Information and Computation 252 (2017) 161–175

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

A decomposition theorem and two algorithms for reticulation-visible networks

Andreas D.M. Gunawan [a], Bhaskar DasGupta [b], Louxin Zhang [a,*]

[a] Department of Mathematics, National University of Singapore, Singapore 119076, Singapore
[b] Department of Computer Science, University of Illinois at Chicago, Chicago, IL 60607, USA

ARTICLE INFO

Article history:
Received 8 January 2016
Received in revised form 26 May 2016
Available online 5 November 2016

Keywords:
Phylogenetic networks
Reticulation-visibility
Tree containment problem
Cluster containment problem
Galled networks

ABSTRACT

In studies of molecular evolution, phylogenetic trees are rooted binary trees, whereas phylogenetic networks are rooted acyclic digraphs. Edges are directed away from the root and leaves are uniquely labeled with taxa in phylogenetic networks. For the purpose of validating evolutionary models, biologists check whether or not a phylogenetic tree (resp. cluster) is contained in a phylogenetic network on the same taxa. These tree and cluster containment problems are known to be NP-complete. A phylogenetic network is reticulation-visible if every reticulation node separates the root of the network from at least a leaf. We answer an open problem by proving that the tree containment problem is solvable in quadratic time for reticulation-visible networks. The key tool used in our answer is a powerful decomposition theorem. It also allows us to design a linear-time algorithm for the cluster containment problem for networks of this type and to prove that every galled network with n leaves has $2(n-1)$ reticulation nodes at most.

1. Introduction

How life came to existence and evolved has been a key scientific question in the past hundreds of years. Traditionally, a phylogenetic tree has been used to model the evolutionary history of species, in which an internal node represents a *speciation event* and the leaves represent the extant species under study. These evolutionary trees are often reconstructed from the gene or protein sequences sampled from the extant species. Since genomic studies have demonstrated that genetic material is often transferred between organisms in a non-reproductive manner [4,25], it has been commonly accepted that phylogenetic networks are more suitable than phylogenetic trees for modeling reticulation events such as horizontal gene transfer, introgression, recombination and hybridization events in genome evolution [6,7,14,23,24]. Mathematically, a phylogenetic network is a rooted acyclic digraph with leaves that are uniquely labeled with extant species. The algorithmic and combinatorial aspects of networks have been intensively studied over the past two decades (*e.g.*, see [8,14,18,27]).

In phylogenetics, an important task is checking the "consistency" of two evolutionary models. A somewhat simpler (but nonetheless very important) version of this task asks whether a given network is consistent with an existing tree model or not. This has motivated researchers to study the problem of determining whether a tree is displayed by a network or not, which is called the tree containment problem (TCP). The cluster containment problem (CCP) is related algorithmic problem that asks whether or not a subset of taxa is a cluster in a tree displayed by a network [18]. Both the TCP and CCP have also been investigated in the development of metrics for phylogenetic networks [3,19].

* Corresponding author.
E-mail addresses: a0054645@u.nus.edu (A.D.M. Gunawan), bdasgup@uic.edu (B. DasGupta), matlzx@nus.edu.sg (L. Zhang).

http://dx.doi.org/10.1016/j.ic.2016.11.001
0890-5401/

The TCP and CCP are NP-complete [19], even on a very restricted class of networks [26]. Because of their importance, a lot of effort has been made to find network classes for which the TCP and CCP are solvable in polynomial time. For example, the TCP has been shown to be solvable in polynomial time for two rather restricted subclasses of reticulation-visible networks[12,26]. Here, the reticulation-visibility property was originally introduced to capture an important feature of galled networks [17]. A network is reticulation-visible if every reticulation node separates the network root from at least a leaf. The evolutionary network of violet species appearing in [22] is reticulation-visible, where each reticulation node represents a hybridization event that usually produces a new hybrid species. Gambette et al. also proved that the TCP is polynomial time solvable for binary nearly stable networks and that each reticulation-visible network contains at most $4(n-1)$ reticulations [11]. Other studies related to the TCP include [5,21] and [28].

In this paper, we make three contributions. van Iersel et al. has posed an open problem as to whether or not the TCP is solvable in polynomial time for *binary* reticulation-visible networks [18,26]. In this final version of [13], we present a quadratic time TCP algorithm for *arbitrary* reticulation-visible networks. We would remark that a cubic-time TCP algorithm for *binary* reticulation-visible networks is obtained independently in [2]. In [2], it is also proved that each reticulation-visible network has at most $3(n-1)$ reticulation nodes, which is the tight bound.

Secondly, a quadratic time CCP algorithm for reticulation-visible networks can be found in [18, page 171]. Here, we present a linear time CCP algorithms for this class of networks.

Thirdly, we present a powerful decomposition theorem (Theorem 1) to design our algorithms for the CCP and TCP. Empowered by this, we also prove that every galled network with n leaves has $2(n-1)$ reticulation nodes at most.

The rest of the paper is organized as follows. Section 2 introduces the basic concepts and notation. In Section 3, we develop the decomposition theorem mentioned above. It reveals an important structural property of reticulation-visible networks, based on which the two main theorems (Theorems 3 and 4) are proven in Sections 5 and 6, respectively. Finally, we conclude with a couple of remarks in Section 7.

2. Basic concepts and notation

2.1. Phylogenetic networks

In phylogenetics, *networks* are rooted acyclic digraphs that satisfies (i) edges and paths are directed away from a unique node called the *root*, (ii) there is a path from the root to *every* other node, and (iii) the nodes of indegree one and outdegree zero (the *leaves*) are *uniquely* labeled. The leaf labels represent bio-molecular sequences, extant organisms or species under study.

In a network, the root has indegree zero and outdegree greater than one and each of the other nodes has either indegree one or outdegree one exclusively. A node is called a *reticulation* node if its indegree is strictly greater than one and its outdegree is precisely one. A reticulation node is called a *bicombination* if it has indegree two. Reticulation nodes represent reticulation events occurring in evolution. Non-reticulation nodes are called *tree* nodes, including the root and leaves.

A network is called a *bicombining* network if every reticulation node is of degree three (*i.e.*, indegree two and outdegree one). A network is called *binary* if the root is of degree two, its leaves are of degree one, and all other nodes are of degree three. A *phylogenetic tree* is a binary network without reticulation nodes.

For convenience in describing the algorithms and proofs, we add an *open* incoming edge to the root (Fig. 1). This open edge is not counted into the degree of the root. Let N be a network. For two nodes u, v in N, u is a *parent* of v (alternatively, v is a *child* of u) if (u, v) is an edge in N; u is an *ancestor* of v (alternatively, v is a *descendant* of u) if there is a path from u to v. When u is an ancestor of v, we simply say v is *below* u and u is *above* v sometimes.

Let N be a network. We use the following notation:

- $\rho(N)$: the root of N;
- $\mathcal{L}(N)$: the set of all leaves in N;
- $\mathcal{R}(N)$: the set of all reticulation nodes in N;
- $\mathcal{T}(N)$: the set of the root and other tree nodes of outdegree greater than one in N;
- $\mathcal{V}(N)$: the set of all nodes in N (*i.e.*, $\mathcal{V}(N) = \mathcal{R}(N) \cup \mathcal{T}(N) \cup \mathcal{L}(N)$);
- $\mathcal{E}(N)$: the set of all edges in N;
- $p(u)$: the set of the parents of $u \in \mathcal{R}(N)$ or the unique parent of $u \in \mathcal{T}(N) \setminus \{\rho(N)\}$;
- $c(u)$: the set of the children of $u \in \mathcal{T}(N)$ or the unique child of $u \in \mathcal{R}(N)$;
- $\mathcal{D}_N(u)$: the *subnetwork* node-induced by $u \in \mathcal{V}(N)$ and all the descendants of u;
- $N - E$: the spanning *subnetwork* of N with the node set $\mathcal{V}(N)$ and the edge set $\mathcal{E}(N) \setminus E$ for a subset $E \subseteq \mathcal{E}(N)$;
- $N - V$: the *subnetwork* of N with the node set $\mathcal{V}(N) \setminus V$ and the edge set $\{(x, y) \in \mathcal{E}(N) \mid x \notin V, y \notin V\}$ for a subset $V \subseteq \mathcal{V}(N)$.

Let T be a phylogenetic tree. For $S \subseteq \mathcal{V}(T)$ and $w \in \mathcal{V}(T)$, w is called the lowest common ancestor (LCA) of the nodes in S, denoted $\mathrm{lca}(S)$ if it is an ancestor of every node in S and any other "common" ancestor of the nodes in S is above w in T.