PERFECT TRANSFER NETWORKS

Manuel Lafond,

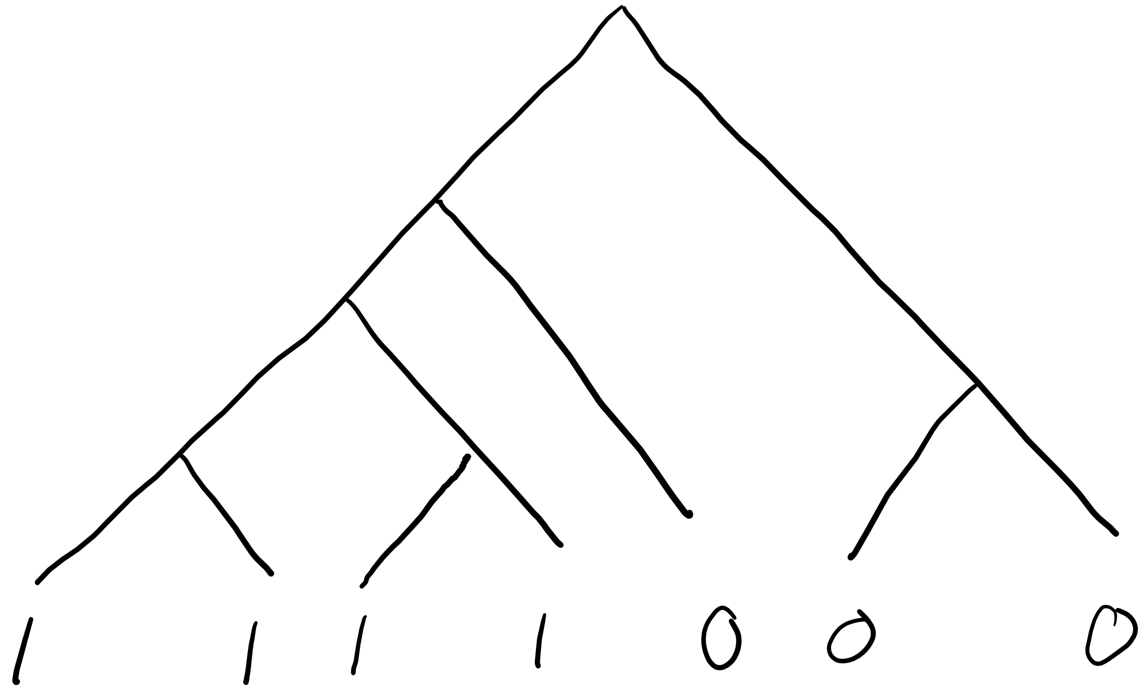
Alitzel Lopez-Sanchez

Université de Sherbrooke, Canada

UNIVERSITÉ DE SHERBROOKE

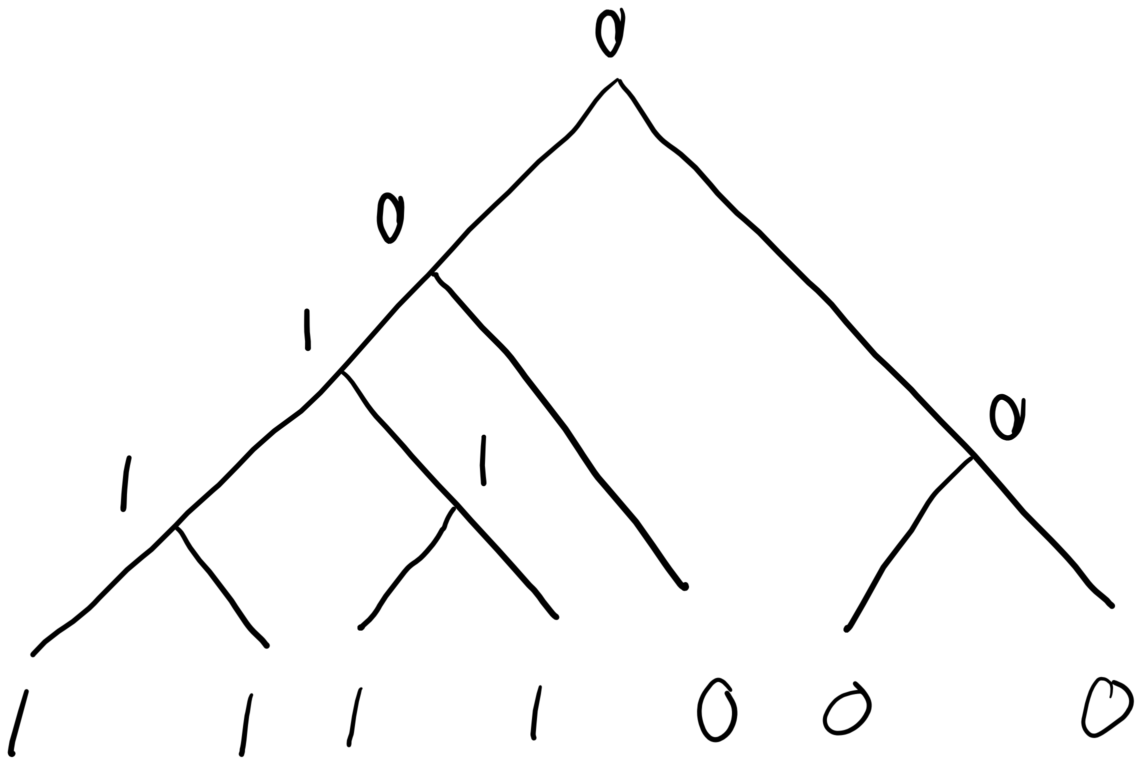
Perfect Phylogeny model

- $\{0, 1\}$ characters evolve on a tree. $0 =$ absence, $1 =$ presence
- **No homoplasy**: only one $0 \rightarrow 1$ transition allowed
- **No losses**: no $1 \rightarrow 0$ transition allowed



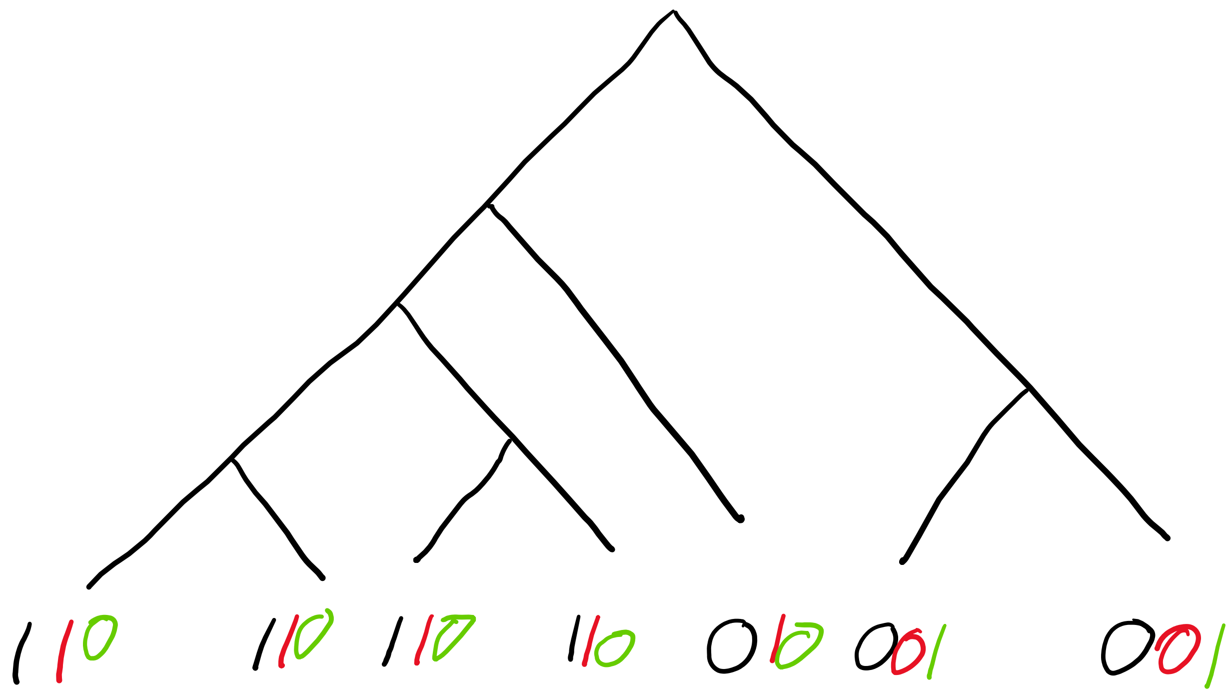
Perfect Phylogeny model

- $\{0, 1\}$ characters evolve on a tree. $0 =$ absence, $1 =$ presence
- **No homoplasy**: only one $0 \rightarrow 1$ transition allowed
- **No losses**: no $1 \rightarrow 0$ transition allowed



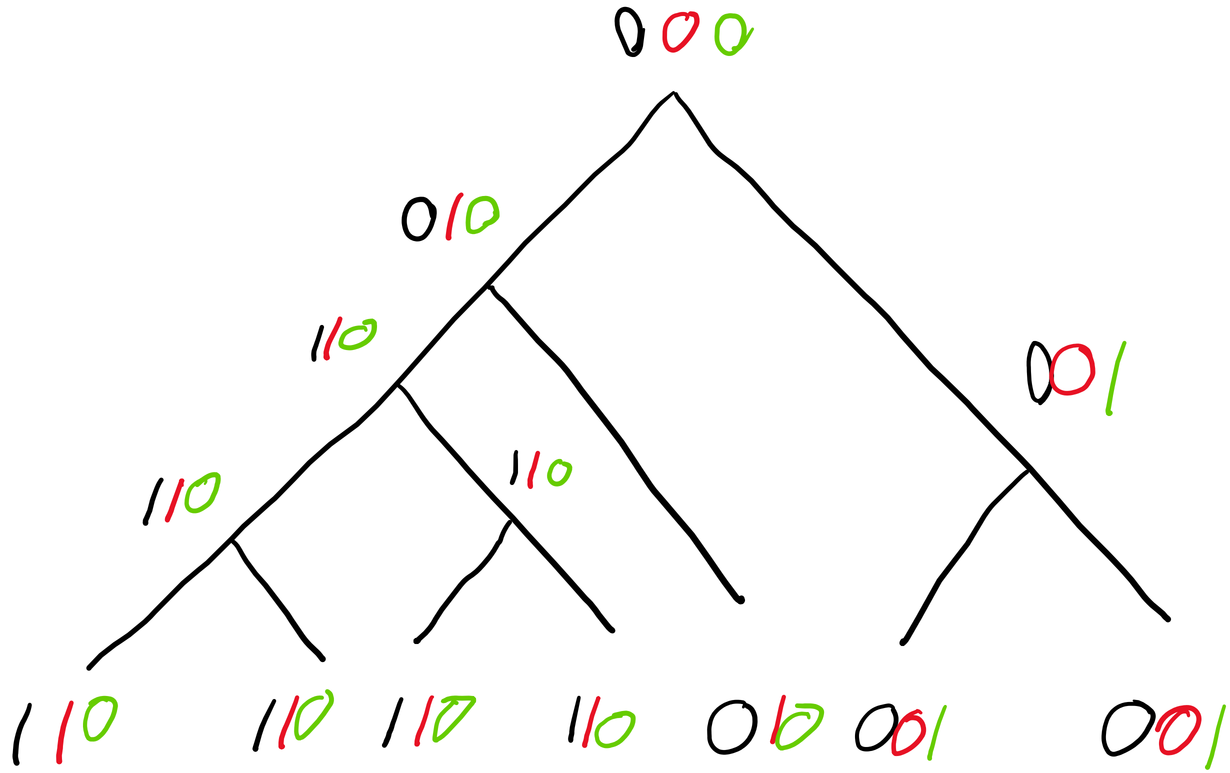
Perfect Phylogeny model

- {0, 1} characters evolve on a tree. 0 = absence, 1 = presence
- **No homoplasy**: only one 0 → 1 transition allowed
- **No losses**: no 1 → 0 transition allowed



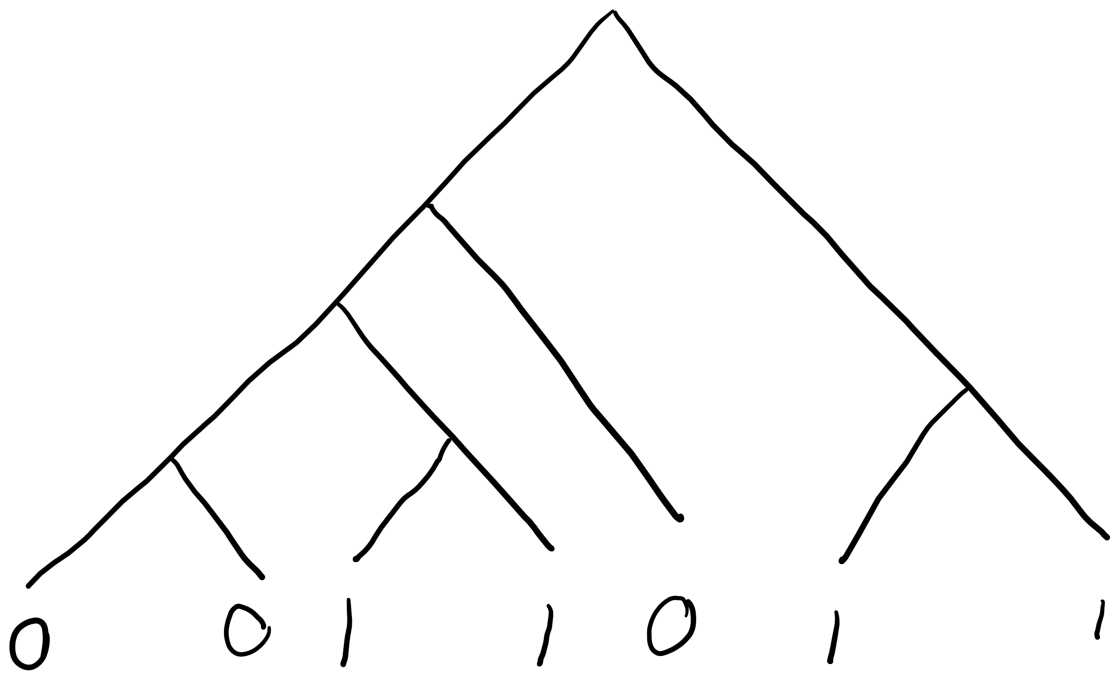
Perfect Phylogeny model

- {0, 1} characters evolve on a tree. 0 = absence, 1 = presence
- **No homoplasy**: only one 0 → 1 transition allowed
- **No losses**: no 1 → 0 transition allowed



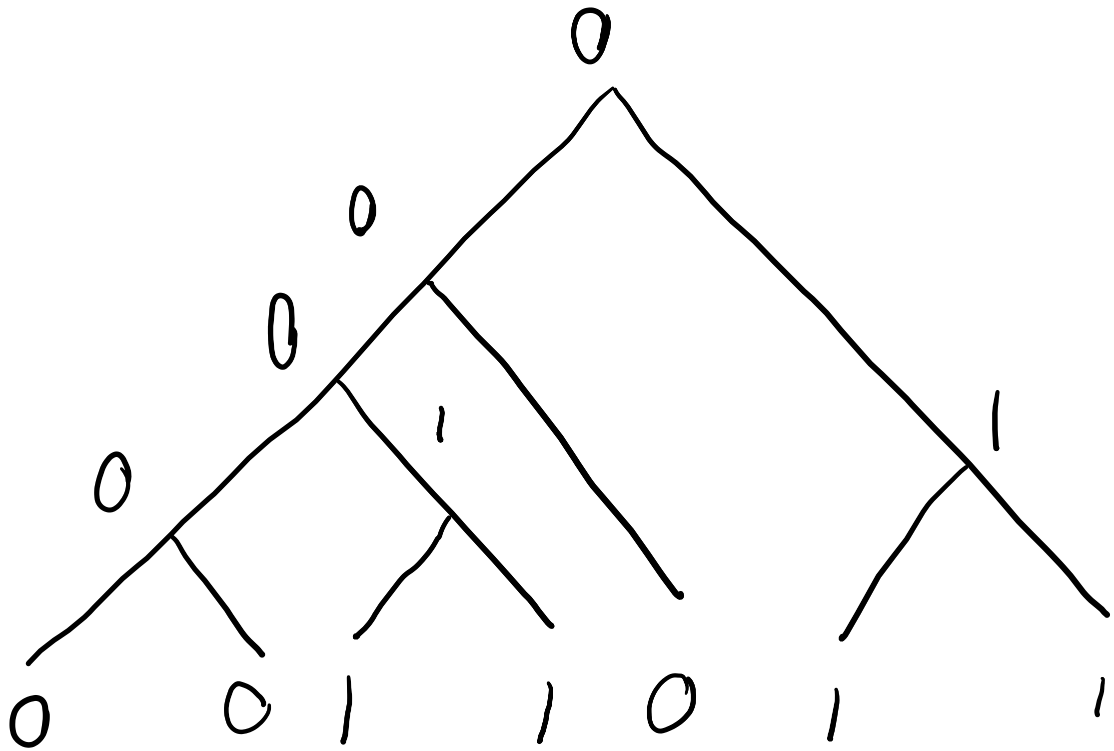
Perfect Phylogeny model

- {0, 1} characters evolve on a tree. 0 = absence, 1 = presence
- **No homoplasy**: only one 0 → 1 transition allowed
- **No losses**: no 1 → 0 transition allowed



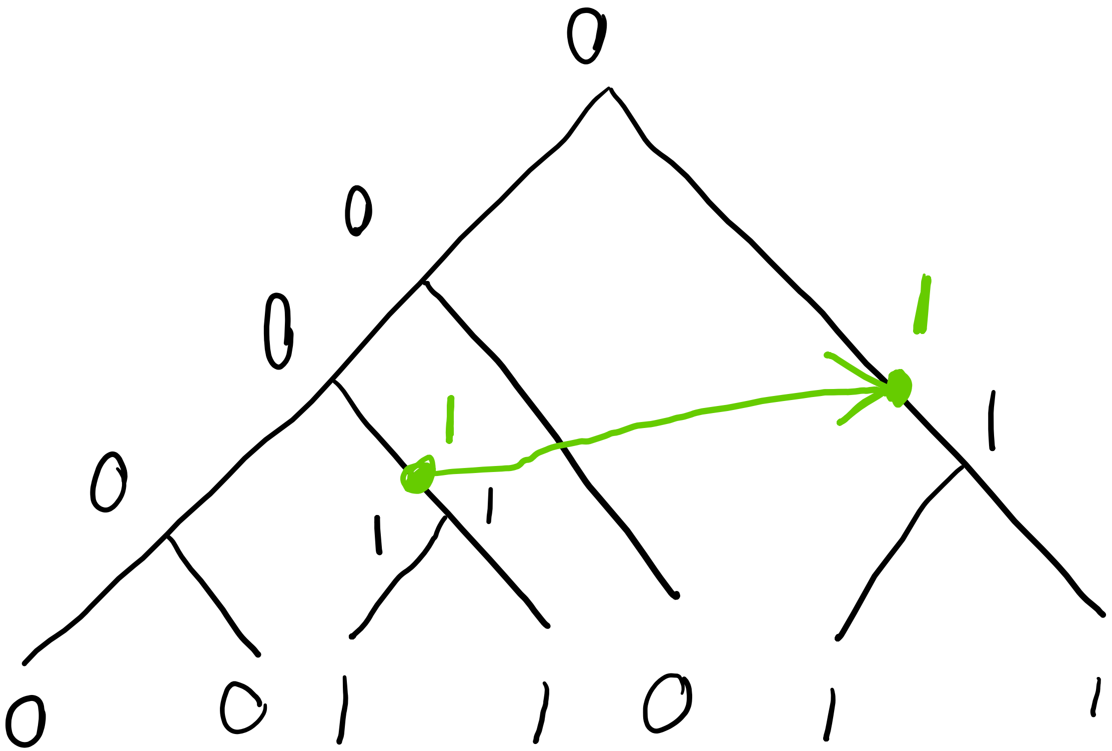
Perfect Phylogeny model

- {0, 1} characters evolve on a tree. 0 = absence, 1 = presence
- **No homoplasy**: only one 0 → 1 transition allowed
- **No losses**: no 1 → 0 transition allowed



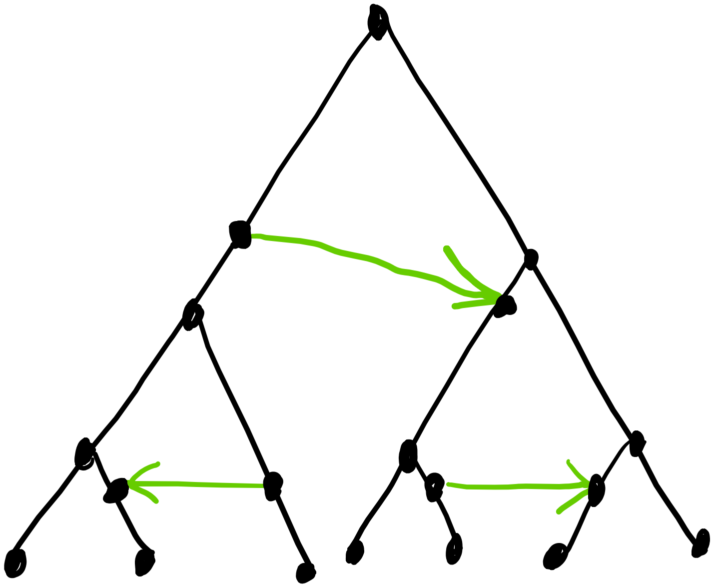
Perfect Phylogeny model

- {0, 1} characters evolve on a tree. 0 = absence, 1 = presence
- **No homoplasy**: only one 0 → 1 transition allowed
- **No losses**: no 1 → 0 transition allowed



Perfect Transfer Network model (PTN)

- {0, 1} characters evolve on an **LGT network.**



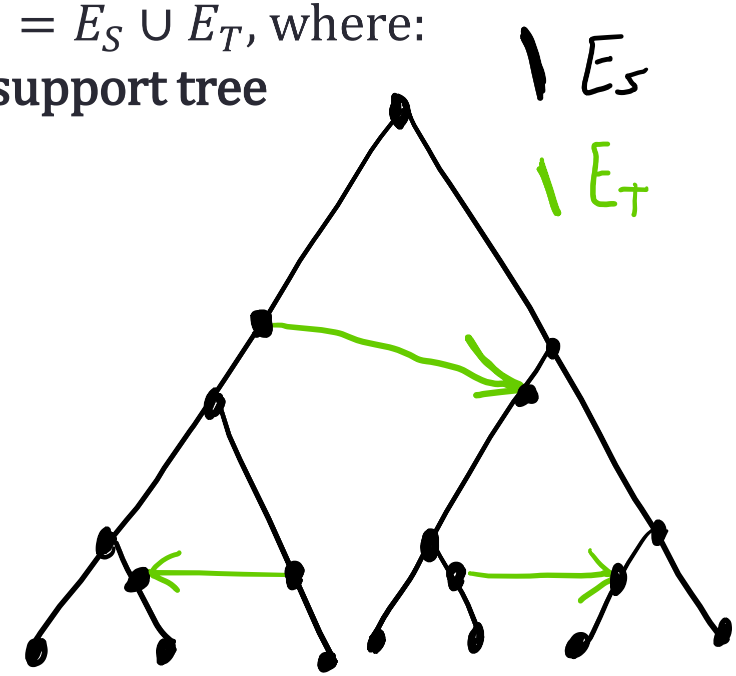
Perfect Transfer Network model (PTN)

- $\{0, 1\}$ characters evolve on an **LGT network.**

LGT network = tree-based network with **specified support tree.**

A network $N = (V, E)$ is an *LGT network* if $E = E_S \cup E_T$, where:

- The subgraph (V, E_S) is a tree, called the **support tree**
- The edges in E_T are called transfer edges

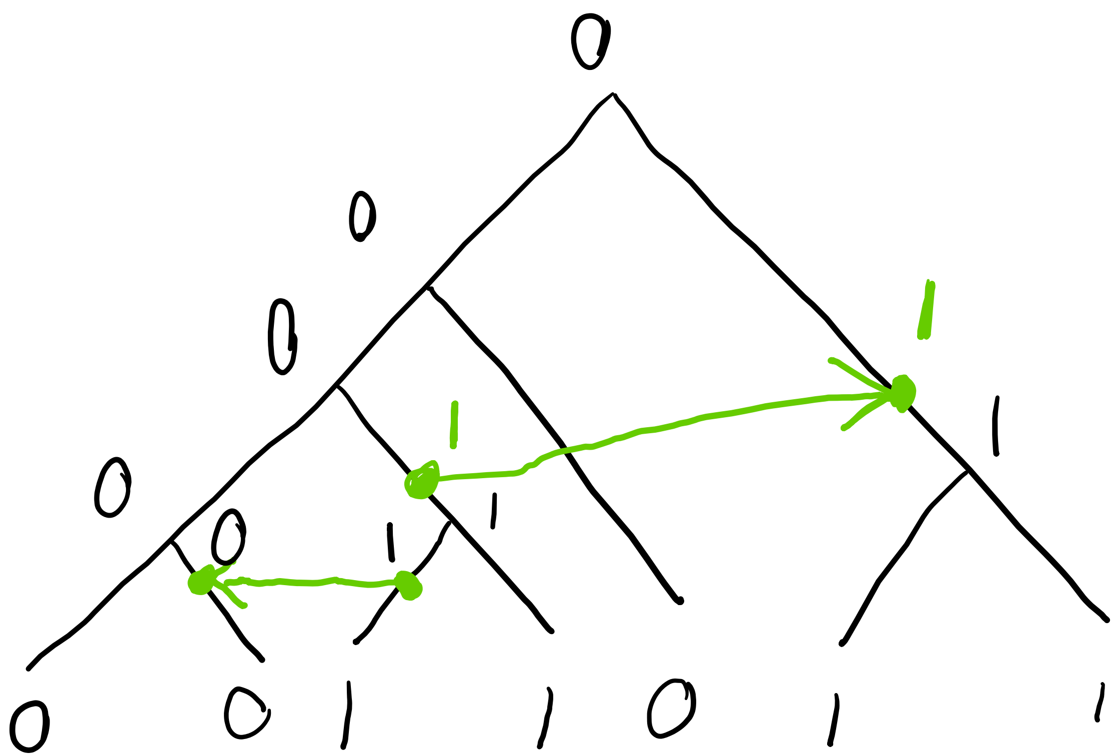


Perfect Transfer Network model (PTN)

- $\{0, 1\}$ characters evolve on an **LGT network** $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E_S} \cup \boldsymbol{E_T})$

Perfect Transfer Network model (PTN)

- $\{0, 1\}$ characters evolve on an **LGT network** $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E_S} \cup \boldsymbol{E_T})$
- **No homoplasy**: let V_1 be the set of nodes labeled 1. Then there is a unique $v \in V_1$ that reaches every node in $N[V_1]$.
- **No losses**: no $1 \rightarrow 0$ transition allowed in the support tree



Perfect Transfer Network model (PTN)

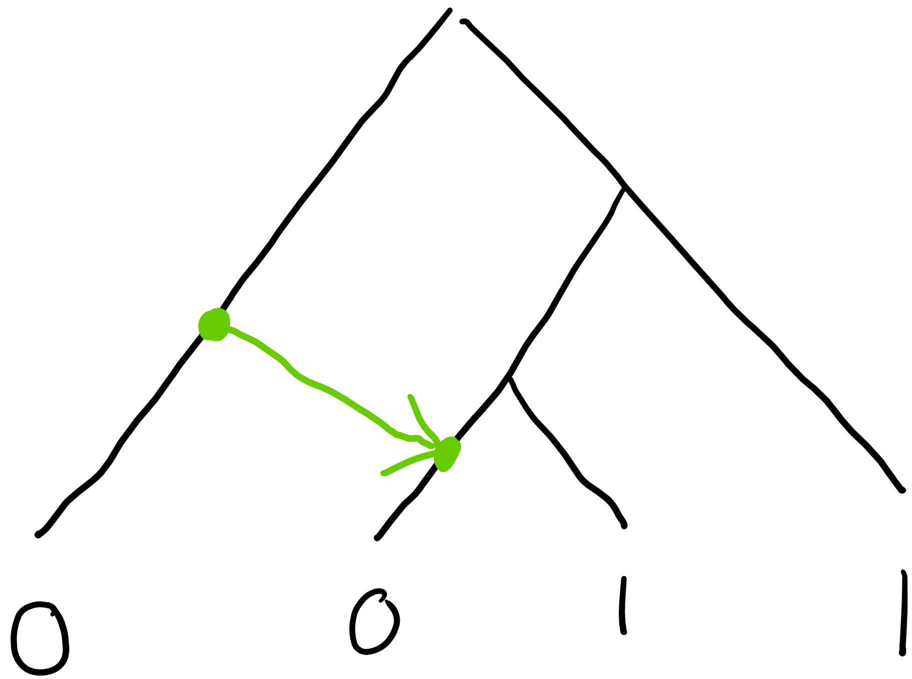
- $\{0, 1\}$ characters evolve on an **LGT network** $\boldsymbol{N = (V, E_S \cup E_T)}$
- **No homoplasy**: let V_1 be the set of nodes labeled 1. Then there is a unique $v \in V_1$ that reaches every node in $N[V_1]$.
- **No losses**: no $1 \to 0$ transition allowed in the support tree

Related work:

- Perfect Phylogenetic Networks [Nakhleh & Warnow, 2004]
- Perfect Recombination Networks [Wang, Zhang & Zhang, 2005]
- Ancestral Recombination Graphs [Gusfield, 2014]
- HGT+losses, averting genome of Eden [van Iersel, Semple & Steel, 2010]

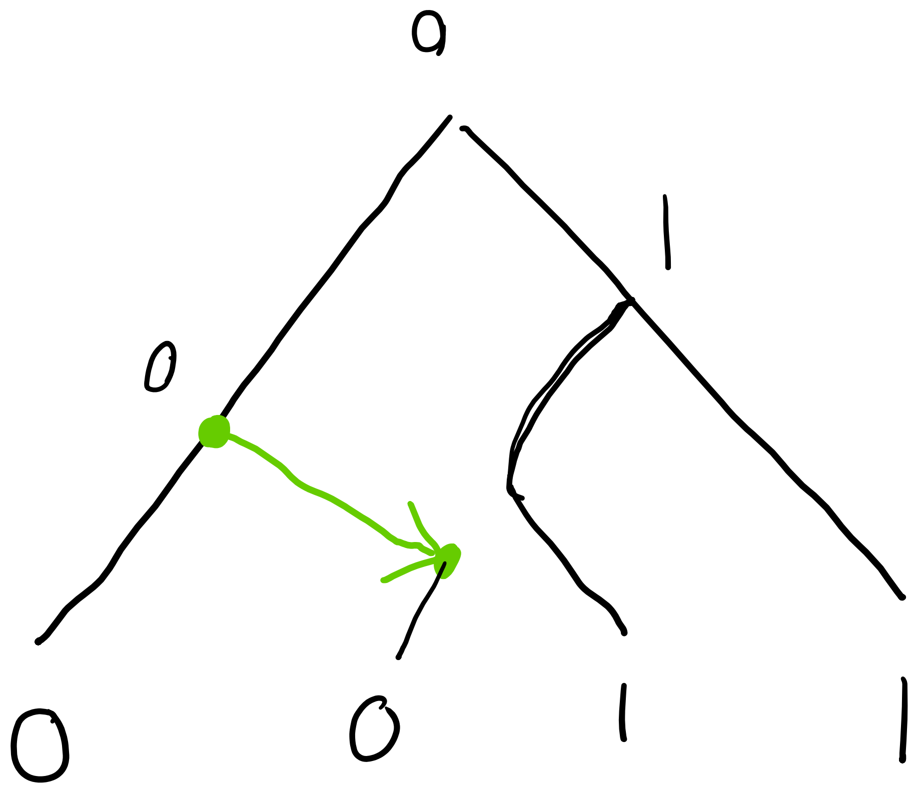
Perfect Phylogenetic Networks (PPN) [Nakhleh & Warnow, 2004]

- multi-state characters evolve on an **LGT network** $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E_S} \cup \boldsymbol{E_T})$
- for each character C, there must be a tree T displayed by N such that C is *convex* on T (nodes in same state induce a connected subgraph).



Perfect Phylogenetic Networks (PPN) [Nakhleh & Warnow, 2004]

- multi-state characters evolve on an **LGT network** $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E_S} \cup \boldsymbol{E_T})$
- for each character C, there must be a tree T displayed by N such that C is *convex* on T (nodes in same state induce a connected subgraph).



Perfect Phylogenetic Networks (PPN) [Nakhleh & Warnow, 2004]

- multi-state characters evolve on an **LGT network** $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E_S} \cup \boldsymbol{E_T})$
- for each character C, there must be a tree T displayed by N such that C is *convex* on T (nodes in same state induce a connected subgraph).

The main difference:

- PPN: all state transitions $i \rightarrow j$ allowed everywhere
- PTN: $1 \rightarrow 0$ forbidden on support tree

Extending a leaf-labelled network N into a PPN is NP-hard
→ Motivation for studying special cases

Perfect Transfer Network model (PTN)

- $\{0, 1\}$ characters evolve on an **LGT network** $\boldsymbol{N = (V, E_S \cup E_T)}$
- **No homoplasy**: let V_1 be the set of nodes labeled 1. Then there is a unique $v \in V_1$ that reaches every node in $N[V_1]$.
- **No losses**: no $1 \rightarrow 0$ transition allowed in the support tree

Perfect Transfer Network model (PTN)

- $\{0, 1\}$ characters evolve on an **LGT network** $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E_S} \cup \boldsymbol{E_T})$
- **No homoplasy**: let V_1 be the set of nodes labeled 1. Then there is a unique $v \in V_1$ that reaches every node in $N[V_1]$.
- **No losses**: no $1 \to 0$ transition allowed in the support tree

Algorithmic questions [Lopez & L, 2022]:

1. **PTN recognition**: given a leaf-labeled LGT network N, can we extend the labeling into a PTN?

Perfect Transfer Network model (PTN)

- $\{0, 1\}$ characters evolve on an **LGT network** $\boldsymbol{N = (V, E_S \cup E_T)}$
- **No homoplasy**: let V_1 be the set of nodes labeled 1. Then there is a unique $v \in V_1$ that reaches every node in $N[V_1]$.
- **No losses**: no $1 \rightarrow 0$ transition allowed in the support tree

Algorithmic questions [Lopez & L, 2022]:

1. **PTN recognition**: given a leaf-labeled LGT network N, can we extend the labeling into a PTN?
2. **PTN completion**: given a tree T with leaf labeled by a set of characters C, is there a (time-consistent) LGT network that explains C and whose support tree is T?

Perfect Transfer Network model (PTN)

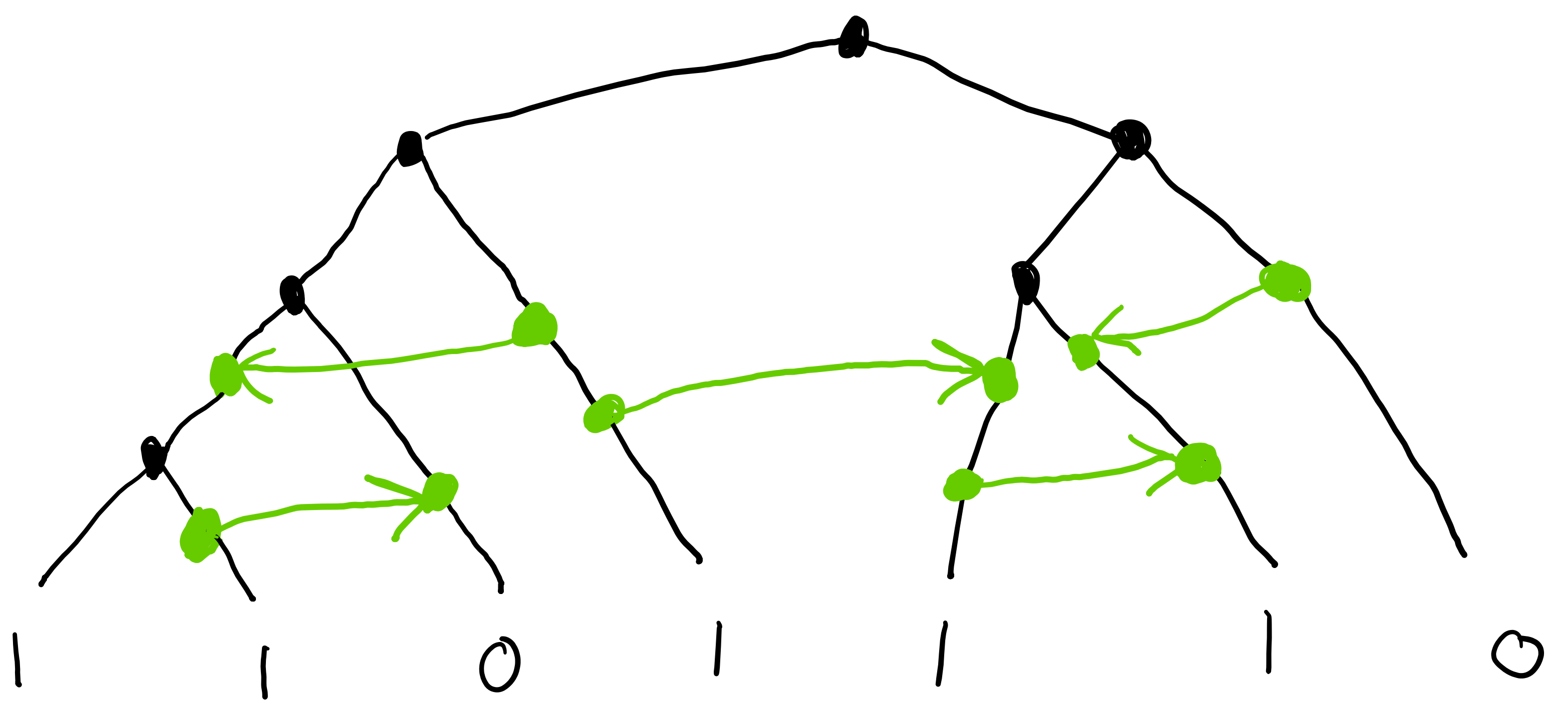
- $\{0, 1\}$ characters evolve on an **LGT network** $\boldsymbol{N = (V, E_S \cup E_T)}$
- **No homoplasy**: let V_1 be the set of nodes labeled 1. Then there is a unique $v \in V_1$ that reaches every node in $N[V_1]$.
- **No losses**: no $1 \rightarrow 0$ transition allowed in the support tree

Algorithmic questions [Lopez & L, 2022]:

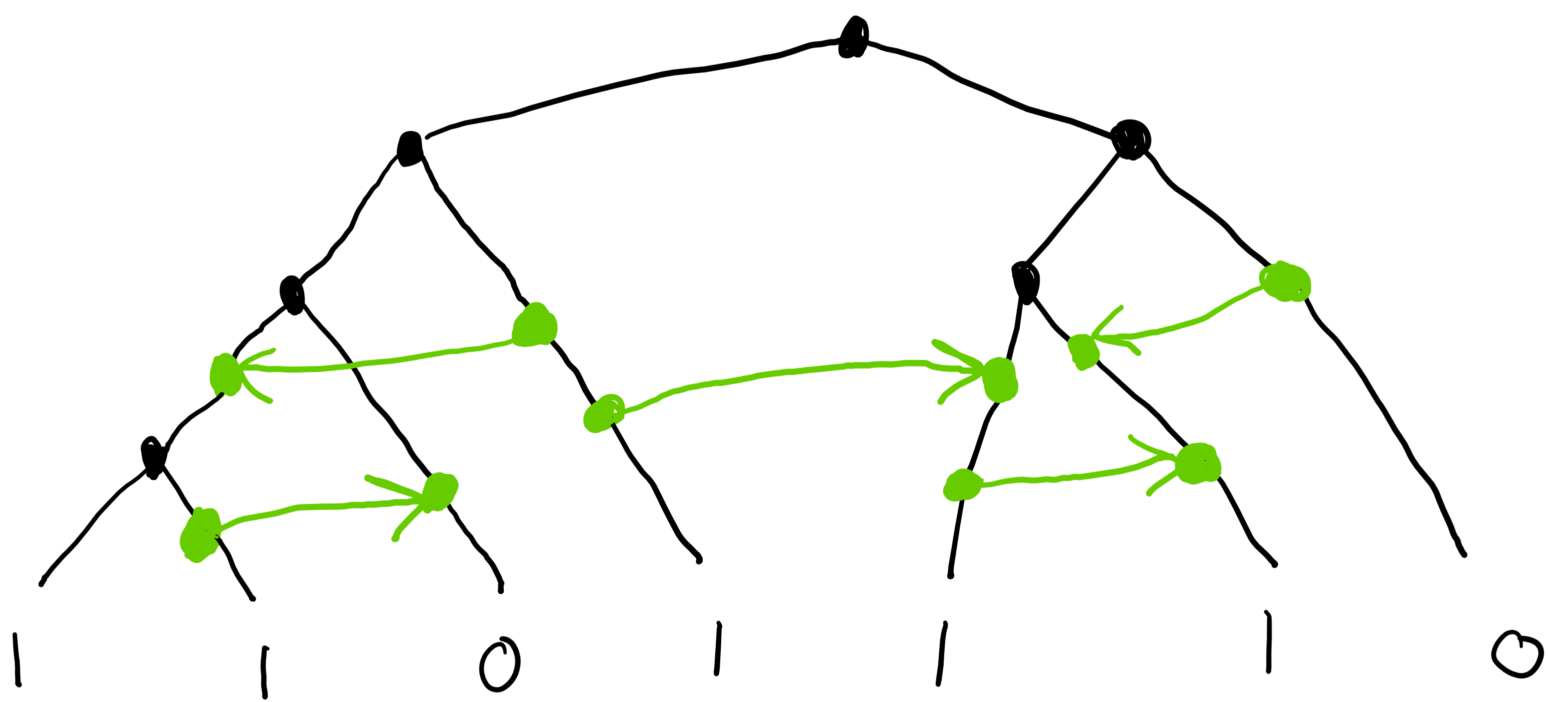
1. **PTN recognition**: given a leaf-labeled LGT network N, can we extend the labeling into a PTN?
2. **PTN completion**: given a tree T with leaf labeled by a set of characters C, is there a (time-consistent) LGT network that explains C and whose support tree is T?
3. **PTN reconstruction**: given a set of characters C, is there a PTN that explains C?

PTN recognition: given an LGT network N with leaves labeled in $\{0,1\}$, can we extend the labels to obtain a PTN?

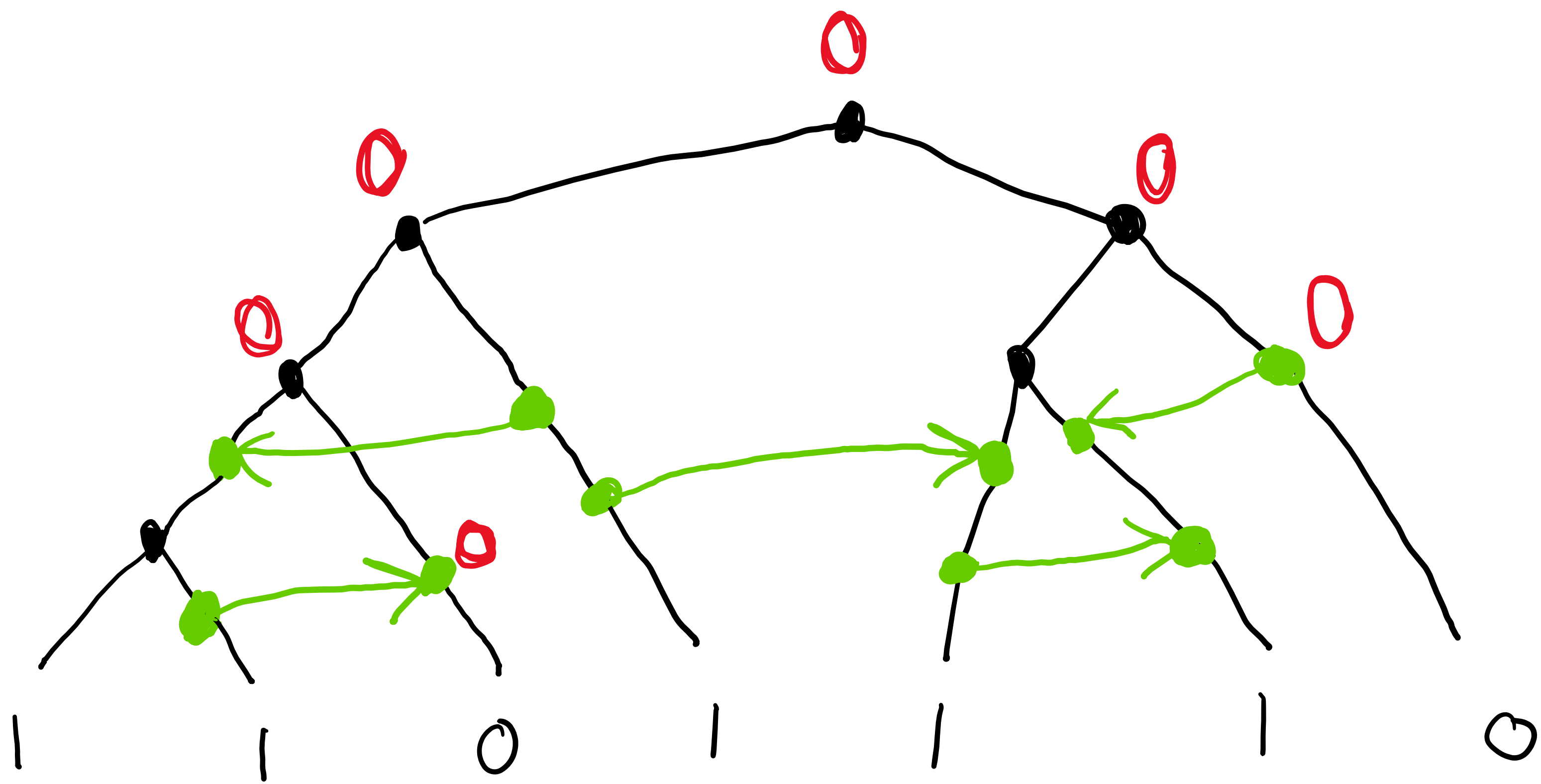
- **No homoplasy**: let V_1 be the set of nodes labeled 1. Then there is a unique $v \in V_1$ that reaches every node in $N[V_1]$.
- **No losses**: no $1 \rightarrow 0$ transition allowed in the support tree



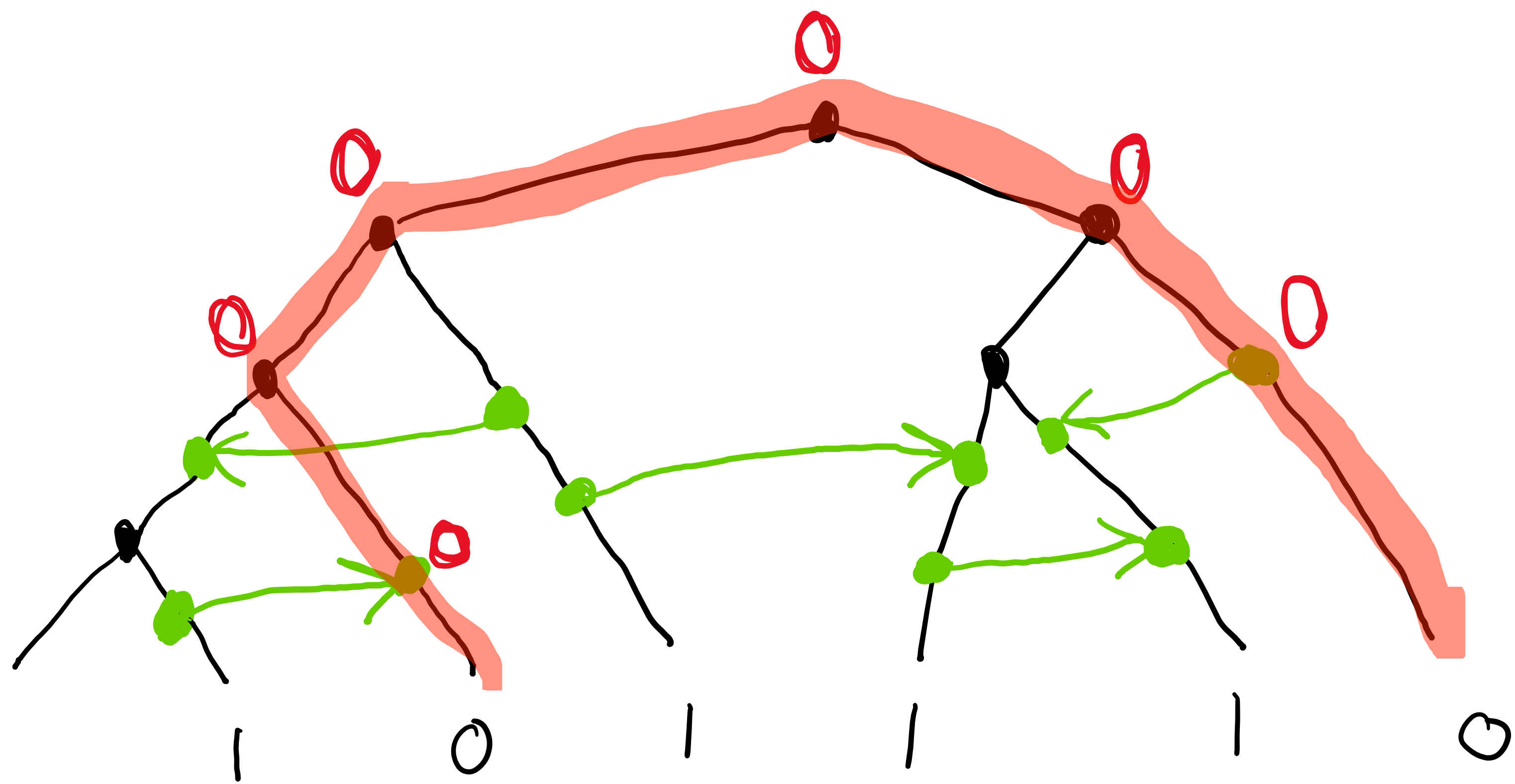
1) Add label 0 to every ancestor of a 0 leaf in the support tree. Let F be the set of nodes labeled 0.
2) Find a node v in G – F that reaches every leaf labeled 1. (if fails, no solution)
3) Add label 1 to every node on a v-to-leaf path in G – F. Put a 0 on unlabeled nodes.



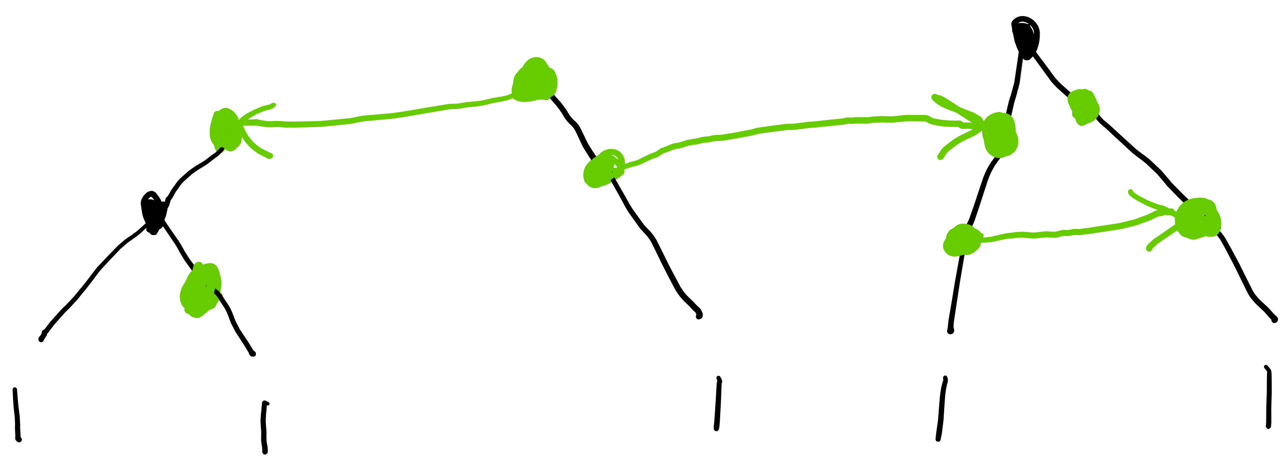
1) Add label 0 to every ancestor of a 0 leaf in the support tree. Let F be the set of nodes labeled 0.
2) Find a node v in G – F that reaches every leaf labeled 1. (if fails, no solution)
3) Add label 1 to every node on a v-to-leaf path in G – F. Put a 0 on unlabeled nodes.



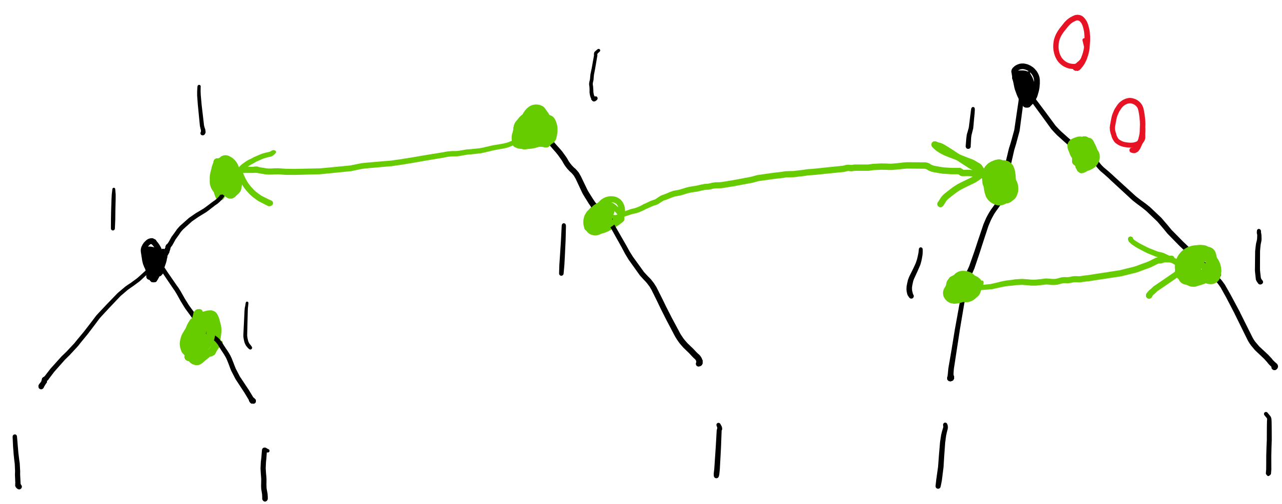
1) Add label 0 to every ancestor of a 0 leaf in the support tree. Let F be the set of nodes labeled 0.
2) Find a node v in G – F that reaches every leaf labeled 1. (if fails, no solution)
3) Add label 1 to every node on a v-to-leaf path in G – F. Put a 0 on unlabeled nodes.



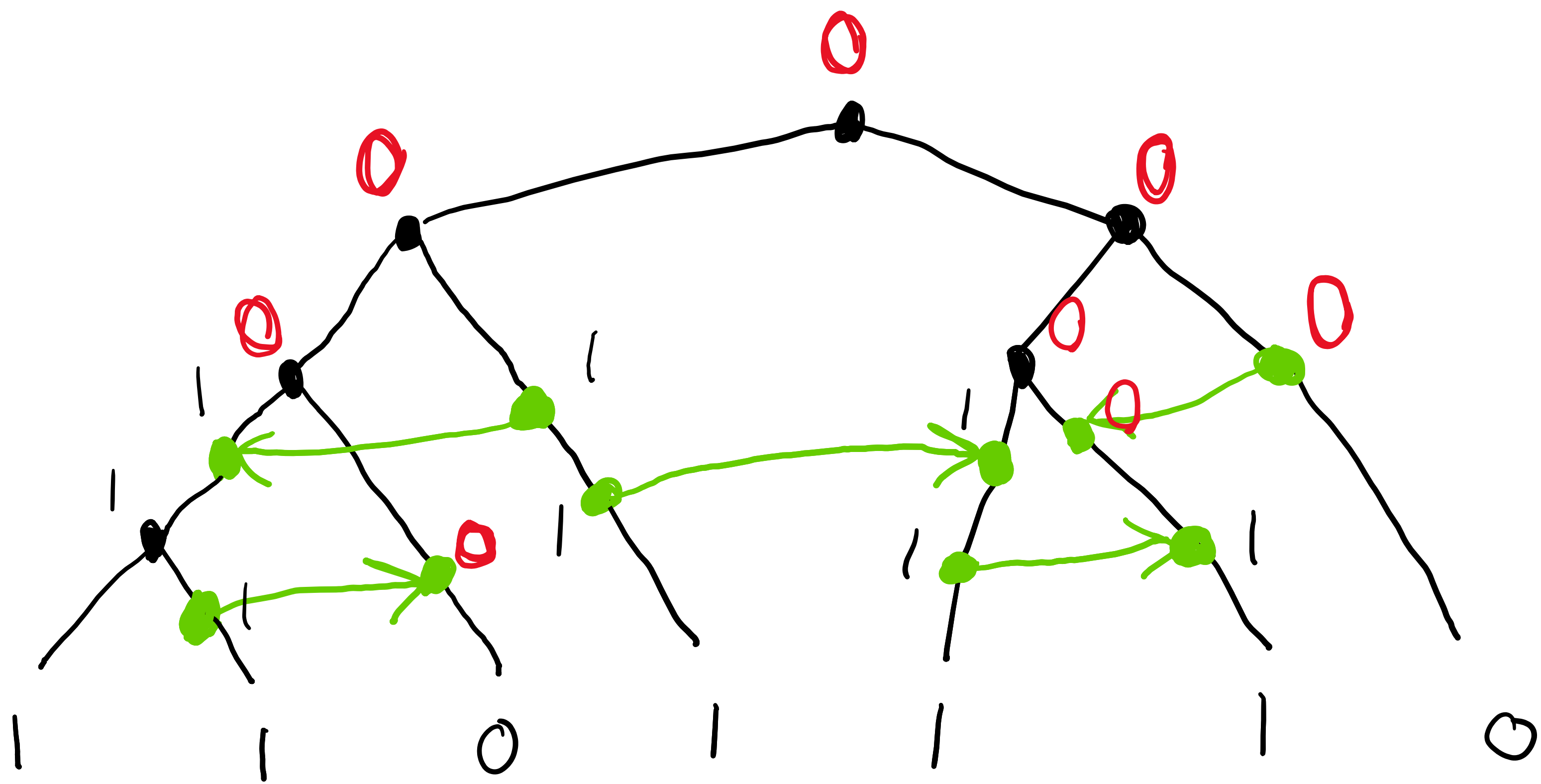
1) Add label 0 to every ancestor of a 0 leaf in the support tree. Let F be the set of nodes labeled 0.
2) Find a node v in G – F that reaches every leaf labeled 1. (if fails, no solution)
3) Add label 1 to every node on a v-to-leaf path in G – F. Put a 0 on unlabeled nodes.



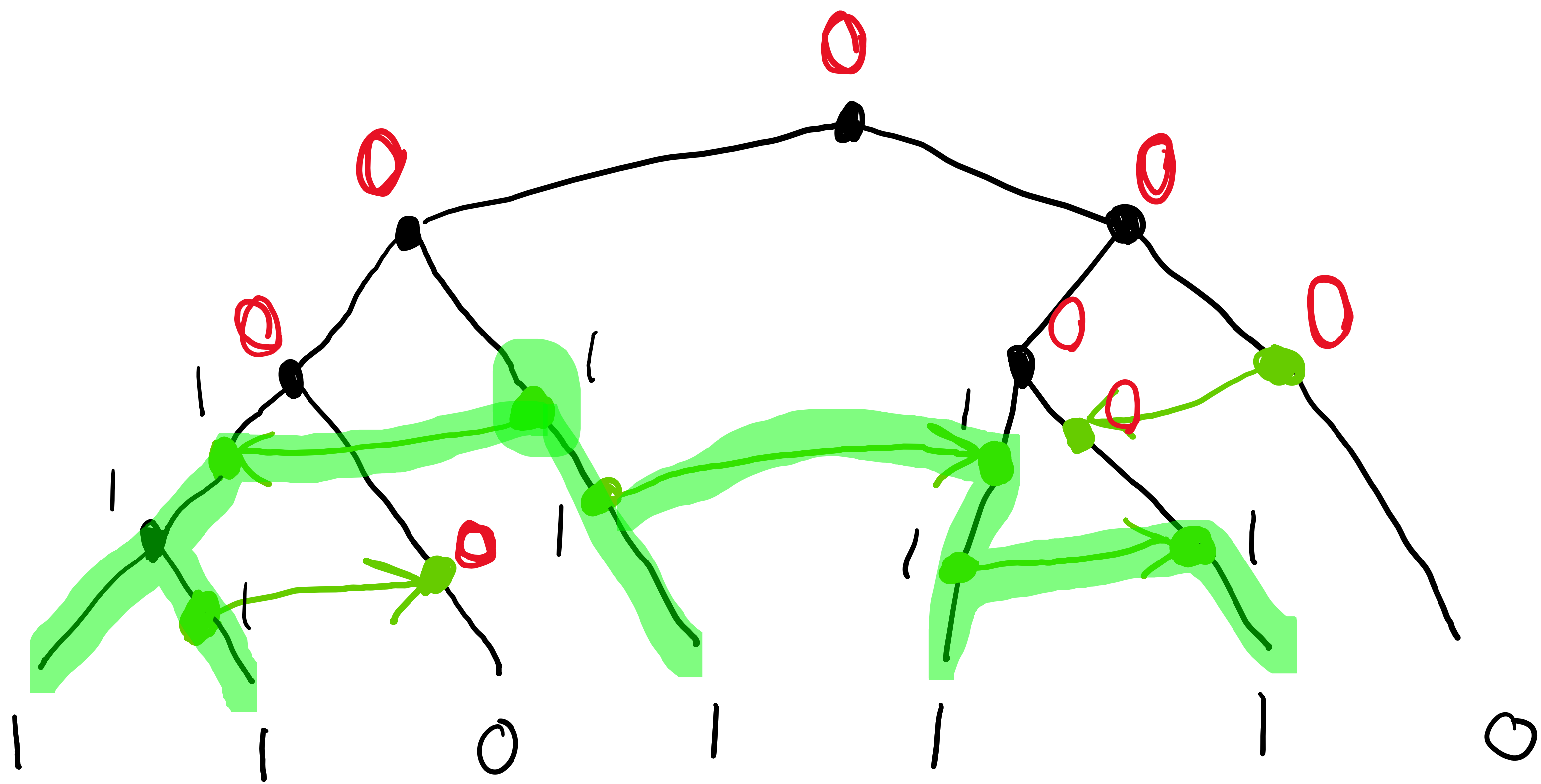
1) Add label 0 to every ancestor of a 0 leaf in the support tree. Let F be the set of nodes labeled 0.
2) Find a node v in G – F that reaches every leaf labeled 1. (if fails, no solution)
3) Add label 1 to every node on a v-to-leaf path in G – F. Put a 0 on unlabeled nodes.



1) Add label 0 to every ancestor of a 0 leaf in the support tree. Let F be the set of nodes labeled 0.
2) Find a node v in G – F that reaches every leaf labeled 1. (if fails, no solution)
3) Add label 1 to every node on a v-to-leaf path in G – F. Put a 0 on unlabeled nodes.



1) Add label 0 to every ancestor of a 0 leaf in the support tree. Let F be the set of nodes labeled 0.
2) Find a node v in G – F that reaches every leaf labeled 1. (if fails, no solution)
3) Add label 1 to every node on a v-to-leaf path in G – F. Put a 0 on unlabeled nodes.



PTN completion: given a tree T and leaves labeled by a set of $\{0,1\}$ characters, is there an LGT network that explains C and whose support tree is T?

PTN completion: given a tree T and leaves labeled by a set of $\{0,1\}$ characters, is there an LGT network that explains C and whose support tree is T?

The answer is **always yes**.

Even if:

1) we must use a given (lossless) labeling of the internal nodes.
2) the resulting network must be time-consistent.

PTN completion: given a tree T and leaves labeled by a set of $\{0,1\}$ characters, is there an LGT network that explains C and whose support tree is T?

The answer is **always yes**.

Even if:

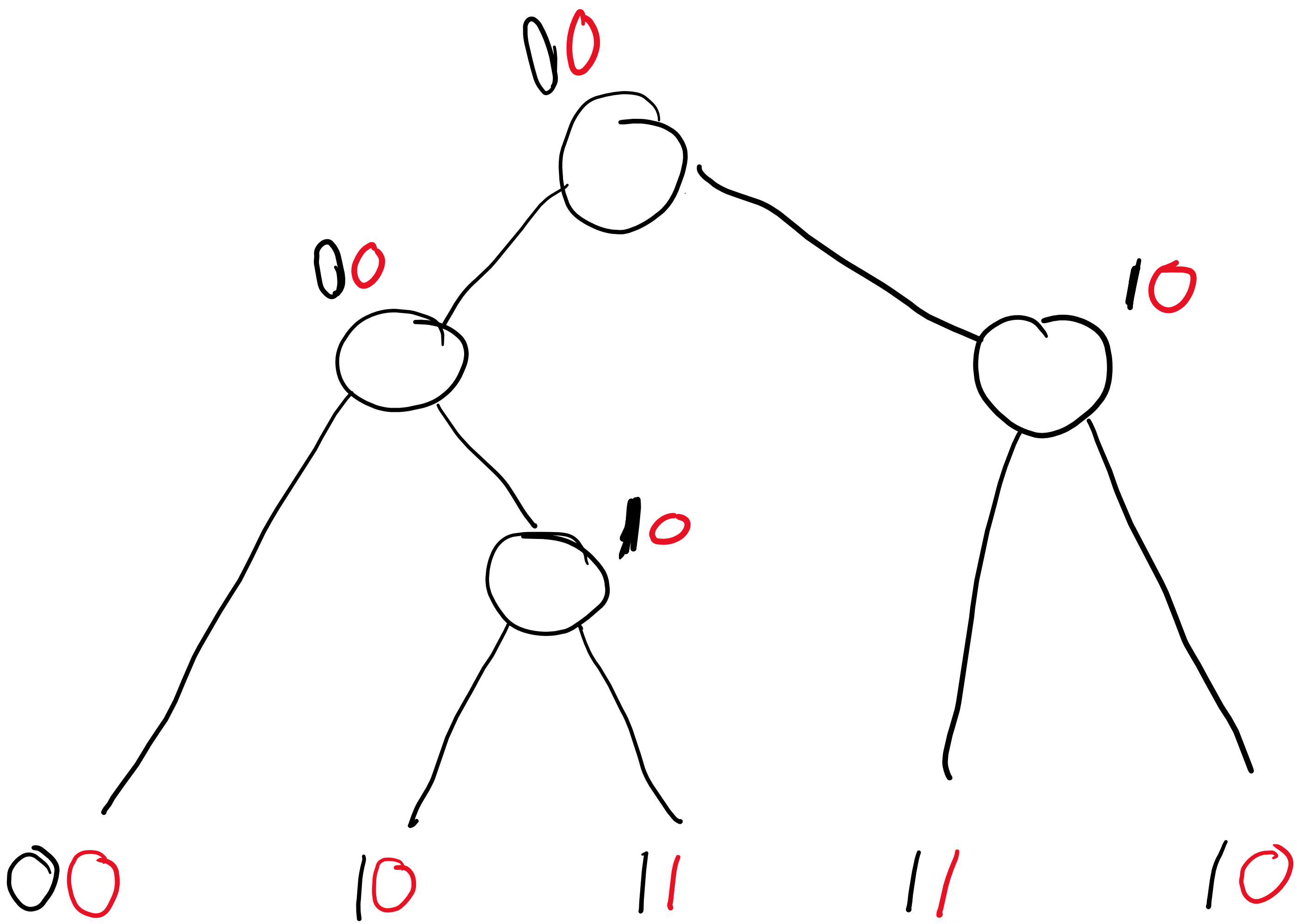
1) we must use a given (lossless) labeling of the internal nodes.
2) the resulting network must be time-consistent.

Time-consistent: nodes v can be assigned a time $t(v)$ such that

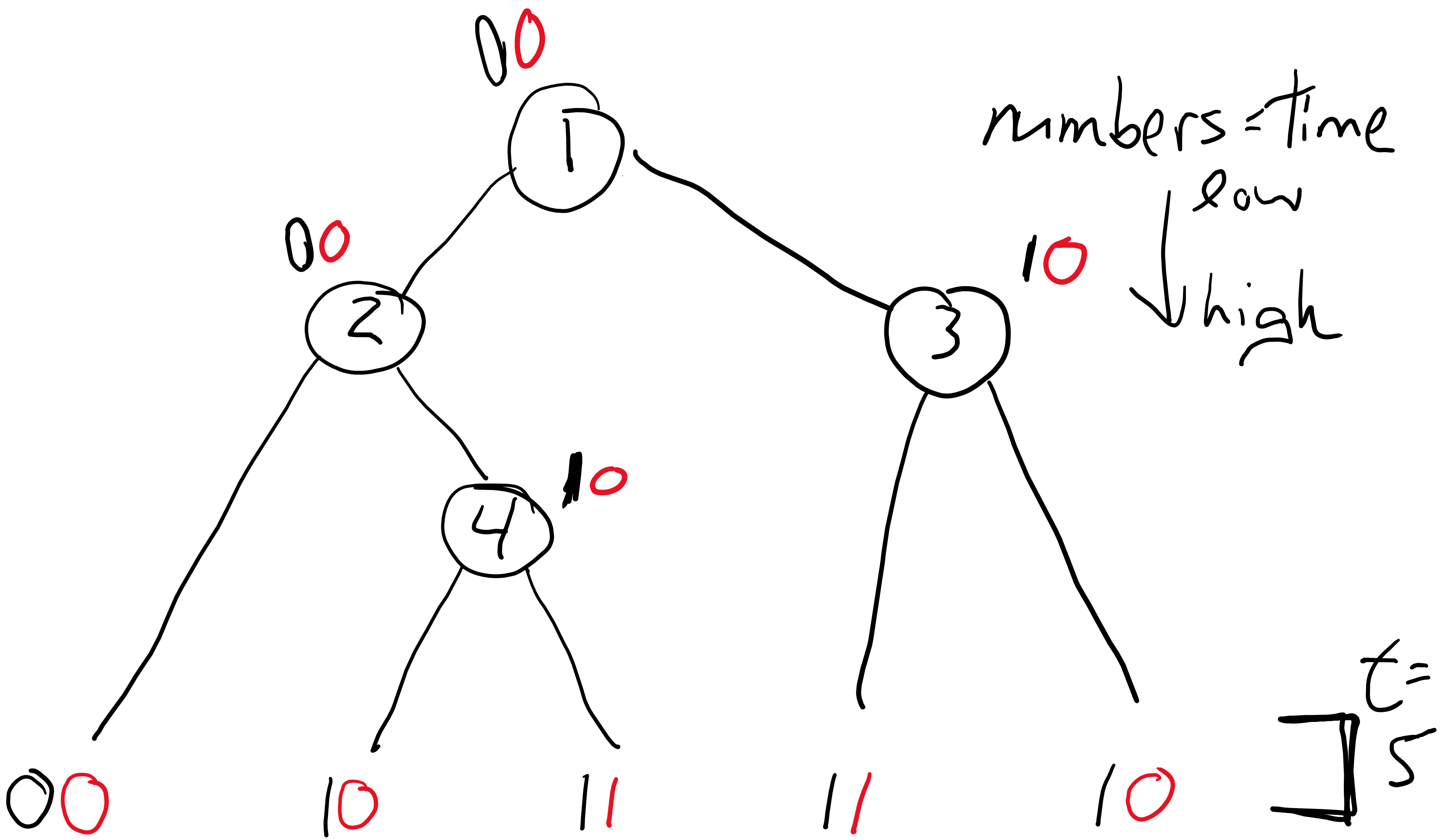
- $(u, v) \in E_S \Rightarrow t(u) < t(v)$
- $(u, v) \in E_T \Rightarrow t(u) = t(v)$

PTN completion: given a tree T and leaves labeled by a set of $\{0,1\}$ characters, is there a (time-consistent) LGT network that explains C and whose support tree is T?

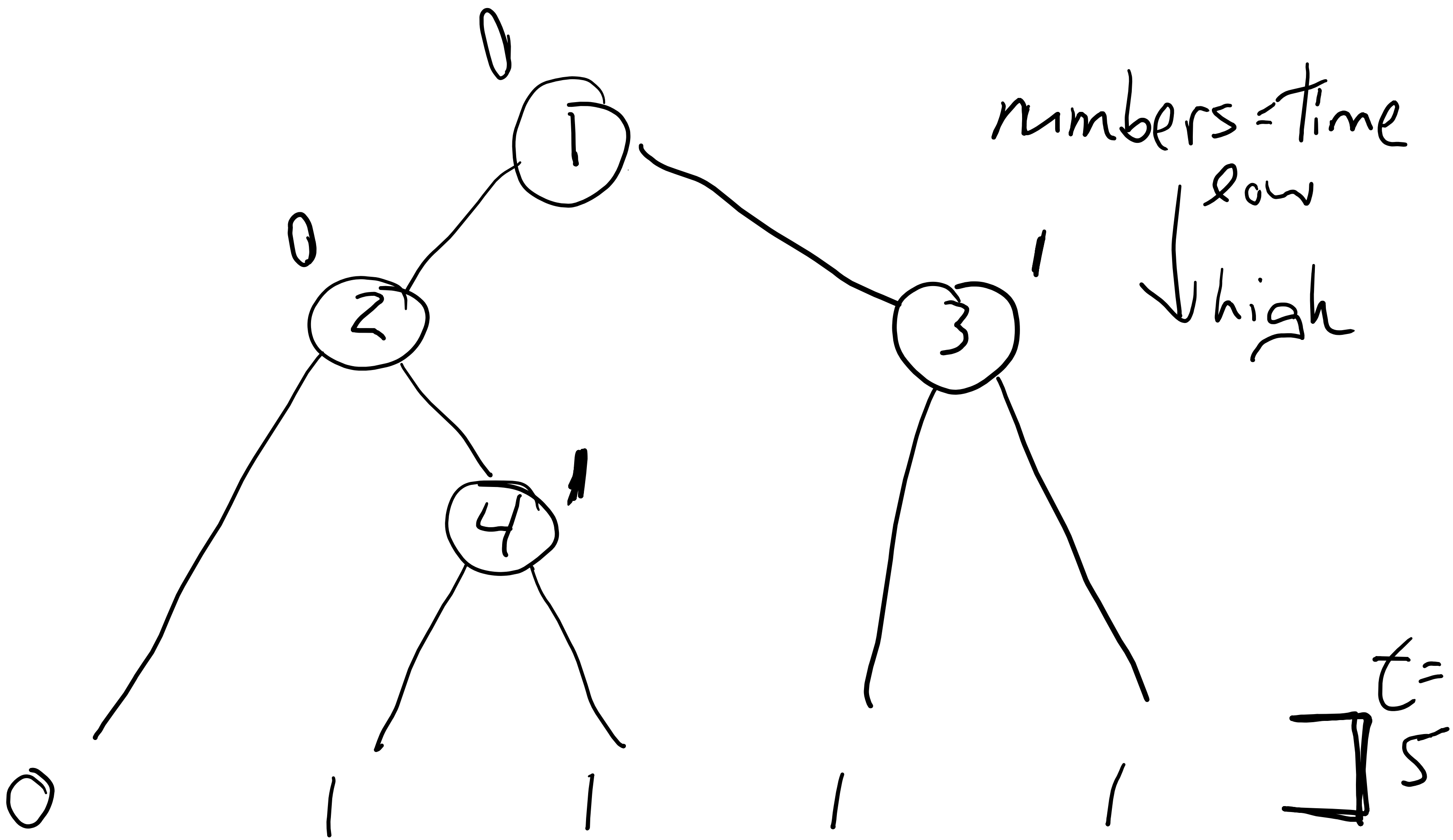
1) Assign arbitrary times to $V(T)$ in a time-consistent manner.
2) For each character $c \in C$:
   1) Order the « 1 » subtrees in decreasing time order of their roots, say $T_1, T_2, \dots, T_k$
   2) Add a transfer from an edge in T_i to the edge above the root of T_{i+1}, in a time-consistent manner.



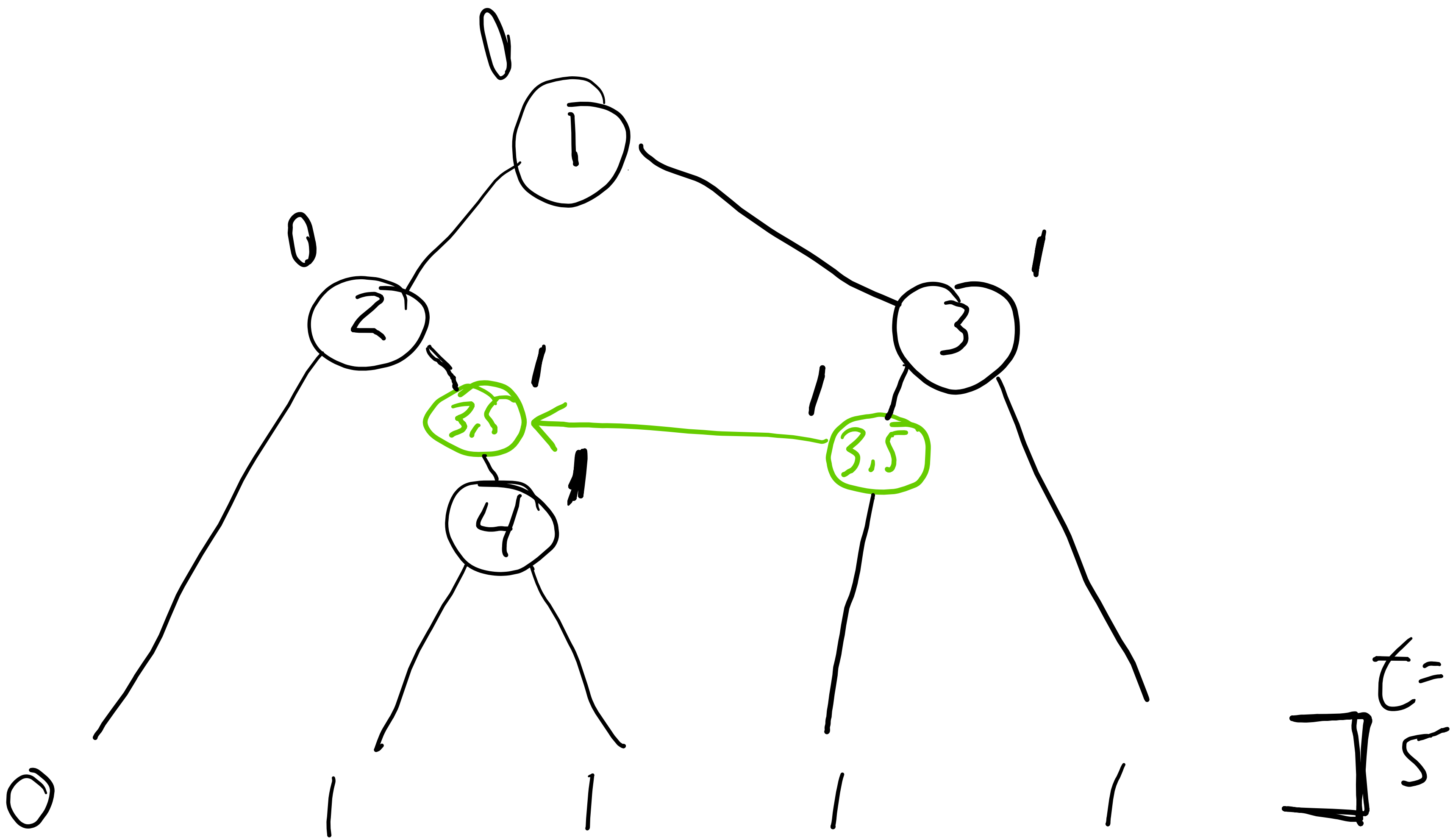
00
00
10
10
00
10
11
11
10



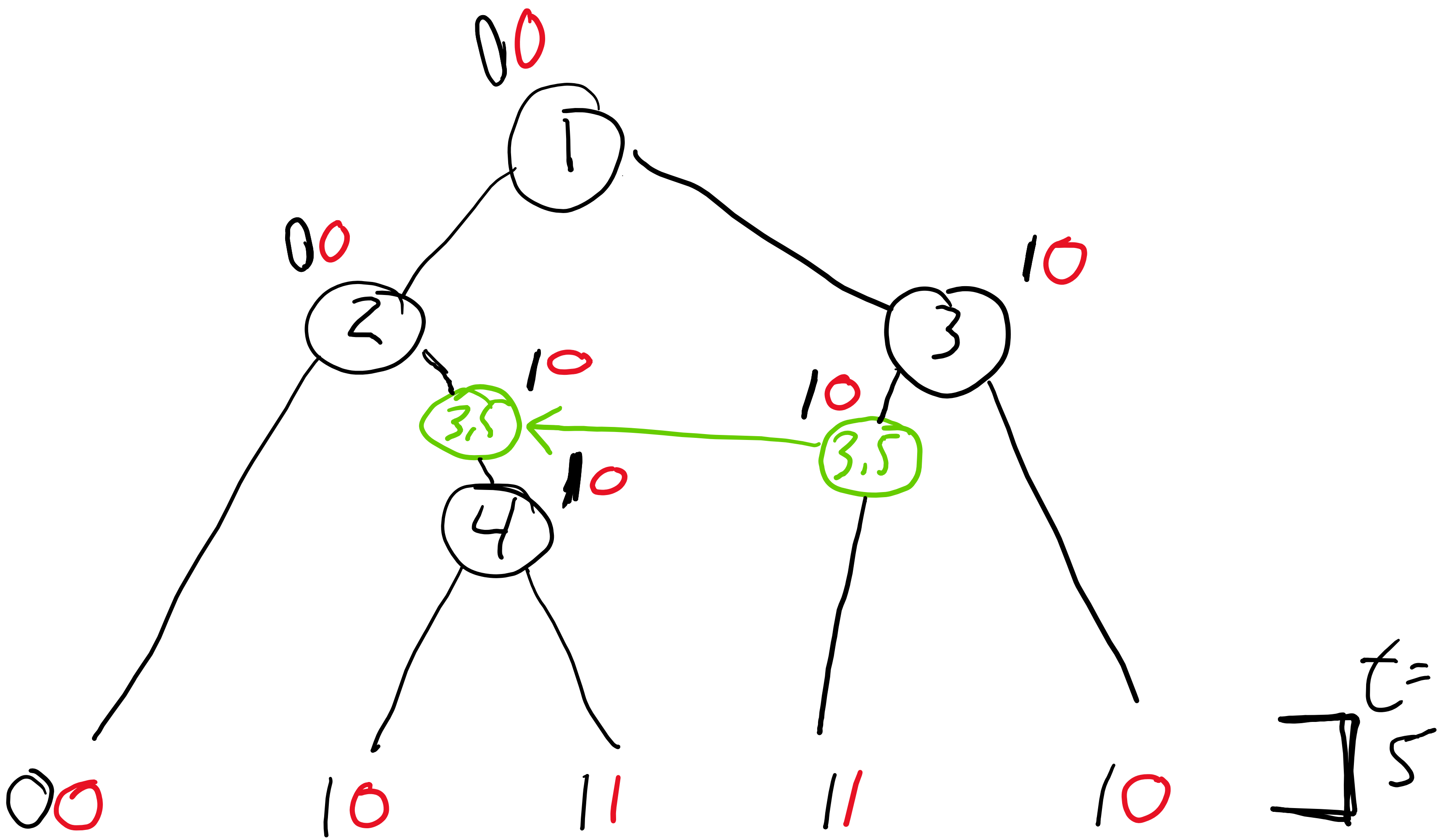
00
1
00
2
10
3
10
4
00
10
11
11
10
numbers = time
low
high
t = 5



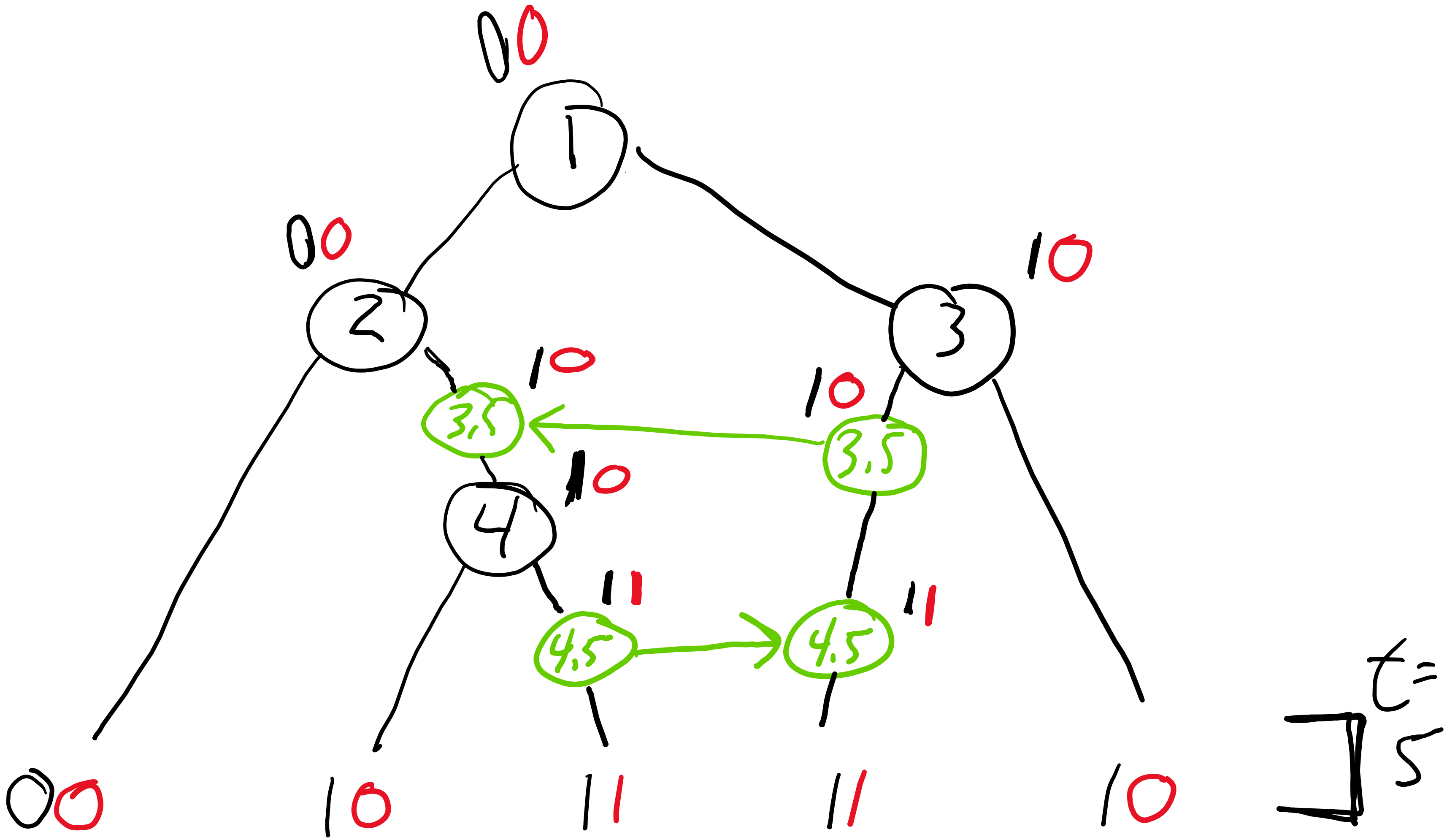
numbers = time
low
high
t = 5



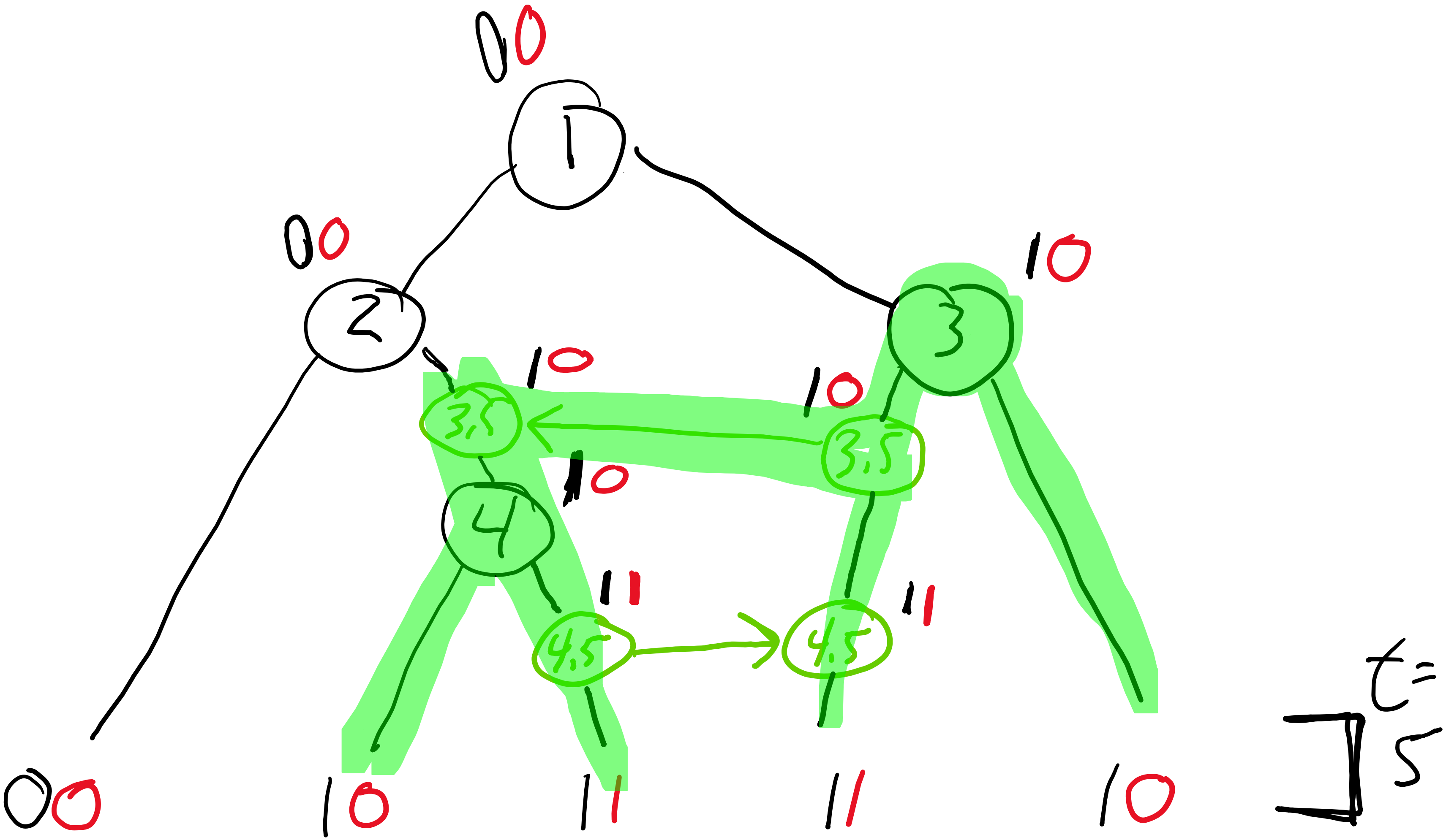
0
1
0
2
1
3
1
3,5
1
3,5
1
4
0
1
1
1
1
t=
5



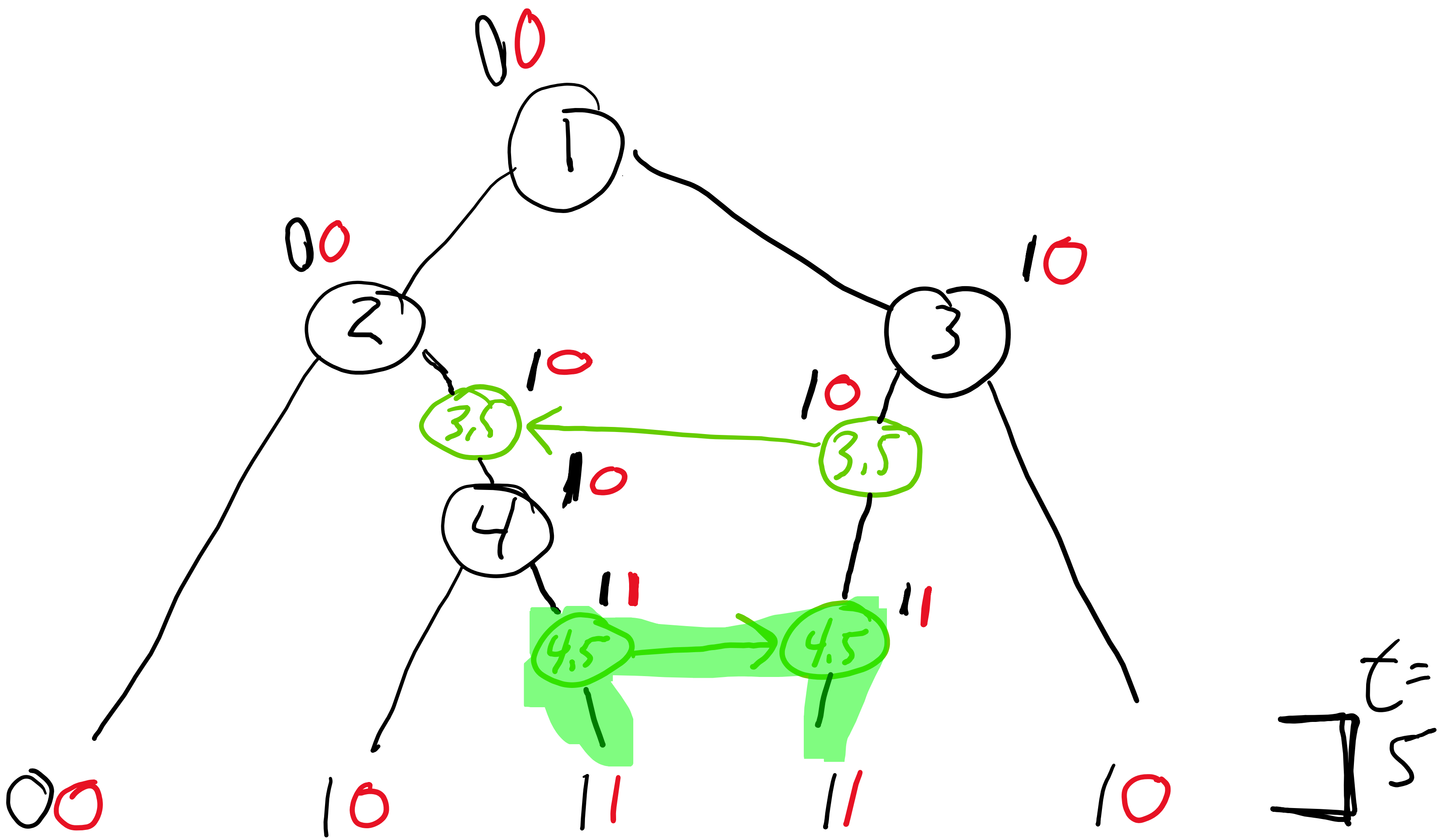
00
1
00
2
10
3
10
3,5
10
3,5
10
4
00
10
11
11
10
t=5



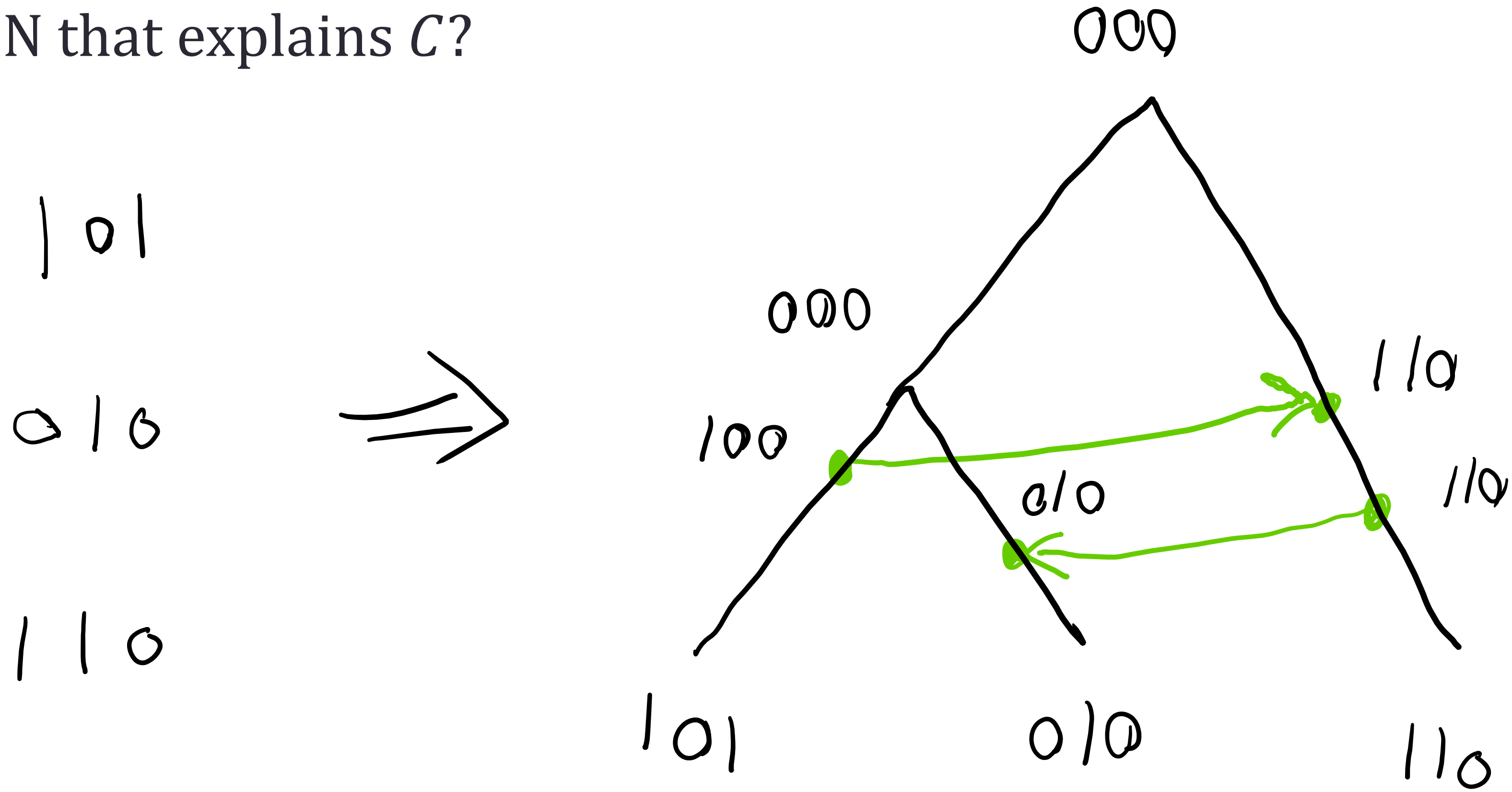
00
1
2
00
3
10
3,5
10
3,5
10
4
10
11
4,5
4,5
11
00
10
11
11
10
t=
5



00
1
00
2
10
3
10
3.5
10
3.5
10
4
11
4.5
11
4.5
00
10
11
11
10
t = 5


1
2
3
4
3,5
3,5
4,5
4,5
00
00
10
10
10
10
11
11
00
10
11
11
10
t=5

PTN reconstruction: given a set of characters C, is there a PTN that explains C?



PTN reconstruction: given a set of characters C, is there a PTN that explains C?

Yes, just build any tree and apply the previous algorithm.

PTN reconstruction: given a set of characters C, is there a PTN that explains C?

Yes, just build any tree and apply the previous algorithm.

Question: let C be a set of k characters (i.e. the $\{0,1\}$ vectors have length k). In the worst case, how many transfer edges are needed on a PTN whose leaves are labeled by C, with respect to k?

PTN reconstruction: given a set of characters C, is there a PTN that explains C?

Yes, just build any tree and apply the previous algorithm.

Question: let C be a set of k characters (i.e. the $\{0,1\}$ vectors have length k). In the worst case, how many transfer edges are needed on a PTN whose leaves are labeled by C, with respect to k?

Theorem: any set C of k characters can be explained using at most $2^k + (k-1)$ transfer edges.
Moreover, there is a set C of k characters that require at least $2^k/(3\text{k}) - 1$ transfer edges to be explained by any PTN.

Open problems

- **PTN recognition**: given N, extend labeling while using a minimum number of transfer edges.

Open problems

- **PTN recognition**: given N, extend labeling while using a minimum number of transfer edges.
- **PTN completion**: completing a tree to a PTN by adding a minimum number of transfer edges.
  - Unfinished draft of NP-hardness, will finish at some point. FPT algorithms?

Open problems

- **PTN recognition**: given N, extend labeling while using a minimum number of transfer edges.
- **PTN completion**: completing a tree to a PTN by adding a minimum number of transfer edges.
  - Unfinished draft of NP-hardness, will finish at some point. FPT algorithms?
- **PPN recognition**: If characters have s possible states and the set of possible $i \rightarrow j$ transitions is given as a directed graph D, when is PPN recognition NP-hard?
  - $s = 2$, $1 \rightarrow 0$ forbidden, easy
  - s unbounded, $D =$ complete directed graph, NP-hard
  - $D =$ directed tree: ???
  - FPT in parameter s?

Ongoing work

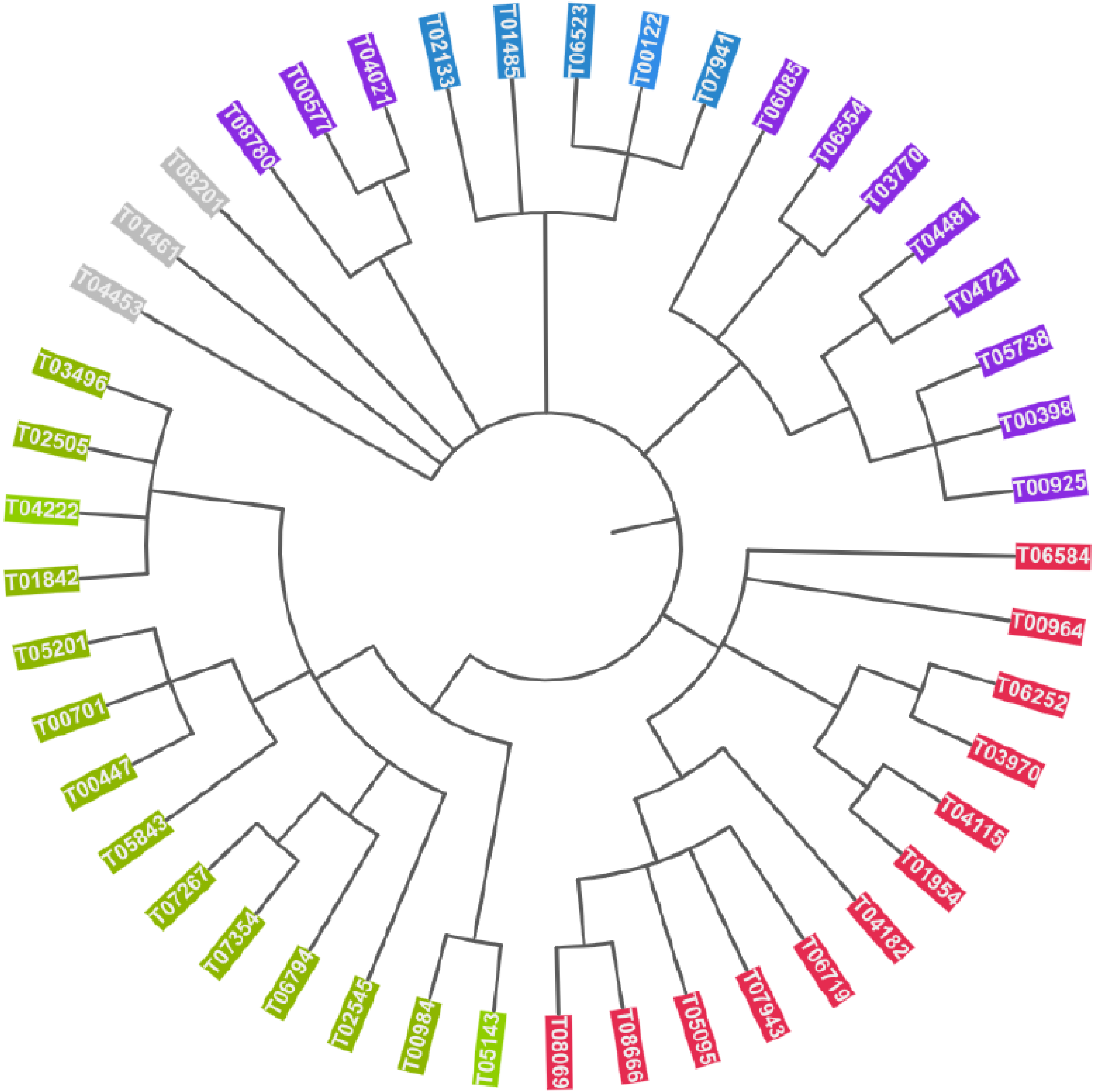
Galled PTNs

Which set of characters can be explained by a PTN that is a galled tree?

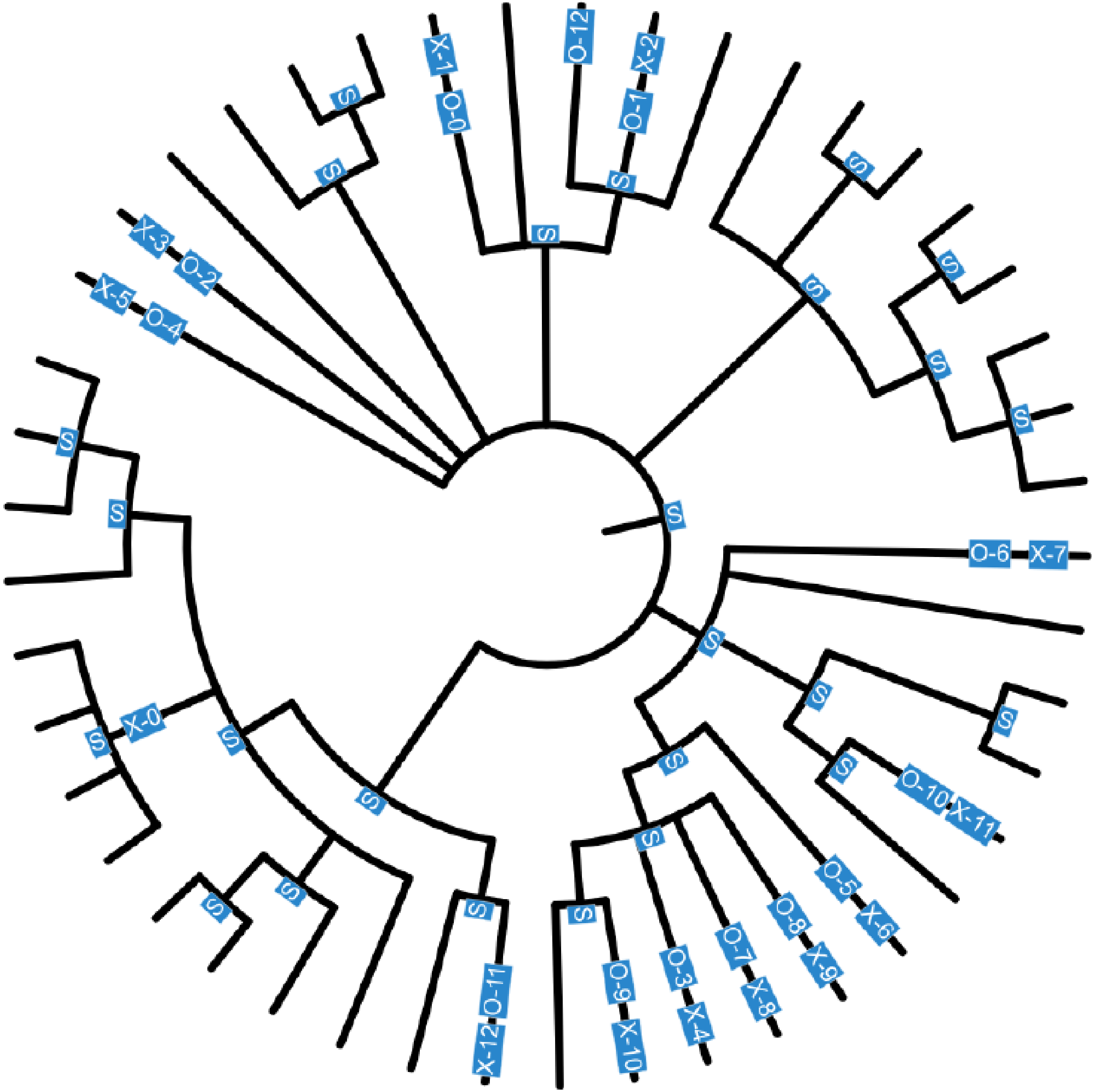
- Characters fit on a tree ⇔ laminar set family
- Characters fit on a Galled PTN ⇔ [?]
    - Tree incompatibility graph is bipartite + extra containment conditions

Attempt on real data

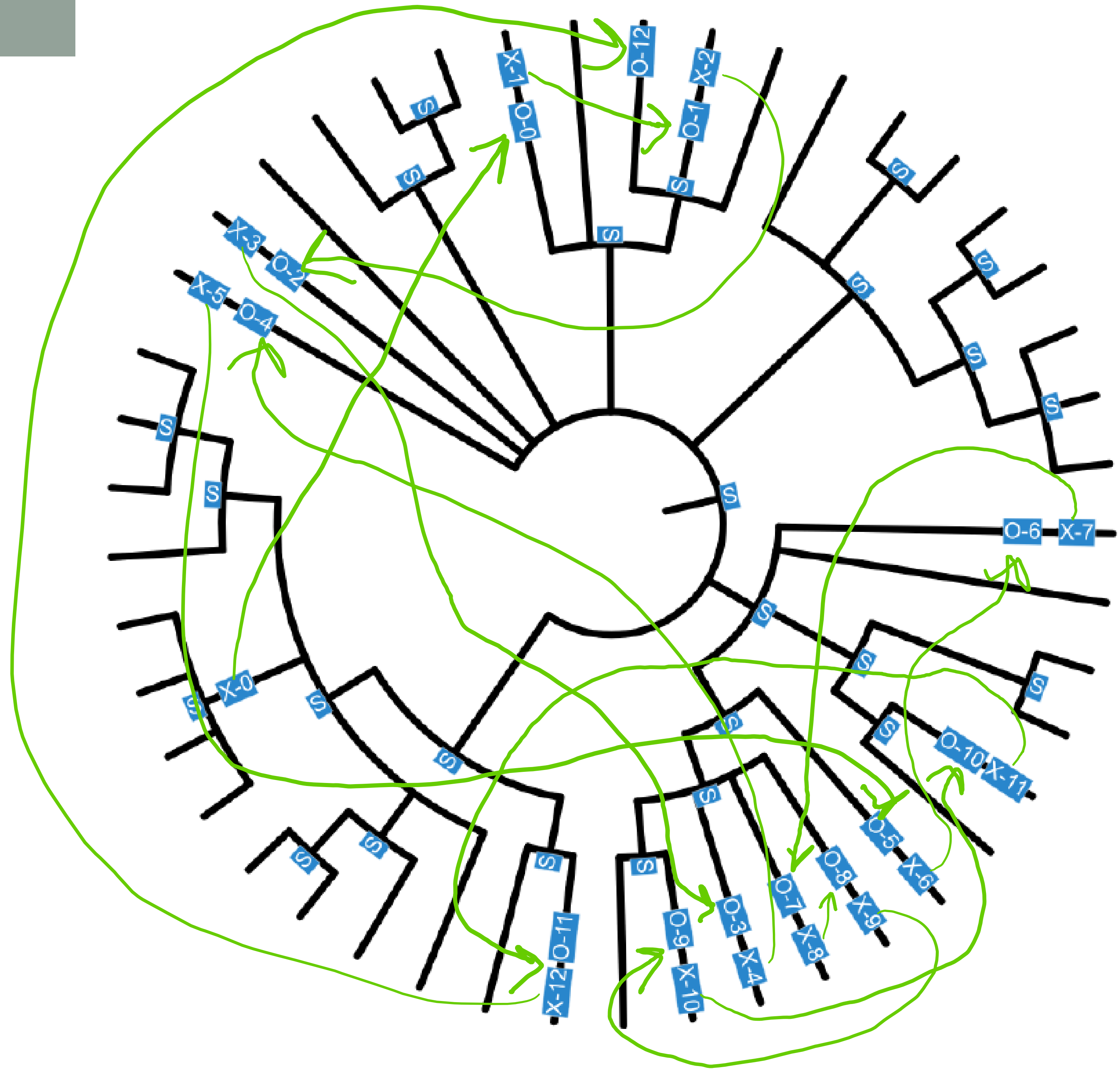
- A. Lopez implemented the tree completion algorithm and tested it on real data.
  - 45 bacterial species
  - ~50 characters = expression data from KEGG database



T08780
T00577
T04021
T02133
T01485
T06523
T00122
T07941
T06085
T06554
T03770
T04481
T04721
T05738
T00398
T00925
T06584
T00964
T06252
T03970
T04115
T01954
T04182
T06719
T07943
T05095
T08666
T08069
T05143
T00984
T02545
T06794
T07354
T07267
T05843
T00447
T00701
T05201
T01842
T04222
T02505
T03496
T04453
T01461
T08201



X-1
O-0
O-12
X-2
O-1
X-3
O-2
X-5
O-4
O-6
X-7
X-0
O-10
X-11
O-5
X-6
O-8
X-9
O-7
X-8
O-3
X-4
O-9
X-10
O-11
X-12
S

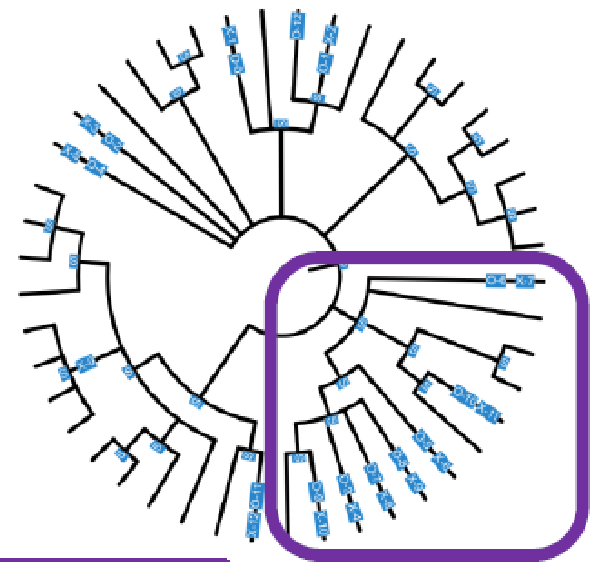


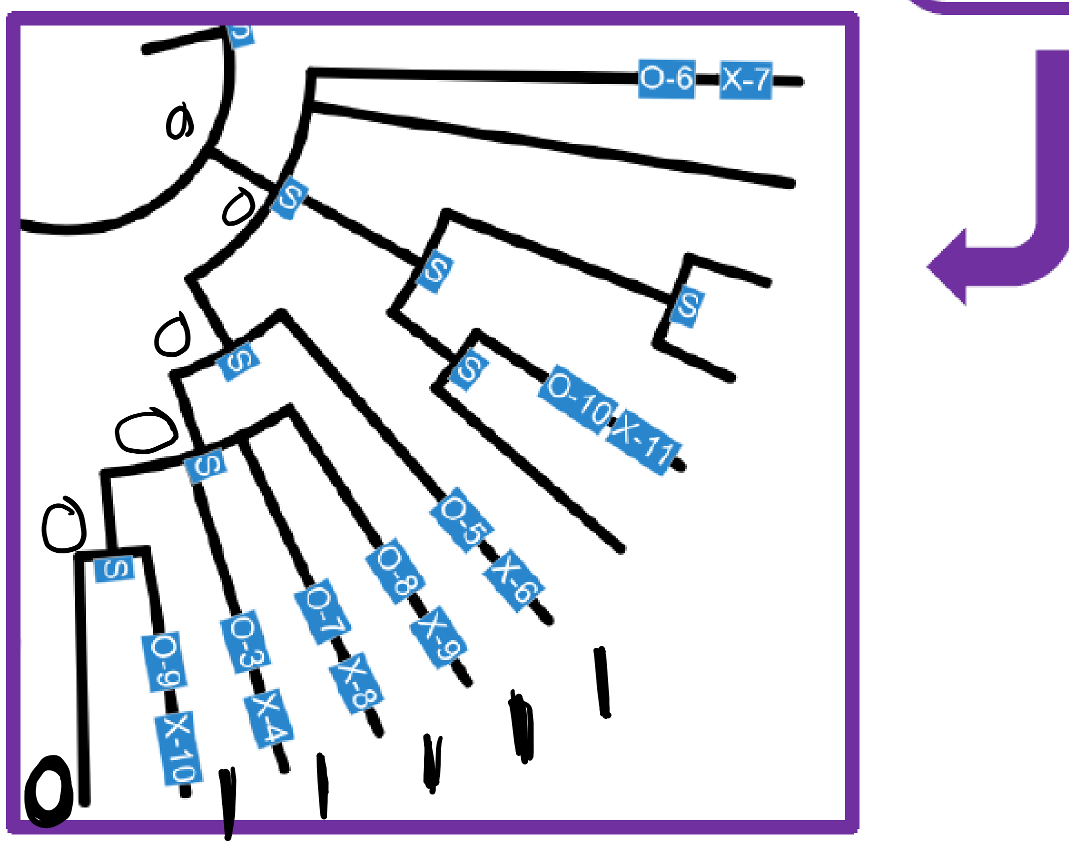
X-1
O-0
O-12
X-2
O-1
X-3
O-2
X-5
O-4
O-6
X-7
X-0
O-10
X-11
O-5
X-6
O-8
X-9
O-7
X-8
O-3
X-4
O-9
X-10
O-11
X-12
S

S Support node

O-n Head of transfer n

X-n Tail of transfer n





Tolerating losses

In practice, $1 \rightarrow 0$ appears to be frequent.

Extend PTNs to Dollo parsimony

- characters can be lost, but never regained

If losses are allowed, no need for transfers.

Assign a cost to transfers and losses.

Tolerating losses

- Given:
  - tree-based network N with leaves labeled in {0, 1}
  - transfer cost τ
  - loss cost λ
- Find:
  - a PTN labeling that minimizes $\tau \cdot (\#transfers) + \lambda \cdot (\#losses)$ is minimum

- Other problems in practice
  - Species trees are not always binary.
    - Polytomy resolution algorithms?
  - Inferred transfers highly dependent of chosen species timing.