Solving the Tree Containment Problem for Genetically Stable Networks in Quadratic Time

Philippe Gambette Andreas D. M. Gunawan **Anthony Labarre**
Stéphane Vialette Louxin Zhang

International Workshop on Combinatorial Algorithms

October 6th, 2015

UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

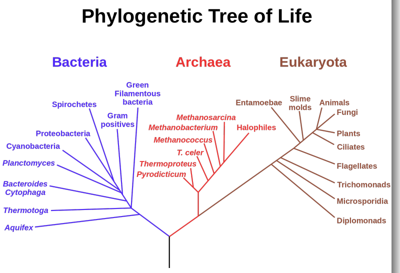
Gaspard - Monge Laboratoire d'informatique

NUS National University of Singapore

Context and motivations

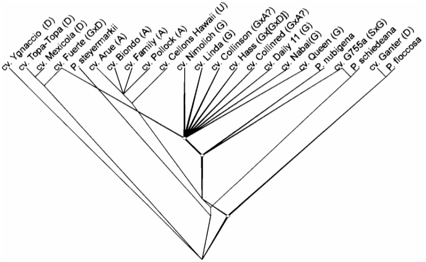
- **Phylogenetic trees** are routinely used to represent evolution, but they cannot display exchanges of genetic material between species;
- When these happen, we rely on **phylogenetic networks** instead;

Example (tree)



(from Wikimedia)

Example (network)

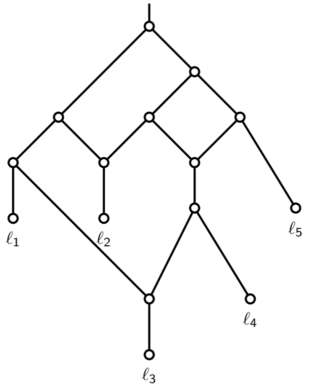


(from The Genealogical World of Phylogenetic Networks)

- We still need to verify that the network “contains” a prescribed set of trees to ensure consistency with previous biological knowledge;

Phylogenetic networks and related concepts

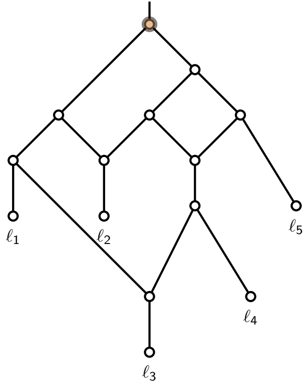
A **phylogenetic network** is a rooted DAG with a labelled leaf set $\{\ell_1, \ell_2, \ldots, \ell_k\}$.



We only consider **binary** networks and trees, i.e. all internal nodes have degree three.

Phylogenetic networks and related concepts

A **phylogenetic network** is a rooted DAG with a labelled leaf set $\{\ell_1, \ell_2, \ldots, \ell_k\}$.

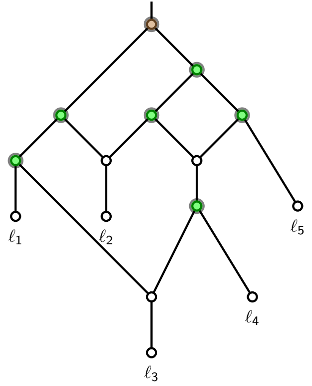


- root: indegree 0;

We only consider **binary** networks and trees, i.e. all internal nodes have degree three.

Phylogenetic networks and related concepts

A **phylogenetic network** is a rooted DAG with a labelled leaf set $\{\ell_1, \ell_2, \ldots, \ell_k\}$.

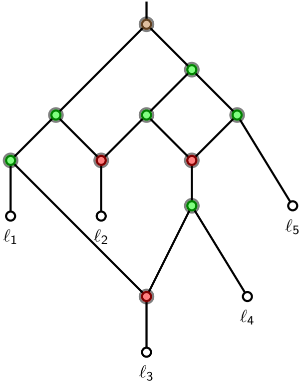


- root: indegree 0;
- tree nodes: indegree 1, outdegree 2;

We only consider **binary** networks and trees, i.e. all internal nodes have degree three.

Phylogenetic networks and related concepts

A **phylogenetic network** is a rooted DAG with a labelled leaf set $\{\ell_1, \ell_2, \ldots, \ell_k\}$.

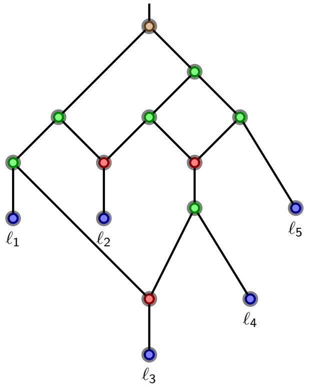


- root: indegree 0;
- tree nodes: indegree 1, outdegree 2;
- reticulations: indegree 2, outdegree 1;

We only consider **binary** networks and trees, i.e. all internal nodes have degree three.

Phylogenetic networks and related concepts

A **phylogenetic network** is a rooted DAG with a labelled leaf set $\{\ell_1, \ell_2, \ldots, \ell_k\}$.



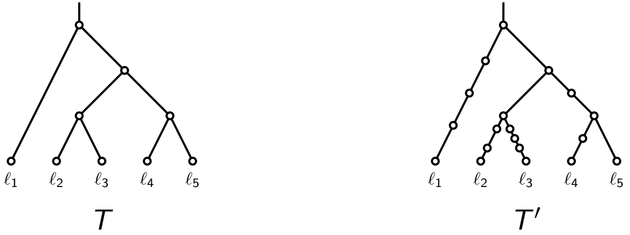
- root: indegree 0;
- tree nodes: indegree 1, outdegree 2;
- reticulations: indegree 2, outdegree 1;
- leaves: outdegree 0;

We only consider **binary** networks and trees, i.e. all internal nodes have degree three.

Tree subdivisions

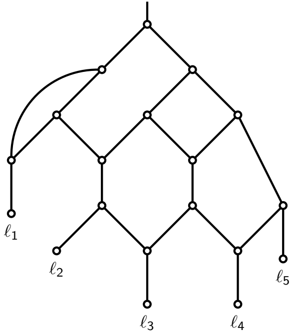
A **subdivision** of a tree T is a tree T' obtained by inserting any number of vertices into the edges of T.

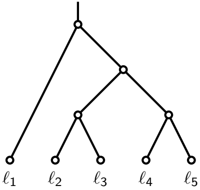
Example (a tree and a subdivision)



The TREE CONTAINMENT problem

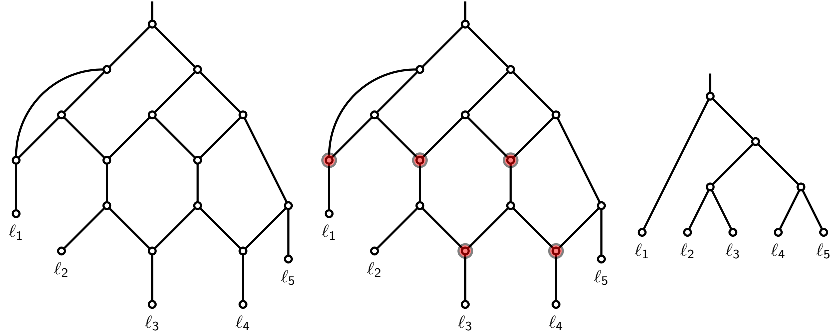
Network N **displays** tree T if we can obtain a subdivision of T by removing incoming edges from reticulations and "dummy leaves".





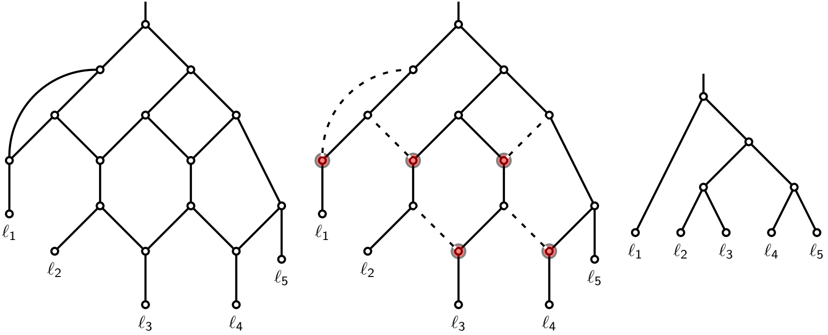
The TREE CONTAINMENT problem

Network N **displays** tree T if we can obtain a subdivision of T by removing incoming edges from reticulations and "dummy leaves".



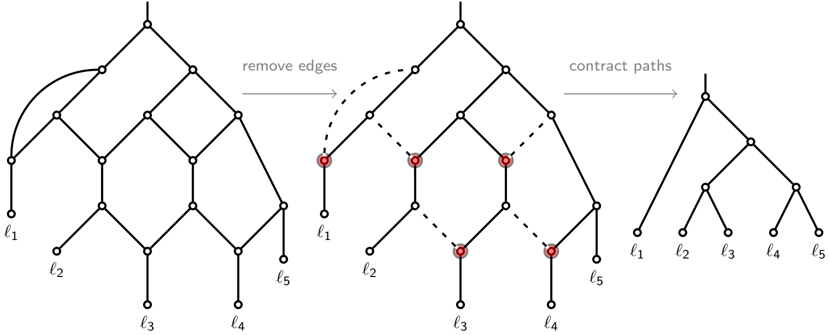
The TREE CONTAINMENT problem

Network N **displays** tree T if we can obtain a subdivision of T by removing incoming edges from reticulations and “dummy leaves”.



The TREE CONTAINMENT problem

Network N **displays** tree T if we can obtain a subdivision of T by removing incoming edges from reticulations and “dummy leaves”.

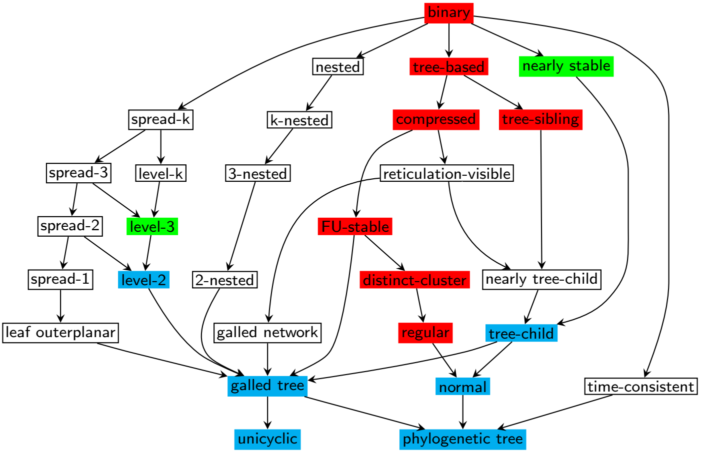


Problem (TREE CONTAINMENT)

Input:	*a phylogenetic network N, a phylogenetic tree T.*
Question:	*does N display T?*

TREE CONTAINMENT prior to this work

$A \to B$ class A contains class B
solvable in polynomial time
in P by class inclusion
NP-complete

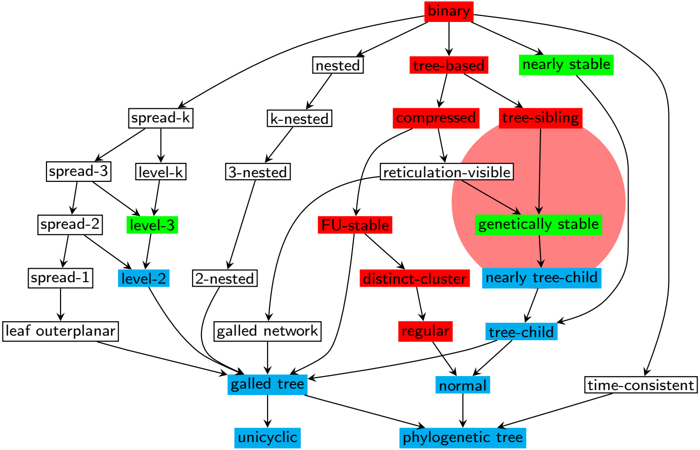


(adapted from `http://phylnet.univ-mlv.fr/isiphync` by Philippe Gambette)

Our contributions

1. **genetically stable** (GS) networks;
2. inclusion relations w.r.t. other classes;
3. TREE CONTAINMENT in P for GS networks;

$A \rightarrow B$ class A contains class B



(adapted from `http://phylnet.univ-mlv.fr/isiphync` by Philippe Gambette)

Genetically stable networks

A node v in a network N is **stable on a leaf** ℓ if every path from the root to ℓ contains v.

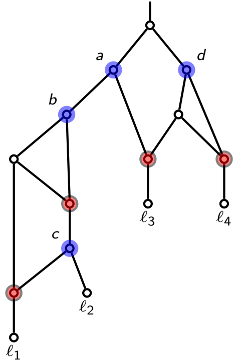
Genetically stable networks

A node v in a network N is **stable on a leaf** ℓ if every path from the root to ℓ contains v. A network N is **genetically stable** if every reticulation has a stable parent (on any leaf).

Genetically stable networks

A node v in a network N is **stable on a leaf** ℓ if every path from the root to ℓ contains v. A network N is **genetically stable** if every reticulation has a stable parent (on any leaf).

A GS network

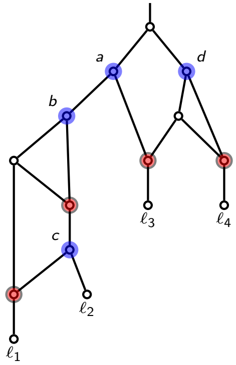


a, b, c stable on ℓ_2
d stable on ℓ_4

Genetically stable networks

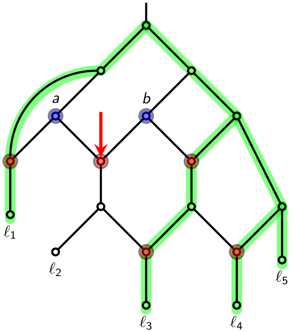
A node v in a network N is **stable on a leaf** ℓ if every path from the root to ℓ contains v. A network N is **genetically stable** if every reticulation has a stable parent (on any leaf).

A GS network



a, b, c stable on ℓ_2
d stable on ℓ_4

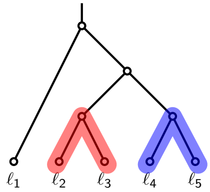
A non-GS network



ℓ_2 can be reached through either a or b
no other leaf "needs" a or b

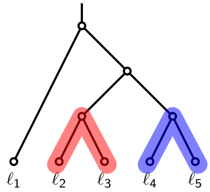
Overview of the algorithm

The subtree induced by two sibling leaves ℓ, ℓ' and their parent α in a tree is called a **cherry**, and is denoted by $\{\alpha, \ell, \ell'\}$.



Overview of the algorithm

The subtree induced by two sibling leaves ℓ, ℓ' and their parent α in a tree is called a **cherry**, and is denoted by $\{\alpha, \ell, \ell'\}$.



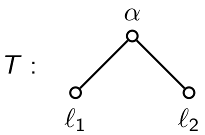
Algorithm for TREE CONTAINMENT in GS networks

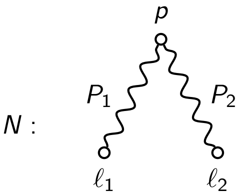
1. Select a cherry $C = \{\alpha, \ell, \ell'\}$ in T;
2. If there is no **match** for C in N, report NO;
3. Otherwise, **remove** the match from N and C from T;
4. If T is now a single node, report YES, otherwise go back to 1;

Matches and removals are such that N displays T if and only if N' displays T'.

Matching cherries: stability helps

Stability narrows down choices for matching α, (α, ℓ_1) and (α, ℓ_2) in N:





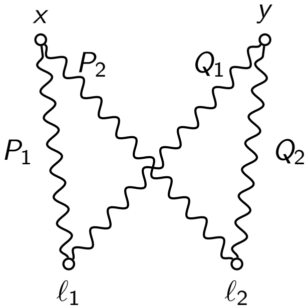
Lemma (1)

If N displays T through some subdivision T', then α must be matched to a node p such that:

1. *ℓ_1 and ℓ_2 are the only leaves on which p can be stable;*
2. *ℓ_1 is the only leaf on which vertices in $P_1 \setminus \{p\}$ can be stable;*
3. *ℓ_2 is the only leaf on which vertices in $P_2 \setminus \{p\}$ can be stable.*

Matching cherries: **genetic** stability helps

Lemma (1) allows us to focus on **specific** paths, i.e. paths P from x to ℓ such that each vertex in $P \setminus \{x\}$ is either stable only on ℓ or not stable at all. What if several choices exist?



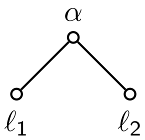
Lemma (2)

If N is genetically stable and contains vertices x and y connected to leaves ℓ_1 and ℓ_2 through specific paths that only intersect at x (resp. y), then either $y \in P_1 \cup P_2$ or $x \in Q_1 \cup Q_2$.

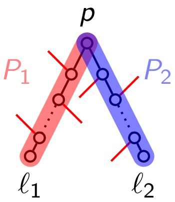
Modifying N and T when N is genetically stable

Lemma (2) allows us to restrict our search to the lowest common ancestor p of ℓ_1 and ℓ_2 such that paths $p \rightsquigarrow \ell_1$ and $p \rightsquigarrow \ell_2$ in N are specific.

T :



N :



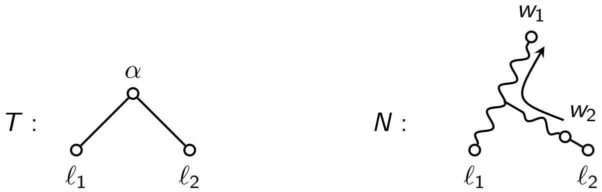
Lemma (3)

If p, P_1 and P_2 match α, (α, ℓ_1) and (α, ℓ_2) in a GS network N, then N displays T if and only if $N \setminus P_1 \setminus P_2$ displays $T \setminus \{\ell_1, \ell_2\}$.

Finding a match for α, (α, ℓ_1) and (α, ℓ_2) in N

1. Move up from ℓ_1 until we find a lowest common ancestor of ℓ_1 and ℓ_2 connected to ℓ_2 by a path free of nodes stable on other leaves;

2. Move up from ℓ_2 to w_1 while remaining in a specific path to ℓ_2;



3. If we succeed, we obtain two specific paths to ℓ_1 and ℓ_2 in N;

Correctness and running time

The previous lemmas prove the correctness of the algorithm.

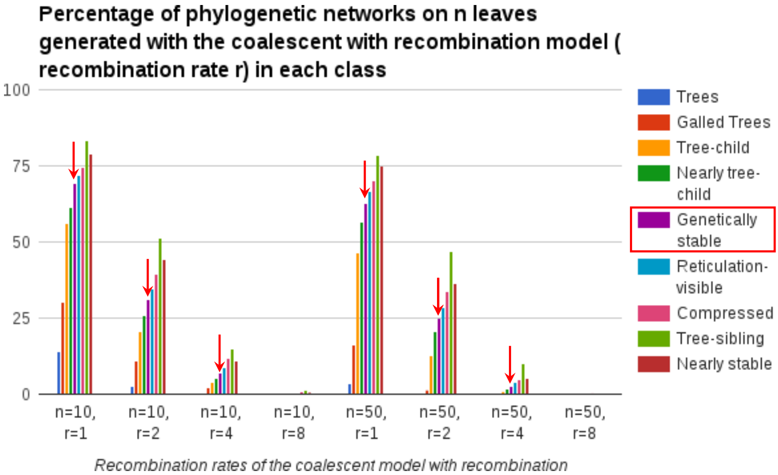
Algorithm for TREE CONTAINMENT in GS networks

1. Select a cherry $C = \{\alpha, \ell, \ell'\}$ in T;
2. If there is no **match** for C in N, report NO;
3. Otherwise, **remove** the match from N and C from T;
4. If T is now a single node, report YES, otherwise go back to 1;

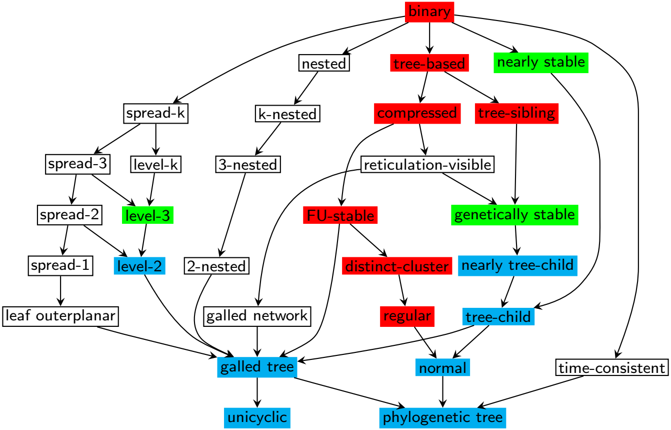
The running time is dominated by checking stability, which implies a running time of $O(|V| \cdot (|E| + |V|)) = O(|L|^2)$ where $|L|$ is the number of leaves of N.

Relevance of GS networks

A fair amount of real-world networks could be genetically stable:



Future work



- Major open problem: complexity for reticulation-visible networks;
- Refine hardness results;
- Improve the complexity for tractable cases;