Which Phylogenetic Networks Are Level-k Networks with Additional Arcs? Structure and Algorithms

Takatora Suzuki
Department of Pure and Applied Mathematics, Graduate School of Fundamental Science and Engineering, Waseda University, Tokyo, Japan

Momoko Hayamizu[1]
Department of Applied Mathematics, Faculty of Science and Engineering, Waseda University, Tokyo, Japan

Abstract

Reticulate evolution gives rise to complex phylogenetic networks, making their interpretation challenging. A typical approach is to extract trees within such networks. Since Francis and Steel's seminal paper, "Which Phylogenetic Networks are Merely Trees with Additional Arcs?" (2015), tree-based phylogenetic networks and their support trees (spanning trees with the same root and leaf-set as a given network) have been extensively studied. However, not all phylogenetic networks are tree-based, and for the study of reticulate evolution, it is often more biologically relevant to identify support networks rather than trees. This study generalizes Hayamizu's structure theorem, which yielded optimal algorithms for various computational problems on support trees of rooted almost-binary phylogenetic networks, to extend the theoretical framework for support trees to support networks. This allows us to obtain a direct-product characterization of each of three sets: all, minimal, and minimum support networks, for a given network. Each characterization yields optimal algorithms for counting and generating the support networks of each type. Applications include a linear-time algorithm for finding a support network with the fewest reticulations (i.e., the minimum tier). We also provide exact and heuristic algorithms for finding a support network with the minimum level, both running in exponential time but practical across a reasonably wide range of reticulation numbers.

2012 ACM Subject Classification Applied computing → Biological networks

Keywords and phrases Phylogenetic networks, Support networks, Level-k networks, Tier-k networks, Structure theorem, Enumeration, Optimization

Digital Object Identifier 10.4230/LIPIcs.WABI.2025.19

Related Version *Previous Version (Preprint)*: `https://doi.org/10.48550/arXiv.2505.11947`

Supplementary Material *Software (Source Code, Datasets and Detailed Experimental Results)*: `https://github.com/hayamizu-lab/structure-support-networks`
archived at `swh:1:dir:a40004b85200c9d46525feb46dc740a1f83ab3eb`

Funding *Takatora Suzuki*: JST SPRING Program Grant Number JPMJSP2128, Japan.
Momoko Hayamizu: JST FOREST Program Grant Number JPMJFR2135, Japan.

Acknowledgements The authors thank Tsuyoshi Urata, Haruki Miyaji, and Yukihiro Murakami for their discussions, Manato Yokoyama for help with the GitHub repository, and anonymous reviewers for their careful reading and useful comments.

[1] Corresponding author

1 Introduction

Evolutionary histories involving reticulate events, such as horizontal gene transfer, hybridization, and recombination, are better represented by networks than trees. However, networks can be far more complex than trees, making their interpretation challenging. A typical approach is to find meaningful subgraphs, particularly spanning trees, of these networks.

Since Francis and Steel's seminal paper "Which Phylogenetic Networks are Merely Trees with Additional Arcs?" [8], tree-based networks and their support trees (spanning trees with the same root and leaf-set as the network, also known as subdivision trees) have been extensively studied (e.g., [9, 10, 16]). Hayamizu [9] developed optimal (linear-time or linear-delay) algorithms for various computational problems, including counting, listing, and optimization of support trees, by establishing a theoretical foundation called a structure theorem for rooted almost-binary phylogenetic networks, a class encompassing binary ones. This theorem provides a canonical way to decompose any such network into its unique maximal zig-zag trails, and characterizes the family of edge-sets of support trees of the network.

While tree-based phylogenetic networks encompass many well-studied subclasses such as tree-child [1], stack-free [14], and orchard networks [4, 12], networks inferred from biological data are not necessarily tree-based. This has led to increased interest in non-tree-based networks (e.g., [2, 6, 7, 15]). Indeed, for the study of reticulate evolution, it is often more biologically relevant to consider networks within networks rather than trees. In such contexts, finding a most concise subgraph, such as one with the fewest reticulations (minimum tier) or with the minimum level, is particularly meaningful. These metrics can be interpreted as measures of deviation from being tree-based [7, 6], as they equal zero for tree-based networks.

In this paper, we generalize Hayamizu's structure theorem to build a theoretical framework for support networks of rooted almost-binary phylogenetic networks. Building on the maximal zig-zag trail decomposition established in [9], we derive direct-product characterizations of three families of support networks for a given network N: all support networks $\mathcal{A}_N$, minimal support networks $\mathcal{B}_N$, and minimum support networks $\mathcal{C}_N$. These characterizations yield closed-form product formulas for their cardinalities, revealing unexpected connections to Fibonacci, Lucas, Padovan, and Perrin numbers. We present a linear-time algorithm for counting each of $|\mathcal{A}_N|$, $|\mathcal{B}_N|$, and $|\mathcal{C}_N|$, as well as a linear-delay algorithm to list the support networks in each family.

Our theoretical results lead to practical algorithms for two key optimization problems: Problem 1 (Reticulation Minimization) and Problem 2 (Level Minimization). Problem 1 asks for a support network with the minimum reticulation number (minimum tier). We present a linear-time optimal algorithm for solving Problem 1, which can be achieved by selecting any element from $\mathcal{C}_N$. Problem 2 asks for a support network with the minimum level. We conjecture that this problem is NP-hard and present exact and heuristic algorithms for solving Problem 2 (Algorithms 1 and 2). Although they are both exponential-time algorithms, they are practical for networks with a reasonably wide range of reticulation numbers. The heuristic is particularly scalable and accurate in most cases.

The remainder of the paper is organized as follows. In Section 2, we define graph theoretical terminology and review the relevant materials on support trees (Section 2.1) and the structure theorem (Section 2.2). In Section 3, we define support networks, their families $\mathcal{A}_N$, $\mathcal{B}_N$, and $\mathcal{C}_N$, and relevant concepts. Section 4 provides characterizations and counting formulas for each family. Numerical results are also presented in Section 4.4. In Section 5, we give a linear-time algorithm for solving Problem 1. Section 6 presents the exact and heuristic algorithms for Problem 2, along with performance evaluations using synthetic data. Section 7 concludes with a summary of our contributions and directions for future research.

2 Preliminaries

The graphs in this paper are finite, simple (i.e. having neither loops nor multiple edges), acyclic directed graphs, unless otherwise stated. For a graph G, $V(G)$ and $E(G)$ denote the sets of vertices and edges of G, respectively. For two graphs G and H, G is a *subgraph* of H if both $V(G) \subseteq V(H)$ and $E(G) \subseteq E(H)$ hold, in which case we write $G \subseteq H$. Two graphs G and H are *isomorphic*, denoted by $G = H$, if there exists a bijection $\varphi : V(G) \to V(H)$ such that $(u, v) \in E(G)$ if and only if $(\varphi(u), \varphi(v)) \in E(H)$ for all $u, v \in V(G)$. A subgraph G of H is *proper* if $G \neq H$. A subgraph G of H is a *spanning* subgraph of H if $V(G) = V(H)$.

Given a graph G and a non-empty subset $S \subseteq E(G)$, the edge-set S is said to *induce the subgraph* $G[S]$ *of* G, that is, the one whose edge-set is S and whose vertex-set is the set of the ends of all edges in S. For a graph G with $|E(G)| \geq 1$ and a partition $\{E_1, \ldots, E_d\}$ of $E(G)$, the collection $\{G[E_1], \ldots, G[E_d]\}$ is a *decomposition* of G, and G is said to be *decomposed into* $G[E_1], \ldots, G[E_d]$. Here we recall that a partition of a set is a collection of pairwise disjoint non-empty subsets whose union is the entire set.

For an edge $e = (u, v)$ of a graph G, u and v are denoted by $\mathrm{tail}(e)$ and $\mathrm{head}(e)$, respectively. For a vertex v of a graph G, the *in-degree of* v *in* G, denoted by $\mathrm{indeg}_G(v)$, is the cardinality of $\{e \in E(G) \mid \mathit{head}(e) = v\}$. The *out-degree of* v *in* G, denoted by $\mathrm{outdeg}_G(v)$, is defined in a similar way. For any graph G, a vertex $v \in V(G)$ with $\mathrm{outdeg}_G(v) = 0$ is called a *leaf* of G. *Subdividing* an edge (u, v) means replacing it with a directed path from u to v of length at least two. *Smoothing* a vertex v where $\mathrm{indeg}_G(v) = \mathrm{outdeg}_G(v) = 1$ means suppressing v from G, namely, the reverse operation of edge subdivision.

An undirected graph is *connected* if there is a path between every pair of vertices. For a connected simple undirected graph G, a *cut vertex* (resp. *cut edge*) of G is a vertex (resp. edge) whose removal disconnects G, and a *block* of G is a maximal connected subgraph of G that contains no cut vertex. In this paper, a *block* of a directed graph refers to a block of its underlying undirected graph.

2.1 Phylogenetic networks and support trees

Suppose X represents a non-empty finite set of present-day species. A *rooted almost-binary phylogenetic network (on a leaf-set X)* is defined to be a finite simple directed acyclic graph N with the following properties (P1)–(P3). The vertex ρ is called *the root* of N, and any vertex v with $\mathrm{indeg}_N(v) > 1$ is called a *reticulation* of N.

(P1) There exists a unique vertex ρ of N with $\mathrm{indeg}_N(\rho) = 0$ and $\mathrm{outdeg}_N(\rho) \in \{1, 2\}$;
(P2) The set of leaves of N is identical to X;
(P3) For any $v \in V(N) \setminus (X \cup \{\rho\})$, $\mathrm{indeg}_N(v) \in \{1, 2\}$ and $\mathrm{outdeg}_N(v) \in \{1, 2\}$;

The above definition allows N to contain a vertex v with $\mathrm{indeg}_N(v) = \mathrm{outdeg}_N(v) = 2$ or $\mathrm{indeg}_N(v) = \mathrm{outdeg}_N(v) = 1$. In this sense, it slightly generalizes the notion of rooted *binary* phylogenetic networks N on X, which is defined by (P1), (P2) and (P4). When N has the properties (P1), (P2) and (P5), N is particularly called a rooted binary phylogenetic *tree* on X.

(P4) For any $v \in V(N) \setminus (X \cup \{\rho\})$, $\{\mathrm{indeg}_N(v), \mathrm{outdeg}_N(v)\} = \{1, 2\}$.
(P5) For any $v \in V(N) \setminus (X \cup \{\rho\})$, $(\mathrm{indeg}_N(v), \mathrm{outdeg}_N(v)) = (1, 2)$.

While the concept of support trees was originally defined for rooted binary phylogenetic networks in [8], the theoretical framework developed in [9], which includes the structure theorem and associated algorithms and forms the foundation of the present work, applies equally to rooted almost-binary networks. Therefore, in this paper, following the approach

taken in [15], we will consider support trees of almost-binary (including binary) networks, unless stated otherwise. For brevity, we omit "rooted almost-binary" when no confusion is likely to arise.

▶ **Definition 1.** *Let N be a rooted almost-binary phylogenetic network on X. If there exists a rooted binary phylogenetic tree $\tilde{T}$ on X and a spanning tree T of N such that $\tilde{T}$ is obtained by smoothing all $v \in V(T)$ with* $\mathrm{indeg}_T(v) = \mathrm{outdeg}_T(v) = 1$*, then N is called a* tree-based *network on X. In this case, T is called a* support tree *of N and $\tilde{T}$ a* base tree *of N.*

As a support tree T is a spanning tree of $N = (V, E)$, each T is specified by its edge-set. This leads to the natural question: Which subsets S of E yield a support tree $N[S] = (V, S)$ of N? Francis and Steel [8] proved that such "admissible" subsets S of E are characterized by the following conditions (C1)–(C3). As in [9], we slightly generalize the original definition in [8] so that we can consider admissible edge selections for any subgraph of N.

▶ **Definition 2.** *Let N be a rooted almost-binary phylogenetic network and let Z be any subgraph of N. A subset S of $E(Z)$ is* admissible *if it satisfies the following conditions:*

(C1) *If (u, v) is an edge of Z with* $\mathrm{outdeg}_N(u) = 1$ *or* $\mathrm{indeg}_N(v) = 1$*, then S contains (u, v).*

(C2) *If e_1 and e_2 are distinct edges of Z with* $\mathrm{tail}(e_1) = \mathrm{tail}(e_2)$*, then S contains at least one of $\{e_1, e_2\}$.*

(C3) *If e_1 and e_2 are distinct edges of Z with* $\mathrm{head}(e_1) = \mathrm{head}(e_2)$*, then S contains exactly one of $\{e_1, e_2\}$.*

▶ **Theorem 3** (almost-binary version of Theorem 1(a) in [8])**.** *Let N be a rooted almost-binary phylogenetic network and let $S \subseteq E(N)$. Then, the subgraph $N[S]$ of N induced by S is a support tree of N if and only if S is an admissible subset of $E(N)$. Moreover, there exists a one-to-one correspondence between the family of admissible subsets S of $E(N)$ and the family of support trees of N.*

▶ Remark 4. In the setting where the internal vertices of N (those except the root and leaves) are unlabeled and thus indistinguishable from one another, two different admissible subsets $S_1 \neq S_2$ of $E(N)$ can induce isomorphic support trees $N[S_1] = N[S_2]$. Due to this subtlety, multiple variations of the support tree counting problem have been explicitly formulated (see Fig. 1 and Section 7.2 in [9]). Therefore, a more precise interpretation of Theorem 3 may be that it establishes a one-to-one correspondence between the family of admissible subsets S of $E(N)$ and the family of support trees of N, under the assumption that all vertices (or edges) of N are distinguishable. In this paper, we adopt this assumption to obtain generalizations of Theorem 3.

Theorem 3 and Remark 4 allow us to identify each support tree T of N with its edge-set $E(T)$. In other words, counting or listing support trees of N refers to counting or listing admissible subsets of $E(N)$.

2.2 Structure theorem for rooted almost-binary phylogenetic networks

We recall the relevant materials from [9, 15]. Given a rooted almost-binary phylogenetic network N, a connected subgraph Z of N with $m := |E(Z)| \geq 1$ is a *zig-zag trail* (*in* N) if the edges of Z can be permuted as $(e_1, \ldots, e_m)$ such that either $\mathrm{head}(e_i) = \mathrm{head}(e_{i+1})$ or $\mathrm{tail}(e_i) = \mathrm{tail}(e_{i+1})$ holds for each $i \in [1, m-1]$. A zig-zag trail is represented by an alternating sequence of (not necessarily distinct) vertices and distinct edges, e.g., $(v_0, (v_0, v_1), v_1, (v_2, v_1), v_2, \ldots, (v_m, v_{m-1}), v_m)$, but this can be more concisely written as $v_0 > v_1 < v_2 > \cdots > v_{m-1} < v_m$ or its reverse, where each edge (v_i, v_{i+1}) is represented by $v_i > v_{i+1}$. A

zig-zag trail Z in N is *maximal* if N contains no zig-zag trail Z' such that Z is a proper subgraph of Z'. Each maximal zig-zag trail in N falls into one of the following four types. A *crown* is a maximal zig-zag trail Z that has even $m := |E(Z)| \geq 4$ and can be written in the cyclic form $v_0 < v_1 > v_2 < v_3 > \cdots > v_{m-2} < v_{m-1} > v_m = v_0$. A *fence* is any maximal zig-zag trail that is not a crown. An *N-fence* is a fence Z with odd $m := |E(Z)| \geq 1$, expressed as $v_0 > v_1 < v_2 > \cdots < v_{m-1} > v_m$. A fence Z with even $m := |E(Z)| \geq 2$ is an *M-fence* if expressed as $v_0 < v_1 > v_2 < \cdots < v_{m-1} > v_m$, and it is a *W-fence* if expressed as $v_0 > v_1 < v_2 > \cdots > v_{m-1} < v_m$.

As established in [9] (and will be restated in Theorem 8), any rooted almost-binary phylogenetic network N admits a unique decomposition $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ into its maximal zig-zag trails ($d \geq 1$), called *the zig-zag trail decomposition* of N (see Figure 1 for an illustration). To understand how this decomposition arises, observe that every edge e of N belongs to an obvious zig-zag trail $\mathrm{tail}(e) > \mathrm{head}(e)$ in N. If $\mathrm{tail}(e)$ has out-degree two in N, or $\mathrm{head}(e)$ has in-degree two in N, then this trail is not maximal. In such cases, by repeatedly extending the zig-zag trail, one can eventually obtain a maximal one. By construction, if each vertex of N has in-degree and out-degree at most two, then a maximal zig-zag trail containing e is unique for each e. In other words, if N is almost-binary, then no two distinct maximal zig-zag trails in N have a common edge. It follows that $\{E(Z_1), \ldots, E(Z_d)\}$ is a partition of $E(N)$, and thus $\mathcal{Z}$ is indeed a decomposition of N.

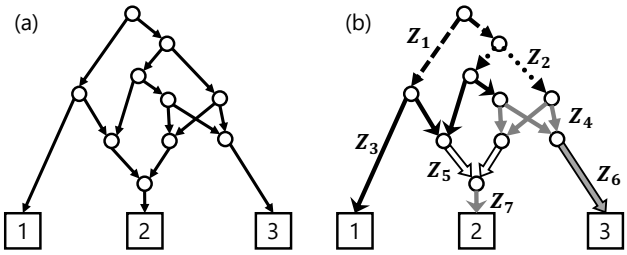

Figure 1 (a) A rooted binary phylogenetic network N. (b) The maximal zig-zag trail decomposition $\mathcal{Z} = \{Z_1, \ldots, Z_7\}$ of N (different arrow styles are used to distinguish the individual trails), where Z_1, Z_2 and Z_3 are M-fences, Z_4 is a crown, Z_5 is a W-fence, and Z_6 and Z_7 are N-fences.

▶ Remark 5. If Z is a maximal zig-zag trail in N, then, the following is equivalent to condition (C1) in Definition 2: If (u, v) is an edge of Z with $\mathrm{outdeg}_Z(u) = 1$ or $\mathrm{indeg}_Z(v) = 1$, then S contains (u, v). In fact, by the maximality of Z, we have $\mathrm{outdeg}_Z(u) = 1$ if and only if $\mathrm{outdeg}_N(u) = 1$ holds. Similarly, $\mathrm{indeg}_Z(v) = 1$ and $\mathrm{indeg}_N(v) = 1$ are equivalent.

▶ **Proposition 6** ([9]). *The maximal zig-zag trail decomposition $\mathcal{Z}$ of N can be computed in $\Theta(|E(N)|)$ time.*

As in the approach taken in [9, 10], we often consider a maximal zig-zag trail Z as a sequence $(e_1, \ldots, e_{|E(Z)|})$ of edges, ordered according to their appearance in the trail, where e_1 and $e_{|E(Z)|}$ are called the *terminal edges* of Z when Z is a fence. For any maximal zig-zag trail $Z = (e_1, \ldots, e_{|E(Z)|})$ in N, any subset S of $E(Z)$ is specified by a 0-1 sequence $\langle b_1\ b_2\ \ldots\ b_{|E(Z)|} \rangle$, where $b_i = 1$ if $e_i \in S$ and $b_i = 0$ otherwise. For example, given a crown $Z = (e_1, e_2, e_3, e_4)$, a subset $\{e_1, e_3\}$ of $E(Z)$ can be encoded as $\langle 1\ 0\ 1\ 0 \rangle$. We often write $\langle (10)^2 \rangle$ to avoid a repetition such as $\langle 1\ 0\ 1\ 0 \rangle$.

For the reader's benefit, we now illustrate how to find admissible subsets using small examples. Let $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ be the maximal zig-zag trail decomposition of N. Suppose $Z_1 = (e_1, e_2, e_3, e_4, e_5)$ is an N-fence with $\mathrm{head}(e_4) = \mathrm{head}(e_5)$. Then, $\{e_1, e_3, e_5\}$ is the only

admissible subset of $E(Z_1)$ because (C1) requires inclusion of both terminal edges e_1 and e_5, (C3) excludes selecting both e_4 and e_5, and (C2) then requires e_3, which in turn excludes e_2 due to (C3). Thus, the family $\mathcal{S}(Z_1)$ of admissible subsets of $E(Z_1)$ consists of a single element $\{e_1, e_3, e_5\}$, and so $\mathcal{S}(Z_1) = \{\langle 1\ 0\ 1\ 0\ 1\rangle\}$. Next, suppose $Z_2 = (e_1, e_2, e_3, e_4)$ is a crown. Then, $\{e_1, e_3\}$ and $\{e_2, e_4\}$ are the only two admissible subsets of $E(Z_2)$, because they are the only subsets that satisfy both (C2) and (C3), and crowns have no edge subject to (C1). Thus, $\mathcal{S}(Z_2) = \{\langle 1\ 0\ 1\ 0\rangle, \langle 0\ 1\ 0\ 1\rangle\}$.

▶ Remark 7. The choice of an admissible subset from $\mathcal{S}(Z_i)$ is independent of and does not influence the selection concerning any other $\mathcal{S}(Z_j)$. Indeed, Remark 5 implies that condition (C1) merely requires both terminal edges of each fence in $\mathcal{Z}$ be always selected (i.e. be marked by 1). In addition, (C2) and (C3) are relevant only to a fence or crown that has a pair of edges expressed as $v_{k-1} < v_k > v_{k+1}$ and $v_{k-1} > v_k < v_{k+1}$, respectively. Recalling that the maximal zig-zag trails are edge-disjoint, whenever such a pair of edges exists, it is contained in a unique maximal zig-zag trail in N. Thus, one can check whether $S \subset E(Z_i)$ is admissible or not even without looking at the entire network N.

Remark 7 provides a key idea behind the direct product decomposition in Theorem 8. After obtaining $\mathcal{Z} = \{Z_1, \dots Z_d\}$ for N, one can construct an admissible subset S of $E(N)$ simply by computing $S = S_1 \cup \cdots \cup S_d$, where S_i is any admissible subset of $E(Z_i)$. The family $\mathcal{T}$ of support trees of N – the family of admissible subsets of $E(N)$ – by listing all such combinations $(S_1, \dots, S_d)$ with each $S_i \in E(Z_i)$.

Using the definitions, notation, and key ideas outlined above, we now state the structure theorem for rooted almost-binary phylogenetic networks. It provides a way to canonically decompose such networks N and characterizes the family of admissible subsets S of $E(N)$, namely, the family of edge-sets of support trees of N.

▶ **Theorem 8** ([9]). *For any rooted almost-binary phylogenetic network N, the following hold:*

1. *The maximal zig-zag trail decomposition $\mathcal{Z} = \{Z_1, \dots, Z_d\}$ of N is unique to N.*
2. *Given $S \subseteq E(N)$, $N[S]$ is a support tree of N if and only if for each $Z_i \in \mathcal{Z}$, $S \cap E(Z_i)$ is an admissible subset of $E(Z_i)$. Here, the family $\mathcal{S}(Z_i)$ of admissible subsets of $E(Z_i)$ is given by Equation* (1).
3. *The family $\mathcal{T}$ of support trees of N is non-empty (i.e., N is tree-based) if and only if each $\mathcal{S}(Z_i)$ is non-empty (i.e., no element of $\mathcal{Z}$ is a W-fence). When $\mathcal{T}$ is non-empty, it is characterized by $\mathcal{T} = \prod_{i=1}^{d} \mathcal{S}(Z_i)$, where $(Z_1, \dots, Z_d)$ is an arbitrary ordering for $\mathcal{Z}$.*

$$\mathcal{S}(Z_i) = \begin{cases} \emptyset & \text{if } Z_i \text{ is a W-fence} \\ \{\langle 1(01)^{(|E(Z_i)|-1)/2}\rangle\} & \text{if } Z_i \text{ is an N-fence} \\ \{\langle (10)^{|E(Z_i)|/2}\rangle, \langle (01)^{|E(Z_i)|/2}\rangle\} & \text{if } Z_i \text{ is a crown} \\ \{\langle 1(01)^p(10)^q 1\rangle \mid p, q \in \mathbb{Z}_{\geq 0}, p+q = (|E(Z_i)|-2)/2\} & \text{if } Z_i \text{ is an M-fence} \end{cases} \tag{1}$$

Theorem 8 and Proposition 6 have furnished optimal algorithms for various computational problems on support trees [9, 10]. For example, the number $|\mathcal{T}|$ of support trees of N is given by $|\mathcal{T}| = \prod_{i=1}^{d} |\mathcal{S}(Z_i)|$, and using (2), it can be computed in $\Theta(|E(N)|)$ time.

$$|\mathcal{S}(Z_i)| = \begin{cases} 0 & \text{if } Z_i \text{ is an W-fence} \\ 1 & \text{if } Z_i \text{ is an N-fence} \\ 2 & \text{if } Z_i \text{ is a crown} \\ |E(Z_i)|/2 & \text{if } Z_i \text{ is an M-fence} \end{cases} \tag{2}$$

3 The family of support networks and its two subfamilies

For a rooted (not-necessarily-binary) phylogenetic network $N = (V, E)$, $r(N)$ denotes the *reticulation number* of N, i.e., the number of reticulations in V, and level(N) denotes the *level* of N, i.e., the maximum number of reticulations of N contained in a block of N, and the *tier* is $|E| - |V| + 1$. By definition, level$(N) \leq r(N)$ holds. In general, $r(N) \leq |E| - |V| + 1$ holds. If N is almost-binary, then $r(N) = |E| - |V| + 1$ holds by the hand-shaking lemma for directed graphs, which states that the sum of the in-degrees over all vertices equals the number of edges gives. Since N is assumed to be almost-binary in this paper, we will use the terms "tier" and "reticulation number" interchangeably.

Let N be a rooted almost-binary phylogenetic network on X and let G be a spanning subgraph of N. If G is also a rooted almost-binary phylogenetic network on X, then G is a *support network* of N. The *base network* $\tilde{G}$ of N is obtained from G by smoothing all vertices $v \in V(G)$ with $\text{indeg}_G(v) = \text{outdeg}_G(v) = 1$. Unlike support and base trees, support and base networks always exist since N itself is a support network of N. The *base tier* of N, denoted by $r^*(N)$, is the minimum value of $r(G) := |E(G)| - |V(G)| + 1$, which equals $r(\tilde{G})$, over all support networks G of N. Similarly, the *base level* of N, denoted by level$^*(N)$, is the minimum value of level(G), which equals level$(\tilde{G})$, over all support networks G of N. If k is the base tier (resp. base level) of N, then N is *tier-k-based* (resp. *level-k-based*).

For unrooted phylogenetic networks, Fischer and Francis [6] has introduced the concept of support networks to discuss tier-k-based and level-k-based networks. While the definitions are analogous, the mathematical properties of these concepts differ between rooted and unrooted networks. Indeed, for unrooted N, it was shown in [6] that $r^*(N) = \text{level}^*(N)$ holds if N has at most one "non-trivial blob" – a maximal connected subgraph without a cut edge and with at least two vertices. For rooted N, however, this equality does not hold in general (see Figure 2 for a counterexample).

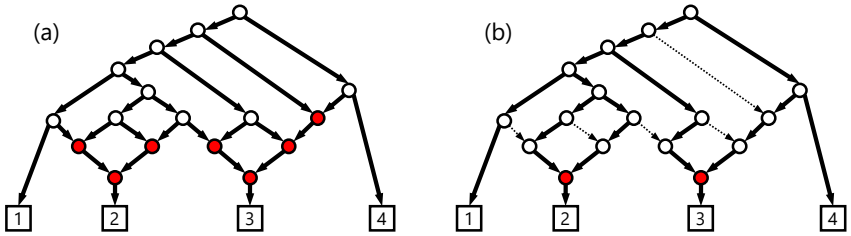

Figure 2 (a) A rooted binary phylogenetic network N with $r^*(N) = 2$ and level$^*(N) = 1$. (b) A support network G of N that attains the optimal values $r(G) = 2$ and level$(G) = 1$. The edges of G are shown by solid arrows. The reticulations in each graph are colored red.

The focus of this paper is on the computation of $r^*(N)$ and level$^*(N)$ for rooted N, with particular emphasis on the latter which presents greater computational challenges. We approach these problems using the families of minimal and minimum support networks of N, which are defined as follows:

▶ **Definition 9.** *For a rooted almost-binary phylogenetic network N,*

- *$\mathcal{A}_N$ is the set of all support networks of N;*
- *$\mathcal{B}_N$ is the set of minimal support networks of N (i.e. those with a minimal edge-set);*
- *$\mathcal{C}_N$ is the set of minimum support networks of N (i.e. those with the fewest edges).*

Definition 9 implies $(\emptyset \neq)\mathcal{A}_N \supseteq \mathcal{B}_N \supseteq \mathcal{C}_N$. As Figure 3 shows, these families can be all distinct. We also note that if N is tree-based, then $\mathcal{C}_N$ is nothing but the set $\mathcal{T}$ of support trees of N characterized in Theorems 3 and 8.

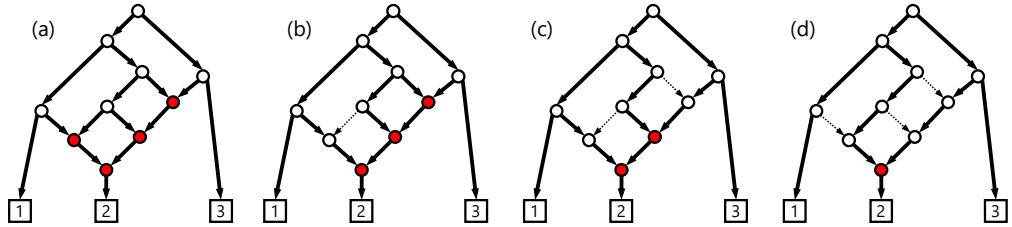

Figure 3 (a): A rooted binary phylogenetic network N. (b): A support network in $\mathcal{A}_N \setminus \mathcal{B}_N$. (c): A support network in $\mathcal{B}_N \setminus \mathcal{C}_N$. (d): A support network in $\mathcal{C}_N$. The reticulations in each graph are colored red.

4 Counting the support networks of three types

Examining the proper subfamilies $\mathcal{B}_N$ and $\mathcal{C}_N$ rather than the entire family $\mathcal{A}_N$ of support networks could reduce the search space for computing $r^*(N)$ or level$^*(N)$. A natural question is how substantial this reduction might be. In this section, we generalize Theorem 8 to derive analogous direct-product characterizations of $\mathcal{A}_N$, $\mathcal{B}_N$, and $\mathcal{C}_N$, and determine their cardinalities. To achieve this, we relax condition (C3) in Definition 2 to introduce different notions of admissibility as follows.

▶ **Definition 10.** *Let N be a rooted almost-binary phylogenetic network and let Z be any subgraph of N. A subset S of $E(Z)$ is* $\mathcal{A}$-admissible *if it satisfies the following conditions:*

(C1*) *If (u, v) is an edge of Z with* $\mathrm{outdeg}_N(u) = 1$ *or* $\mathrm{indeg}_N(v) = 1$*, then S contains (u, v);*

(C2*) *If e_1 and e_2 are distinct edges of Z with* $\mathrm{tail}(e_1) = \mathrm{tail}(e_2)$ *or* $\mathrm{head}(e_1) = \mathrm{head}(e_2)$*, then S contains at least one of $\{e_1, e_2\}$.*

In particular, given an $\mathcal{A}$-admissible subset S of $E(Z)$, S is $\mathcal{B}$-admissible *if it is minimal, i.e. no proper subset of S is an $\mathcal{A}$-admissible subset of $E(Z)$, and S is* $\mathcal{C}$-admissible *if it is smallest among all $\mathcal{A}$-admissible subsets of $E(Z)$.*

The definition of $\mathcal{A}$-admissibility differs from the previous admissibility (Definition 2) in a single aspect: it relaxes the original condition (C3). In fact, (C1*) is identical to the original condition (C1). Similarly to Remark 5, when Z is a maximal zig-zag trail in N, we may replace $\mathrm{outdeg}_N(v)$ and $\mathrm{indeg}_N(v)$ in condition (C1*) with $\mathrm{outdeg}_Z(v)$ and $\mathrm{indeg}_Z(v)$, respectively. The new condition (C2*) is an amalgamation of the original condition (C2) and a relaxed version of (C3). In other words, (C2*) retains the original requirement for edges sharing the same tail, while it replaces the "exactly one" requirement for edges with the same head by a weaker "at least one". This modification of (C3) allows an $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$-admissible subsets of $E(N)$ to contain both incoming edges at a reticulation vertex and to induce a non-tree subgraph of N with a reticulation.

Lemma 11 generalizes Theorem 3 and the second statement of Theorem 8. The proof is given in Appendix A.1.

▶ **Lemma 11.** *Let N be a rooted almost-binary phylogenetic network, and let $\mathcal{X} \in \{\mathcal{A}, \mathcal{B}, \mathcal{C}\}$. Then, there is a one-to-one correspondence between the family $\mathcal{X}_N$ of support networks and the family $\mathcal{S}_\mathcal{X}$ of $\mathcal{X}$-admissible subsets of $E(N)$. Moreover, the subgraph $N[S]$ of N induced by $S \subseteq E(N)$ is a support network in $\mathcal{X}_N$ if and only if $S \cap E(Z_i)$ is an $\mathcal{X}$-admissible subset of $E(Z_i)$ for each maximal zig-zag trail Z_i in N.*

4.1 The number $|\mathcal{A}_N|$ of all support networks

By Lemma 11, one can find an $\mathcal{A}$-admissible subset of $E(N)$ in essentially the same manner as for an admissible subset of $E(N)$: first, compute the zig-zag trail decomposition $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ of N, and then select edges of Z_i according to the conditions (C1*) and (C2*) in Definition 10. As explained in Remark 7, when $Z_i = (e_1, \ldots, e_{|E(Z_i)|})$ is a fence, (C1*) requires that the terminal edges e_1 and $e_{|E(Z_i)|}$ be marked 1. In contrast, new condition (C2*) prohibits unselected consecutive edges of Z_i – consecutive 0's in the 0-1 sequence – regardless of whether Z_i is a crown or a fence.

Thus, the family $\mathcal{S}_\mathcal{A}(Z_i)$ of $\mathcal{A}$-admissible subsets of $E(Z_i)$ is expressed by (3) for each $Z_i \in \mathcal{Z}$. We note that, if Z_i is a crown, different choices of e_1 result in different sequences representing the same subset of $E(Z_i)$. For example, both $\langle 1\ 0\ 0\ 1\rangle$ and $\langle 0\ 1\ 1\ 0\rangle$ represent the same subset that does not satisfy condition (C2*).

$$\mathcal{S}_\mathcal{A}(Z_i) = \begin{cases} \{\langle b_1 \cdots b_{|E(Z_i)|}\rangle \mid b_1 = b_{|E(Z_i)|} = 1, \text{ and no } \langle 00\rangle \text{ occurs}\} \\ \qquad\qquad \text{if } Z_i \text{ is a fence} \\ \{\langle b_1 \cdots b_{|E(Z_i)|}\rangle \mid \text{no } \langle 00\rangle \text{ occurs in any circular ordering}\} \\ \qquad\qquad \text{if } Z_i \text{ is a crown} \end{cases} \tag{3}$$

Moreover, by Lemma 11, $\mathcal{A}_N$ is characterized by $\mathcal{A}_N = \prod_{i=1}^{d} \mathcal{S}_\mathcal{A}(Z_i)$, similarly to $\mathcal{T}$ in Theorem 8. Therefore, $|\mathcal{A}_N| = \prod_{i=1}^{d} |\mathcal{S}_\mathcal{A}(Z_i)|$. As in (3), each number $|\mathcal{S}_\mathcal{A}(Z_i)|$ depends on whether Z_i is a fence or not. This number can be more explicitly represented using Fibonacci and Lucas numbers (OEIS A000045 and OEIS A000032, respectively) as follows. The proof of Theorem 12 is given in Appendix A.2.

▶ **Theorem 12.** *Let N be a rooted almost-binary phylogenetic network and $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ be the maximal zig-zag trail decomposition of N. Let $\{F_n\}$ be the Fibonacci sequence defined by $F_1 = F_2 = 1$ and $F_n = F_{n-1} + F_{n-2}$ $(n \geq 3)$, and $\{L_n\}$ be the Lucas sequence defined by $L_1 = 1, L_2 = 3$ and $L_n = L_{n-1} + L_{n-2}$ $(n \geq 3)$. Then, the number of support networks $|\mathcal{A}_N|$ is given by* (4).

$$|\mathcal{A}_N| = \prod_{Z_i : fence} F_{|E(Z_i)|} \cdot \prod_{Z_i : crown} L_{|E(Z_i)|} \tag{4}$$

Moreover, $|\mathcal{A}_N| = \Theta(\phi^{|E(N)|})$ holds, where $\phi = (1 + \sqrt{5})/2 = 1.6180\ldots$ is the golden ratio.

Theorem 12 yields an obvious algorithm for counting $|\mathcal{A}_N|$. It first computes the maximal zig-zag trail decomposition $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ of N, and then calculates $|\mathcal{A}_N|$ using (4).

▶ **Proposition 13.** *$|\mathcal{A}_N|$ can be computed by in $\Theta(|E(N)|)$ time.*

Proof. By Proposition 6, $\mathcal{Z}$ can be computed in $\Theta(|E(N)|)$ time. For each $Z_i \in \mathcal{Z}$, deciding whether Z_i is a crown or not takes $O(|E(Z_i)|)$ time, and $F_{|E(Z_i)|}$ and $L_{|E(Z_i)|}$ can be computed in $O(|E(Z_i)|)$ time. Since $|\mathcal{Z}| = O(|E(N)|)$, one can count $|\mathcal{A}_N|$ using (4) in $O(|E(N)|)$ time. Loading N takes $\Omega(|E(N)|)$ time, so the overall complexity is $\Theta(|E(N)|)$. ◀

Similarly, we can develop an algorithm to generate all (or a desired number of) elements of $\mathcal{A}_N$ by sequentially outputting each element of $\prod_{i=1}^{d} \mathcal{S}_\mathcal{A}(Z_i)$, achieving $\Theta(|E(N)|)$ delay. We refer the reader to [9] or [10] for the basics on the complexity analysis of listing algorithms.

4.2 The number $|\mathcal{B}_N|$ of minimal support networks

We can see that $\mathcal{A}$-admissible subset S of $E(Z_i)$ is $\mathcal{B}$-admissible if and only if S contains no three consecutive edges in the edge sequence $(e_1, \ldots, e_{|E(Z_i)|})$ of Z_i. Thus, the family $\mathcal{S}_{\mathcal{B}}(Z_i)$ of $\mathcal{B}$-admissible subsets of $E(Z_i)$ is expressed as in (5) for each $Z_i \in \mathcal{Z}$.

$$\mathcal{S}_{\mathcal{B}}(Z_i) = \begin{cases} \{\langle b_1 \cdots b_{|E(Z_i)|} \rangle \mid b_1 = b_{|E(Z_i)|} = 1, \text{ and no } \langle 00 \rangle \text{ or } \langle 111 \rangle \text{ occurs}\} \\ \qquad\qquad \text{if } Z_i \text{ is a fence} \\ \{\langle b_1 \cdots b_{|E(Z_i)|} \rangle \mid \text{no } \langle 00 \rangle \text{ or } \langle 111 \rangle \text{ occurs in any circular ordering}\} \\ \qquad\qquad \text{if } Z_i \text{ is a crown} \end{cases} \tag{5}$$

By Lemma 11, $\mathcal{B}_N$ is characterized by $\mathcal{B}_N = \prod_{i=1}^{d} \mathcal{S}_{\mathcal{B}}(Z_i)$. Therefore, $|\mathcal{B}_N| = \prod_{i=1}^{d} |\mathcal{S}_{\mathcal{B}}(Z_i)|$. This number is expressed using Padovan numbers (OEIS A000931) and Perrin numbers (OEIS A001608) as follows. The proof of Theorem 14 is in Appendix A.3.

▶ **Theorem 14.** *Let N be a rooted almost-binary phylogenetic network and $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ be the maximal zig-zag trail decomposition of N. Let $\{P_n\}$ be the Padovan sequence defined by $(P_1, P_2, P_3) = (1, 1, 1)$ and $P_n = P_{n-2} + P_{n-3}$ $(n \geq 4)$, and $\{Q_n\}$ be the Perrin sequence defined by $(Q_1, Q_2, Q_3) = (0, 2, 3)$ and $Q_n = Q_{n-2} + Q_{n-3}$ $(n \geq 4)$. Then, the number of minimal support networks $|\mathcal{B}_N|$ is given by* (6).

$$|\mathcal{B}_N| = \prod_{Z_i : fence} P_{|E(Z_i)|} \cdot \prod_{Z_i : crown} Q_{|E(Z_i)|} \tag{6}$$

Moreover, $|\mathcal{B}_N| = \Theta(\psi^{|E(N)|})$ holds, where $\psi = \sqrt[3]{(9 + \sqrt{69})/18} + \sqrt[3]{(9 - \sqrt{69})/18} = 1.3247\ldots$ is the plastic number.

▶ **Proposition 15.** *$|\mathcal{B}_N|$ can be computed in $\Theta(|E(N)|)$ time.*

Proof. The only difference between (4) and (6) is that $F_{|E(Z_i)|}$ and $L_{|E(Z_i)|}$ are replaced by $P_{|E(Z_i)|}$ and $Q_{|E(Z_i)|}$, respectively. For each $Z_i \in \mathcal{Z}$, both $P_{|E(Z_i)|}$ and $Q_{|E(Z_i)|}$ can also be computed in $O(|E(Z_i)|)$ time. The remainder is the same as the proof of Proposition 13. ◀

As noted in Section 4.1, the elements of $\mathcal{B}_N$ can also be listed with $\Theta(|E(N)|)$ delay.

4.3 The number $|\mathcal{C}_N|$ of minimum support networks

The construction of $\mathcal{C}$-admissible subsets of each $E(Z_i)$ is the same as that of admissible subsets for support trees, with the only difference that $\mathcal{C}$-admissible subsets allow W-fences.

$$\mathcal{S}_{\mathcal{C}}(Z_i) = \begin{cases} \{\langle (10)^{|E(Z_i)|/2} \rangle, \langle (01)^{|E(Z_i)|/2} \rangle\} & \text{if } Z_i \text{ is a crown} \\ \{\langle 1(01)^{(|E(Z_i)|-1)/2} \rangle\} & \text{if } Z_i \text{ is an N-fence} \\ \{\langle 1(01)^p(10)^q 1 \rangle \mid p, q \in \mathbb{Z}_{\geq 0}, p + q = (|E(Z_i)| - 2)/2\} & \\ \qquad\qquad \text{if } Z_i \text{ is an M-fence or a W-fence} \end{cases} \tag{7}$$

By Lemma 11, $\mathcal{C}_N$ is characterized by $\mathcal{C}_N = \prod_{i=1}^{d} \mathcal{S}_{\mathcal{C}}(Z_i)$. Therefore, $|\mathcal{C}_N| = \prod_{i=1}^{d} |\mathcal{S}_{\mathcal{C}}(Z_i)|$. From (7), we have $|\mathcal{S}_{\mathcal{C}}(Z_i)| = 2$ for any crown Z_i, $|\mathcal{S}_{\mathcal{C}}(Z_i)| = 1$ for any N-fence Z_i, and $|\mathcal{S}_{\mathcal{C}}(Z_i)| = |E(Z_i)|/2$ for any other Z_i. Thus, we obtain Theorem 16 and Proposition 17.

▶ **Theorem 16.** *Let N be a rooted almost-binary phylogenetic network and $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ be the maximal zig-zag trail decomposition of N. Then, $|\mathcal{C}_N| = 2^c \cdot \prod_{Z_i \in \mathcal{Z}_{MW}} (|E(Z_i)|/2)$ holds, where c is the number of crowns in $\mathcal{Z}$, and $\mathcal{Z}_{MW} := \{Z_i \in \mathcal{Z} \mid Z_i$ is an M-fence or W-fence$\}$.*

▶ **Proposition 17.** *$|\mathcal{C}_N|$ can be computed in $\Theta(|E(N)|)$ time.*

As noted in Section 4.1, the elements of $\mathcal{C}_N$ can also be listed with $\Theta(|E(N)|)$ delay.

4.4 Case study: Comparison of the numbers $|\mathcal{A}_N|$, $|\mathcal{B}_N|$, and $|\mathcal{C}_N|$

One can solve any optimization problem on support networks by listing all elements of $\mathcal{A}_N$, but this approach is generally infeasible due to its enormous size. Although all of $|\mathcal{A}_N|$, $|\mathcal{B}_N|$, and $|\mathcal{C}_N|$ grow exponentially with the size of N, we present a comparison of their practical sizes to motivate our approach to Problem 2 described in Section 6. The full implementation including the code used to generate the networks, along with the generated samples and detailed count data, is available in our GitHub repository.

For this case study, we generated rooted binary phylogenetic networks with n leaves and $r = 2(n-1)$ reticulations. Such a network has $2n - 2 + 3r = 8(n-1)$ edges, and thus 50% of its edges are reticulation edges, representing a high level of reticulation. For each $3 \leq n \leq 10$, we created 100 such networks by the following procedure: starting from a rooted binary phylogenetic tree N_0, the procedure repeatedly adds a new edge between two arbitrary edges of N_i, directing it so as to preserve the acyclicity of N_{i+1}, and returns N_r. Note that networks generated in this manner may or may not be tree-based, since every new edge is added not between edges of the initial tree N_0 but between edges of the current network N_i.

For each generated network N, we computed $|\mathcal{A}_N|$, $|\mathcal{B}_N|$, and $|\mathcal{C}_N|$ using the linear-time algorithms described in Sections 4.1–4.3. The minimum, maximum, and median counts across the 100 samples of each size are summarized in Table 1.

Table 1 shows that $|\mathcal{B}_N|$ is consistently much smaller than $|\mathcal{A}_N|$, and $|\mathcal{C}_N|$ is even smaller. This indicates that, even though $|\mathcal{A}_N|$ can be prohibitively large, both $|\mathcal{B}_N|$ and $|\mathcal{C}_N|$ are typically small enough to allow exhaustive enumeration. We also see that the number of support networks varies widely even among networks of the same size, suggesting that the computational cost of exhaustive search within $\mathcal{B}_N$ or $\mathcal{C}_N$ may vary substantially depending on the structure of N. For example, the number $|\mathcal{C}_N|$ of minimum support networks of N with $r = 18$ was 1728 for a certain sample but was only 4 for another sample.

Table 1 Summary of the case study results: minimum, maximum, and median of $|\mathcal{A}_N|$, $|\mathcal{B}_N|$, and $|\mathcal{C}_N|$ for 100 rooted binary phylogenetic networks with n leaves and $r = 2(n-1)$ reticulations.

n	r	$\lvert\mathcal{A}_N\rvert$			$\lvert\mathcal{B}_N\rvert$			$\lvert\mathcal{C}_N\rvert$		
		Min	Max	Median	Min	Max	Median	Min	Max	Median
3	4	15	64	30	2	10	4	1	9	3
4	6	55	336	160	4	28	9	1	18	4
5	8	288	2,640	900	8	72	24	1	36	8
6	10	900	15,840	4,478	6	256	48	1	108	12
7	12	6,435	134,784	24,720	24	768	142	2	192	24
8	14	34,272	571,914	134,400	40	1,680	270	2	576	38
9	16	93,600	4,186,080	699,920	90	2,880	576	2	1,152	68
10	18	449,280	21,715,200	3,861,936	144	7,056	1,372	4	1,728	124

5 Finding a support network with the fewest reticulations

As defined in Section 3, the base tier $r^*(N)$ of N is the minimum value of $r(G)$ over all support networks G of N. This leads us to Problem 1.

▶ **Problem 1** (Reticulation Minimization)**.** Given a rooted almost-binary phylogenetic network N, compute the base tier $r^*(N)$, and find a support network G of N with $r(G) = r^*(N)$.

Since $r(G) := |E(G)| - |V(G)| + 1$ and G is a spanning subgraph of N, the support networks G of N minimize $r(G)$ if and only if it has the fewest edges among all support networks, i.e., it is a minimum support network. Therefore, an optimal solution of Problem 1 can be obtained by simply selecting an arbitrary element from $\mathcal{C}_N = \prod_{i=1}^{d} \mathcal{S}_{\mathcal{C}}(Z_i)$. This is done by computing the maximal zig-zag trail decomposition $\{Z_1, \ldots, Z_d\}$ of N, and then selecting an arbitrary element of $\mathcal{S}_{\mathcal{C}}(Z_i)$ expressed in (7) for each Z_i.

For example, the network N in Figure 1 is decomposed into the maximal zig-zag trails Z_1 through Z_7. According to (7), the sequence $\langle 11 \rangle$ is $\mathcal{C}$-admissible for $E(Z_1)$, $E(Z_2)$, and $E(Z_5)$; $\langle 1010 \rangle$ is $\mathcal{C}$-admissible for $E(Z_3)$ and $E(Z_4)$; and $\langle 1 \rangle$ is $\mathcal{C}$-admissible for $E(Z_6)$ and $E(Z_7)$. Piecing these $\mathcal{C}$-admissible subsets together yields the edge-set of a support network G^* of N with the minimum reticulation number, that is, $r(G^*) = 1$.

▶ **Theorem 18.** *The above algorithm solves Problem 1 in $\Theta(|E(N)|)$ time.*

When N is tree-based, $r^*(N) = 0$ holds and a support network G with $r(G^*) = 0$ is simply its support tree. In general, $r(G^*)$ equals the number of W-fences in the maximal trail decomposition of N, as easily shown by induction on the number of W-fences.

6 Finding a support network with the minimum level

In Section 3, we have also defined level$^*(N)$, i.e., the base level of N. In contrast to Problem 1, we conjecture that Problem 2 is NP-hard. Here, we provide exact and heuristic exponential algorithms for solving Problem 2.

▶ **Problem 2** (Level Minimization)**.** Given a rooted almost-binary phylogenetic network N, compute the base level, level$^*(N)$, and find a support network G with level$(G) =$ level$^*(N)$.

6.1 Exact algorithm

An obvious exact algorithm for Problem 2 computes level(G) of each element G in $\mathcal{A}_N$ to obtain the minimum value, but this method is clearly impractical. Algorithm 1 performs an exhaustive search in $\mathcal{B}_N$ instead of $\mathcal{A}_N$. Recalling the numerical results in Table 1, we know that this search space reduction is significant. We can prove that $\mathcal{B}_N$ contains a correct G^*, yielding Theorem 19. The proof is given in Appendix A.4.

Algorithm 1 Exact method for solving Level Minimization (Problem 2).

Input: A rooted almost-binary phylogenetic network N
Output: A support network G^* of N with the minimum level, the value of level$^*(N)$
foreach $G \in \mathcal{B}_N$ **do**
  compute level(G)
let G^* be a support network G with the minimum level(G) in $\mathcal{B}_N$;
return G^*, and level(G^*) as level$^*(N)$

▶ **Theorem 19.** *Algorithm 1 returns a correct solution of Problem 2. It runs in $O(|\mathcal{B}_N| \cdot |E(N)|) = O(\psi^{|E(N)|} \cdot |E(N)|)$ time, where ψ is as in Theorem 14.*

6.2 Heuristic algorithm

Algorithm 2 performs an exhaustive search in an even smaller search space $\mathcal{C}_N$. It repeats the same procedure as Algorithm 1, except its iteration number is reduced from $|\mathcal{B}_N|$ to $|\mathcal{C}_N|$. By Theorem 19, Algorithm 2 therefore runs in $O(|\mathcal{C}_N| \cdot |E(N)|)$ time.

Algorithm 2 does not always output a correct solution of Problem 2, because minimizing the number of reticulations does not imply minimizing the level of support networks. Consider a level-1-based network N in Figure 4(a), which has the property that any $G \in \mathcal{C}_N$ satisfies $r(G) = 2$ and $\mathrm{level}(G) = 2$. Then, Algorithm 2 outputs a level-2 support network as illustrated in Figure 4(b). An optimal support network G^* with $\mathrm{level}(G^*) = 1$ exists in $\mathcal{B}_N \setminus \mathcal{C}_N$ while it is not optimal in terms of reticulation numbers, as shown in Figure 4(c).

Algorithm 2 Heuristic method for solving Level Minimization (Problem 2).

Input: A rooted almost-binary phylogenetic network N
Output: A support network G^* of N with the minimum level, the value of $\mathrm{level}^*(N)$
foreach $G \in \mathcal{C}_N$ **do**
  compute $\mathrm{level}(G)$
let G^* be a support network G with the minimum $\mathrm{level}(G)$ in $\mathcal{C}_N$;
return G^*, and $\mathrm{level}(G^*)$ as $\mathrm{level}^*(N)$

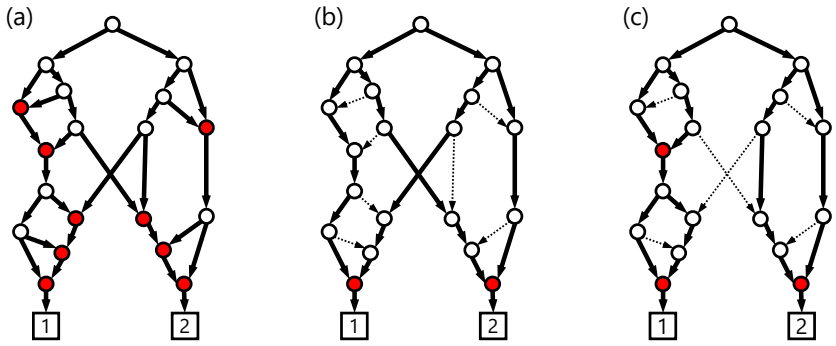

Figure 4 (a) An instance of Problem 2 for which Algorithm 2 cannot find a correct solution. (b) An example of a support network output by Algorithm 2 with two reticulations (red) and level 2. (c) An optimal support network that has three reticulations (red) yet achieves level 1.

6.3 Performance evaluation of the exact and heuristic algorithms

We evaluated the performance of our exact algorithm (Algorithm 1) and heuristic algorithm (Algorithm 2) using rooted binary phylogenetic networks with $n = 8$ leaves and various reticulation numbers r. Although our experiment focuses on only binary networks, this simplification does not undermine the validity of our evaluation, as the search space sizes $|\mathcal{B}_N|$ for Algorithm 1 and $|\mathcal{C}_N|$ for Algorithm 2 are not affected by whether N is binary or almost-binary.

For each r ranging from 1 to 50, we generated 25 instances using the procedure described in Section 4.4. The experiments were conducted on a MacBook Pro equipped with an Apple M3 Max CPU (4.05 GHz) and 36 GB of memory, with a timeout limit of 30 minutes per instance. The Python code and experimental results, including the outputs of both algorithms for all instances, are available in our GitHub repository.

We first assessed computational efficiency of Algorithms 1 and 2 by measuring runtime on each instance. Figure 5 shows average runtime on a log scale with minimum and maximum values across 25 instances for each r. Both algorithms exhibit expected exponential runtime growth as r increases. Algorithm 1 completed up to $r = 36$, beyond which timeouts occurred, while Algorithm 2 successfully handled all instances up to $r = 49$. The performance gap is particularly pronounced for larger networks. For example, at $r = 35$, Algorithm 1 approaches its limits while Algorithm 2 completes within seconds on average.

Notably, both algorithms exhibit substantial runtime variability even for the same r, as shown by the wide error bars in Figure 5. This variation reflects runtime dependence on network structure beyond just reticulation number, consistent with our observations in Section 4.4 regarding the significant variability of $|B_N|$ and $|C_N|$ among instances with identical r values.

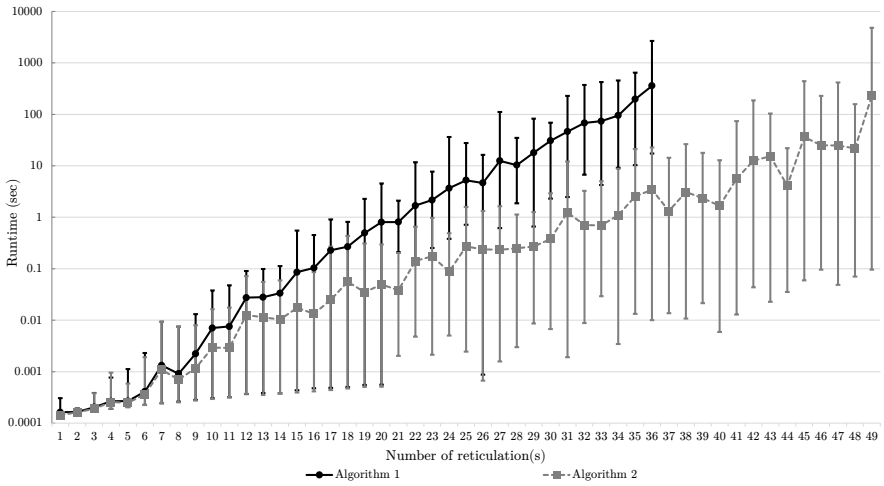

Figure 5 Runtimes (log scale) of Algorithms 1 and 2 on rooted binary phylogenetic networks with 8 leaves and various reticulation numbers r ($1 \leq r \leq 36$ for exact, $1 \leq r \leq 49$ for heuristic). Runtime is averaged over 25 instances per r, with error bars showing minimum and maximum.

While Algorithm 1 guarantees optimal solutions (Theorem 19), Algorithm 2 provides no such theoretical guarantee. To evaluate the practical quality of the heuristic solutions, we compared the outputs of Algorithm 2 against the optimal values level$^*(N)$ computed by Algorithm 1 for all instances with $r \leq 36$ where both algorithms completed successfully within the time limit. Figure 6 provides a breakdown of the output differences Δ for each r, where Δ is defined as the difference between the value returned by Algorithm 2 and the optimal value level$^*(N)$ computed by Algorithm 1. The bars show the proportion of instances (out of 25) with $\Delta = 0$ (exact match), $\Delta = 1$, $\Delta = 2$, and $\Delta \geq 3$.

Although the accuracy gradually declines as r increases, Algorithm 2 maintains practical effectiveness across the entire range of tested networks. In fact, up to $r \leq 17$, Algorithm 2 achieved exact solutions ($\Delta = 0$) for all instances. Even at $r = 30$, it still returned optimal values for approximately 80% of instances. Large deviations ($\Delta \geq 3$) remain rare throughout the tested range, occurring in less than 10% of instances even for the largest networks.

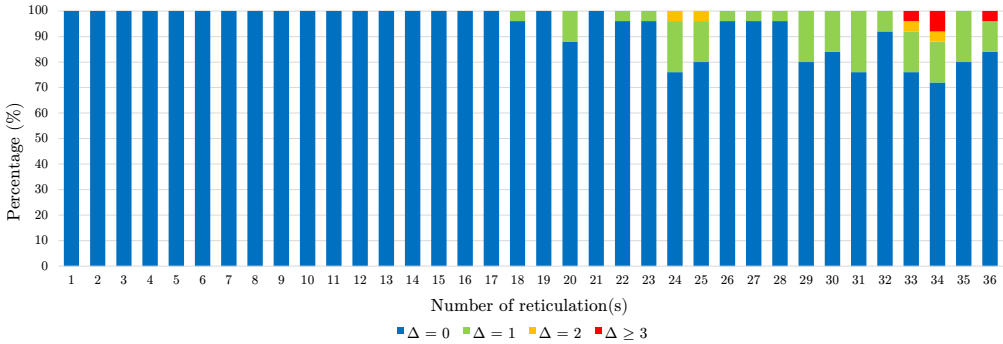

Figure 6 Accuracy of Algorithm 2 on randomly generated rooted binary phylogenetic networks with 8 leaves and $r \in \{1, \ldots, 36\}$ (25 instances per r). Bars show the percentage of instances with difference Δ between Algorithm 2's output and optimal level$^*(N)$ found by Algorithm 1: $\Delta = 0$ (blue), $\Delta = 1$ (green), $\Delta = 2$ (orange), $\Delta \geq 3$ (red).

7 Conclusion and future directions

In this paper, we have extended Hayamizu's structure theorem to develop a theoretical foundation for support networks of rooted almost-binary phylogenetic networks. The extension from support trees allows us to analyze not only tree-based networks but also general, non-tree-based rooted phylogenetic networks. Our main contributions are threefold:

First, we established direct-product characterizations of three families of support networks for a given network N: all support networks $\mathcal{A}_N$, minimal support networks $\mathcal{B}_N$, and minimum support networks $\mathcal{C}_N$. These characterizations yielded closed-form counting formulas that can be computed in linear time, revealing interesting connections to well-known integer sequences such as Fibonacci, Lucas, Padovan, and Perrin numbers. We also described and implemented a linear-delay algorithm for listing the support networks in each family.

Second, we developed a linear-time algorithm for Reticulation Minimization, which finds a support network with the minimum reticulation number (minimum tier). We proved that an optimal solution can be found simply by picking any element from the set $\mathcal{C}_N$.

Third, we proposed both exact and heuristic algorithms for Level Minimization. While both algorithms have exponential time complexity, our experimental evaluations demonstrated their practicality across a wide range of reticulation levels. Notably, the heuristic method found optimal solutions in most cases, and when it did not, it still produced good approximations.

There are some promising avenues for future research. In this paper, we have discussed support networks of almost-binary networks, which form a subclass of non-binary networks. However, extending this framework to fully non-binary networks is challenging because the uniqueness of the maximal zig-zag trail decomposition of N, a key component of Hayamizu's structure theorem that underpins our work, is not guaranteed beyond the almost-binary case. It is also important to explore whether support networks with the minimum tier or level can gain meaningful insights into reticulate evolutionary processes from biological data. As our heuristic algorithm empirically produces good approximate solutions, theoretical analysis of its approximation ratio and the design of improved approximation algorithms also remain valuable pursuits.

References

1 Gabriel Cardona, Francesc Rosselló, and Gabriel Valiente. Comparison of Tree-Child Phylogenetic Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 6(4):552–569, 2009. doi:10.1109/TCBB.2007.70270.

2 Nathan Davidov, Amanda Hernandez, Justin Jian, Patrick McKenna, K.A. Medlin, Roadra Mojumder, Megan Owen, Andrew Quijano, Amanda Rodriguez, Katherine St. John, Katherine Thai, and Meliza Uraga. Maximum Covering Subtrees for Phylogenetic Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 18(6):2823–2827, November 2021. doi:10.1109/TCBB.2020.3040910.

3 Tomislav Došlić and Ivana Zubac. Counting maximal matchings in linear polymers. *Ars Mathematica Contemporanea*, 11:255–276, January 2016. doi:10.26493/1855-3974.851.167.

4 Péter L Erdős, Charles Semple, and Mike Steel. A class of phylogenetic networks reconstructable from ancestral profiles. *Mathematical Biosciences*, 313:33–40, 2019. doi:10.1016/j.mbs.2019.04.009.

5 E. J. Farrell. On the Occurrences of Fibonacci Sequences in the Counting of Matchings in Linear Polygonal Chains. *The Fibonacci Quarterly*, 24(3):238–246, 1986.

6 Mareike Fischer and Andrew Francis. How tree-based is my network? Proximity measures for unrooted phylogenetic networks. *Discrete Applied Mathematics*, 283:98–114, September 2020. doi:10.1016/j.dam.2019.12.019.

7 Andrew Francis, Charles Semple, and Mike Steel. New characterisations of tree-based networks and proximity measures. *Advances in Applied Mathematics*, 93:93–107, February 2018. doi:10.1016/j.aam.2017.08.003.

8 Andrew Francis and Mike Steel. Which Phylogenetic Networks are Merely Trees with Additional Arcs? *Systematic Biology*, 64(5):768–777, September 2015. doi:10.1093/sysbio/syv037.

9 Momoko Hayamizu. A Structure Theorem for Rooted Binary Phylogenetic Networks and Its Implications for Tree-Based Networks. *SIAM Journal on Discrete Mathematics*, 35(4):2490–2516, January 2021. doi:10.1137/19M1297403.

10 Momoko Hayamizu and Kazuhisa Makino. Ranking Top-k Trees in Tree-Based Phylogenetic Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 20(3):2349–2355, May 2023. doi:10.1109/TCBB.2022.3229827.

11 John Hopcroft and Robert Tarjan. Algorithm 447: Efficient Algorithms for Graph Manipulation. *Communications of the ACM*, 16(6):372–378, June 1973. doi:10.1145/362248.362272.

12 Remie Janssen and Yukihiro Murakami. On cherry-picking and network containment. *Theoretical Computer Science*, 856:121–150, 2021. doi:10.1016/j.tcs.2020.12.031.

13 Thomas Koshy. *Fibonacci and Lucas Numbers with Applications*. John Wiley & Sons, September 2001. doi:10.1002/9781118033067.

14 Charles Semple and Jack Simpson. When is a Phylogenetic Network Simply an Amalgamation of Two Trees? *Bulletin of Mathematical Biology*, 80(9):2338–2348, 2018. doi:10.1007/s11538-018-0463-x.

15 Takatora Suzuki, Han Guo, and Momoko Hayamizu. Bridging Between Deviation Indices for Non-Tree-Based Phylogenetic Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 21(6):2226–2234, 2024. doi:10.1109/TCBB.2024.3456575.

16 Louxin Zhang. On Tree-Based Phylogenetic Networks. *Journal of Computational Biology*, 23(7):553–565, July 2016. doi:10.1089/cmb.2015.0228.

A Appendix

In what follows, $N = (V, E)$ is a rooted almost-binary phylogenetic network on the leaf-set X, $\mathcal{Z} = \{Z_1, \ldots, Z_d\}$ is the maximal zig-zag trail decomposition of N, and $\vec{\mathcal{Z}} = (Z_1, \ldots, Z_d)$ is any ordered set of the elements of $\mathcal{Z}$. In addition, for each $Z_i \in \mathcal{Z}$, we let $m_i := |E(Z_i)|$ and write $Z_i = (e_1, \ldots, e_{m_i})$.

A.1 Proof of Lemma 11

We begin by proving the first statement. To show the one-to-one correspondence between $\mathcal{X}_N$ and $\mathcal{S}_\mathcal{X}$ in the case of $\mathcal{X} = \mathcal{A}$, we first prove that $N[S]$ is a support network of N if and only if S is an $\mathcal{A}$-admissible subset of E. Assume that G is a support network of N. Then, $V(G) = V$ by definition. We will now verify that $\tilde{S} := E(G)$ satisfies conditions (C1*) and (C2*). Recall that Z in Definition 10 is any subgraph of N, so we may let Z be N. For any $(u, v) \in E$ with $\mathrm{outdeg}_N(u) = 1$, we have $(u, v) \in \tilde{S}$ since otherwise $\mathrm{outdeg}_G(u) = 0$ would hold and thus G would have a leaf u not in the leaf-set X of N. Also, for any $(u, v) \in E$ with $\mathrm{indeg}_N(v) = 1$, we have $(u, v) \in \tilde{S}$ since otherwise $\mathrm{indeg}_G(v) = 0$ would hold and thus G would have a root v other than the unique root of N. Hence, $\tilde{S}$ satisfies (C1*). Next, for any distinct $e_1, e_2 \in E$ with either $\mathrm{tail}(e_1) = \mathrm{tail}(e_2)$ or $\mathrm{head}(e_1) = \mathrm{head}(e_2)$, at least one of $\{e_1, e_2\}$ is in $\tilde{S}$ since otherwise, by the assumption that N is almost-binary, $\mathrm{tail}(e_1) \notin X$ would be a leaf of G or $\mathrm{head}(e_1)$ would be an extra root of G. Hence, $\tilde{S}$ satisfies (C2*).

To prove the converse, assume that S is an $\mathcal{A}$-admissible subset of E. Then, S satisfies (C1*) and (C2*). This implies that $N[S]$ is a spanning subgraph of N. Indeed, if there were $v \in V$ with $v \notin \{\mathrm{head}(e), \mathrm{tail}(e)\}$ for any $e \in S$, then (C1*) or (C2*) (or both) would be violated, as easily verified for each possible case of $\mathrm{indeg}_N(v) \in \{0, 1, 2\}$. Let $\tilde{G} := (V, S)$ be the spanning subgraph of N. Since S satisfies both (C1*) and (C2*), $\mathrm{outdeg}_N(v) = 0$ if and only if $\mathrm{outdeg}_{\tilde{G}}(v) = 0$, so the root of N remains the unique root of $\tilde{G}$. Similarly, $\mathrm{outdeg}_N(v) = 0$ if and only if $\mathrm{outdeg}_{\tilde{G}}(v) = 0$, so the leaf-set of $\tilde{G}$ remains X. Hence, $\tilde{G} = (V, S)$ is a support network of N.

By the assumption on N described in Remark 4, if S_1 and S_2 are different $\mathcal{A}$-admissible subsets of E, then $N[S_1] = (V, S_1)$ and $N[S_2] = (V, S_2)$ are different support networks of N, and the converse also holds. Thus, the map $\mu : \mathcal{S}_\mathcal{A} \to \mathcal{A}_N$, defined by $\mu(S) = N[S]$, is a bijection.

The above arguments $\mathcal{X} = \mathcal{A}$ apply for similarly to $\mathcal{X} \in \{\mathcal{B}, \mathcal{C}\}$. Indeed, by Definitions 9 and 10, an $\mathcal{A}$-admissible subset $S \in \mathcal{S}_\mathcal{A}$ belongs to $\mathcal{S}_\mathcal{B}$ if and only if the support network $N[S] \in \mathcal{A}_N$ belongs to $\mathcal{B}_N$. Thus, the restriction of μ to $\mathcal{S}_\mathcal{B}$ gives a bijection between $\mathcal{S}_\mathcal{B}$ and $\mathcal{B}_N$. The same holds for $\mathcal{X} = \mathcal{C}$. This completes the proof of the first statement.

To prove the second statement for $\mathcal{X} = \mathcal{A}$, we show that a subset $S \subseteq E$ satisfies conditions (C1*) and (C2*) with $Z := N$ if and only if $S \cap E(Z_i)$ satisfies (C1*) and (C2*) with $Z := Z_i$ for each $i \in [1, d]$. First, observe that if $(u, v) \in E$ satisfies $\mathrm{outdeg}_N(u) = 1$ or $\mathrm{indeg}_N(v) = 1$, then there exists a unique $Z_i \in \mathcal{Z}$ such that $(u, v) \in E(Z_i)$. Since $\{E(Z_1), \ldots, E(Z_d)\}$ is a partition of E, the condition (C1*) for S with $Z := N$ holds if and only if each $S \cap E(Z_i)$ satisfies the condition (C1*) with $Z := Z_i$. Similarly, if two distinct edges $e_1, e_2 \in E$ share the same head or the same tail, then both edges must belong to a unique $Z_i \in \mathcal{Z}$, i.e., $\{e_1, e_2\} \subseteq E(Z_i)$. Then, the requirement that S contains e_1 or e_2 is equivalent to requiring that $S \cap E(Z_i)$ contains e_1 or e_2. Thus, S satisfies (C2*) with $Z := N$ if and only if each $S \cap E(Z_i)$ satisfies (C2*) with $Z := Z_i$. Hence, S is an $\mathcal{A}$-admissible subset of E (i.e., $N[S]$ is a support network of N) if and only if each $S_i := S \cap E(Z_i)$ is an $\mathcal{A}$-admissible subset of $E(Z_i)$.

The result just proved for $\mathcal{X} = \mathcal{A}$ implies that any $\mathcal{A}$-admissible subset $S \subseteq E$ can be obtained by independently selecting, for each $Z_i \in \mathcal{Z}$, an $\mathcal{A}$-admissible subset $S_i \subseteq E(Z_i)$, and then taking their union: $S = S_1 \cup \cdots \cup S_d$. Then, similarly to Remark 7, the admissible subset S is minimal (resp. minimum) if and only if each S_i is minimal (resp. minimum). This completes the proof.

A.2 Proof of Theorem 12

We prove that $|\mathcal{S}_{\mathcal{A}}(Z_i)| = F_{m_i}$ holds if Z_i is a fence and that $|\mathcal{S}_{\mathcal{A}}(Z_i)| = L_{m_i}$ holds if Z_i is a crown. Consider a fence $Z_i = (e_1, \ldots, e_{m_i}) \in \mathcal{Z}$. As the case of $m_i \leq 2$ is trivial, we may assume $m_i \geq 3$. Let $Z'_i = (e'_1, \ldots, e'_{m_i})$ be the undirected graph obtained from Z_i by ignoring all edge orientations, and let $Z''_i = (e'_2, \ldots, e'_{m_i-1})$ be its subgraph induced by the non-terminal edges of Z'_i. We can assume without loss of generality that Z''_i is a path. Indeed, when N is binary, this holds trivially because the fence Z_i visits each internal vertex exactly once. If N is almost-binary but not binary, Z''_i may fail to be a path due to the presence of a vertex v with both $\mathrm{indeg}_{Z_i}(v) = 2$ and $\mathrm{outdeg}_{Z_i}(v) = 2$. However, we can transform N by a preprocessing step that splits each such vertex v into two vertices connected by a new edge, which ensures that Z''_i is a path. This operation preserves $|\mathcal{A}_N|$ and does not affect the set $\mathcal{S}_{\mathcal{A}}(Z_i)$.

Let S be a subset of $E(Z_i)$ and let $\overline{S} := E(Z_i) \setminus S$. By Lemma 11 and Equation (3), S is $\mathcal{A}$-admissible if and only if $e_1, e_2 \notin \overline{S}$ and for any $k \in [1, m_i - 1]$, $\{e_k, e_{k+1}\} \not\subseteq \overline{S}$. Then, there exists a bijection between $\mathcal{S}_{\mathcal{A}}(Z_i)$ and the family of matchings in the path Z''_i. For example, when $m_i = 5$, an $\mathcal{A}$-admissible subset $S = \{e_1, e_3, e_5\}$ of $E(Z_i)$ corresponds to a matching $\overline{S} = \{e'_2, e'_4\}$ in $Z''_i = (e'_2, e'_3, e'_4)$. Theorem 1 in [5] says that the number of (possibly empty) matchings in a path with $\ell \geq 1$ edges equals $F_{\ell+2}$. Since $|E(Z''_i)| = m_i - 2$, the number of matchings in Z''_i equals F_{m_i}. Hence, $|\mathcal{S}_{\mathcal{A}}(Z_i)| = F_{m_i}$ holds.

Consider a crown $Z_i = (e_1, \ldots, e_{m_i})$. Since a crown has no terminal edges, $S \subseteq E(Z_i)$ is $\mathcal{A}$-admissible if and only if for any $k \in [1, m_i - 1]$, $\{e_k, e_{k+1}\} \not\subseteq \overline{S}$. Let Z'_i be the undirected graph obtained from Z_i by ignoring all edge orientations. Using the pre-processing described above, we can treat Z'_i as a cycle graph without loss of generality. Similarly to above, there is a bijection between $\mathcal{S}_{\mathcal{A}}(Z_i)$ and the family of matchings in the cycle Z'_i. Theorem 2 in [5] says that the number of (possibly empty) matchings in a cycle with $\ell \geq 3$ edges equals L_ℓ. Hence, $|\mathcal{S}_{\mathcal{A}}(Z_i)| = L_{m_i}$ holds.

Theorems 5.6 and 5.8 in [13] say that F_n and L_n are expressed by the closed-forms $F_n = (\phi^n - (-\phi)^{-n})/\sqrt{5}$ and $L_n = \phi^n + (1-\phi)^n$, respectively. Then, $F_n = \Theta(\phi^n)$ and $L_n = \Theta(\phi^n)$ hold. This implies $\Theta(|\mathcal{A}_N|) = \Theta(\phi^{|E(Z_1)|} \times \cdots \times \phi^{|E(Z_d)|}) = \Theta(\phi^{|E(N)|})$ holds. This completes the proof.

A.3 Proof of Theorem 14

We prove that $|\mathcal{S}_{\mathcal{B}}(Z_i)| = P_{m_i}$ holds if Z_i is a fence and $|\mathcal{S}_{\mathcal{B}}(Z_i)| = Q_{m_i}$ holds if Z_i is a crown. Consider a fence $Z_i = (e_1, \ldots, e_{m_i})$. As the case of $m_i \leq 2$ is trivial, we may assume $m_i \geq 3$. Let Z''_i be the undirected path as in the proof of Theorem 12. An $\mathcal{A}$-admissible subset S of $E(Z_i)$ is minimal (namely, $\mathcal{B}$-admissible) if and only if a matching in Z''_i corresponding to $\overline{S} := E(Z_i) \setminus S$ is maximal. Then, there exists a bijection between $\mathcal{S}_{\mathcal{B}}(Z_i)$ and the family of maximal matchings in Z''_i. For example, when $m_i = 5$, a $\mathcal{B}$-admissible subset $S = \{e_1, e_2, e_4, e_5\}$ of $E(Z_i)$ corresponds to a maximal matching $\overline{S} = \{e'_3\}$ in $Z''_i = (e'_2, e'_3, e'_4)$. Proposition 3.1 in [3] say that the number of maximal matchings in a path with ℓ edges equals $P_{\ell+2}$. Since $|E(Z''_i)| = m_i - 2$, the number of maximal matchings in Z''_i equals P_{m_i}. Hence, $|\mathcal{S}_{\mathcal{B}}(Z_i)| = P_{m_i}$ holds.

Consider a crown $Z_i = (e_1, \ldots, e_{m_i})$. Let Z'_i be the undirected cycle as in the proof of Theorem 12. Then, there exists a bijection between $\mathcal{S}_{\mathcal{B}}$ and the family of maximal matchings in the cycle Z'_i. By the statement after Proposition 3.9 in [3], the number of maximal matchings in a cycle with $\ell \geq 3$ edges equals Q_ℓ. Thus, $|\mathcal{S}_{\mathcal{B}}(Z_i)| = Q_{m_i}$.

By [3], $P_n = \Theta(\psi^n)$ and $Q_n = \Theta(\psi^n)$ hold. Hence, $\Theta(|\mathcal{B}_N|) = \Theta(\psi^{|E(Z_1)|} \times \cdots \times \psi^{|E(Z_d)|}) = \Theta(\psi^{|E(N)|})$ holds. This completes the proof.

A.4 Proof of Theorem 19

As Algorithm 1 checks the level of each support network in $\mathcal{B}_N$, our goal is to show that $\mathcal{B}_N$ contains a level-k support network of N. To obtain a contradiction, assume that $\mathrm{level}(G) > k$ holds for any $G \in \mathcal{B}_N$. This implies that any level-k support network G' of N is in $\mathcal{A}_N \setminus \mathcal{B}_N$. Since G' is not minimal, there exists a minimal one $G \in \mathcal{B}_N$ with $G \subsetneq G'$. Then, $\mathrm{level}(G) \leq \mathrm{level}(G') = k$ since each block of G is a subgraph of its corresponding block of G', which is a contradiction. This completes the proof of the correctness of Algorithm 1.

Since N is a binary network, we have $O(|V(N)|) = O(|E(N)|)$. Also, as any minimal support network $G \in \mathcal{B}_N$ is a subgraph of N, we have $O(|E(G)|) = O(|E(N)|)$. For each $G \in \mathcal{B}_N$, it takes $O(|V(G)| + |E(G)|) = O(|E(N)|)$ time to compute its block decomposition $\{G_1, \ldots, G_k\}$ using depth-first search [11]. For any block G_i, one can compute $r_i = |E(G_i)| - |V(G_i)| + 1$ in $O(|E(G_i)|)$ time. Then, for each $G \in \mathcal{B}_N$, it takes $O(|E(G_1)| + \cdots + |E(G_k)|) = O(|E(G)|) = O(|E(N)|)$ time to calculate $\mathrm{level}(G)$, namely the maximum r_i across all blocks G_i of G. Overall, it takes $O(|\mathcal{B}_N| \cdot |E(N)|)$ time to obtain the minimum value of $\mathrm{level}(G)$ across all $G \in \mathcal{B}_N$. By Theorem 14, $O(|\mathcal{B}_N| \cdot |E(N)|) = O(\psi^{|E(N)|} \cdot |E(N)|)$ holds.