $P_{\mathrm{vis}}(v)$.

Assume, for the converse, that $P_{\mathrm{vis}}(v)$ contains a unique $\preceq_N$-minimal vertex p and that $C_{\mathrm{vis}}(v)$ contains a unique $\preceq_N$-maximal vertex c. We first show that if an arc $v \to u$ of $\mathrm{Cov}(N)$ exist, then $u \in C_{\mathrm{vis}}(v)$. To this end, note first that if $v \to u$ is an arc of $\mathrm{Cov}(N)$, then, by definition, $u \prec_N v$. Let P denote any path from v to u in N. If there would exist a visible vertex $w \in V(P) \setminus \{u, v\}$, then $u \prec_N w \prec_N v$ and Eq. (1) would imply that $u \prec_{\mathrm{Cov}(N)} w \prec_{\mathrm{Cov}(N)} v$; but that would make the arc $v \to u$ a shortcut of $\mathrm{Cov}(N)$, contradicting $\mathrm{Cov}(N)$ being shortcut-free. Hence $V(P) \cap \mathfrak{vis}(N) = \{u, v\}$ and, therefore, $u \in C_{\mathrm{vis}}(v)$. Thus, any arc $v \to u$ that potentially exists in $\mathrm{Cov}(N)$ must satisfy $u \in C_{\mathrm{vis}}(v)$. Now, by definition, each $u \in C_{\mathrm{vis}}(v)$ satisfies $u \prec_N v$. Hence, for each $u \in C_{\mathrm{vis}}(v)$, the arc $v \to u$ is initially added to the DAG G as in Definition 4.1(1) from which $\mathrm{Cov}(N) = G^-$ is obtained from. Since c is the unique $\preceq_N$-maximal vertex in $C_{\mathrm{vis}}(v)$ it follows that any other vertex $w \neq c$ in $C_{\mathrm{vis}}(v)$ (if there is one), must satisfy $w \prec_N c$. By the latter arguments, the arcs $v \to c$, $c \to w$ and $v \to w$ exists in G. However, since the arcs $v \to w$ is a shortcut in G, it will thus

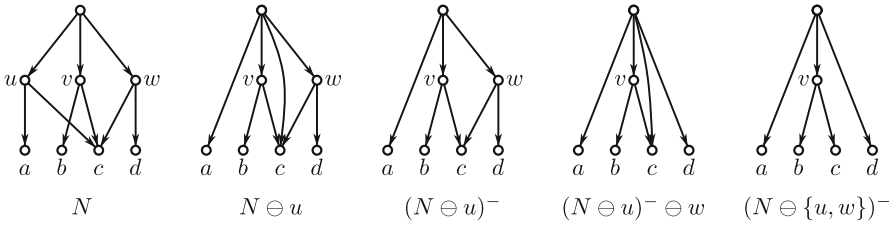


Fig. 9 Some networks resulting from the N on the very left after application of the $\ominus$-operator and removal of shortcuts. In particular, note that $(N \ominus u)^- \ominus w$ is not isomorphic to $(N \ominus \{u, w\})^-$

not remain in $\mathrm{Cov}(N) = G^-$. As this holds for all vertices $w \in C_{\mathrm{vis}}(v)\backslash\{c\}$ it follows that only the arc $v \to c$ exists in $\mathrm{Cov}(N)$ and therefore, that c is the unique child of v in $\mathrm{Cov}(N)$. In summary, $\mathrm{outdeg}_{\mathrm{Cov}(N)} = 1$. Using similar arguments, one can show that the unique $\preceq_N$-minimal vertex p in $P_{\mathrm{vis}}(v)$ is the unique parent of v in $\mathrm{Cov}(N)$ and thus, that $\mathrm{indeg}_{\mathrm{Cov}(N)} = 1$. By definition, $v \in \mathfrak{dvis}(N)$. Therefore Statement (1) and (2) are equivalent.

Now consider any $v \in \mathfrak{dvis}(N)$. Since $\mathrm{outdeg}_{\mathrm{Cov}(N)}(v) = 1$, one easily verifies that there is a path $P = v \rightsquigarrow u$ in $\mathrm{Cov}(N)$ where u satisfies $\mathrm{outdeg}_{\mathrm{Cov}(N)}(u) \neq 1$, and where each vertex w along P distinct from u (if there is one) satisfies $w \in \mathfrak{dvis}(N)$. Clearly, $\mathrm{C}_{\mathrm{Cov}(N)}(v) = \mathrm{C}_{\mathrm{Cov}(N)}(u)$ and $u \notin \mathfrak{dvis}(N)$. By Lemma 4.2, the vertices in X remain vertices in $\mathrm{Cov}(N)$. This and Eq. (1) implies that, for each $x \in X$, we have $x \preceq_{\mathrm{Cov}(N)} u$ if and only if $x \preceq_N u$. Consequently, $\mathrm{C}_{\mathrm{Cov}(N)}(u) = \mathrm{C}_N(u)$. Similarly, $\mathrm{C}_{\mathrm{Cov}(N)}(v) = \mathrm{C}_N(v)$. Since, additionally, $u \prec_N v$, we have shown that v has a proper descendant u in N with the same cluster as v. Consequently, $v \notin \mathrm{LCA}_N(\mathrm{C}_N(v))$ must hold, and this together with (Lindeberg and Hellmuth, 2025, Lem 3.4) implies that $v \notin \mathrm{LCA}_N(A)$ and, in particular, $v \neq \mathrm{lca}_N(A)$ for all non-empty subsets $A \subseteq X$. As the latter holds for all vertices $v \in \mathfrak{dvis}(N)$, we conclude that $\mathfrak{dvis}(N) \subseteq \overline{\mathfrak{lca}_{|X|}}(N)$. By Lemma 3.1, we thus have $\mathfrak{dvis}(N) \subseteq \overline{\mathfrak{lca}_i}(N)$ for all $i \geq 2$. □

By definition, suppression can be formulated in terms of the $\ominus$-operator (c.f. the discussion right after Definition 2.5), hence the following is an immediate result of Lemma 4.2 and the definition of $\mathrm{Norm}(N)$.

Theorem 4.4 *For all networks N we have* $\mathrm{Norm}(N) = \mathrm{Cov}(N) \ominus \mathfrak{dvis}(N) = (N \ominus \overline{\mathfrak{vis}}(N))^- \ominus \mathfrak{dvis}(N)$.

Note that $(N \ominus U) \ominus W = N \ominus (U \cup W)$ always holds by Lemma 2.6. However, the $\ominus$-operator could result in networks $N \ominus U$ that contain shortcuts even if the original network N is shortcut-free; for example, see the network $N \ominus u$ in Fig. 9. Hence, in general, $(N \ominus U)^- \ominus W \neq (N \ominus (U \cup W))^-$, since $(N \ominus U)^- \ominus W$ may contain shortcuts that, clearly, do not appear in $(N \ominus (U \cup W))^-$; see Fig. 9 again for an example. Intriguingly, as we shall see in Theorem 4.6, $\mathrm{Norm}(N) = (N \ominus (\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N)))^-$. In order to prove this theorem and for completeness, we now show that Theorem 3.3 of Francis et al. (2021) also holds for the more general setting of networks that are not necessarily phylogenetic and separated.

Proposition 4.5 *Let N be a network on X. The following statements hold.*

(1) $\mathrm{NORM}(N)$ *is a normal network on* X *for which* $V(\mathrm{NORM}(N)) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N) = V(N)\backslash(\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N))$.
(2) For all $u, v \in V(\mathrm{NORM}(N))$, $v \preceq_{\mathrm{NORM}(N)} u \iff v \preceq_N u$.
(3) If $v \in V(\mathrm{NORM}(N))$, *then* $\mathsf{C}_{\mathrm{NORM}(N)}(v) = \mathsf{C}_N(v)$.
(4) No vertex v *of* $\mathrm{NORM}(N)$ *satisfy* $\mathrm{indeg}_{\mathrm{NORM}(N)}(v) = \mathrm{outdeg}_{\mathrm{NORM}(N)}(v) = 1$. *In particular,* $\mathrm{NORM}(N)$ *is phylogenetic if and only if its root has out-degree distinct from 1.*
(5) $\mathrm{NORM}(N) = N$ *if and only if* N *is normal and there exists no* $v \in V(N)$ *such that* $\mathrm{indeg}_N(v) = \mathrm{outdeg}_N(v) = 1$.
(6) If N *is separated, then* $\mathrm{NORM}(N)$ *is separated.*
(7) $\mathrm{NORM}(N)$ *is a tree if and only if* $\{\mathsf{C}_N(v) \mid v \in \mathfrak{vis}(N)\}$ *is a hierarchy.*

Proof Since the proof is straightforward and resembles (Francis et al., 2021, Thm. 3.3), we have placed it in Appendix A. □

We are now in the position to extend Theorem 4.4 and to characterize those networks N for which $\mathrm{NORM}(N)$ and $\mathrm{REG}(N)$ coincide.

Theorem 4.6 *Let* N *be a network. Then* $\mathrm{NORM}(N) = (N \ominus (\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N)))^{-}$. *Moreover, for each* $i \geq 2$, *we have*

$$V(\mathrm{REG}_i(N)) \cap V(\mathrm{NORM}(N)) = \mathfrak{lca}_i(N) \cap \mathfrak{vis}(N) \tag{2}$$
$$V(\mathrm{REG}_i(N)) \setminus V(\mathrm{NORM}(N)) = \mathfrak{lca}_i(N) \setminus \mathfrak{vis}(N) \tag{3}$$
$$V(\mathrm{NORM}(N)) \setminus V(\mathrm{REG}_i(N)) = \mathfrak{vis}(N) \setminus (\mathfrak{lca}_i(N) \cup \mathfrak{dvis}(N)). \tag{4}$$

In particular, $\mathrm{NORM}(N) = \mathrm{REG}_i(N)$ *if and only if* $\mathfrak{lca}_i(N) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$ *and, therefore,* $\mathrm{NORM}(N) = \mathrm{REG}(N)$ *if and only if* $\mathfrak{lca}_{|X|}(N) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$.

Proof Let N be a network on X. For simplicity, put $N' := (N \ominus (\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N)))^{-}$. By definition of $\overline{\mathfrak{vis}}(N)$ and $\mathfrak{dvis}(N)$, $X \subseteq V(N)\backslash(\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N))$. This together with Observation 2.13 ensures that N' is a shortcut-free DAG on X such that $V(N') = (V(N)\backslash\overline{\mathfrak{vis}}(N))\backslash\mathfrak{dvis}(N) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$. In addition, Proposition 4.5 shows, in particular, that $\mathrm{NORM}(N)$ is a shortcut-free network on X with $V(\mathrm{NORM}(N)) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$. To summarize so far, the graphs $\mathrm{NORM}(N)$ and N' have the same set of vertices and are both shortcut-free. Moreover, by Observation 2.13, we have $u \preceq_{N'} v$ if and only if $u \preceq_N v$ for all $u, v \in V(N')$. By Proposition 4.5, $u \preceq_{\mathrm{NORM}(N)} v$ if and only if $u \preceq_N v$ for all $u, v \in V(\mathrm{NORM}(N))$. Combining the latter two arguments allow us to conclude that, for all $u, v \in \mathfrak{vis}(N) \setminus \mathfrak{dvis}(N)$, we have $u \preceq_{N'} v$ if and only if $u \preceq_{\mathrm{NORM}(N)} v$. By Lemma 2.1, we thus have $\mathrm{NORM}(N) = N' = (N \ominus (\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N)))^{-}$.

Now, fix some integer $i \geq 2$. By definition, we have $V(\mathrm{REG}_i(N)) = \mathfrak{lca}_i(N)$. By Proposition 4.5, $V(\mathrm{NORM}(N)) = \mathfrak{vis}(N) \setminus \mathfrak{dvis}(N)$. In addition, Lemma 4.3 implies that $\mathfrak{dvis}(N) \subseteq \overline{\mathfrak{lca}_i}(N)$ and, therefore, $\mathfrak{lca}_i(N) \cap \mathfrak{dvis}(N) = \emptyset$. It now readily follows that $V(\mathrm{NORM}(N)) \cap V(\mathrm{REG}_i(N)) = \mathfrak{vis}(N) \cap \mathfrak{lca}_i(N)$. Hence, Equation (2) holds. Furthermore, $V(\mathrm{REG}_i(N)) \setminus V(\mathrm{NORM}(N)) = \mathfrak{lca}_i(N) \setminus (\mathfrak{vis}(N) \setminus \mathfrak{dvis}(N))$. Since $\mathfrak{lca}_i(N) \cap \mathfrak{dvis}(N) = \emptyset$, the latter simplifies to $V(\mathrm{REG}_i(N))\backslash V(\mathrm{NORM}(N)) =$

$\mathfrak{lca}_i(N)\backslash\mathfrak{vis}(N)$, i.e., Equation (3) holds. Moreover, $V(\text{REG}_i(N)) = \mathfrak{lca}_i(N)$ and $V(\text{NORM}(N)) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$ implies that Equation (4) holds.

To see that the last statement in the theorem holds, first assume that $\text{NORM}(N) = \text{REG}_i(N)$. Since $V(\text{NORM}(N)) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$ and $V(\text{REG}_i(N)) = \mathfrak{lca}_i(N)$, we must in particular have $\mathfrak{vis}(N)\backslash\mathfrak{dvis}(N) = V(\text{NORM}(N)) = V(\text{REG}_i(N)) = \mathfrak{lca}_i(N)$. Conversely, assume that $\mathfrak{lca}_i(N) = \mathfrak{vis}(N) \setminus \mathfrak{dvis}(N)$. Recall that $\text{NORM}(N) = (N \ominus (\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N)))^-$ and so

$$\begin{aligned}\text{REG}_i(N) = (N \ominus \overline{\mathfrak{lca}_i}(N))^- &= (N \ominus (V \setminus \mathfrak{lca}_i(N)))^- \\ &= (N \ominus (V \setminus (\mathfrak{vis}(N) \setminus \mathfrak{dvis}(N))))^- \\ &= (N \ominus ((V \setminus \mathfrak{vis}(N)) \cup (V \cap \mathfrak{dvis}(N))))^- \\ &= (N \ominus (\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N)))^- = \text{NORM}(N).\end{aligned}$$

The latter, in particular, shows that $\mathfrak{lca}_i(N) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$ implies that $\text{REG}_i(N) = \text{NORM}(N)$. Hence setting $i = |X|$ implies that $\text{NORM}(N) = \text{REG}(N)$ if and only if $\mathfrak{lca}_{|X|}(N) = \mathfrak{vis}(N)\backslash\mathfrak{dvis}(N)$. □

Corollary 4.7 *For a network N and an integer $i \geq 2$, the following statements are equivalent.*

(1) $\text{NORM}(N) = \text{REG}_i(N)$
(2) $\mathfrak{vis}(N) \setminus \mathfrak{dvis}(N) = \mathfrak{lca}_i(N)$
(3) $\mathfrak{vis}(N) = \mathfrak{lca}_i(N) \cup \mathfrak{dvis}(N)$
(4) $\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N) = \overline{\mathfrak{lca}_i}(N)$
(5) $\mathfrak{lca}_i(N) \cap \overline{\mathfrak{vis}}(N) = \emptyset$ *and* $\overline{\mathfrak{lca}_i}(N) \cap \mathfrak{vis}(N) = \mathfrak{dvis}(N)$.

Proof Since the proof is tedious but straightforward, we have placed it in Appendix A. □

Corollary 4.8 *Given a network N, testing if $\text{REG}(N) = \text{NORM}(N)$ can be done in polynomial time in $|V(N)|$.*

Proof Let N be a network on X. As outlined in the appendix of Francis et al. (2021), $\text{NORM}(N)$ can be computed in polynomial time, whenever the network N is separated. In fact, the exact same approach may be used to compute $\text{NORM}(N)$ for a network that is not necessarily separated which, in particular, provides access to the set $V(N) = \mathfrak{vis}(N) \setminus \mathfrak{dvis}(N)$. As outlined in the proof of Proposition 3.9, the set $\overline{\mathfrak{lca}_{|X|}}(N)$ and, thus, $\mathfrak{lca}_{|X|}(N) = V(N) \setminus \overline{\mathfrak{lca}_{|X|}}(N)$ can be computed in polynomial time in $|V(N)|$. Since, by Theorem 4.6, $\text{NORM}(N) = \text{REG}(N)$ if and only if $\mathfrak{lca}_{|X|}(N) = \mathfrak{vis}(N) \setminus \mathfrak{dvis}(N)$, we can thus test in polynomial time whether $\text{NORM}(N) = \text{REG}(N)$ by a straightforward set comparison in $O(|V(N)|^2)$ time. □

5 Discussion and Outlook

In this paper, we have introduced and explored a framework for simplifying DAGs, where the simplified DAGs have the form $(G \ominus W)^-$ for certain subsets W of the

vertices of G. To be more precise, in Sect. 3, we studied the i-regularization $\textsc{Reg}_i(G)$ of a DAG G, where W is the set $\overline{\mathfrak{lca}_i}(G)$ of vertices that are not the unique least common ancestor of any subset of leaves of size at most i. In particular, we noted that $\textsc{Reg}_i(G)$ is both i-lca-relevant and regular (c.f. Theorem 3.3). We then proceeded to describe the clusters of $\textsc{Reg}_i(G)$ in Theorem 3.5 which, due to regularity, fully determine the structure of $\textsc{Reg}_i(G)$. In Sect. 3.2, we studied regularizations of normal networks, in particular showing that regularizations preserve the property of being a normal network (Theorem 3.14). Finally, in Sect. 4, we studied the normalization $\textsc{Norm}(N)$ of a network N and showed, in Theorem 4.6, that $\textsc{Norm}(N) = (N \ominus W)^-$ for $W = \overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N)$. In other words, the set W contains all vertices of N that are not visible and all vertices of N that are visible but have in- and out-degree 1 in $\textsc{Cov}(N)$ (where the latter vertices, in particular, can be determined by properties in N itself, see Lemma 4.3). Moreover, Theorem 4.6 also characterizes those networks N for which $\textsc{Norm}(N)$ and $\textsc{Reg}_i(N)$ coincide.

The comparison between regularization and normalization also highlights that neither simplification is uniformly more informative than the other. Their behavior depends on which type of leaf-based information is regarded as relevant. There are regular networks for which regularization retains essentially the entire structure while normalization retains only very little, since most vertices are not visible. Conversely, there are networks for which normalization retains substantial structure although few, or no, internal vertices occur as unique LCAs of subsets of leaves, so that regularization removes most of the network (refer back to Fig. 8, from which larger examples can easily be obtained). Thus, the two approaches should be viewed as complementary rather than competing simplification procedures. This difference also comes with a potential complexity trade-off. Regular DAGs may in general contain many more vertices than normal or strongly normal networks (see, for example, (Willson, 2010, Thm. 5.1)), and hence a regularization that retains a large collection of LCA-induced clusters may be substantially more complex than a corresponding normalization. On the other hand, such additional vertices may be relevant when they are supported by the available data, for example through clusters or inferred LCA relationships. Which simplification is preferable therefore depends not only on structural considerations but also on the type of information that can be reliably inferred.

It could be worthwhile to investigate how different choices of subsets W affect the simplification $(G \ominus W)^-$ of a given DAG or network G. By the results developed in Lindeberg and Hellmuth (2025), cf. Observation 2.13, the DAG $H := (G \ominus W)^-$ always preserves the $\preceq_G$-relation i.e. $u \preceq_H v$ if and only if $u \preceq_G v$ for all $u, v \in V(H)$. Consequently, $\mathfrak{C}(H) \subseteq \mathfrak{C}(G)$, provided that W contains no leaves of G. Thus, the simplification H never introduces clusters that were not already present in G.

In another more general direction, in (Francis, 2025, p.4), Francis states

> "The normalization captures the key information in *any* given network: the information that is in fact reconstructable, that is, the vertices and edges whose evolutionary signal may be visible in the leaves. If one has data that evolved from a process on a network, reconstructing the normalization of the network is the sweet spot: both the most information reasonable to hope for, and the most that can practically be obtained."

This provides a compelling argument for the normalization procedure in particular, and for normal networks more generally. Although normalization is designed to remove vertices that are not visible, visibility is a path-based notion of relevance that depends on the topology of the entire network. From the perspective of data-driven simplification, it may therefore not be always clear which observable data would directly support such a topological property, especially when the underlying network is unknown. For example, depending on the data at hand, it is conceivable that one may be able to infer the existence of a vertex v with a particular cluster $C = \mathsf{C}_N(v)$, even if no vertex u with $\mathsf{C}_N(u) = C$ is visible in the network N. Such a cluster would disappear under normalization, since no vertex with cluster C would be retained; cf. Proposition 4.5(3).

In general, such considerations could be very important when trying to take into account the data when simplifying networks. For example, there is evidence that relationships between LCAs can, to some extent, be inferred from so-called best hits (Geiß et al. 2020; Stadler et al. 2020). More precisely, pairwise similarities between genomic sequences associated with leaves a, b, and b', where a belongs to some species A and b, b' belong to a species B distinct from A, may support conclusions of the form $\mathrm{lca}_N(a, b) \preceq_N \mathrm{lca}_N(a, b')$ even without prior knowledge of the structure of the underlying network N. The vertices $\mathrm{lca}_N(a, b)$ and $\mathrm{lca}_N(a, b')$ need not be visible, however, and may therefore be removed by the normalization $\textsc{Norm}(N)$. Thus, it may be useful to consider more general simplifications of the form $(N \ominus W)^-$, where W need not contain all non-visible vertices. Such a choice of W allows one to retain non-visible vertices whose relevance is nevertheless supported by the available data, for instance because they induce observed clusters or occur as unique LCAs of pairs, or more generally subsets, of leaves.

In summary, the main point is that, depending on the type of data under consideration, there may be different notions of what constitutes the *key information* of a network and, consequently, of what makes a vertex relevant for simplification. Thus, network simplifications should aim to retain those vertices whose existence is supported, or witnessed, by the available data. If the relevant signal is visibility, then this requires information about paths from vertices to leaves. If the data are clusters, then vertices v with $\mathsf{C}_N(v)$ among the supported clusters should be retained, since their relevance is witnessed by the cluster system. Similarly, if the data support the existence or relative position of LCAs of pairs of leaves, then the corresponding LCA vertices should be retained. In this sense, the appropriate simplification of a network should be guided by the kind of evolutionary signal present in the data, rather than by a single universal notion of vertex relevance and, indeed, this information could also be useful for the more challenging problem of constructing a network in the first place.

A Outsourced Proofs

Proof of Lemma 2.6 Let G be a DAG and $u, v \in V(G)$ be distinct vertices. By definition, the vertex sets of $(G \ominus v) \ominus u$ and $(G \ominus u) \ominus v$ coincide. We show now that their arc sets coincide. To this end, let $p \to q$ be an arc of $(G \ominus v) \ominus u$ and thus, $\{p, q\} \cap \{u, v\} = \emptyset$ and, in particular, $|\{p, q, u, v\}| = 4$. We will show that $p \to q$ is an arc of $(G \ominus u) \ominus v$.

If $(p, q) \in E(G)$ then it is, by definition, also an arc in $(G \ominus u) \ominus v$. Suppose now that $(p, q) \notin E(G)$. Now, one of the following cases occurs.

Case: $(p, q) \in E(G \ominus v)$. Since $(p, q) \notin E(G)$, it holds that $p \in \mathrm{par}_G(v)$ and $q \in \mathrm{child}_G(v)$. Since $p, q, v \neq u$, the arcs $p \to v$ and $v \to q$ remain in $G \ominus u$ i.e. $p \in \mathrm{par}_{G \ominus u}(v)$ and $q \in \mathrm{child}_{G \ominus u}(v)$. Thus, by definition, $(p, q) \in E((G \ominus u) \ominus v)$.
Case: $(p, q) \notin E(G \ominus v)$. In this case, $(p, q) \in E((G \ominus v) \ominus u) \setminus E(G \ominus v)$ hold. Hence, by definition, $p \in \mathrm{par}_{G \ominus v}(u)$ and $q \in \mathrm{child}_{G \ominus v}(u)$. In other words, $(p, u), (u, q) \in E(G \ominus v)$. Now one of the following subcases occurs.

Subcase: $(p, u), (u, q) \in E(G)$. By definition, $(p, q) \in E(G \ominus u)$. Since, $p, q \neq v$ it follows that $(p, q) \in E((G \ominus u) \ominus v)$.
Subcase: $(p, u) \notin E(G), (u, q) \in E(G)$. Since (p, u) is an arc of $G \ominus v$ but not of G, the arcs $p \to v$ and $v \to u$ must exist in G. Since also $u \to q$ is an arc of G, it follows that there is a path $p \to v \to u \to q$ in G. By definition, $p \to v$ and $v \to q$ are arcs of $G \ominus u$, and it follows that $p \to q$ is an arc of $(G \ominus u) \ominus v$.
Subcase: $(p, u) \in E(G), (u, q) \notin E(G)$. Since $u \to q$ is an arc of $G \ominus v$ but not of G, $u \to v$ and $v \to q$ must be arcs of G. Thus, there is a path $p \to u \to v \to q$ in G. By similar arguments as in the previous case, $p \to q$ is an arc of $(G \ominus u) \ominus v$.
Subcase: $(p, u), (u, q) \notin E(G)$. As argued before, since $(p, u) \in E(G \ominus v)$ and $(p, u) \notin E(G)$, the arc $v \to u$ must exist in G. Moreover, since $(u, q) \in E(G \ominus v)$ and $(u, q) \notin E(G)$, $u \to v$ is an arc of G. The existence of the two arcs $u \to v$ and $v \to u$ in G, however, contradicts the fact that G is a DAG. Hence, this case cannot occur.

In summary, in all cases that can occur, $(p, q) \in E((G \ominus v) \ominus u)$ implies $(p, q) \in E((G \ominus u) \ominus v)$. By analogous argumentation, $E((G \ominus u) \ominus v) \subseteq E((G \ominus v) \ominus u)$ holds and thus, the arc sets of the graphs coincide. Therefore, $(G \ominus v) \ominus u = (G \ominus u) \ominus v$. □

Proof of Lemma 3.10 Let G be a strongly normal DAG on X. If G has $k \geq 2$ roots $r_1, \ldots, r_k$, we construct the network N by adding a new root ρ and the arc $\rho \to r_i$ for each $1 \leq i \leq k$. If G, instead, is already a network, we put $N := G$ and let ρ denote its unique root. It is easy to see that in both cases, $N = (V, E)$ is a strongly normal network on X. We will now use (Willson, 2010, Thm 3.9) established by Willson to prove the result. To this end, we define the *base-set* $\widetilde{X}_N$ of N as the set of vertices in N comprising the root, the leaves, and all vertices with out-degree 1. In our setting $\widetilde{X}_N = X \cup \{\rho\}$, since strongly normal networks by definition have no vertices with out-degree 1. Willson has shown that each vertex $v \in V \setminus \widetilde{X}_N$ satisfies $v = \mathrm{lca}_N(x, y)$ for some $x, y \in \widetilde{X}_N$. Note that $\mathrm{lca}_N(\{x\}) = x$ for all $x \in X$. By the latter arguments,

all vertices $v \in V \setminus \{\rho\}$ satisfy $v = \mathrm{lca}_N(x, y)$ for some $x, y \in X \cup \{\rho\}$. As $v \prec_N \rho$ for all $v \in V \setminus \{\rho\}$, we obtain $v = \mathrm{lca}_N(x, y)$ for some $x, y \in X$.

Hence, if G has multiple roots, then the latter result together with the fact that $u \preceq_G v$ if and only if $u \preceq_N v$ for all $u, v \in V(G) = V \setminus \{\rho\}$ implies that, for all $v \in V(G)$, there are some $x, y \in X$ such that $v = \mathrm{lca}_N(x, y)$. Therefore, G is 2-lca-relevant in case $G \neq N$.

We now consider the case when $N = G$. Hence, in order to show that N is 2-lca-relevant, it remains to show that $\rho = \mathrm{lca}_N(x, y)$ for some $x, y \in X$. To this end, we extend N by adding a vertex r and the arc $r \to \rho$, resulting in the network N^*. It is easy to verify that N^* remains a normal network on X with base-set $\widetilde{X}_{N^*} = X \cup \{r\}$. We can now apply (Willson, 2010, Thm 3.9) again to conclude that $\rho = \mathrm{lca}_{N^*}(x, y)$ for some $x, y \in X \cup \{r\}$. By definition of least common ancestors we have $x, y \preceq_{N^*} \mathrm{lca}_{N^*}(x, y) = \rho \prec_{N^*} r$, which ensures that $r \neq x$ and $r \neq y$. In other words, $\rho = \mathrm{lca}_{N^*}(x, y)$ implies $x, y \in X$. It is now straight-forward to check that $\rho = \mathrm{lca}_N(x, y)$ holds. In summary, G is 2-lca-relevant, in the case that $G = N$.

Thus, G is 2-lca-relevant in both cases. In particular, G is k-lca-relevant for all $k \in \{2, \ldots, |X|\}$. □

Proof of Proposition 3.12 Let $G = (V, E)$ be a normal DAG on X and put $U := \{v \in V \mid \mathrm{outdeg}_G(v) = 1\}$. Let $v \in U$ and denote with c the unique child of v. By Lemma 2.7, $G \ominus v$ is a DAG on X. We start with showing that $G \ominus v$ remains normal. Since G is tree-child and since c is the unique child of v it must be a tree-vertex, i.e., $\mathrm{indeg}_G(c) = 1$, which implies that v is the unique parent of c.

We continue with showing that $G \ominus v$ remains tree-child. Let $u \in V(G \ominus v) \setminus X$. We consider now the following cases for u: (i) $u = c$ and (ii) u is a parent of v and (iii) $u \neq c$ and u is not a parent of v.

Case (i): $u = c$. Since G is tree-child and $u \notin X$, $u = c$ has a tree-child c' in G. Since the children of c in $G \ominus v$ are precisely the children of c in G it follows that c' is a child of c in $G \ominus v$. Moreover, as v is the unique parent of c and $u \neq v$, it follows that c' is not a child of (and since G is a DAG also no parent of) v. Hence, the out-degree and in-degree of c' in G is the same as in $G \ominus v$, and we can conclude that c' remains a tree-child of $u = c$ in $G \ominus v$.

Case (ii): u is a parent of v in G. If u is the unique parent of v in G, then u is the unique parent of c in $G \ominus v$, in which case c becomes the new tree-child of u in $G \ominus v$. If u is one of at least two parents, then v is not a tree-child of u in G since $\mathrm{indeg}_G(v) > 1$. However, u must have a tree-child c' in G. As v is the unique parent of c it follows that $c \neq c'$ and, therefore, c' cannot be a child of v in G. Moreover, c' cannot be a parent of v in G as otherwise the arc $u \to v$ would be a shortcut. Hence, the out-degree and in-degree of c' in G is the same as in $G \ominus v$, and we can conclude that c' remains a tree-child of u in $G \ominus v$.

Case (iii): u is neither a parent nor a child of v. Since G is tree-child, u must have a tree-child c' in G. Since u is not a parent of v, it follows that $c' \neq v$ and, therefore, $c' \in V(G \ominus v)$. Since $u \neq v$ and v is the unique parent of c, we can conclude that $c \neq c'$. It immediately follows, by construction of $G \ominus v$ that the in-degree of c' in G and $G \ominus v$ remain the same, i.e., c' remains a tree-child of u. In summary, all vertices $u \in V(G \ominus v) \setminus X$ have a tree-child and thus, $G \ominus v$ is tree-child.

We continue by showing that $G \ominus v$ remains shortcut-free. Assume, for contradiction, that $G \ominus v$ contains a shortcut (a, b). Hence, there is an alternative path $P = a \rightsquigarrow b$ in $G \ominus v$ that does not contain the arc (a, b). Note that $b \prec_{G \ominus v} a$, $v \notin X$ and Observation 2.13 implies that $b \prec_G a$ holds. Suppose first that $(a, b) \notin E(G)$. The latter is, by construction of $G \ominus v$, only possible if $c = a$ or $c = b$ holds for the unique child c of v. However, since the children of c and their descendants in G and $G \ominus v$ are identical, $a = c$ is not possible as then $(a, b) \in E(G)$. Thus, $c = b$ must hold, in which case a must be a parent of v in G, i.e., there is an arc $a \to v$ in G. Since c is the unique child of v in G and v the unique parent of c in G, it follows that any path from a to $b = c$ must contain the arc $v \to b$. However, there is the alternative P from a to b in $G \ominus v$. One easily verifies that this is only possible if there is a path $P' = a \rightsquigarrow v \to b$ in G where the subpath $a \rightsquigarrow v$ contains more than the single arc $a \to v$. But then G is not shortcut-free; a contradiction. Therefore, $(a, b) \in E(G)$ must hold. However, since $P = a \rightsquigarrow b$ is a directed path from a to b in $G \ominus v$ it follows that v is not contained in P and Observation 2.13 implies that all vertices w in P satisfy $b \preceq_G w \preceq_G a$. As at least one such vertex $w \neq a, b$ exists it follows that there is an alternative path $a \rightsquigarrow b$ in G, a contradiction to G being shortcut-free. Thus, $G \ominus v$ is shortcut-free.

In summary, we have shown that $G \ominus v$ is a shortcut-free and tree-child DAG on X, i.e., $G \ominus v$ is a normal DAG on X. Recall that, since G is tree-child, the unique child c of v satisfies $\mathrm{indeg}_G(c) = 1$. Therefore, Lemma 2.14 implies that the out-degrees of vertices in $G \ominus v$ are the same in G and $G \ominus v$, the previous arguments and induction can be used to show that for all $W \subseteq U$ it holds that $G \ominus W$ is a normal DAG on X. As U is the set of all vertices of out-degree 1 in G and each vertex in U has a tree-child, Lemma 2.14 also implies that $G \ominus U$ cannot have vertices of out-degree 1. Hence, $G \ominus U$ is, in particular, strongly normal.

Finally, we will show that $\overline{\mathrm{lca}_i}(G) = U$ for all $i \geq 2$. We first show that $\overline{\mathrm{lca}_2}(G) = U$ by verifying that $V \setminus \overline{\mathrm{lca}_2}(G) = V \setminus U$. Since, by Lemma 3.1, $U \subseteq \overline{\mathrm{lca}_2}(G)$ it follows that $V \setminus \overline{\mathrm{lca}_2}(G) \subseteq V \setminus U$. Now let $v \in V \setminus U$. By definition, $v \in V(G \ominus U)$; see also Lemma 2.7. As shown in the previous paragraph, $G \ominus U$ is strongly normal and so Lemma 3.10 implies that $G \ominus U$ is 2-lca-relevant. Hence, there are leaves $x, y \in X$ such that $v = \mathrm{lca}_{G \ominus U}(x, y)$. In other words, $\mathrm{LCA}_{G \ominus U}(\{x, y\}) = \{v\}$. This together with Lemma 2.14 implies that $\mathrm{LCA}_G(\{x, y\}) = \{v\}$ and, therefore, $v = \mathrm{lca}_G(x, y)$. By definition, $v \in \mathrm{lca}_2(G) = V \setminus \overline{\mathrm{lca}_2}(G)$. In conclusion, $U = \overline{\mathrm{lca}_2}(G)$ holds. This together with Lemma 3.1 implies that $\overline{\mathrm{lca}_i}(G) = U$ for all integers $i \geq 2$. □

Proof of Lemma 4.2 Let N be a network on X with unique root ρ. Let G denote the digraph as specified in Definition 4.1(1). Hence, $\mathrm{Cov}(N) = G^-$. Since $V(G) = \mathfrak{vis}(N)$ and $u \to v$ is an arc of G if and only if $v \prec_N u$, it follows that there is a directed uv-path in G if and only if $u, v \in \mathfrak{vis}(N)$ and $v \preceq_N u$. Consequently, G is a DAG such that

$$v \preceq_G u \iff v \preceq_N u, \quad \text{for all } u, v \in \mathfrak{vis}(N). \tag{5}$$

Note that, by definition of visibility, $\rho \in \mathfrak{vis}(N)$ and $X \subseteq \mathfrak{vis}(N)$. Moreover, for all v in N, $v \preceq_N \rho$ and there is a leaf $x \in X$ with $x \preceq_N v$. The latter two arguments

together with Eq. (5) ensure that ρ remains the unique root of G and that the set of leaves of G is X and, in particular, that G is network on X

Now, by definition, $\mathrm{Cov}(N) = G^-$ and $\mathrm{Cov}(N)$ is, by Lemma 2.1 and the arguments in the previous paragraph, a shortcut-free network on X which satisfy $V(\mathrm{Cov}(N)) = V(G) = \mathfrak{vis}(N)$. Moreover, Lemma 2.1 together with Eq. (5) also ensures that

$$v \preceq_{\mathrm{Cov}(N)} u \iff v \preceq_N u, \quad \text{for all } u, v \in \mathfrak{vis}(N). \tag{6}$$

Now consider $N' := (N \ominus \overline{\mathfrak{vis}}(N))^-$. Since $\overline{\mathfrak{vis}}(N) \subseteq V(N) \backslash X$, we can apply Lemma 2.1 and Lemma 2.7 to conclude that $V(N') = V(N) \setminus \overline{\mathfrak{vis}}(N) = \mathfrak{vis}(N)$ and that

$$v \preceq_N u \iff v \preceq_{N'} u, \quad \text{for all } u, v \in \mathfrak{vis}(N). \tag{7}$$

By, Lemma 2.1, N' is shortcut-free. We have thus shown that N' and $\mathrm{Cov}(N)$ are shortcut-free DAGs with common vertex set $\mathfrak{vis}(N)$. Moreover, Eqs. (6) and (7) together shows that $u \preceq_{\mathrm{Cov}(N)} v$ if and only if $u \preceq_{N'} v$ for all $u, v \in \mathfrak{vis}(N)$. Thus Lemma 2.1 implies that $\mathrm{Cov}(N) = N' = (N \ominus \overline{\mathfrak{vis}}(N))^-$.

Finally, we show that $\mathrm{Cov}(N)$ is normal network on X. Recall that $\mathrm{Cov}(N)$ is a network on X. We continue by showing that every vertex of $\mathrm{Cov}(N)$ is visible. Assume, for contradiction, that there is a vertex v in $\mathrm{Cov}(N)$ that is not visible. Let $x \in X$ be a leaf with $x \preceq_N v$ and $P_1 = \rho \rightsquigarrow v \rightsquigarrow x$ be directed path in $\mathrm{Cov}(N)$. Since, v is not visible in $\mathrm{Cov}(N)$ there is an alternative path $P_2 = \rho \rightsquigarrow x$ that does not contain v. By Eq. (6), there are paths $P_1' = \rho \rightsquigarrow v \rightsquigarrow x$ and $P_2' = \rho \rightsquigarrow x$ in N where P_2' does not contain v. Hence, v is not visible in N; a contradiction to $v \in V(\mathrm{Cov}(N)) = \mathfrak{vis}(N)$. Thus, all vertices of $\mathrm{Cov}(N)$ are visible. Since $\mathrm{Cov}(N) = G^-$ is a shortcut-free network, Theorem 2.4 implies that $\mathrm{Cov}(N)$ is normal. □

Proof of Proposition 4.5 Let N be a network on X and consider $\mathrm{Norm}(N)$. By Theorem 4.4, $\mathrm{Norm}(N) = \mathrm{Cov}(N) \ominus \mathfrak{dvis}(N) = (N \ominus \overline{\mathfrak{vis}}(N))^- \ominus \mathfrak{dvis}(N)$. By definition, $\mathrm{Cov}(N)$ has vertex set $\mathfrak{vis}(N)$ and, by Lemma 4.2, $\mathrm{Cov}(N)$ is a normal network on X. Furthermore, by definition, $\mathfrak{dvis}(N) \subseteq \{v \in V(\mathrm{Cov}(N)) \mid \mathrm{outdeg}_{\mathrm{Cov}(N)}(v) = 1\}$. The latter two facts together with Proposition 3.12 ensure that $\mathrm{Cov}(N) \ominus \mathfrak{dvis}(N)$ is a normal DAG on X. Since, by Theorem 4.4, $\mathrm{Norm}(N) = \mathrm{Cov}(N) \ominus \mathfrak{dvis}(N)$, it follows that $\mathrm{Norm}(N)$ is a normal DAG on X. In particular, Lemma 2.7 ensures that $V(\mathrm{Norm}(N)) = V(\mathrm{Cov}(N)) \setminus \mathfrak{dvis}(N) = \mathfrak{vis}(N) \setminus \mathfrak{dvis}(N)$. In addition, Theorem 4.4 implies that $V(\mathrm{Norm}(N)) = V(N) \setminus (\overline{\mathfrak{vis}}(N) \cup \mathfrak{dvis}(N))$, and Lemma 2.7 and Eq. (6) together ensure that

$$v \preceq_{\mathrm{Norm}(N)} u \iff v \preceq_{\mathrm{Cov}(N)} u \iff v \preceq_N u \quad \text{for all } u, v \in V(\mathrm{Norm}(N)) \subseteq V(\mathrm{Cov}(N)).$$

In particular, the unique root ρ of $\mathrm{Cov}(N)$ will, by definition, remain a vertex in $\mathrm{Norm}(N)$ and satisfies $v \preceq_{\mathrm{Norm}(N)} \rho$ for all $v \in V(\mathrm{Norm}(N))$. Therefore, $\mathrm{Norm}(N)$ has a unique root and is indeed a normal *network* on X. We have thus proven statement (1) and (2) in the proposition. Statement (3) is an immediate consequence of (2), since for all $x \in X$ and $v \in V(\mathrm{Norm}(N))$ we have $x \preceq_N v$ if and only if $x \preceq_{\mathrm{Norm}(N)} v$.

For the first part of Statement (4), recall that, by definition, $\mathrm{Norm}(N) = \mathrm{Cov}(N) \ominus \mathfrak{dvis}(N)$. It is easily verified that if $v \in \mathfrak{dvis}(N)$ i.e. if $\mathrm{indeg}_{\mathrm{Cov}(N)}(v) =$

$\mathrm{outdeg}_{\mathrm{Cov}(N)}(v) = 1$, then $\mathrm{outdeg}_{\mathrm{Cov}(N)\ominus v}(u) = \mathrm{outdeg}_{\mathrm{Cov}(N)}(u)$ and $\mathrm{indeg}_{\mathrm{Cov}(N)\ominus v}(u) = \mathrm{indeg}_{\mathrm{Cov}(N)}(u)$ for all $u \in V(\mathrm{Cov}(N) \ominus v)$. Consequently, $\mathrm{Norm}(N) = \mathrm{Cov}(N) \ominus \mathfrak{dvis}(N)$ contains no vertex of in- and out-degree 1. The second part of Statement (4) is an immediate consequence of the first part and the definition of phylogenetic networks.

Next consider Statement (5). The *only-if* part is an immediate consequence of Statement (1) and (4). For the *if* part, suppose that N is normal and that there exists no $v \in V(N)$ such that $\mathrm{indeg}_N(v) = \mathrm{outdeg}_N(v) = 1$. By Theorem 2.4, $V(N) = \mathfrak{vis}(N)$, so Lemma 4.2 implies that $\mathrm{Cov}(N) = (N \ominus \overline{\mathfrak{vis}}(N))^- = (N \ominus \emptyset)^- = N$, in particular applying that N is shortcut-free. Since $\mathrm{Cov}(N) = N$ and there is no $v \in V(N)$ such that $\mathrm{indeg}_N(v) = \mathrm{outdeg}_N(v) = 1$, we also have $\mathfrak{dvis}(N) = \emptyset$. Hence, $\mathrm{Norm}(N) = \mathrm{Cov}(N) \ominus \mathfrak{dvis}(N) = \mathrm{Cov}(N) = N$ follows.

The final two statements can be proved verbatim as in the proof of (Francis et al., 2021, Thm. 3.3) (see Appendix A therein) and are therefore left to the reader. □

Proof of Corollary 4.7 Let $N = (V, E)$ be a network. The equivalence between Condition (1) and (2) has been shown in Theorem 4.6. Suppose Condition (2) holds. Then, $\mathfrak{vis}(N) = (\mathfrak{vis}(N) \setminus \mathfrak{dvis}(N)) \cup \mathfrak{dvis}(N) = \mathfrak{lca}_i(N) \cup \mathfrak{dvis}(N)$. Since, by Lemma 4.3, $\mathfrak{dvis}(N) \subseteq \overline{\mathfrak{lca}_i}(N)$ it follows that $\mathfrak{lca}_i(N)$ and $\mathfrak{dvis}(N)$ are disjoint. Taken the latter two arguments together, $\mathfrak{vis}(N) = \mathfrak{lca}_i(N) \uplus \mathfrak{dvis}(N)$ and, thus, Condition (3) holds. Suppose Condition (3) holds. Hence, $V \setminus \mathfrak{vis}(N) = V \setminus (\mathfrak{lca}_i(N) \uplus \mathfrak{dvis}(N))$. Note that $V \setminus \mathfrak{vis}(N) = \overline{\mathfrak{vis}}(N)$ and that $V \setminus (\mathfrak{lca}_i(N) \uplus \mathfrak{dvis}(N)) = (V \setminus \mathfrak{lca}_i(N)) \setminus \mathfrak{dvis}(N) = \overline{\mathfrak{lca}_i}(N) \setminus \mathfrak{dvis}(N)$. Hence, $\overline{\mathfrak{vis}}(N) = \overline{\mathfrak{lca}_i}(N) \setminus \mathfrak{dvis}(N)$ and consequently, $\overline{\mathfrak{vis}}(N) \uplus \mathfrak{dvis}(N) = \overline{\mathfrak{lca}_i}(N)$, i.e., Condition (4) holds. If Condition (4) holds, then $N \ominus (\overline{\mathfrak{vis}}(N) \uplus \mathfrak{dvis}(N)) = N \ominus \overline{\mathfrak{lca}_i}(N)$. Thus, $(N \ominus (\overline{\mathfrak{vis}}(N) \uplus \mathfrak{dvis}(N)))^- = (N \ominus \overline{\mathfrak{lca}_i}(N))^-$. By definition, $\mathrm{Reg}_i(N) = (N \ominus \overline{\mathfrak{lca}_i}(N))^-$ and, by Theorem 4.6, $\mathrm{Norm}(N) = (N \ominus (\overline{\mathfrak{vis}}(N) \uplus \mathfrak{dvis}(N)))^-$. Therefore, $\mathrm{Norm}(N) = \mathrm{Reg}_i(N)$ and Condition (1) holds. In summary, Condition (1)–(4) are equivalent.

For the final equivalence, first assume that Condition (3), or equivalently, (4) holds. By Condition (3), $\mathfrak{vis}(N) = \mathfrak{lca}_i(N) \uplus \mathfrak{dvis}(N)$. Hence, $\overline{\mathfrak{vis}}(N) = V \setminus \mathfrak{vis}(N) = V \setminus (\mathfrak{lca}_i(N) \uplus \mathfrak{dvis}(N)) \subseteq V \setminus \mathfrak{lca}_i(N)$. Therefore, $\mathfrak{lca}_i(N) \cap \overline{\mathfrak{vis}}(N) = \emptyset$. Moreover, by Condition (4), $\overline{\mathfrak{lca}_i}(N) \cap \mathfrak{vis}(N) = (\overline{\mathfrak{vis}}(N) \uplus \mathfrak{dvis}(N)) \cap \mathfrak{vis}(N) = (\overline{\mathfrak{vis}}(N) \cap \mathfrak{vis}(N)) \cup (\mathfrak{dvis}(N) \cap \mathfrak{vis}(N)) = \mathfrak{dvis}(N)$ where the latter equation follows from the fact that, by definition, $\overline{\mathfrak{vis}}(N) \cap \mathfrak{vis}(N) = \emptyset$ and $\mathfrak{dvis}(N) \subseteq \mathfrak{vis}(N)$. Thus, Condition (5) holds. Assume that Condition (5) holds. By definition, $V(N) = \mathfrak{lca}_i(N) \uplus \overline{\mathfrak{lca}_i}(N)$ and $V(N) = \mathfrak{vis}(N) \uplus \overline{\mathfrak{vis}}(N)$. Hence, $\mathfrak{vis}(N) = (\mathfrak{lca}_i(N) \uplus \overline{\mathfrak{lca}_i}(N)) \setminus \overline{\mathfrak{vis}}(N) = (\mathfrak{lca}_i(N) \setminus \overline{\mathfrak{vis}}(N)) \uplus (\overline{\mathfrak{lca}_i}(N) \setminus \overline{\mathfrak{vis}}(N))$. Since, by assumption, $\mathfrak{lca}_i(N) \cap \overline{\mathfrak{vis}}(N) = \emptyset$, we obtain $\mathfrak{vis}(N) = \mathfrak{lca}_i(N) \uplus (\overline{\mathfrak{lca}_i}(N) \setminus \overline{\mathfrak{vis}}(N))$. Moreover, since $V(N) = \mathfrak{vis}(N) \uplus \overline{\mathfrak{vis}}(N)$, we obtain $\overline{\mathfrak{lca}_i}(N) \setminus \overline{\mathfrak{vis}}(N) = \overline{\mathfrak{lca}_i}(N) \cap \mathfrak{vis}(N) = \mathfrak{dvis}(N)$. Hence, $\mathfrak{vis}(N) = \mathfrak{lca}_i(N) \uplus \mathfrak{dvis}(N)$, i.e., Condition (3) holds. □

Acknowledgements We sincerely thank the anonymous reviewers for their careful reading of the manuscript and for their constructive and detailed comments. Their suggestions helped us to improve the presentation, clarify the relationship to previous work, and sharpen the exposition of the main contributions.

Funding Open access funding provided by Stockholm University.

Data Availability No data sets are associated with this manuscript.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

van der Aalst WMP (1998) The application of petri nets to workflow management. J Circuits Syst Comput 8(1):21–66. https://doi.org/10.1142/S0218126698000043

van der Aalst WMP (2004) Workflow mining: discovering process models from event logs. IEEE Trans Knowl Data Eng 16(9):1128–1142. https://doi.org/10.1109/TKDE.2004.47

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 J Comput 10(3):405–421. https://doi.org/10.1137/0210030

Baroni M, Steel M (2006) Accumulation phylogenies. Ann Comb 10(1):19–30. https://doi.org/10.1007/s00026-006-0271-0

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

Baroni M, Semple C, Steel M (2006) Hybrids in real time. Syst Biol 55(1):46–56. https://doi.org/10.1080/10635150500431197

Cardona G, Rosselló F, Valiente G (2009) Comparison of tree-child phylogenetic networks. IEEE/ACM Trans Comp Biol Bioinf 6:552–569. https://doi.org/10.1109/TCBB.2007.70270

Döcker J, Linz S, Semple C (2019) Displaying trees across two phylogenetic networks. Theoret Comput Sci 796:129–146. https://doi.org/10.1016/j.tcs.2019.09.003

Dress A, Moulton V, Steel M, Wu T (2010) Species, clusters and the 'tree of life': a graph-theoretic perspective. J Theor Biol 265(4):535–542. https://doi.org/10.1016/j.jtbi.2010.05.031

Francis A (2025) "Normal" phylogenetic networks may be emerging as the leading class. J Theor Biol 614:112236. https://doi.org/10.1016/j.jtbi.2025.112236

Francis A, Huson DH, Steel M (2021) Normalising phylogenetic networks. Mol Phylogenet Evol 163:107215. https://doi.org/10.1016/j.ympev.2021.107215

Geiß M, Stadler PF, Hellmuth M (2020) Reciprocal best match graphs. J Math Biol 80(3):865–953. https://doi.org/10.1007/s00285-019-01444-2

Gusfield D (2014) ReCombinatorics: the algorithmics of ancestral recombination graphs and explicit phylogenetic networks. The MIT Press. https://doi.org/10.7551/mitpress/9432.001.0001

Heiss J, Huson DH, Steel M (2025) Transformations to simplify phylogenetic networks. Bull Math Biol 87(2):20. https://doi.org/10.1007/s11538-024-01398-7

Hellmuth M, Lindeberg A (2026) Characterizing and transforming DAGs within the I-lca framework. Discret Appl Math 378:584–593. https://doi.org/10.1016/j.dam.2025.08.037

Hellmuth M, Schaller D, Stadler PF (2023) Clustering systems of phylogenetic networks. Theory Biosci 142(4):301–358. https://doi.org/10.1007/s12064-023-00398-w

Huber KT, Moulton V, Wu T (2019) Hierarchies from lowest stable ancestors in nonbinary phylogenetic networks. J Classif 36(2):200–231. https://doi.org/10.1007/s00357-018-9279-5

Huson DH, Scornavacca C (2011) A survey of combinatorial methods for phylogenetic networks. Genome Biol Evol 3:23–35. https://doi.org/10.1093/gbe/evq077

Koller D, Friedman N (2009) Probabilistic graphical models: principles and techniques. MIT Press

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

Linz S, Semple C (2020) Caterpillars on three and four leaves are sufficient to reconstruct binary normal networks. J Math Biol 81(4):961–980. https://doi.org/10.1007/s00285-020-01533-7

Moret B, Nakhleh L, Warnow T, Linder C, Tholse A, Padolina A, Sun J, Timme R (2004) Phylogenetic networks: modeling, reconstructibility, and accuracy. IEEE/ACM Trans Comput Biol Bioinf 1(1):13–23. https://doi.org/10.1109/TCBB.2004.10

Pearl J (2009) Causality: models, reasoning, and inference, 2nd edn. Cambridge University Press https://doi.org/10.1142/S0218126698000043

Semple C, Steel M (2003) Phylogenetics, oxford lecture series in mathematics and its applications. Oxford University Press, Oxford, UK

Shanavas AV, Changat M, Hellmuth M, Stadler PF (2024) Unique least common ancestors and clusters in directed acyclic graphs. In: Kalyanasundaram S, Maheshwari A (eds) Algorithms and discrete applied mathematics, Springer Nature Switzerland, Cham, pp 148–161, https://doi.org/10.1007/978-3-031-52213-0_11

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 Mol Biol 15(1):5. https://doi.org/10.1186/s13015-020-00165-2

Willson S (2012) CSD homomorphisms between phylogenetic networks. IEEE/ACM Trans Comput Biol Bioinf 9(4):1128–1138. https://doi.org/10.1109/TCBB.2012.52

Willson SJ (2010) Properties of normal phylogenetic networks. Bull Math Biol 72(2):340–358. https://doi.org/10.1007/s11538-009-9449-z

Willson SJ (2011) Restricted trees: simplifying networks with bottlenecks. Bull Math Biol 73(10):2322–2338. https://doi.org/10.1007/s11538-010-9624-2

Willson SJ (2016) Comparing and simplifying distinct-cluster phylogenetic networks. Ann Comb 20(4):917–938. https://doi.org/10.1007/s00026-016-0324-y

Willson SJ (2022) Merging arcs to produce acyclic phylogenetic networks and normal networks. Bull Math Biol 84(2):26. https://doi.org/10.1007/s11538-021-00986-1

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.