• **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
Week 8

– Covering maps
Covering map

- Networks $N = (V, P, p)$ and $N' = (V', P', p')$

- Surjection $\varphi: V \rightarrow V'$ that preserves inputs, degrees, connections, port numbers

- “Fools” any deterministic PN-algorithm: cannot distinguish between $N$ and $N'$
Networks $N$ and $N'$
Covering map $\varphi: V \to V'$
Preserves degrees

$N$: 1 2 3

$\varphi$

$N'$:
Preserves degrees

$N$: 1 2 3

$\varphi$

$N'$: 1 2 3
Preserves connections & port numbers

$N$:  

$\varphi$

$N'$:
Preserves connections & port numbers

$N$: 

$\varphi$

$N'$:
Preserves connections & port numbers

$N$: 

$\varphi$

$N'$:
Theorem: preserves outputs!
Covering map

- $\varphi$ covering map from $N$ to $N'$,
  A deterministic PN-algorithm

- Run $A$ on $N$ and $N'$

- Theorem: $\nu$ and $\varphi(\nu)$ always in the same state
Covering map

• Theorem: \( \nu \) and \( \varphi(\nu) \) always in the same state

• Proof: by induction
  • before round 1: map \( \varphi \) preserves local states
  • during round 1: map \( \varphi \) preserves messages
  • after round 1: map \( \varphi \) preserves local states
Covering map

• Theorem: \( \nu \) and \( \varphi(\nu) \) always in the same state

• Proof: by induction
  • before round 2: map \( \varphi \) preserves local states
  • during round 2: map \( \varphi \) preserves messages
  • after round 2: map \( \varphi \) preserves local states
Covering map

- **Theorem:** $v$ and $\varphi(v)$ always in the same state
- **Proof:** by induction
  - before round $t$: map $\varphi$ preserves local states
  - during round $t$: map $\varphi$ preserves messages
  - after round $t$: map $\varphi$ preserves local states
Before round $t$: local states agree
During round $t$: outgoing messages agree
During round $t$: incoming messages agree
After round $t$: local states agree
Covering map

- \( \varphi \) covering map from \( N \) to \( N' \),
  A deterministic PN-algorithm

- Run \( A \) on \( N \) and \( N' \)

- Theorem: \( v \) and \( \varphi(v) \) always in the same state

- Corollary: \( v \) and \( \varphi(v) \) have the same output
Application:
Path graphs

\[ G: \quad \circ \quad \cdash \quad \circ \]
Application: Path graphs

\[ G: \quad \text{N:} \quad 1 \quad 1 \quad N': \quad 1 \]

Diagram:

- Graph \( G \):
- Node labels: 1, 1
- 

- Node labels: 1
- Connections:
  - From 1 to 1
  - From 1 to 1

Diagram:

- Graph \( G \):
- Node labels: 1, 1
- Connections:
  - From 1 to 1

Diagram:

- Graph \( G \):
- Node labels: 1, 1
- Connections:
  - From 1 to 1

Diagram:

- Graph \( G \):
- Node labels: 1, 1
- Connections:
  - From 1 to 1

Diagram:

- Graph \( G \):
- Node labels: 1, 1
- Connections:
  - From 1 to 1
Application: Path graphs

$G$: 

\[
\begin{array}{c}
\text{\includegraphics[width=0.5\textwidth]{path_graph.png}}
\end{array}
\]
Application: Path graphs

$G$:  

$N$:  

$N'$:
Application: Cycle graphs

\[ G: \]
Application: Cycle graphs

$G:$

$N:$

$N':$
Application: Cycle graphs

- Cannot break symmetry in cycles

- Deterministic PN algorithms cannot find:
  - vertex colouring, edge colouring
  - maximal independent set, maximal matching
  - 1.99-approximation of minimum vertex cover
...
Covering maps and symmetry

$G$: 

$N$: 

$N'$:
Summary

• Covering map: preserves inputs, degrees, connections, port numbers

• Fools any deterministic PN-algorithm

• Can be used to prove that many problems cannot be solved at all in 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