Reconstructing Phylogenetic Networks

Mareike Fischer, Leo van Iersel, Steven Kelk, Nela Lekić, Simone Linz, Celine Scornavacca, Leen Stougie

Centrum Wiskunde & Informatica (CWI)
Amsterdam

MCW Prague, 30 July 2013
Definition

Let $X$ be a finite set. A **(rooted) phylogenetic tree** on $X$ is a rooted tree with no indegree-1 outdegree-1 vertices whose leaves are bijectively labelled by the elements of $X$.
Definition

Let $X$ be a finite set. A (rooted) phylogenetic network on $X$ is a rooted directed acyclic graph with no indegree-1 outdegree-1 vertices whose leaves are bijectively labelled by the elements of $X$. 

Parasitic Jaeger  
Pomarine Skua  
Great Skua
The first phylogenetic network (Buffon, 1755)
Definition

A reticulation is a vertex with indegree at least 2.
Tree-based methods

1. Compute trees from DNA sequences.
   - Different parts of DNA might give different trees.

2. Try to induce a phylogenetic network from the trees.

Definition

A phylogenetic tree $T$ is **displayed** by a phylogenetic network $N$ if $T$ can be obtained from a subgraph of $N$ by contracting edges.
Example: tree $T$ is displayed by network $N$
The other binary tree $T'$ displayed by network $N$
Challenge: try to reconstruct the network from the trees
Definition

The **reticulation number** of a phylogenetic network $N$ is

$$\sum_{v \in V \setminus \{\text{root}\}} d^-(v) - 1.$$  

Problem

**Minimum Reticulation**

- **Instance**: phylogenetic trees $T_1, T_2$
- **Solution**: phylogenetic network that displays $T_1$ and $T_2$
- **Minimize**: reticulation number of the network.

Theorem

*There exists a constant factor approximation algorithm for Minimum Reticulation if and only if there exists a constant factor approximation algorithm for Directed Feedback Vertex Set.*

Open question: how to handle more than two trees (efficiently)?
Reconstructing phylogenetic networks

- **Tree-based methods**
  1. Construct trees from DNA sequences.
  2. Find a network that displays the trees and has minimum reticulation number.

- **Sequence-based methods**
  - Find a network directly from the DNA sequences.
  - Optimize **Parsimony** or Likelihood score of network.
Maximum Parsimony for trees

Small parsimony problem: given a tree and a sequence for each leaf, assign sequences to the internal vertices in order to minimize the total number of mutations.

Example input

ACCTG  ATCTG  ATCTC  GTAAA  TTACT
Maximum Parsimony for trees

**Small parsimony problem:** given a tree and a sequence for each leaf, assign sequences to the internal vertices in order to **minimize** the total number of **mutations**.

Example labelling of internal vertices
Maximum Parsimony for trees

Small parsimony problem: given a tree and a sequence for each leaf, assign sequences to the internal vertices in order to minimize the total number of mutations.

Example of one mutation

```
ACCTG
ATCTG
ATCTG
ATCTA
ATCTC
TTCTA
TTAAA
GTAAA
TTACT
```

Leo van Iersel (CWI)  Reconstructing Phylogenetic Networks  MCW Prague, 30 July 2013  18 / 44
Maximum Parsimony for trees

**Small parsimony problem**: given a tree and a sequence for each leaf, assign sequences to the internal vertices in order to **minimize** the total number of **mutations**.

![Diagram showing a tree with sequences and numbers indicating mutations.]

All 9 mutations.
Maximum Parsimony for trees

**Small parsimony problem**: given a tree and a sequence for each leaf, assign sequences to the interior vertices in order to **minimize** the total number of **mutations**.

- Polynomial-time solvable:
  - Consider each character separately.
  - Use dynamic programming (Fitch, 1971).

- Two possible extensions to networks:
  - hardwired
  - softwired
A \( p \)-state \textbf{character} on \( X \) is a function \( \alpha : X \rightarrow \{1, \ldots, p\} \).

The \textbf{change} \( c_{\tau}(e) \) on edge \( e = (u, v) \) w.r.t. a \( p \)-state character \( \tau \) on \( V(N) \) is defined as:

\[
c_{\tau}(e) = \begin{cases} 
0 & \text{if } \tau(u) = \tau(v) \\
1 & \text{if } \tau(u) \neq \tau(v) 
\end{cases}
\]

The \textbf{hardwired parsimony score} of a phylogenetic network \( N \) and \( p \)-state character \( \alpha \) is given by

\[
PS_{\text{hw}}(N, \alpha) = \min_{\tau} \sum_{e \in E(N)} c_{\tau}(e),
\]

where the minimum is taken over all \( p \)-state characters \( \tau \) on \( V(N) \) that extend \( \alpha \).
Example input: \((N, \alpha)\)
A 3-state character $\tau$ on $V(N)$ that extends $\alpha$. 
$PS_{hw}(N, \alpha) = 4$
The **softwired parsimony score** of a phylogenetic network $N$ and $p$-state character $\alpha$ is given by

$$PS_{sw}(N, \alpha) = \min_{T \in \mathcal{T}(N)} PS(T, \alpha),$$

where $\mathcal{T}(N)$ is the set of trees on $X$ displayed by $N$. 

---

**Softwired Maximum Parsimony on Networks**

Leo van Iersel (CWI)  
Reconstructing Phylogenetic Networks  
MCW Prague, 30 July 2013  25 / 44
One of the two trees on $X$ displayed by the network
A 3-state character $\tau$ on $V(T)$ that extends $\alpha$. 

Leo van Iersel (CWI)  
Reconstructing Phylogenetic Networks  
MCW Prague, 30 July 2013  
27 / 44
There are 3 changes
The other tree needs 4 changes

The minimum over the two trees is 3, so $PS_{sw}(N, \alpha) = 3$. 
Proposition

$PS_{hw}(N, \alpha)$ is not an $o(n)$-approximation of $PS_{sw}(N, \alpha)$.
Softwired Parsimony Score

\[ PS_{sw}(N, \alpha) = 2 \]
Hardwired Parsimony Score

$PS_{hw}(N, \alpha) = 4 = r + 1$

with $r$ the number of reticulations.
Proposition

Let $G$ be the graph obtained from network $N$ by merging all leaves $x$ with $\alpha(x) = i$ into a single node $\gamma_i$, for $i = 1, \ldots, p$. Then, $PS_{hw}(N, \alpha)$ equals the size of a minimum multiterminal cut in $G$ with terminals $\gamma_1, \ldots, \gamma_p$.

Corollary

Computing the hardwired parsimony score of a phylogenetic network and a binary character is polynomial-time solvable.

Corollary

Computing the hardwired parsimony score of a phylogenetic network and a $p$-state character, for $p \geq 3$, is NP-hard and APX-hard but fixed-parameter tractable (FPT) in the parsimony score, and there exists a polynomial-time $1.3438$-approximation for all $p$ and a $\frac{12}{11}$-approximation for $p = 3$. 
Example
Merge 0-leaves and 1-leaves
Hardwired Parsimony Score is 4
Observation

There exists a (trivial) $|X|$-approximation for computing the softwired parsimony score of a phylogenetic network.

Theorem

For every constant $\epsilon > 0$ there is no polynomial-time approximation algorithm that approximates $PS_{sw}(N, \alpha)$ to a factor $|X|^{1-\epsilon}$, for a phylogenetic network $N$ and a binary character $\alpha$, unless $P = NP$.

Definition

A phylogenetic network is binary if the root has outdegree 2 and all other vertices have total degree 1 or 3.

Theorem

For every constant $\epsilon > 0$ there is no polynomial-time approximation algorithm that approximates $PS_{sw}(N, \alpha)$ to a factor $|X|^{\frac{1}{3}-\epsilon}$, for a binary phylogenetic network $N$ and a binary character $\alpha$, unless $P = NP$. 
Proof: reduction from 3SAT
Binary case
Theorem

There is no **FPT** algorithm for computing the softwired parsimony score, with the score as parameter, unless $P = NP$.

Definition

A phylogenetic network is **level-$k$** if each biconnected component has reticulation number at most $k$.

Theorem

There is an **FPT** algorithm for computing the softwired parsimony score, with the **level** of the network as parameter.
ILP for softwired parsimony score

\[
\begin{align*}
\min \sum_{e \in E} c_e \\
\text{s.t. } \sum_{s \in \mathcal{P}} x_{v,s} &= 1 \quad \text{for all } v \in V \\
& c_e \geq x_{u,s} - x_{v,s} - (1 - y_e) \quad \text{for all } e = (u, v) \in E, s \in \mathcal{P} \\
& c_e \geq x_{v,s} - x_{u,s} - (1 - y_e) \quad \text{for all } e = (u, v) \in E, s \in \mathcal{P} \\
& \sum_{v: (v, r) \in E} y_{(v, r)} = 1 \quad \text{for each reticulation } r \\
& y_e = 1 \quad \text{for each non-reticulate edge } e \\
& x_{v, \alpha(v)} = 1 \quad \text{for each leaf } v \\
& c_e, y_e \in \{0, 1\} \quad \text{for all } e \in E \\
& x_{v,s} \in \{0, 1\} \quad \text{for all } v \in V, s \in \mathcal{P}
\end{align*}
\]

with \( \mathcal{P} = \{1, \ldots, p\} \) and \( \alpha(v) \) the given character state of a leaf \( v \).
Both parsimony scores can be computed quickly using ILP

| \(|X|\) | Avg. num. of retic. | Average computation time (s) |
|-------|--------------------|------------------------------|
|       | Hardwired PS       | Softwired PS                |
|       | 2-state | 3-state | 4-state | 2-state | 3-state | 4-state |
| 50    | 17.0    | 0.0     | 0.0     | 0.1     | 0.1     | 0.3     |
| 100   | 37.0    | 0.0     | 0.0     | 0.2     | 0.0     | 0.6     |
| 150   | 54.1    | 0.0     | 0.1     | 0.6     | 0.1     | 0.8     |
| 200   | 72.8    | 0.0     | 0.1     | 1.1     | 0.1     | 1.4     |
| 250   | 91.3    | 0.0     | 0.1     | 3.5     | 0.1     | 2.2     |
| 300   | 112.6   | 0.0     | 0.2     | 5.2     | 0.1     | 3.7     |
Future Work

- Are there approximation or FPT algorithms for computing the softwired parsimony score of restricted classes of networks?

- How to search for an optimal network?

- What if the different characters are not independent?
Thanks

- Mareike Fischer (Greifswald)
- Steven Kelk (Maastricht)
- Celine Scornavacca (Montpellier)
- Simone Linz (Christchurch)
- Leen Stougie (Amsterdam)
- Nela Lekić (Maastricht)