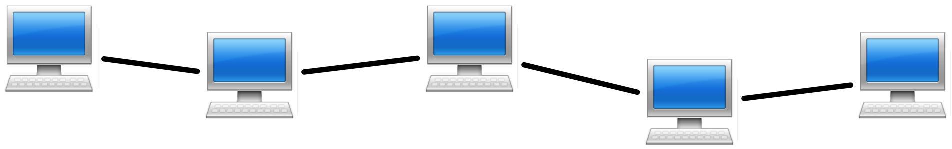
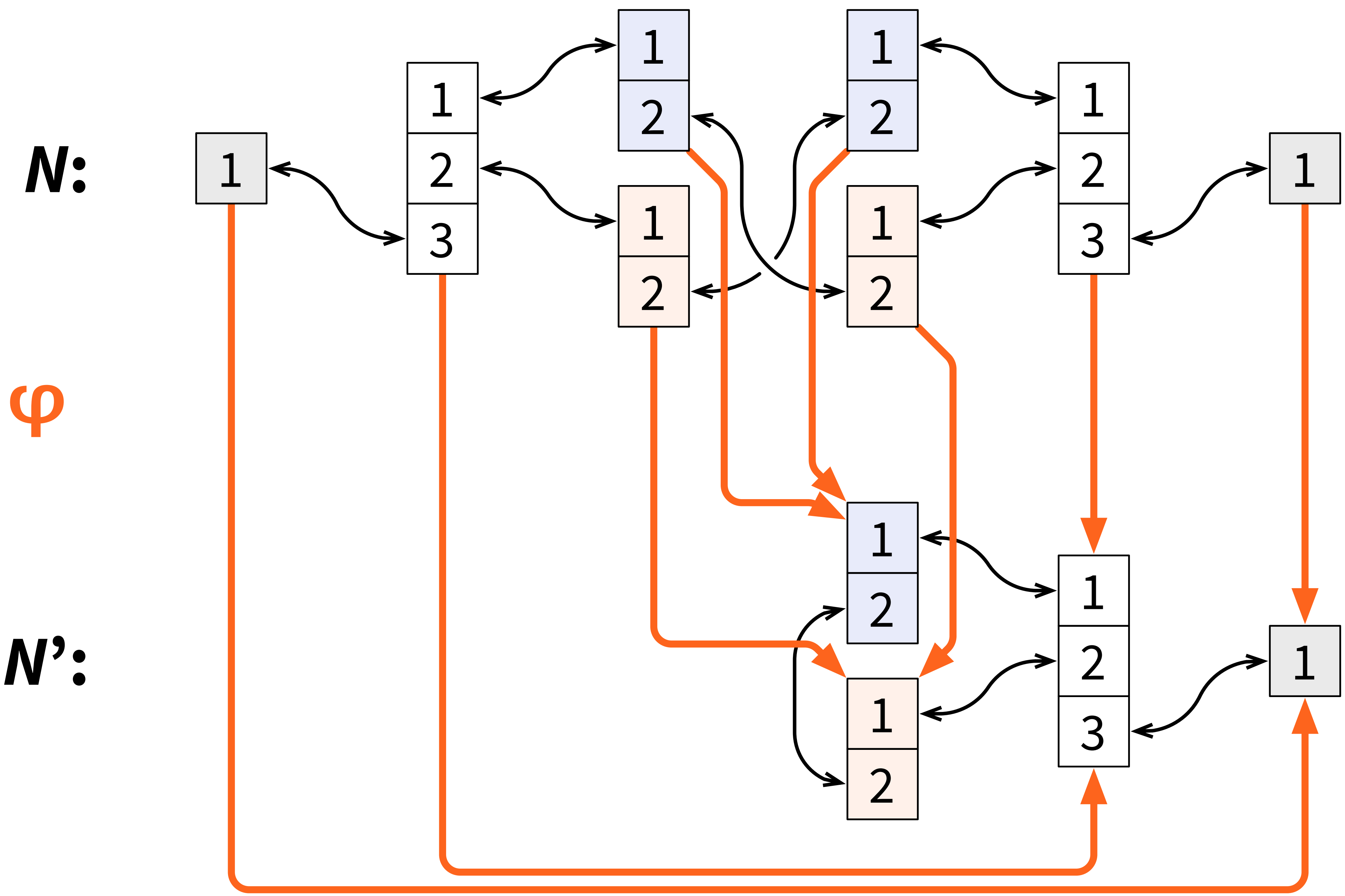
- **Weeks 1–2: informal introduction**
    - network = path 
- **Week 3: graph theory**
- **Weeks 4–7: models of computing**
    - what can be computed (efficiently)?
- **Weeks 8–11: lower bounds**
    - what cannot be computed (efficiently)?
- **Week 12: recap**

Recap:
Covering map

- **Networks $N = (V, P, p)$ and $N' = (V', P', p')$**
- **Surjection ϕ: $V \to V'$ that preserves inputs, degrees, connections, port numbers**
- **Theorem: preserves outputs for any PN-algorithm**

Covering map φ: *V* → *V'*

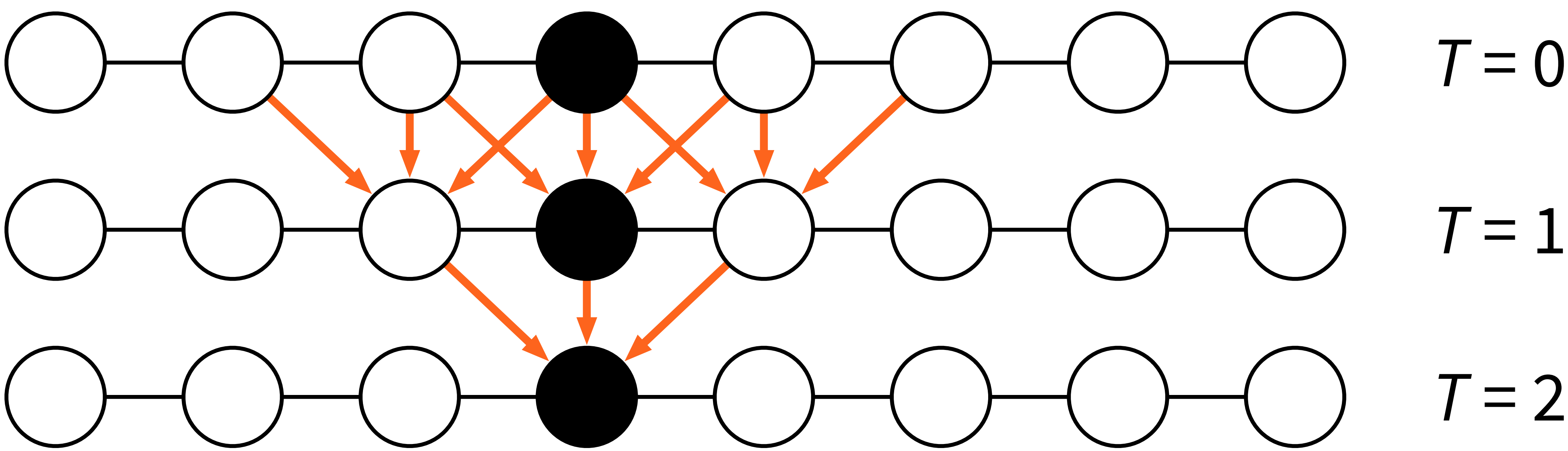


Week 9

– Local neighbourhoods

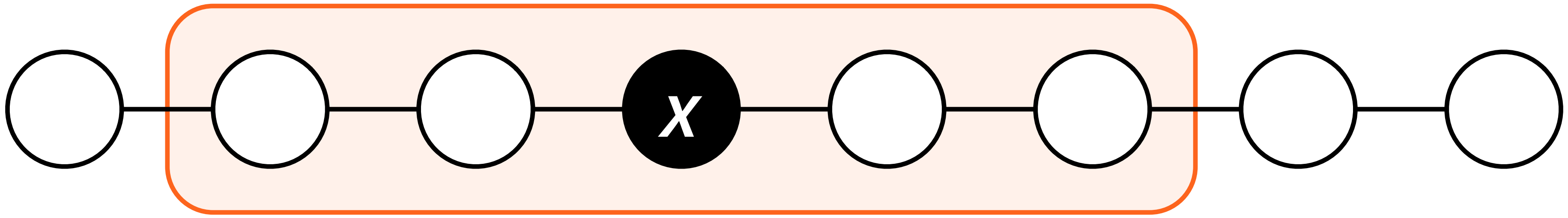
Recap:
Locality

- **State at time *T* only depends on initial information within distance *T***



Recap: **Locality**

- **After *T* communication rounds, node *x* can only know about other nodes that are within distance *T* from it**
    - distance = “number of hops”



Recap: **Locality**

- **Typical application:**
  - two possible worlds,
    need to produce ***different local outputs***
  - isomorphic local neighbourhoods
  - fast algorithm → ***same local output***

Example:
Distance to nearest leaf

- **Graph family:** trees
- **Problem:** each node outputs the distance to the ***nearest degree-1 node***
- **Prove:** not possible to solve in $o(n)$ time in the LOCAL model

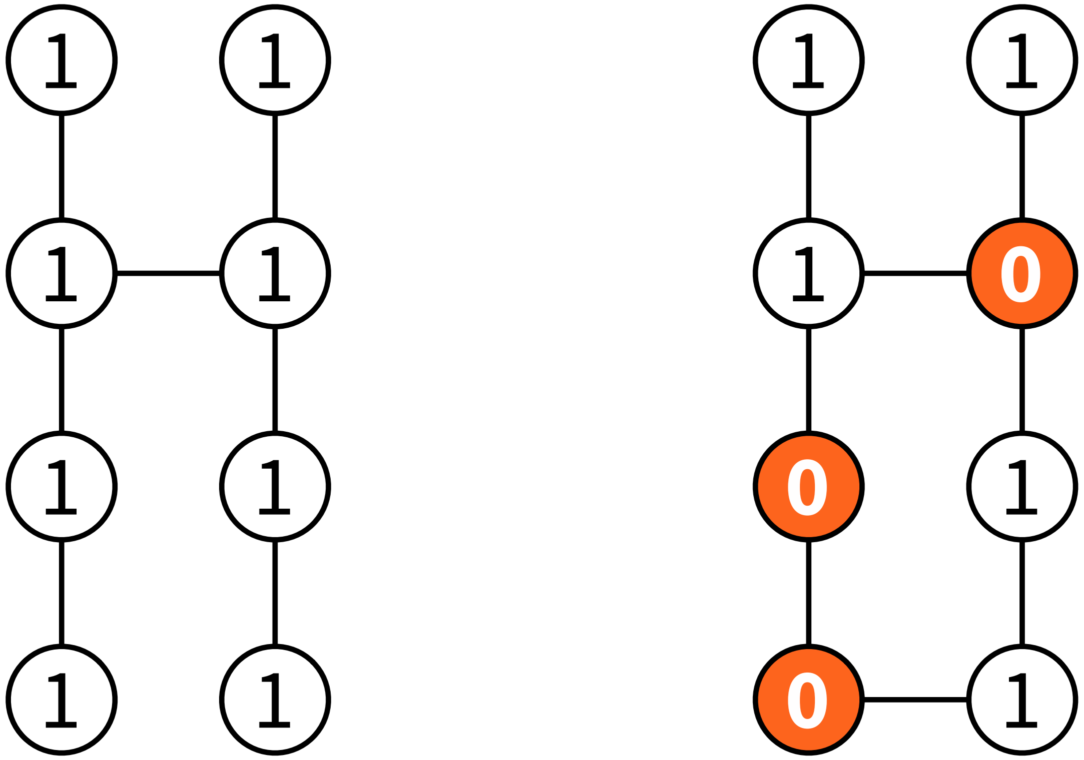
Common pitfalls

- **Don’t forget port numbering**
- **Don’t forget unique identifiers**
  - identifiers must also be ***sufficiently small***
- **Don’t forget local inputs**
  - did you e.g. ***assume that all nodes know n***?

Example:
Detecting forests

- **Problem:**
    - if G is a forest: **all nodes** output "***yes***"
    - otherwise: **at least one node** outputs "***no***"

Example:
Detecting forests



Example:
Detecting forests

- **Problem:**
    - if G is a forest: **all nodes** output "***yes***"
    - otherwise: **at least one node** outputs "***no***"
- **Can we solve this in PN model?**
- **How fast we can solve this in LOCAL model?**

Example:
Detecting forests

- **PN model:**
  - cannot be solved at all if we do not know n
  - can be solved in $O(n)$ rounds if we know n
  - cannot be solved in $o(n)$ rounds,
    even if we know n

Example:
Detecting forests

- **LOCAL model:**
  - can be solved in $O(n)$ rounds even if we do not know n
  - cannot be solved in $o(n)$ rounds even if we know n

Example:
Detecting forests

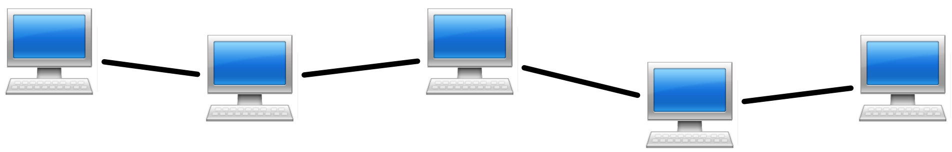
- **LOCAL model:**
    - what is the exact running time if we know n?
    - can we solve it in $\boldsymbol{n/2 + 2}$ rounds?
    - can we solve it in $\boldsymbol{n/2 - 2}$ rounds?

Example:
Detecting forests

- **LOCAL model:**
  - what is the exact running time if we know n?
  - can we solve it in $\boldsymbol{n/2 + 2}$ rounds?
  - can we solve it in $\boldsymbol{n/2 - 2}$ rounds?
  - what if we do not know n?

Summary

- **Two powerful lower-bound techniques:**
  - covering maps → PN, computability
  - locality → LOCAL, computational complexity
- **Sometimes we need to use both techniques together to argue about the PN model**

- **Weeks 1–2: informal introduction**
  - network = path 
- **Week 3: graph theory**
- **Weeks 4–7: models of computing**
  - what can be computed (efficiently)?
- **Weeks 8–11: lower bounds**
  - what cannot be computed (efficiently)?
- **Week 12: recap**