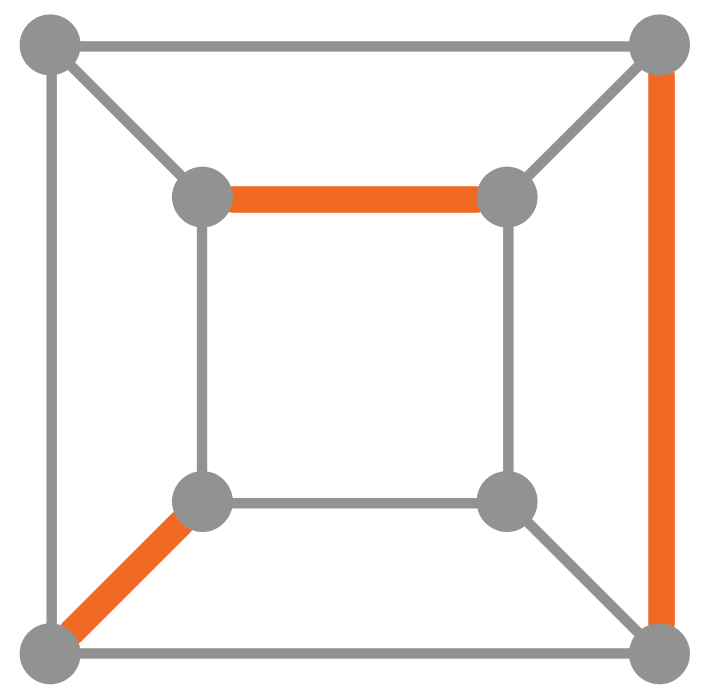
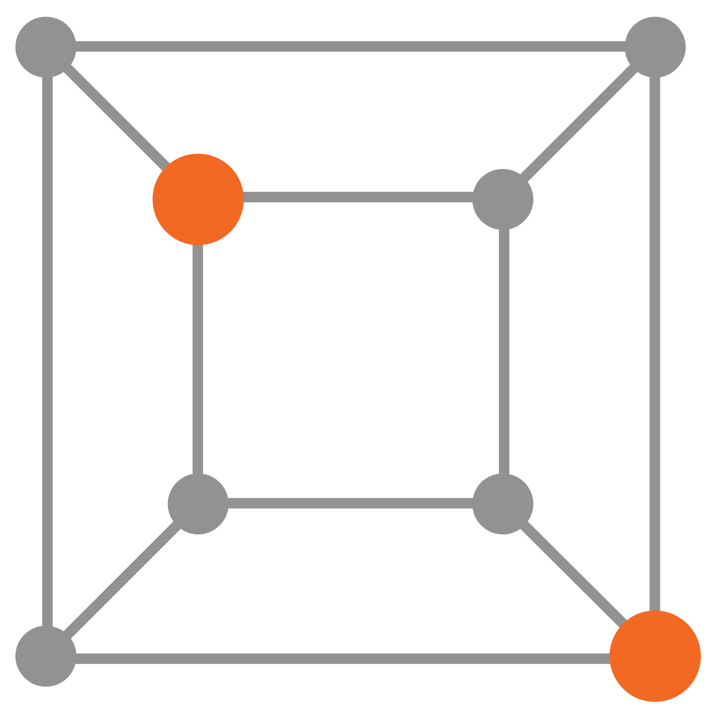
Lower Bounds for Maximal Matchings and Maximal Independent Sets

Alkida Balliu

Aalto University, Finland





Joint work with

Sebastian Brandt · ETH Zurich

Juho Hirvonen · Aalto University

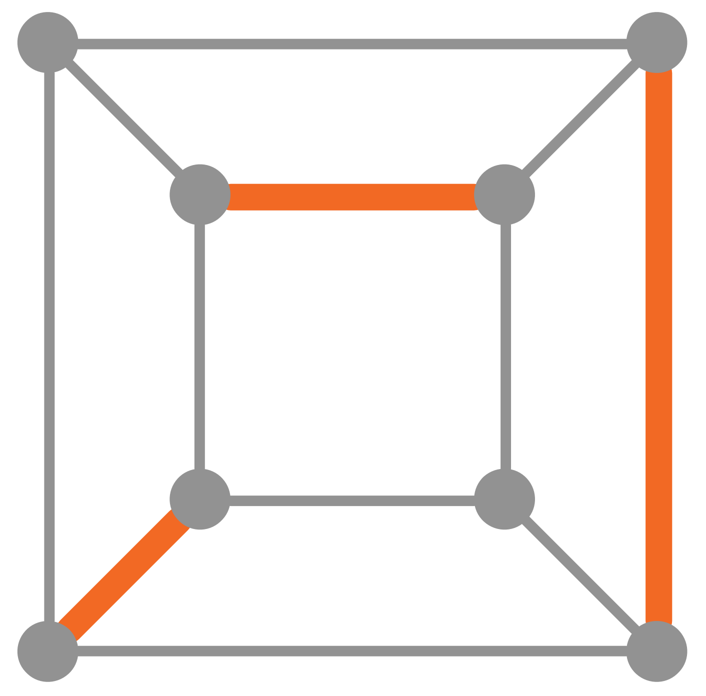
Dennis Olivetti · Aalto University

Mikaël Rabie · LIP6 - Sorbonne University

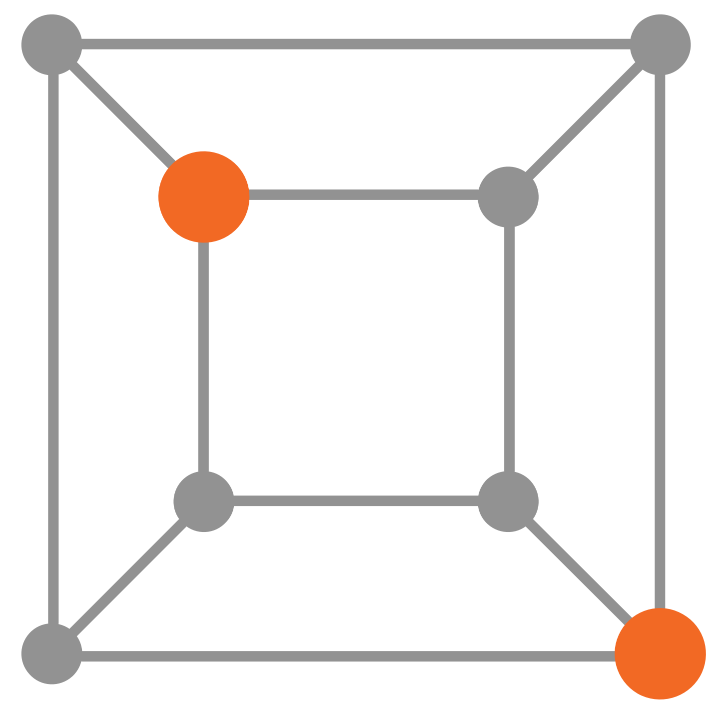
Jukka Suomela · Aalto University

Overview

Maximal matching

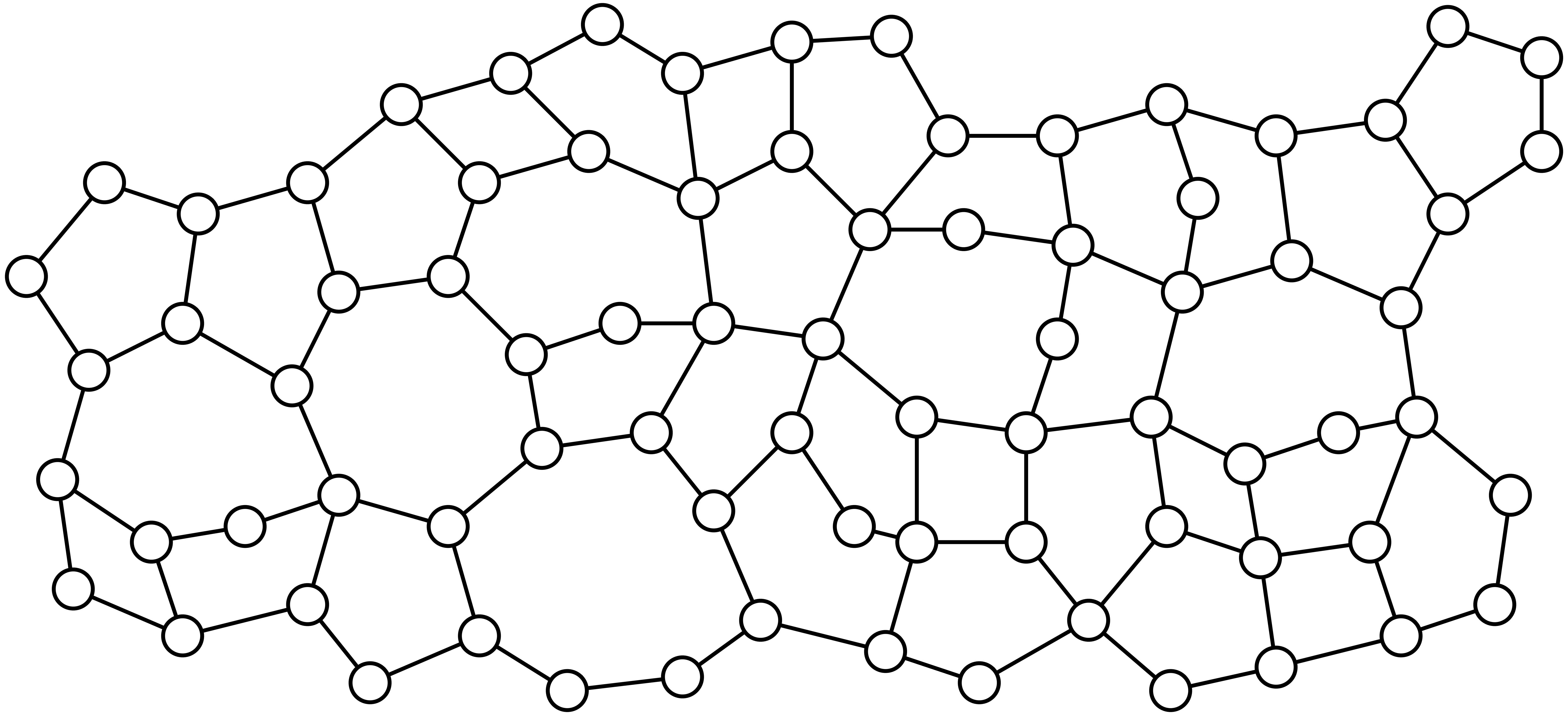


Maximal independent set



We will talk about **lower bounds** for solving these problems in the **distributed setting**

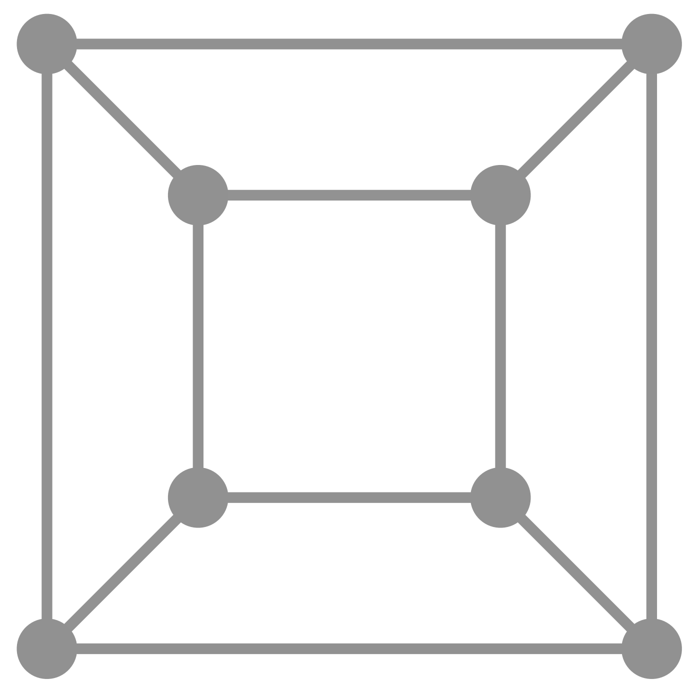
Distributed setting



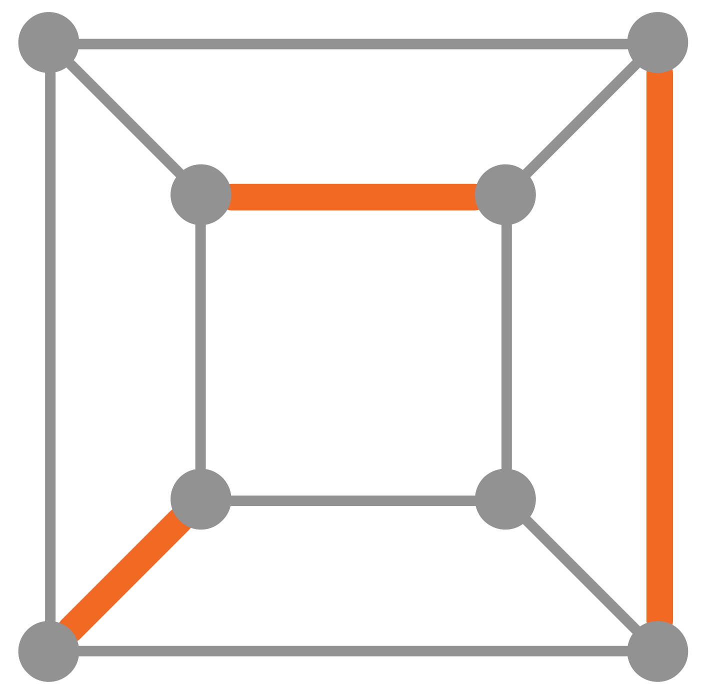
Graph = communication network; **synchronous** rounds; **time** = number of communication rounds

Maximal matching problem

Input



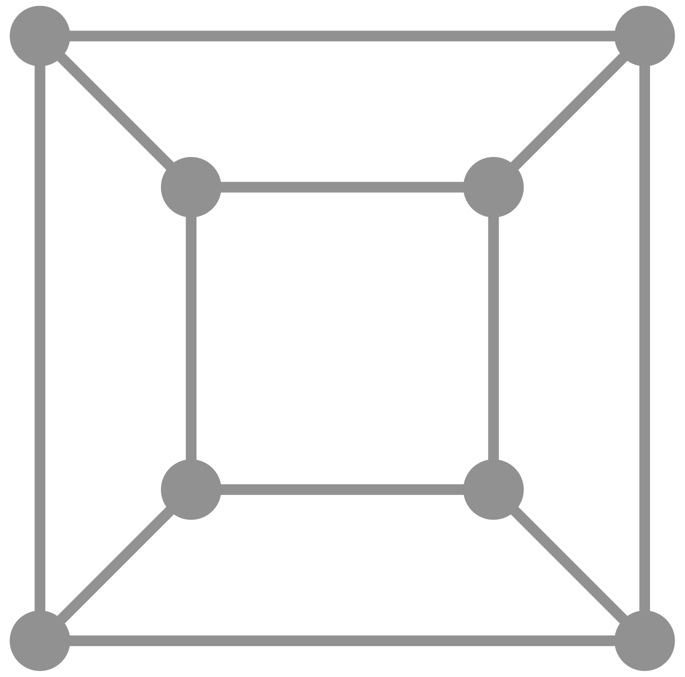
Output



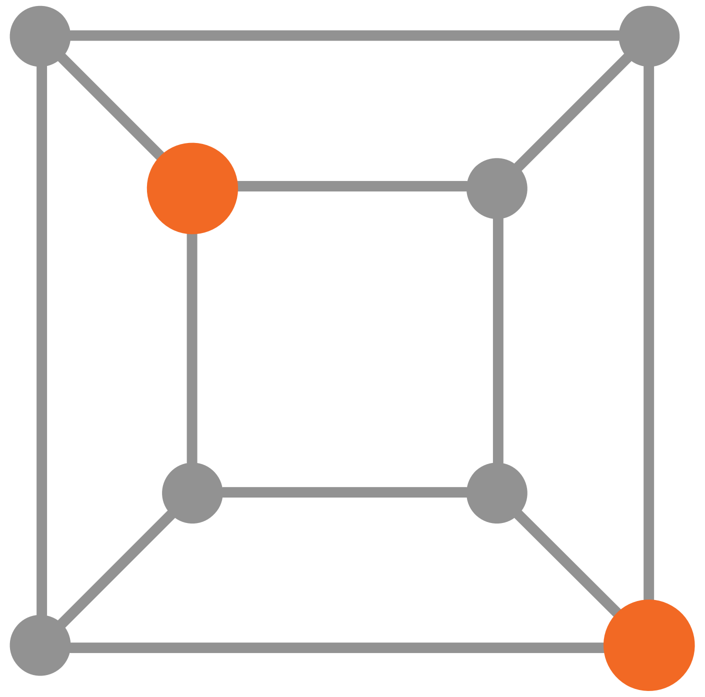
- **Matching**: edges in the matching do not share a node
- **Maximality**: if we add any other edge in the matching, than it is not a matching anymore
- We say that **a node is matched**: it is an endpoint of an edge in the matching

Maximal independent set problem

Input



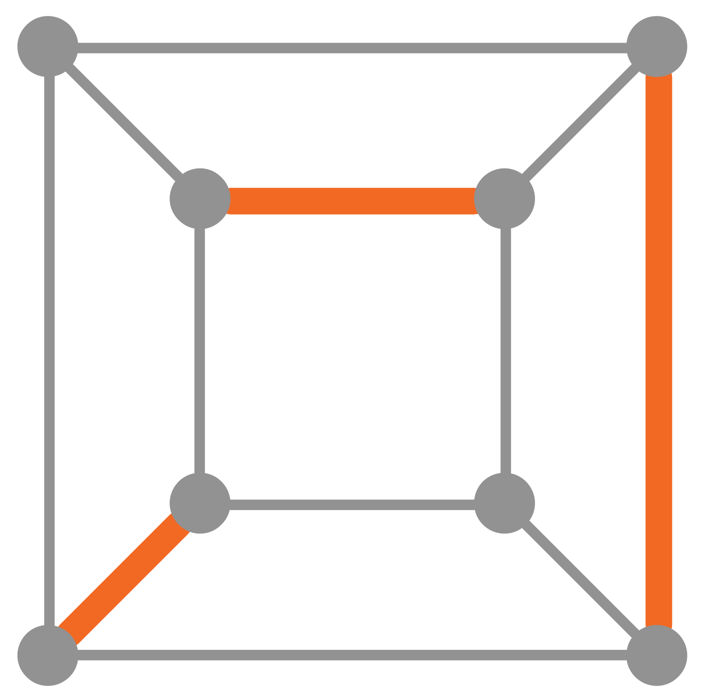
Output



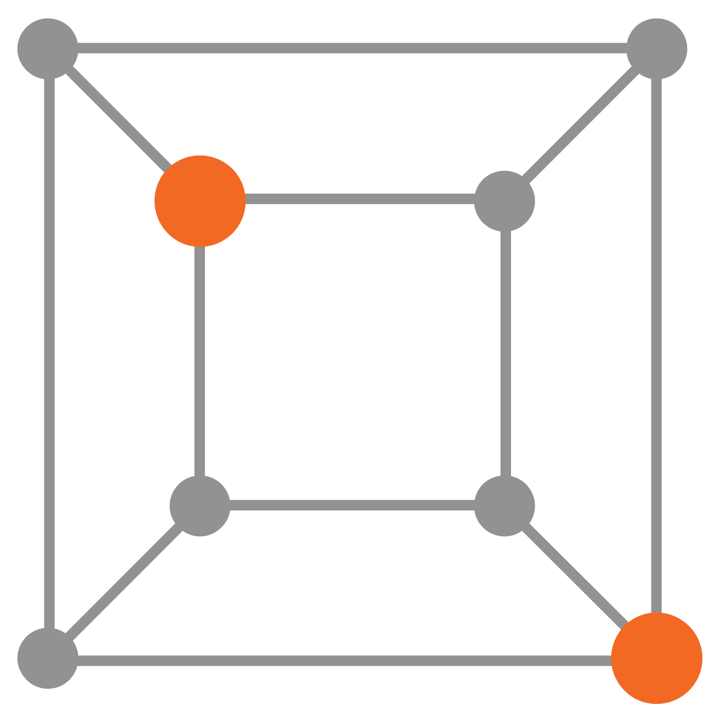
- **Independent set**: nodes in the IS do not share an edge
- **Maximality**: if we add any other node in the IS, than it is not independent anymore

Two classical graph problems

Maximal matching



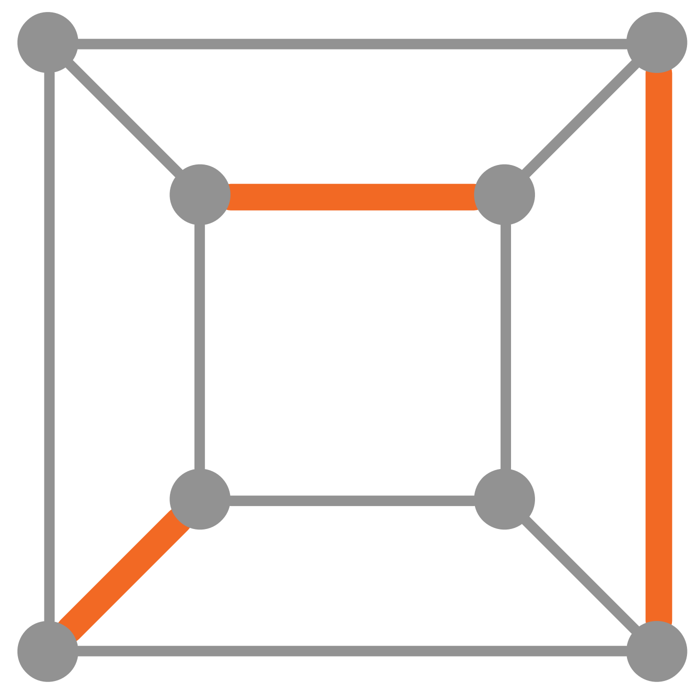
Maximal independent set



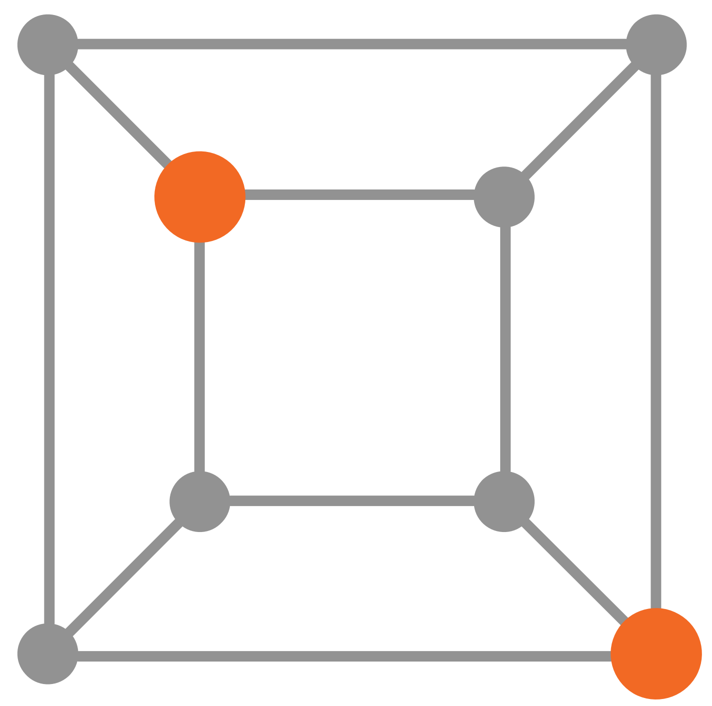
Easy linear-time **centralized** algorithm:
add edges/nodes until stuck

Two classical graph problems

Maximal matching



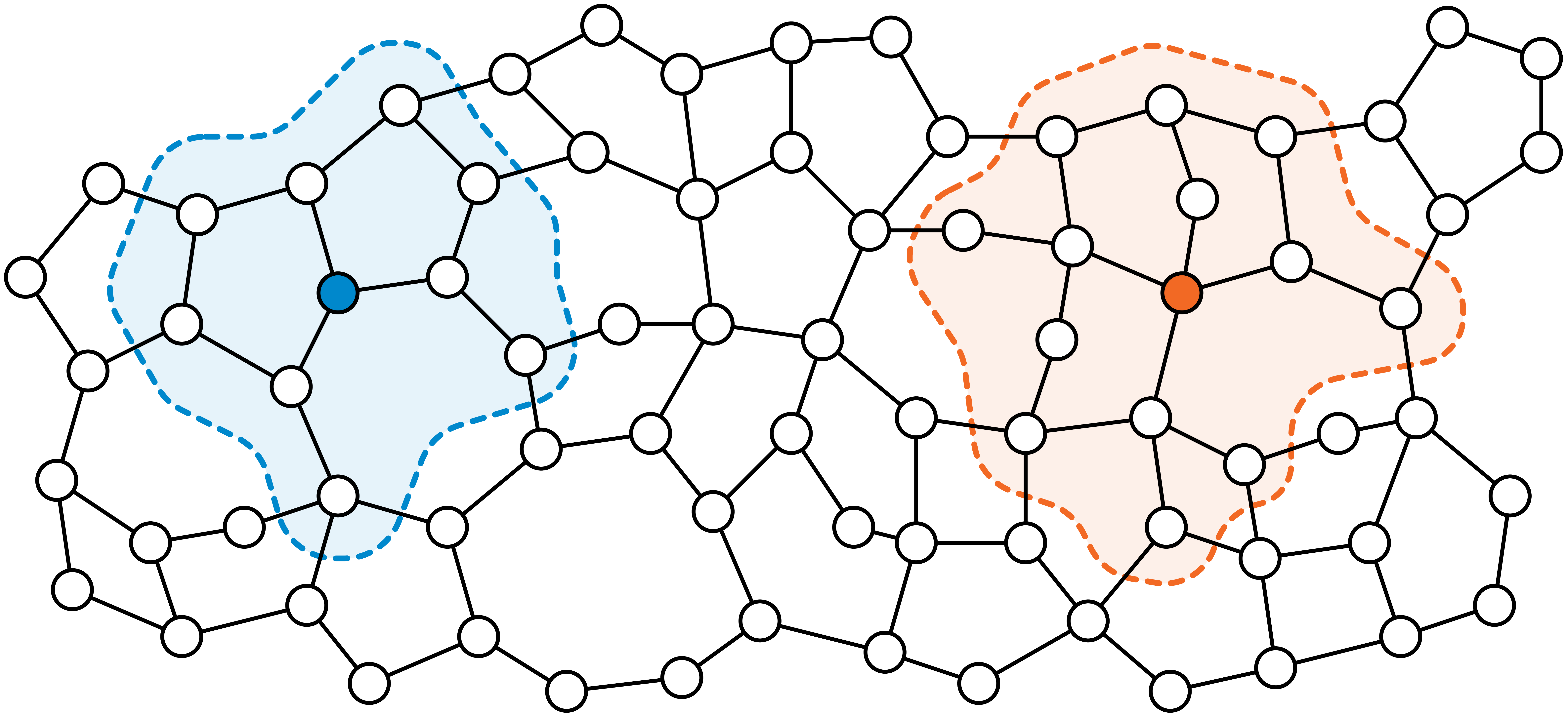
Maximal independent set



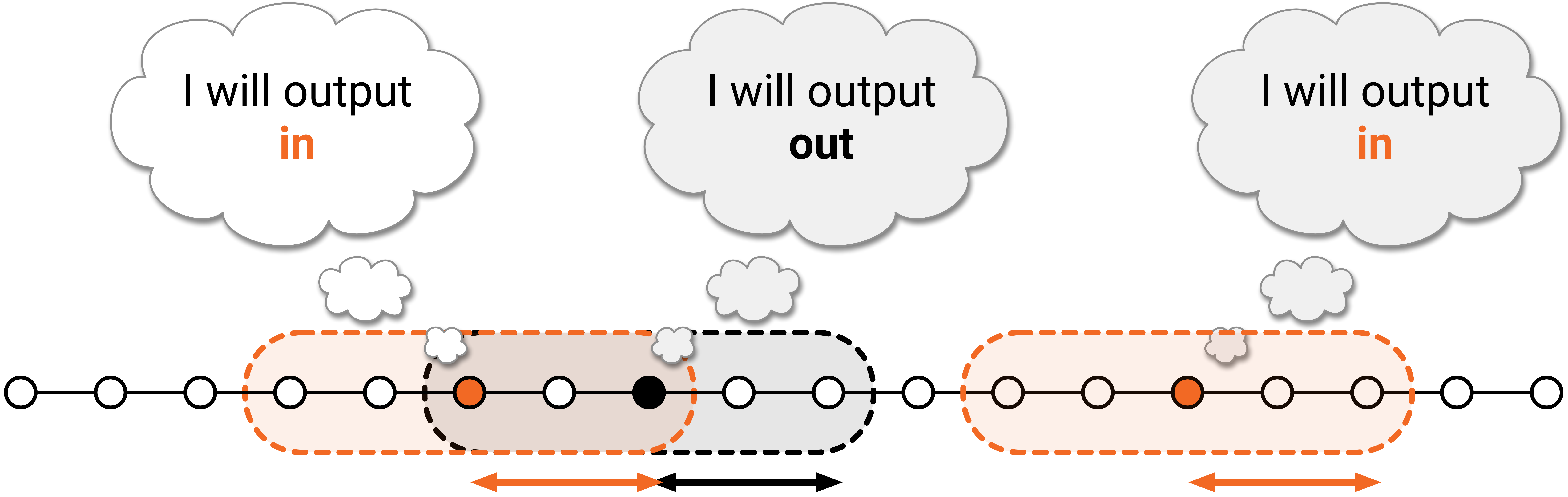
Can be ***verified locally***: if it looks correct everywhere locally, it is also feasible globally

Can these problems be ***solved locally***?

Locality = how far do I need to see to produce my own part of the solution?



Locality = how far do I need to see to produce my own part of the solution?



Locality = how far do I need to see to produce my own part of the solution?

Local outputs form a globally consistent solution

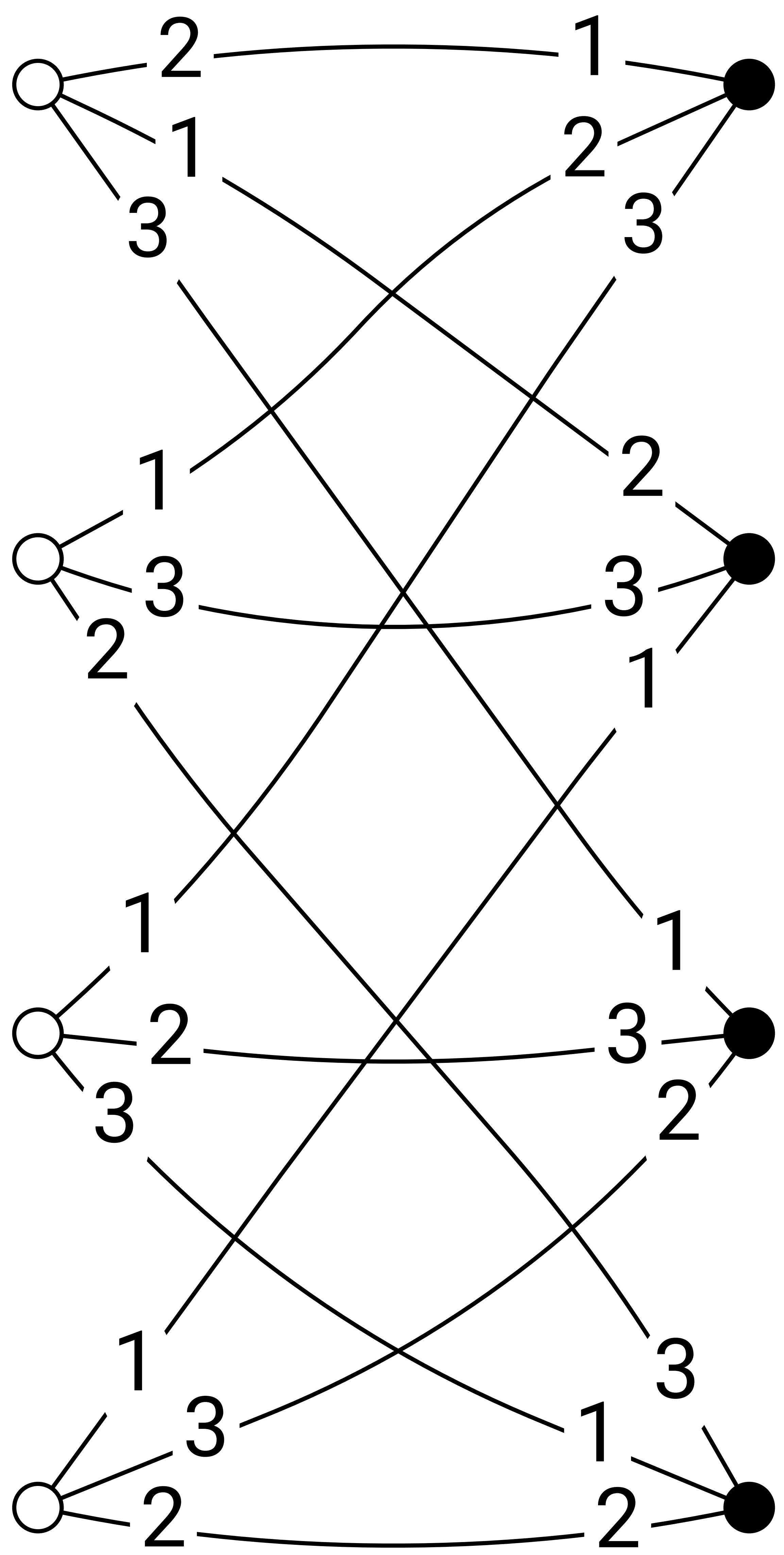
Warmup: toy example

Bipartite graphs & port-numbering model

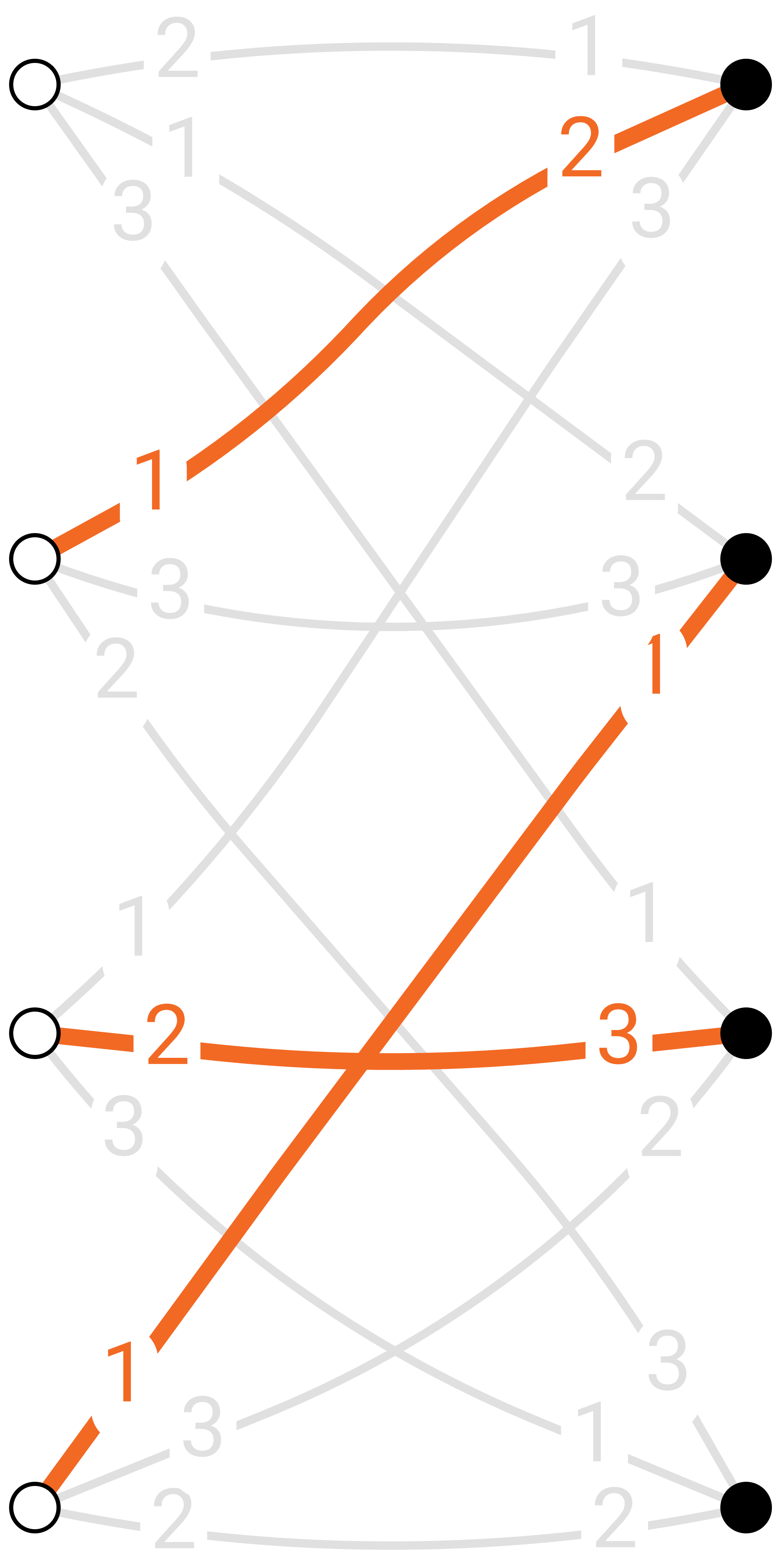
computer network with port numbering

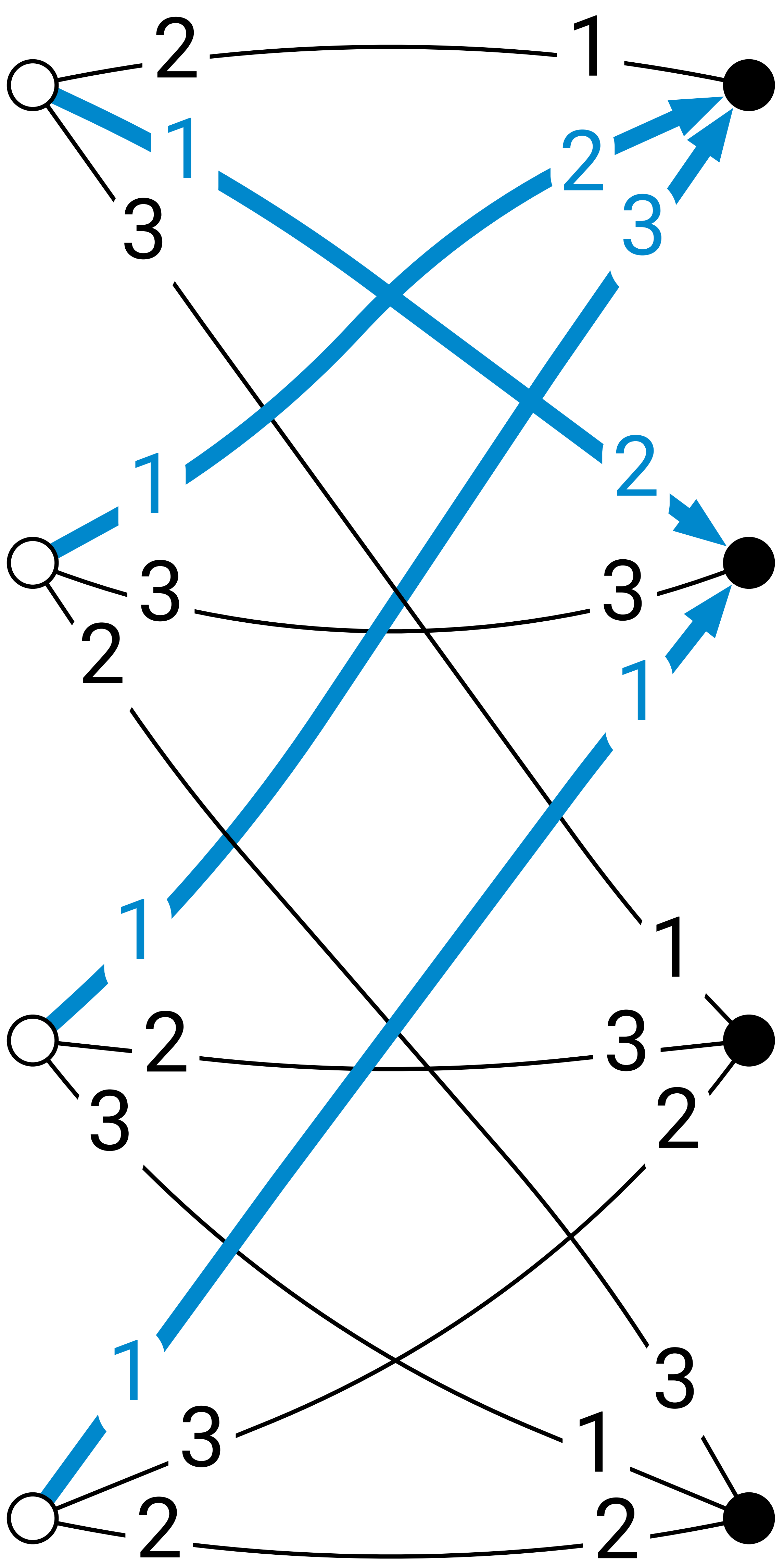
bipartite, 2-colored graph

Δ-regular (here Δ = 3)



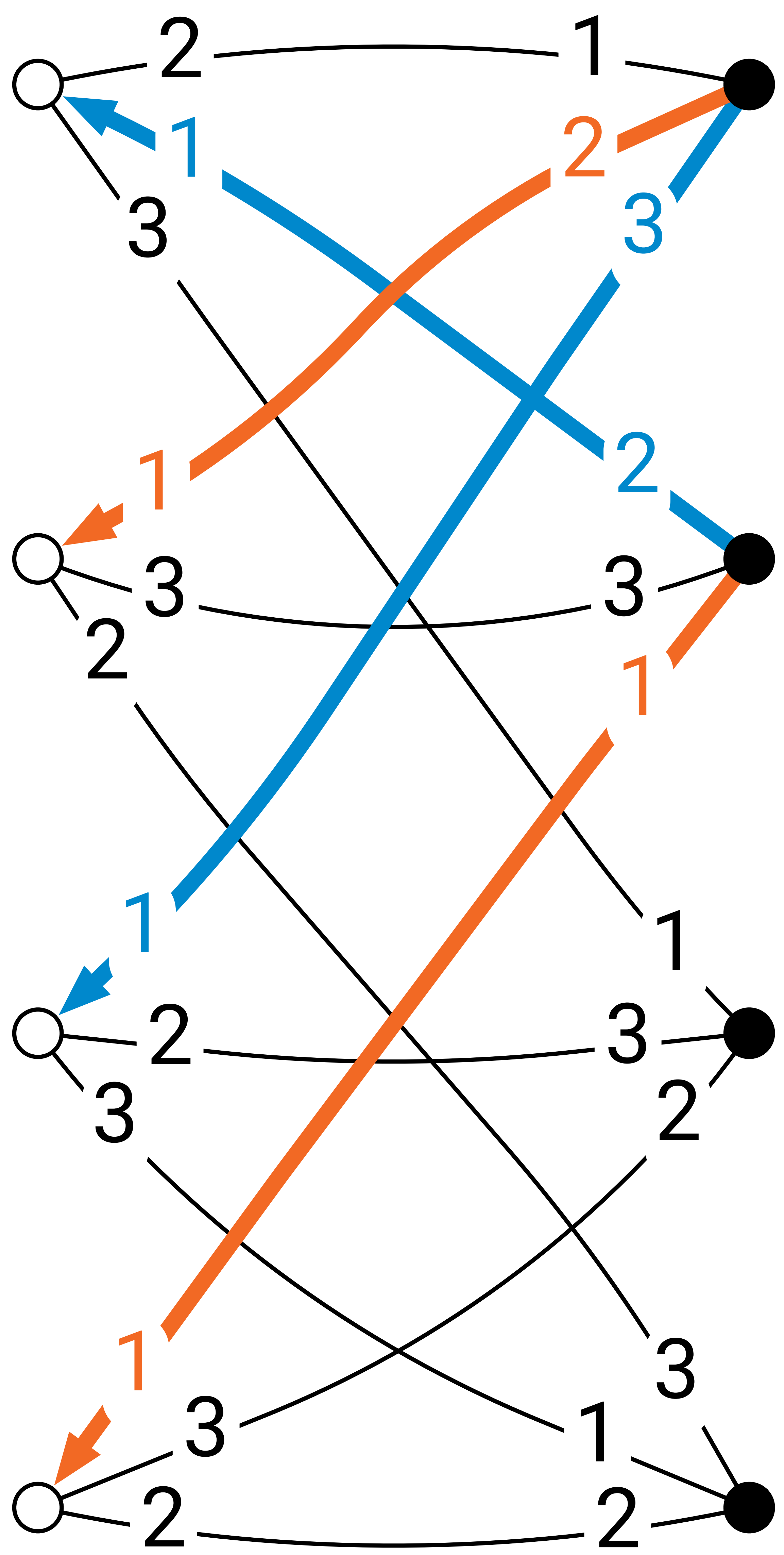
output: ***maximal matching***





Very simple algorithm

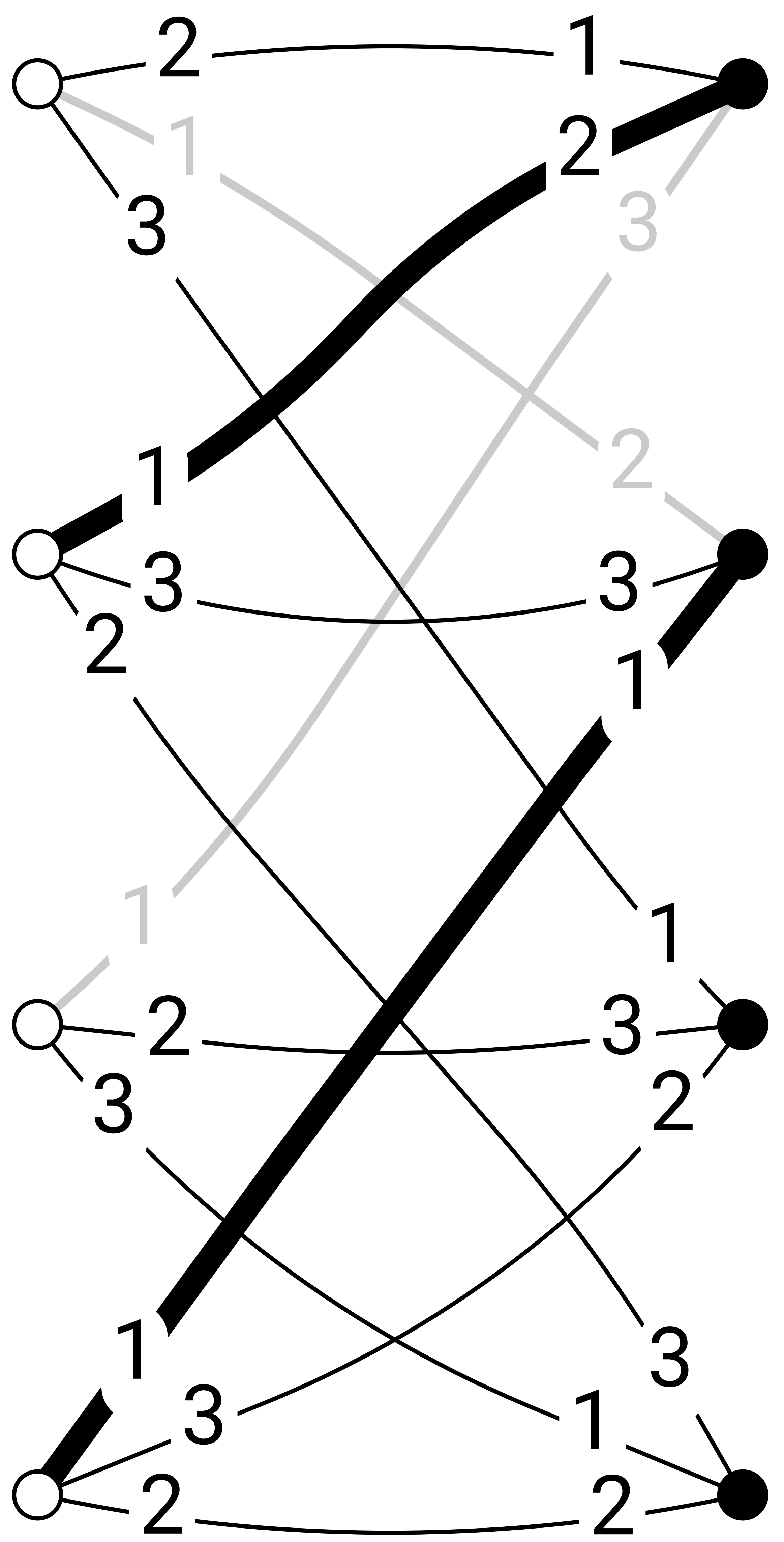
unmatched white nodes:
send ***proposal*** to port 1



Very simple algorithm

unmatched white nodes:
send ***proposal*** to port 1

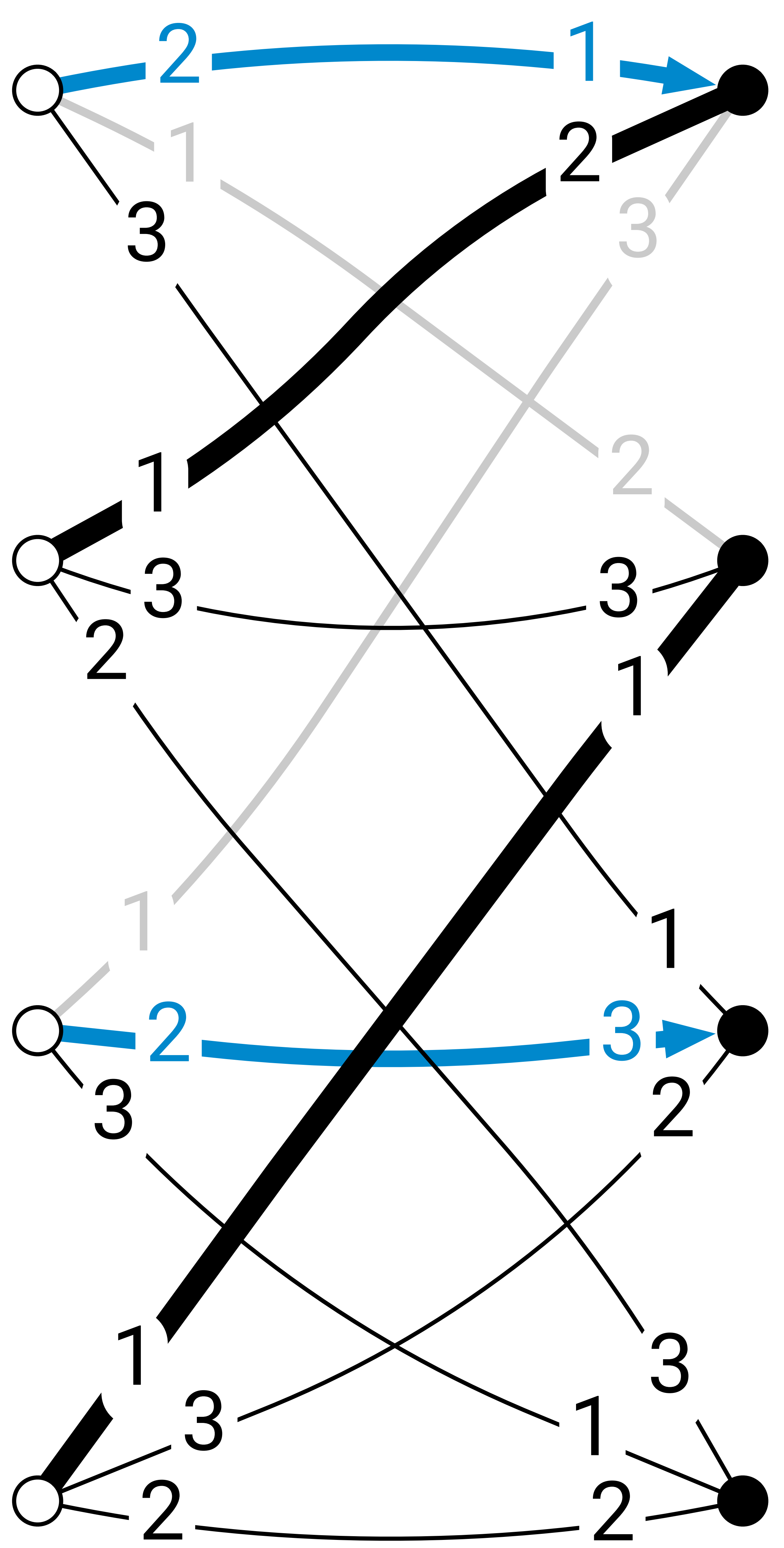
black nodes:
accept the first proposal you get, ***reject*** everything else (break ties with port numbers)



Very simple algorithm

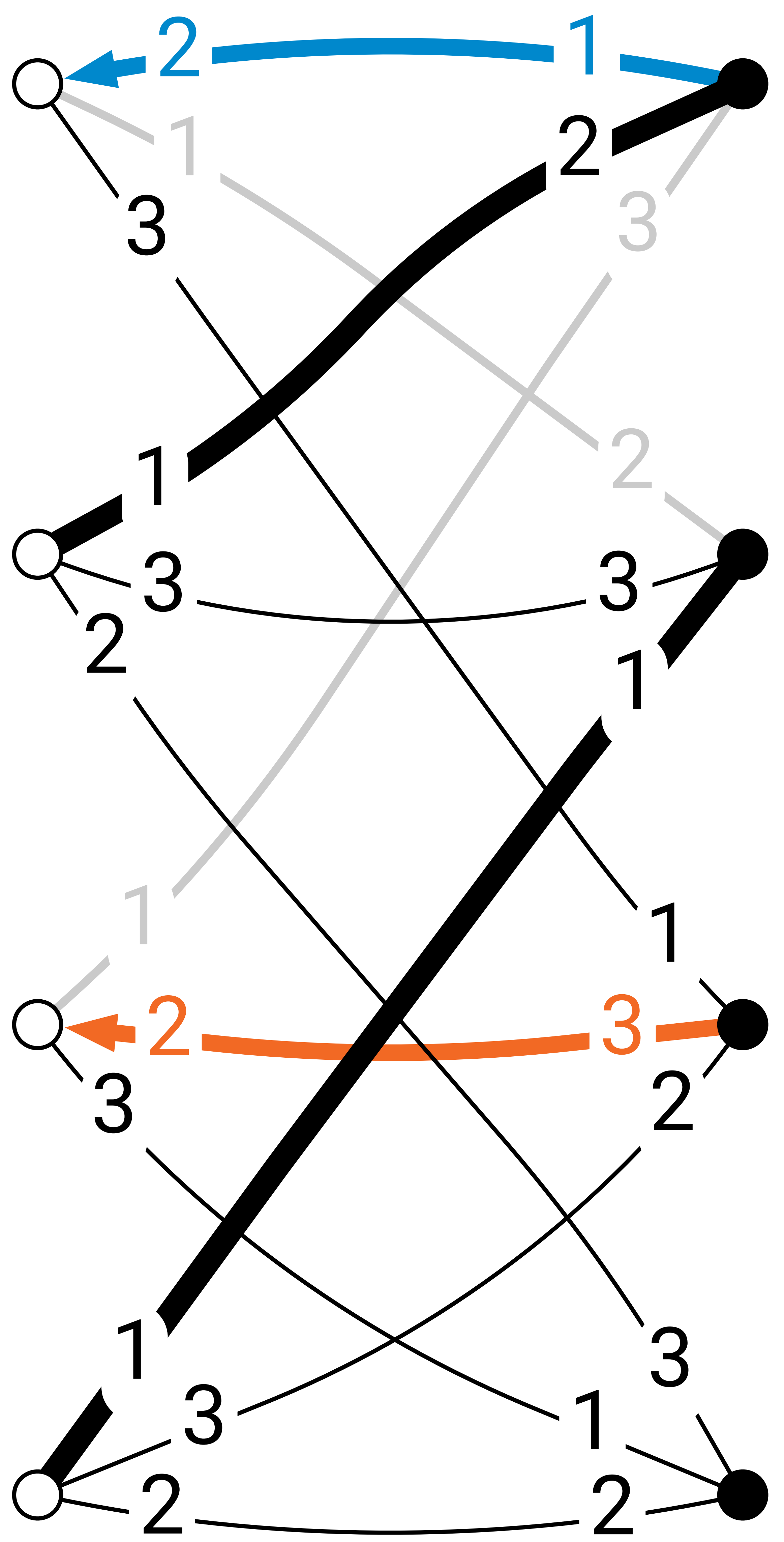
unmatched white nodes:
send ***proposal*** to port 1

black nodes:
accept the first proposal you get, ***reject*** everything else
(break ties with port numbers)



Very simple algorithm

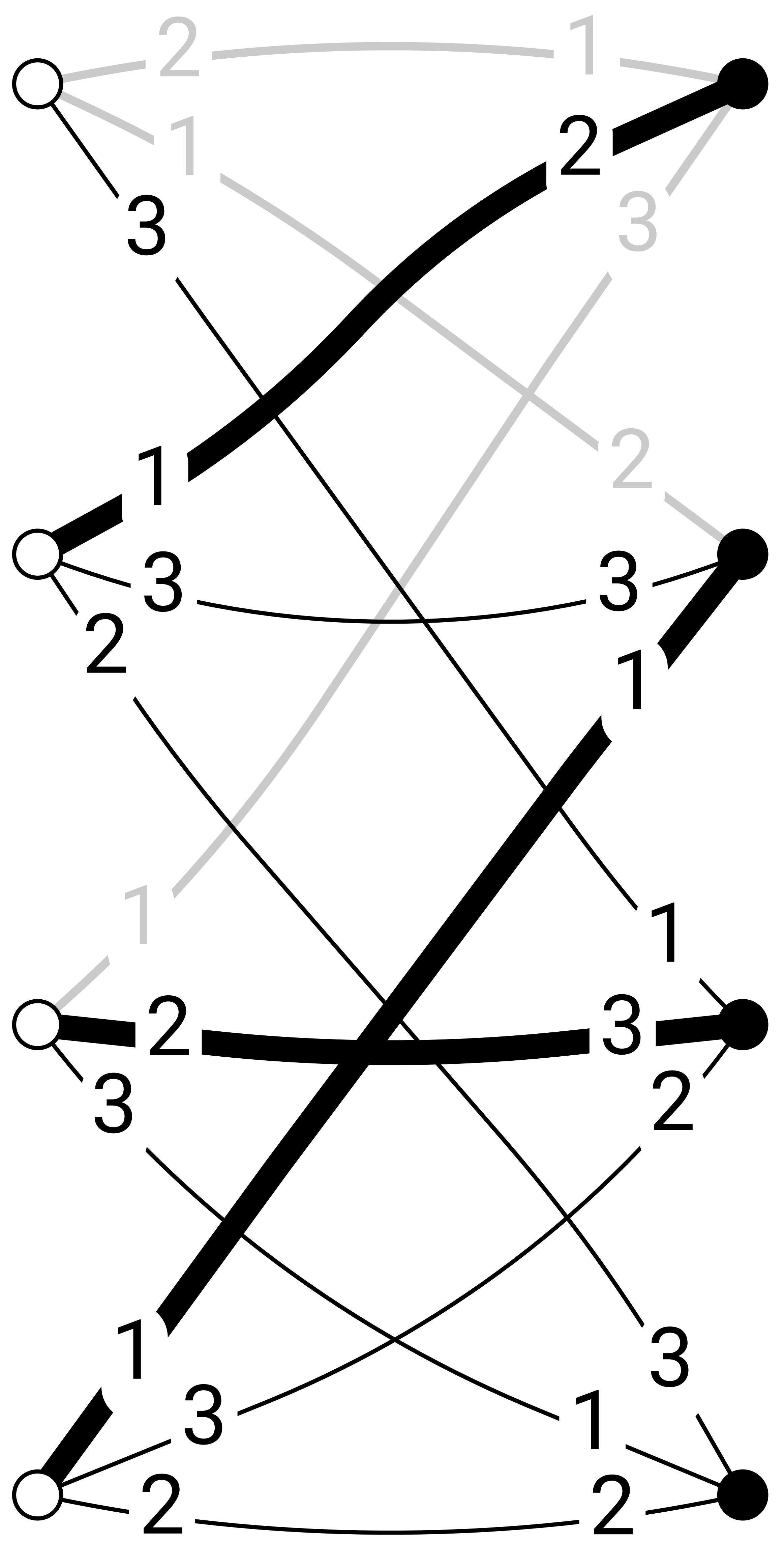
unmatched white nodes:
send ***proposal*** to port 2



Very simple algorithm

unmatched white nodes:
send ***proposal*** to port 2

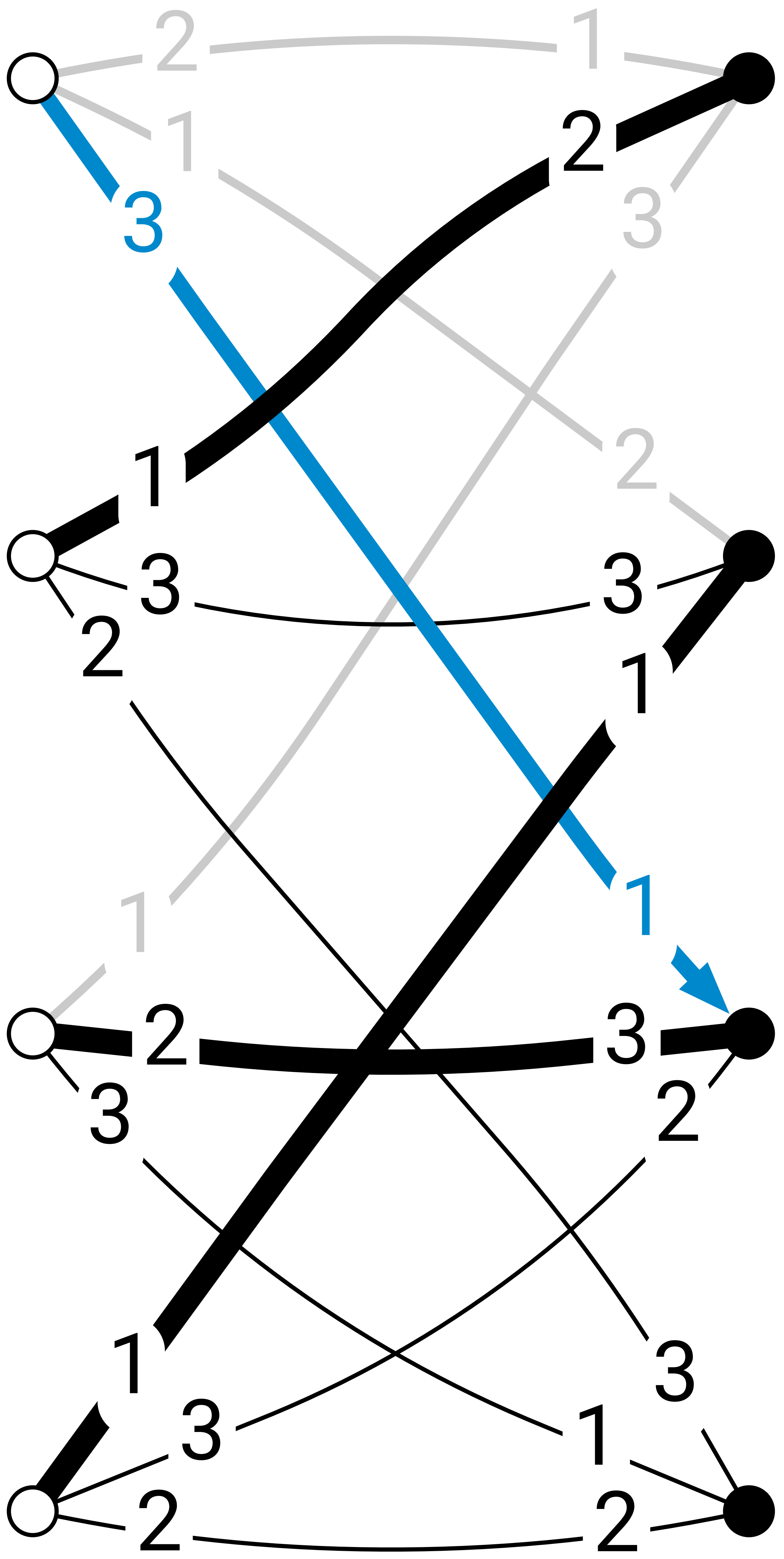
black nodes:
accept the first proposal you get, ***reject*** everything else (break ties with port numbers)



Very simple algorithm

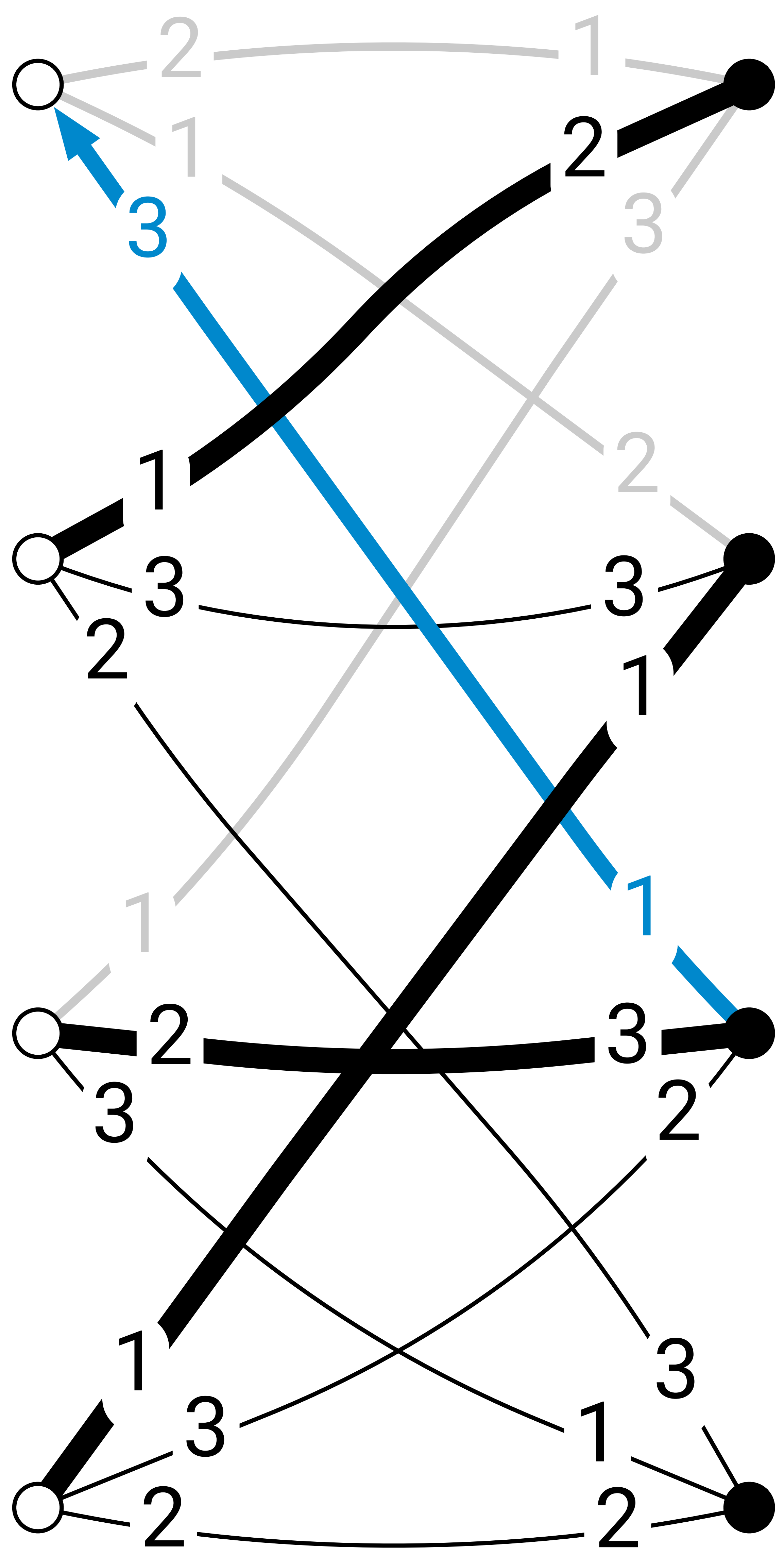
unmatched white nodes:
send ***proposal*** to port 2

black nodes:
accept the first proposal you get, ***reject*** everything else (break ties with port numbers)



Very simple algorithm

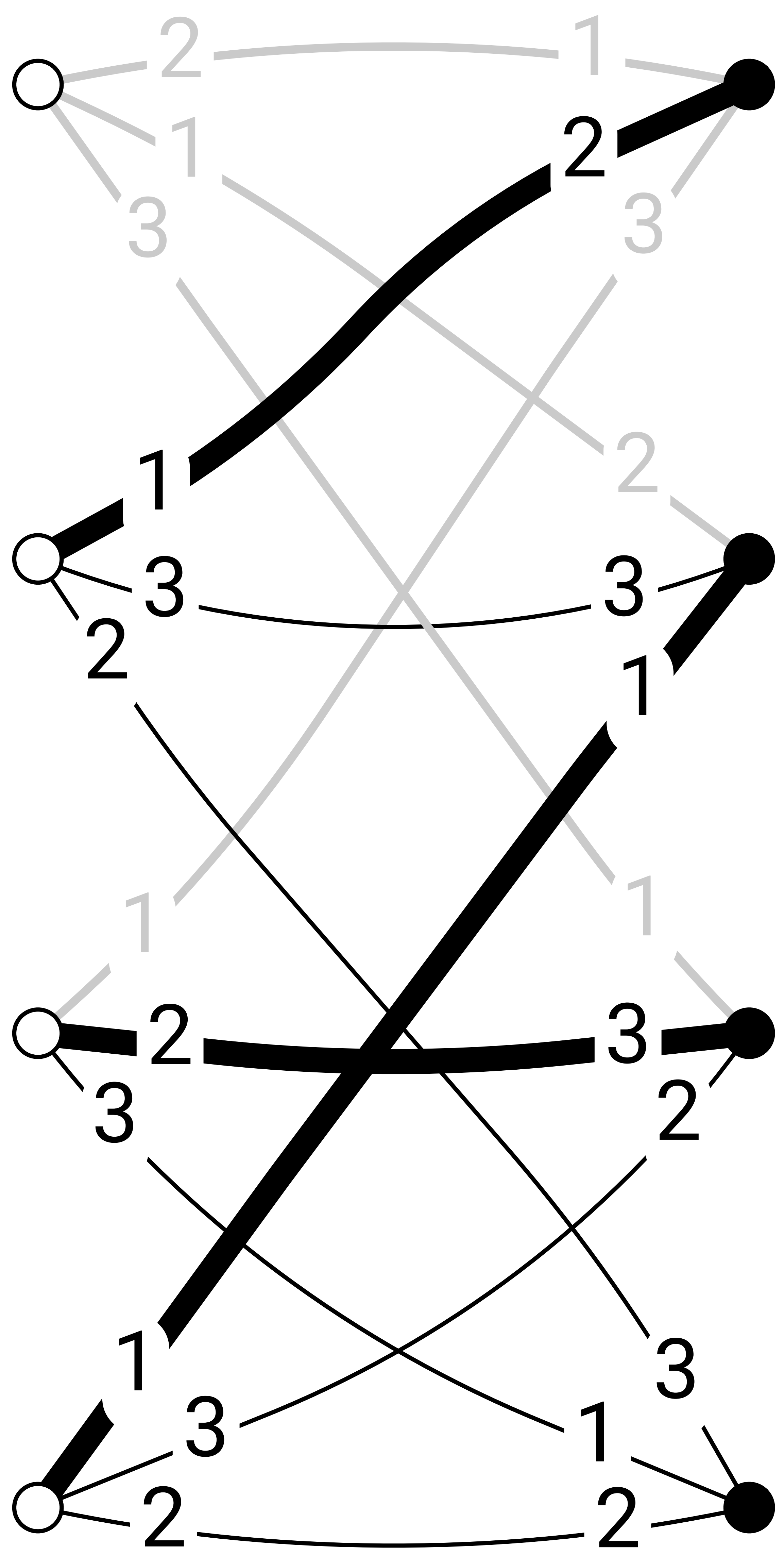
unmatched white nodes:
send ***proposal*** to port 3



Very simple algorithm

unmatched white nodes:
send ***proposal*** to port 3

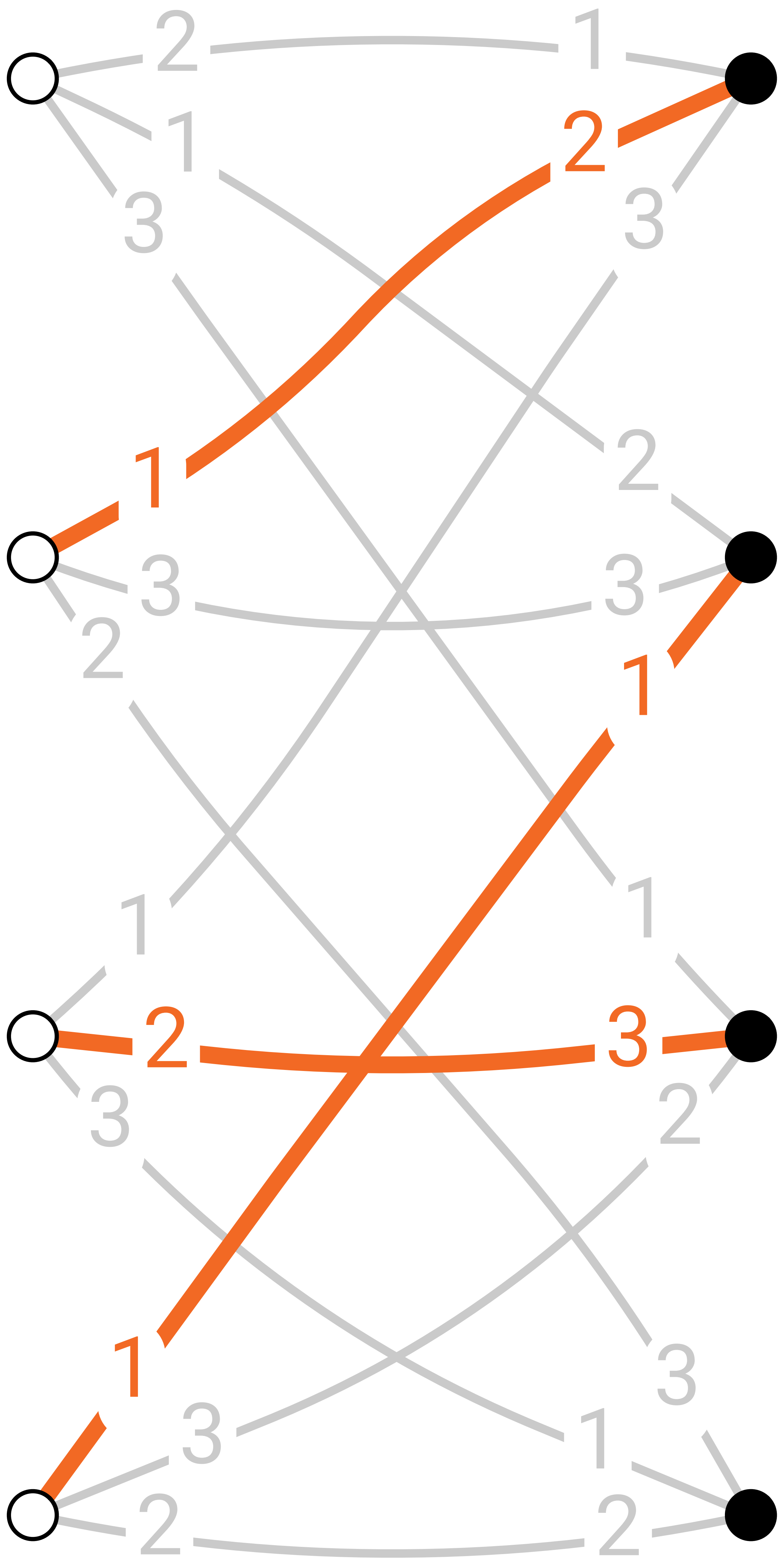
black nodes:
accept the first proposal you get, ***reject*** everything else
(break ties with port numbers)



Very simple algorithm

unmatched white nodes:
send ***proposal*** to port 3

black nodes:
accept the first proposal you get, ***reject*** everything else
(break ties with port numbers)



Very simple algorithm

Finds a ***maximal matching*** in $O(\Delta)$ communication rounds

Note: running time does not depend on n

Bipartite maximal matching

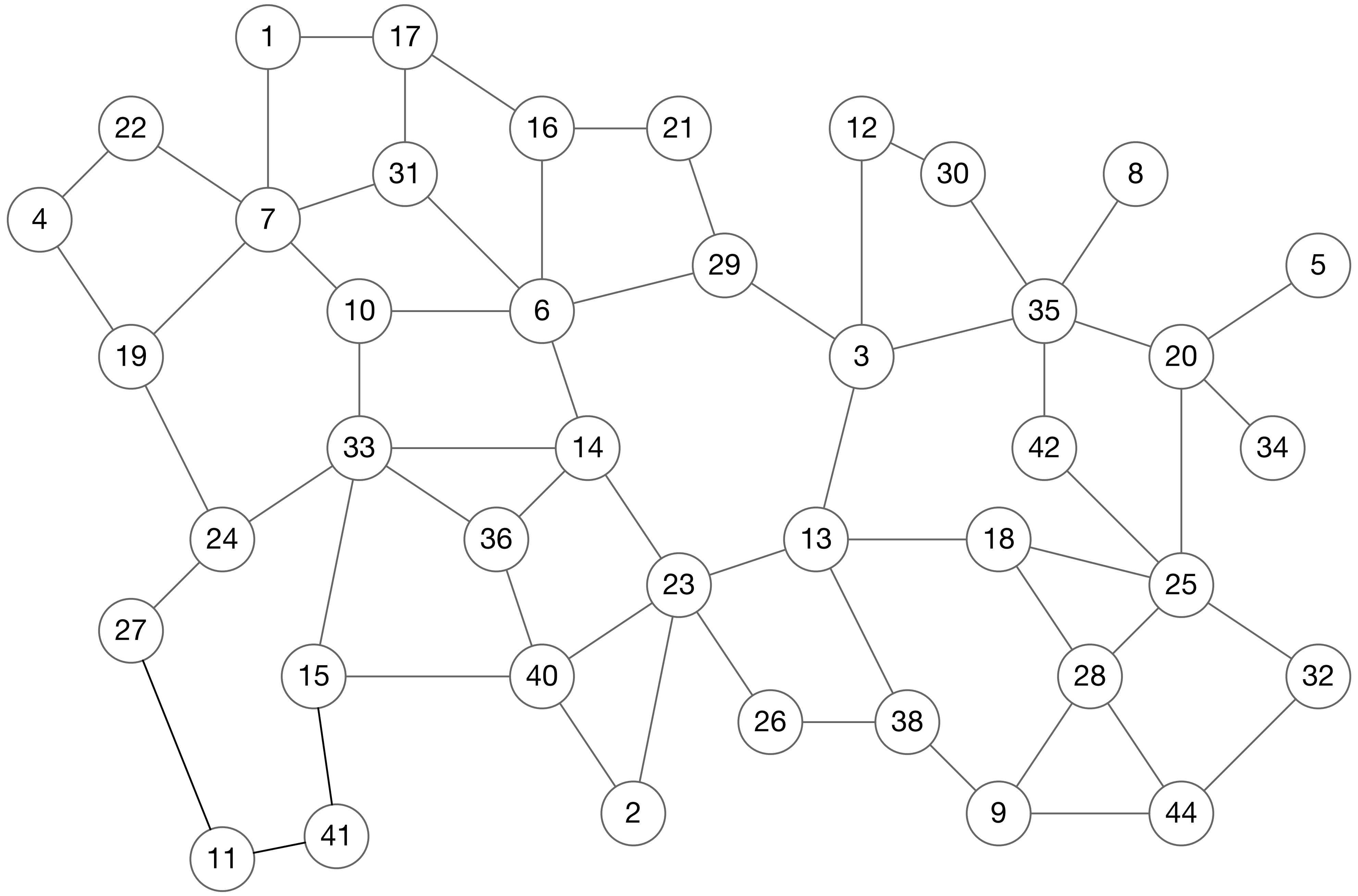
- Maximal matching in very large 2-colored Δ-regular graphs
- Simple algorithm: ***$O(\Delta)$ rounds***, independently of n
- ***Is this optimal?***
  - $o(\Delta)$ rounds?
  - $O(\log \Delta)$ rounds?
  - 4 rounds??

Big picture

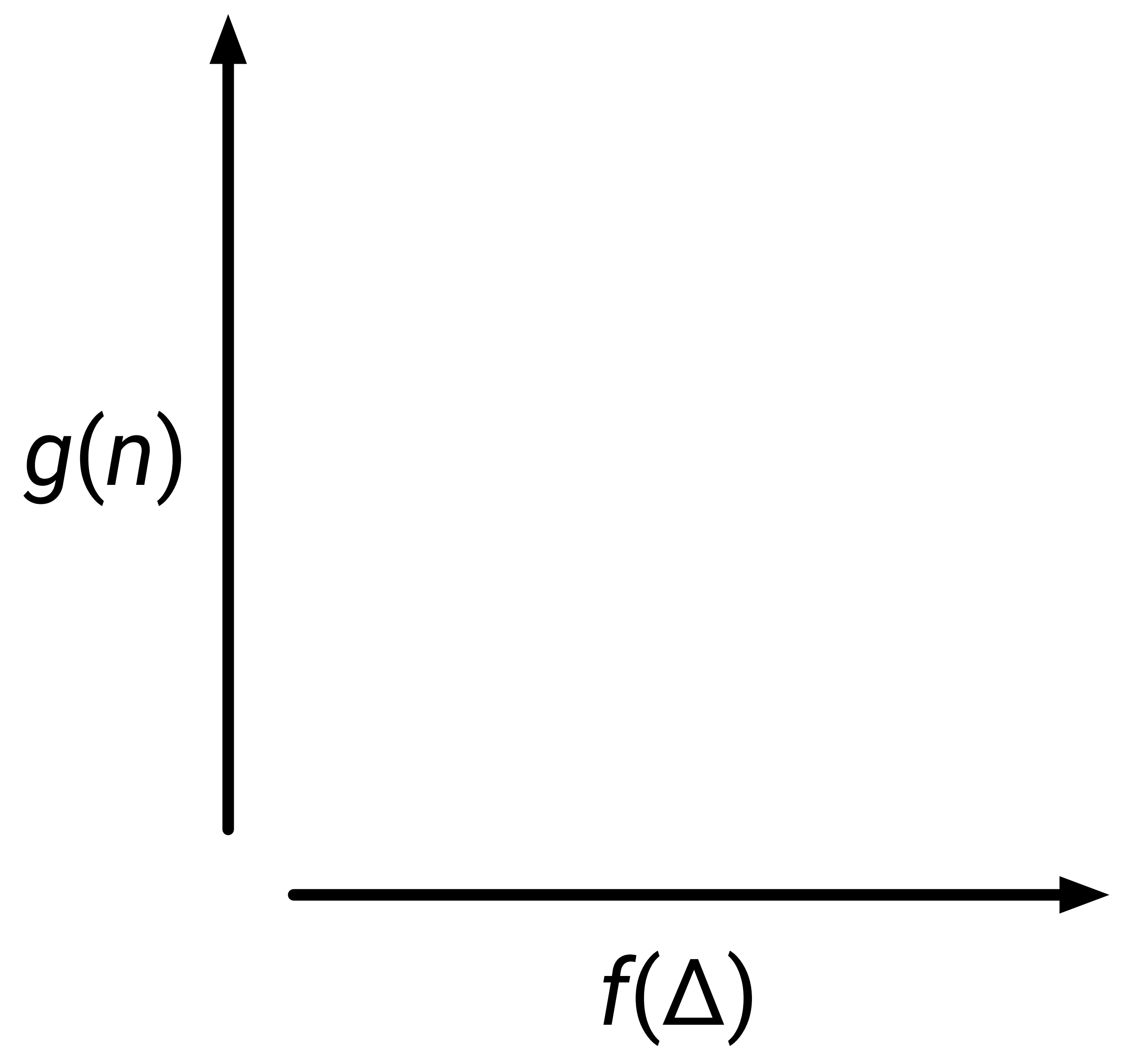
Bounded-degree graphs & LOCAL model

LOCAL model

- Each node has a **unique identifier** from 1 to poly(n)
- **No bounds** on the computational power
- **No bounds** on the bandwidth
- **Synchronous** model
- **Everything** can be solved in **Diameter** time



Strong model — lower bounds widely applicable



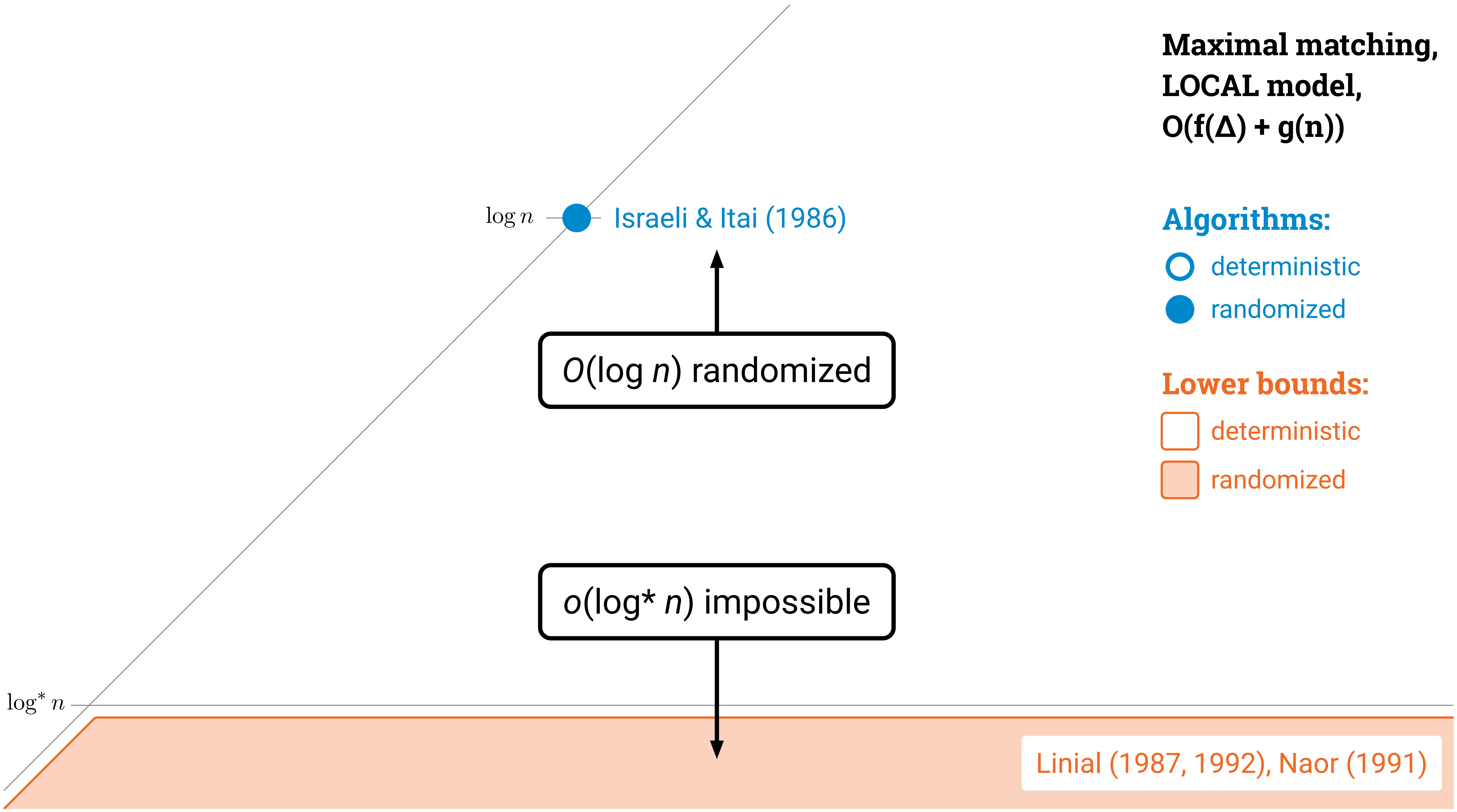
Maximal matching, LOCAL model, O(f(Δ) + g(n))

Algorithms:

- deterministic
- randomized

Lower bounds:

- deterministic
- randomized



Maximal matching,
LOCAL model,
O(f(Δ) + g(n))

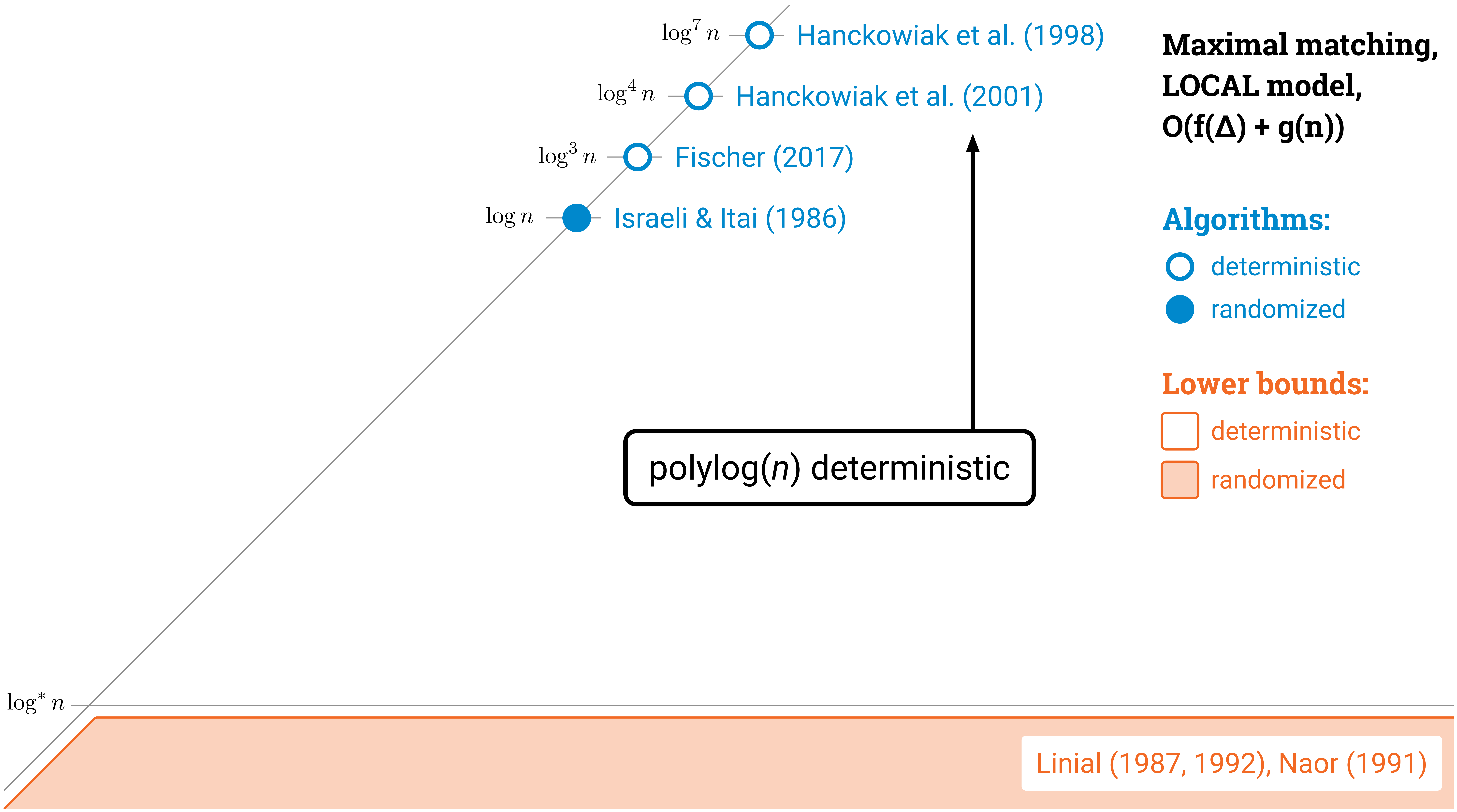
Algorithms:

- deterministic
- randomized

Lower bounds:

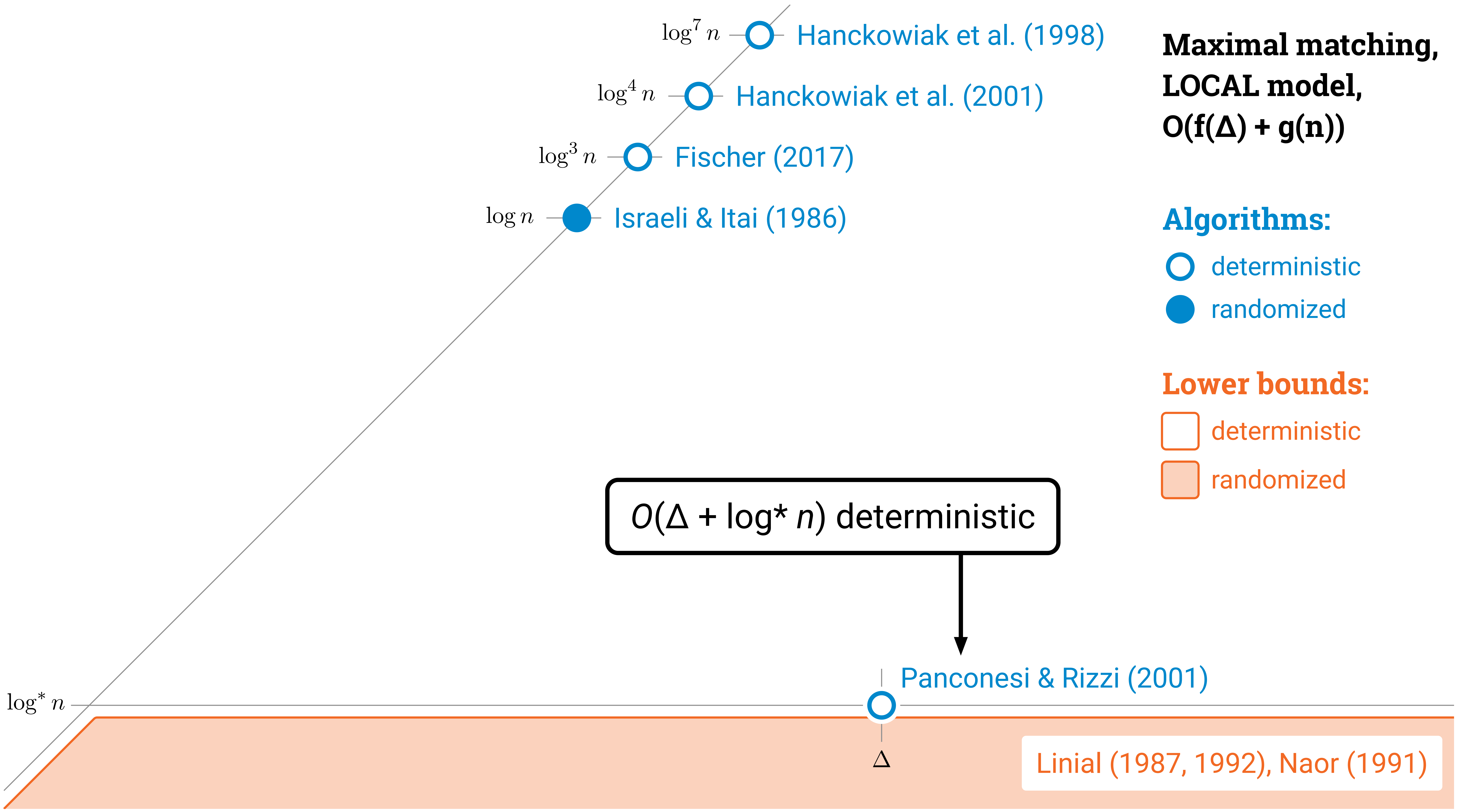
- deterministic
- randomized

Maximal matching,
LOCAL model,
O(f(Δ) + g(n))



Algorithms:
- deterministic
- randomized

Lower bounds:
- deterministic
- randomized



Maximal matching, LOCAL model, O(f(Δ) + g(n))

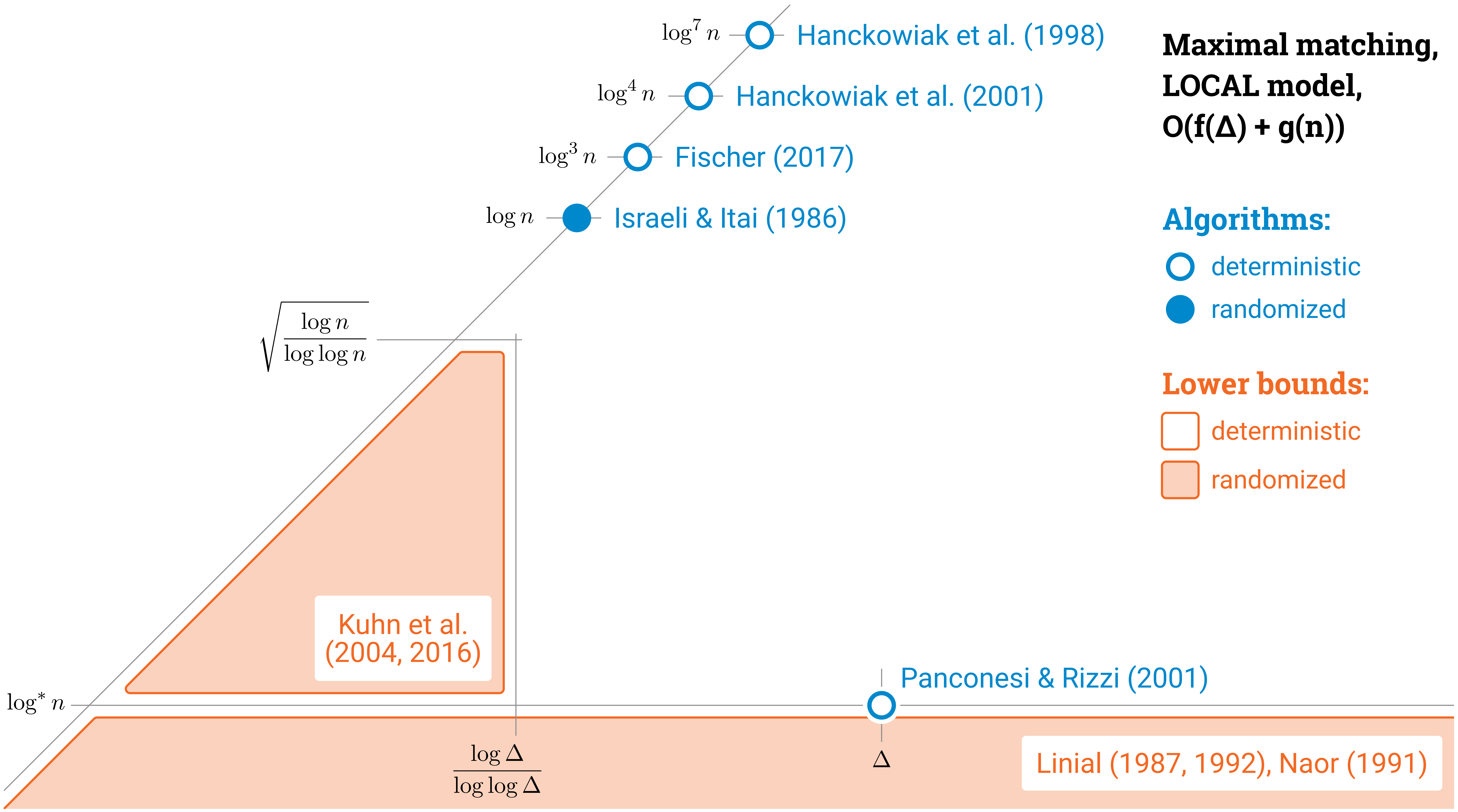
Algorithms:

- deterministic
- randomized

Lower bounds:

- deterministic
- randomized

Maximal matching, LOCAL model, O(f(Δ) + g(n))



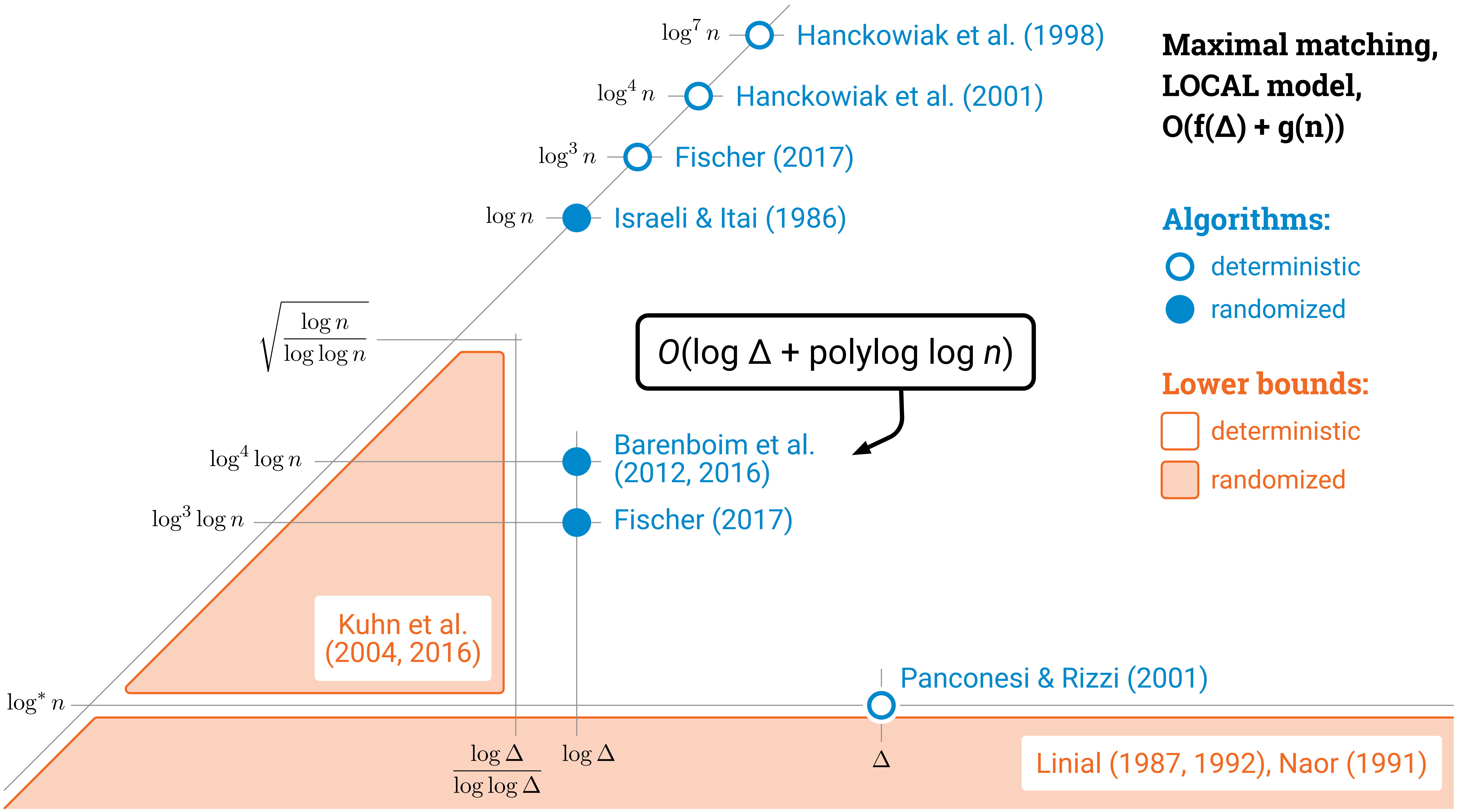
Algorithms:

- deterministic
- randomized

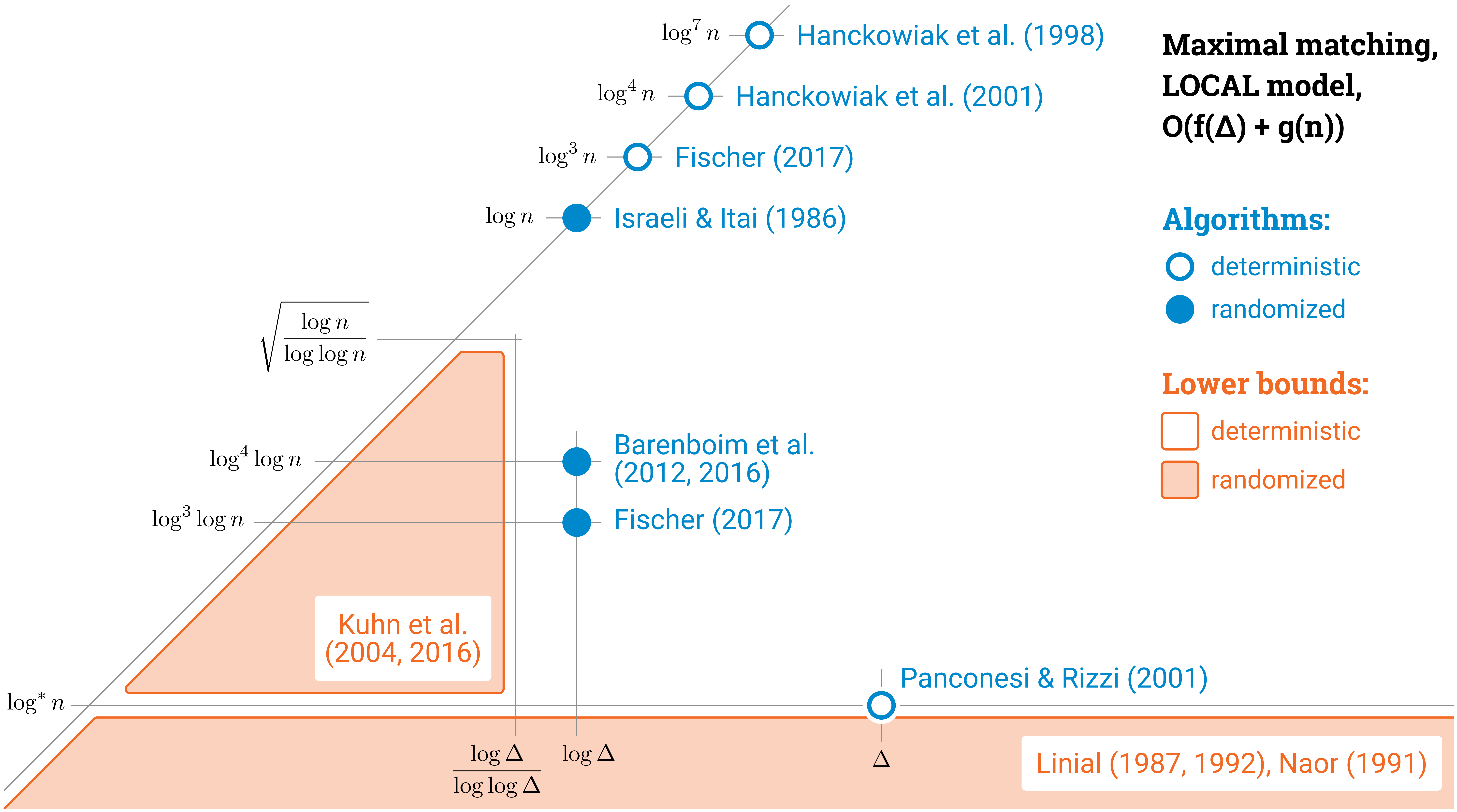
Lower bounds:

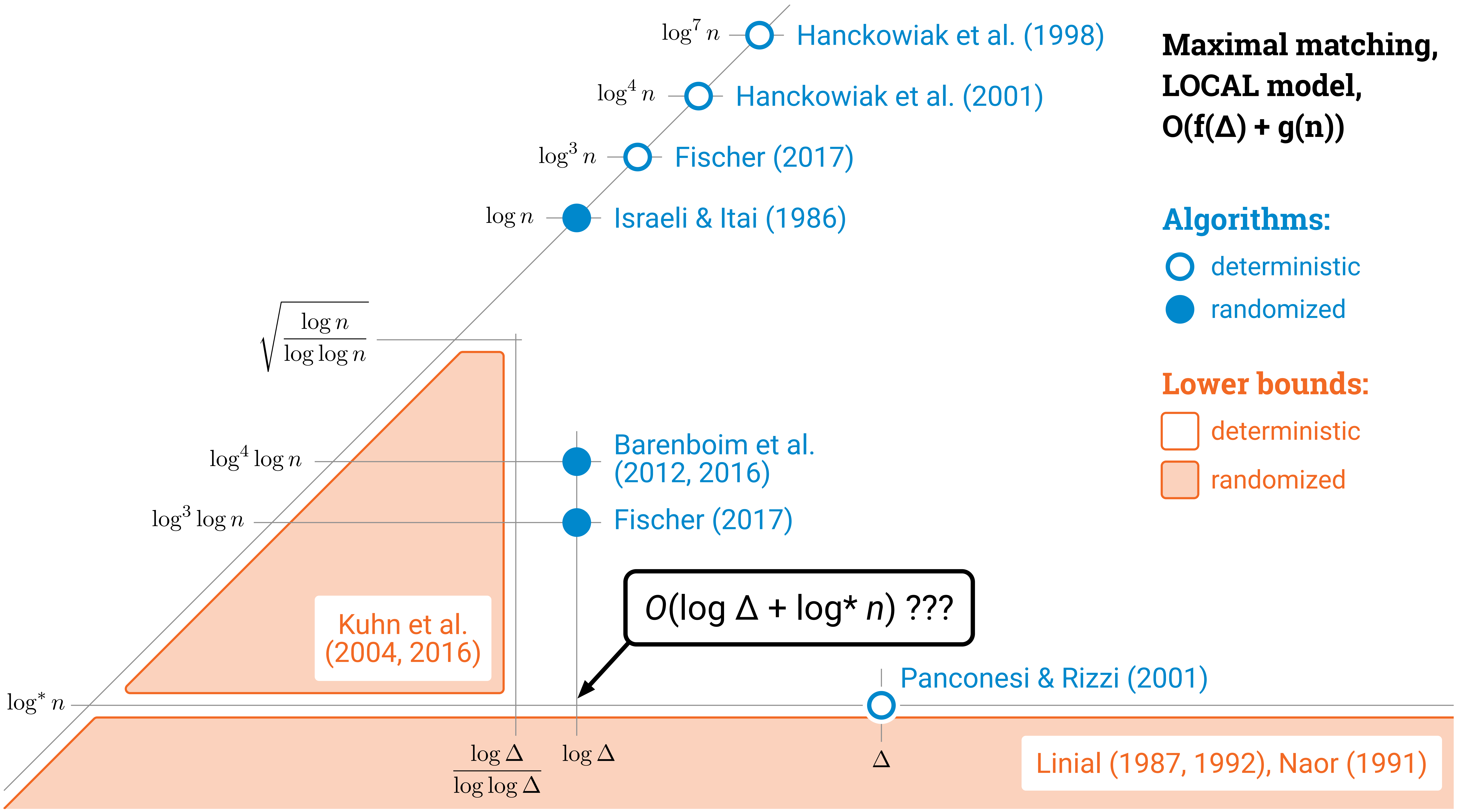
- deterministic
- randomized

Maximal matching, LOCAL model, $O(f(\Delta) + g(n))$



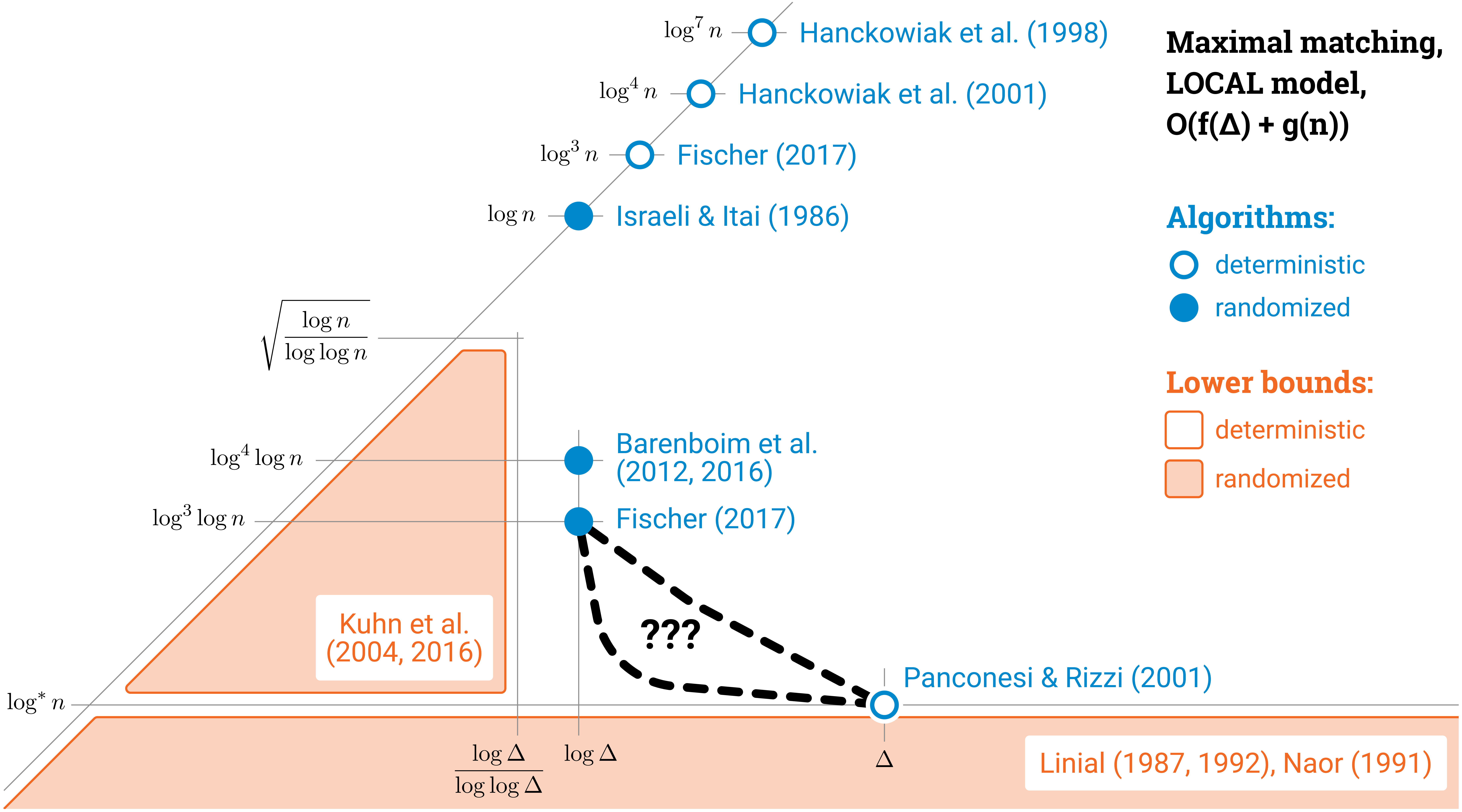
Maximal matching,
LOCAL model,
O(f(Δ) + g(n))

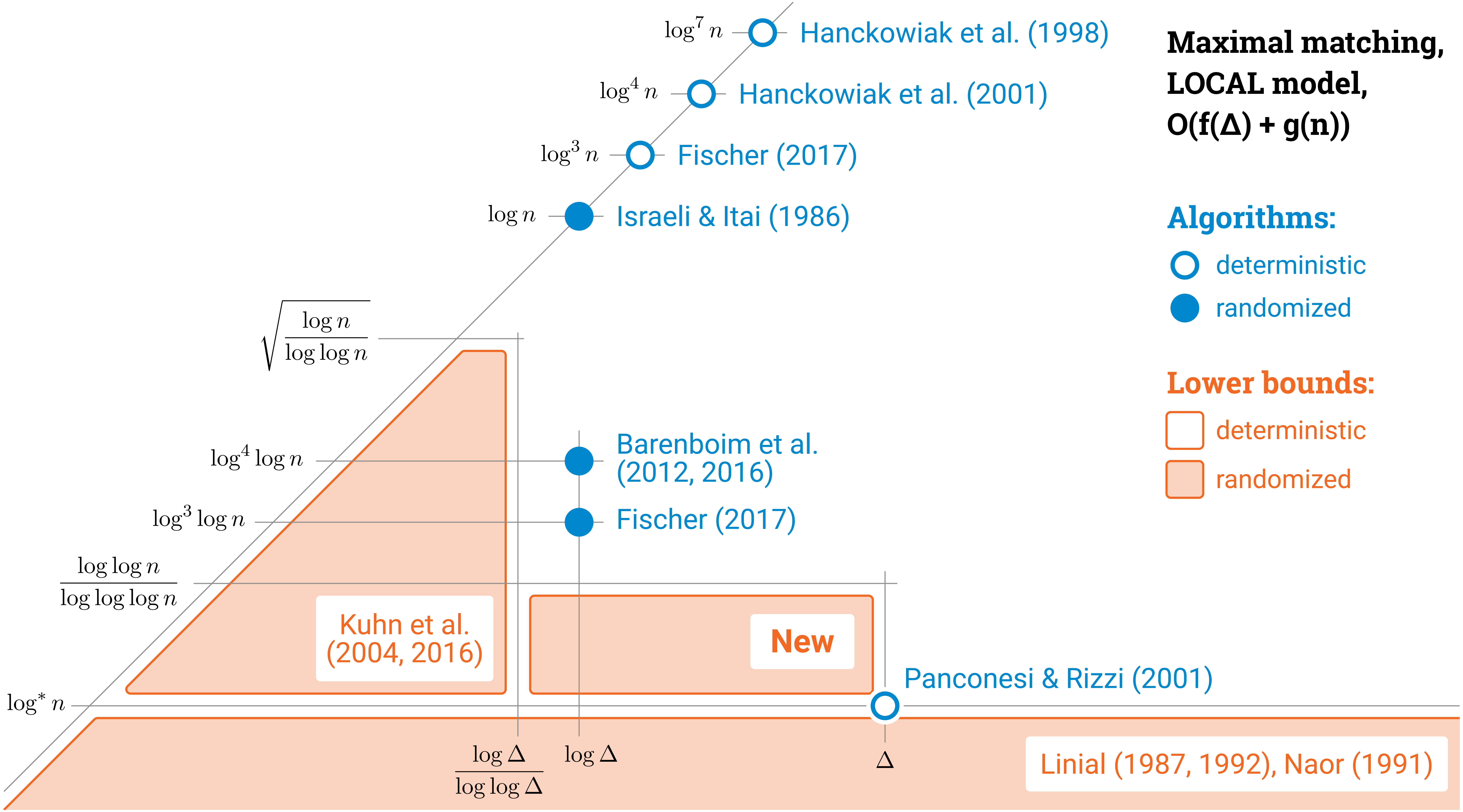




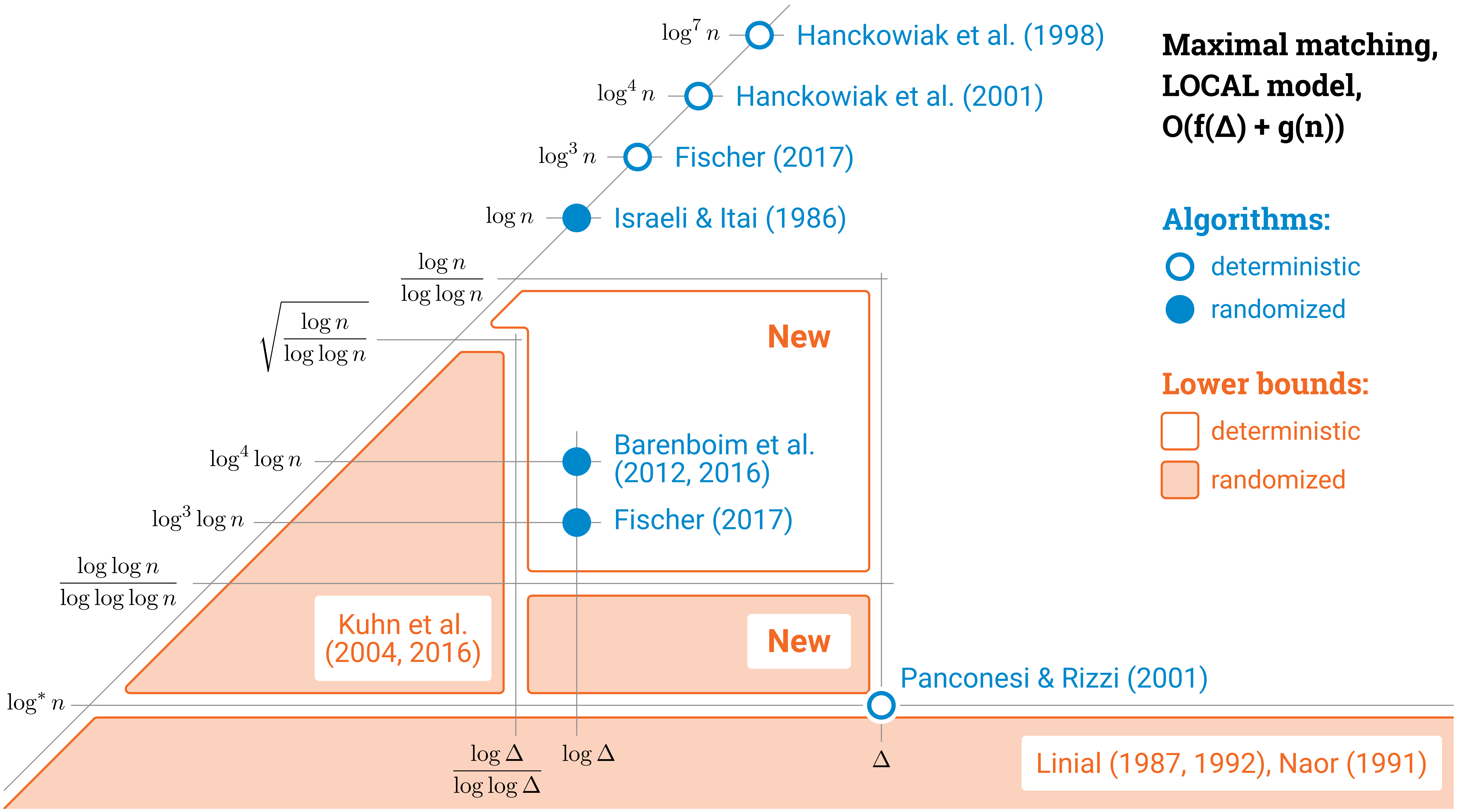
$\log^7 n$
Hanckowiak et al. (1998)
$\log^4 n$
Hanckowiak et al. (2001)
$\log^3 n$
Fischer (2017)
$\log n$
Israeli & Itai (1986)
$\sqrt{\frac{\log n}{\log \log n}}$
$\log^4 \log n$
Barenboim et al. (2012, 2016)
$\log^3 \log n$
Fischer (2017)
$O(\log \Delta + \log^* n)$???
Kuhn et al. (2004, 2016)
Panconesi & Rizzi (2001)
$\log^* n$
$\frac{\log \Delta}{\log \log \Delta}$
$\log \Delta$
Δ
Linial (1987, 1992), Naor (1991)
Maximal matching, LOCAL model, O(f(Δ) + g(n))
Algorithms:
deterministic
randomized
Lower bounds:
deterministic
randomized

Maximal matching, LOCAL model, O(f(Δ) + g(n))




Maximal matching,
LOCAL model,
O(f(Δ) + g(n))
Algorithms:
deterministic
randomized
Lower bounds:
deterministic
randomized
$\log^7 n$
Hanckowiak et al. (1998)
$\log^4 n$
Hanckowiak et al. (2001)
$\log^3 n$
Fischer (2017)
$\log n$
Israeli & Itai (1986)
$\sqrt{\frac{\log n}{\log \log n}}$
$\log^4 \log n$
Barenboim et al. (2012, 2016)
$\log^3 \log n$
Fischer (2017)
$\frac{\log \log n}{\log \log \log n}$
Kuhn et al. (2004, 2016)
New
Panconesi & Rizzi (2001)
$\log^* n$
$\frac{\log \Delta}{\log \log \Delta}$
$\log \Delta$
Δ
Linial (1987, 1992), Naor (1991)

Maximal matching, LOCAL model, O(f(Δ) + g(n))



Algorithms:

- deterministic
- randomized

Lower bounds:

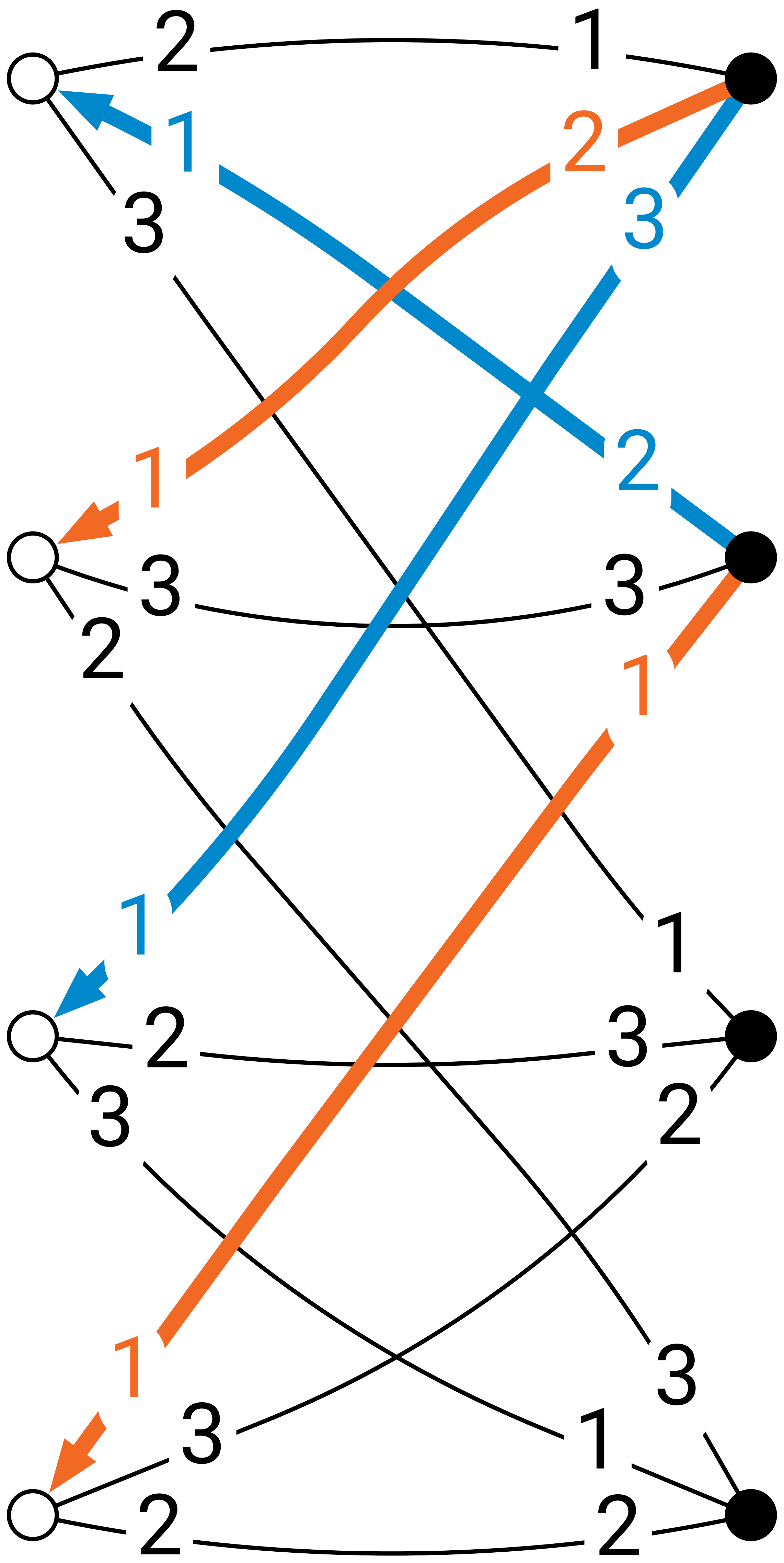
- deterministic
- randomized

Main results

Maximal Matching and **Maximal Independent Set** cannot be solved in

- $o(\Delta + \log \log n / \log \log \log n)$ rounds with randomized algorithms, in the LOCAL model
- $o(\Delta + \log n / \log \log n)$ rounds with deterministic algorithms, in the LOCAL model

Upper bound:
$O(\Delta + \log^* n)$



This is optimal!

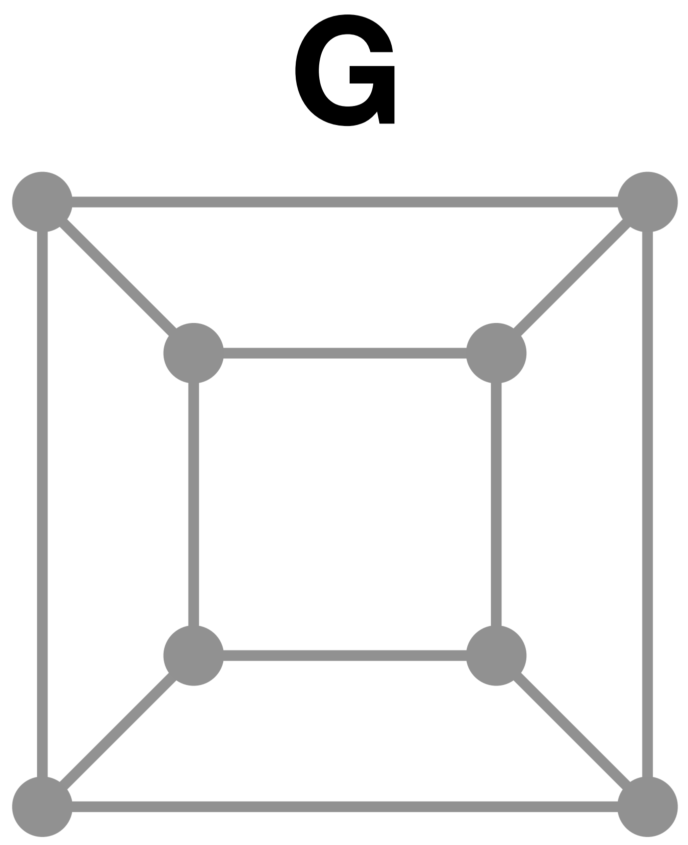
Very simple algorithm

unmatched white nodes:
send ***proposal*** to port 1

black nodes:
accept the first proposal you get, ***reject*** everything else
(break ties with port numbers)

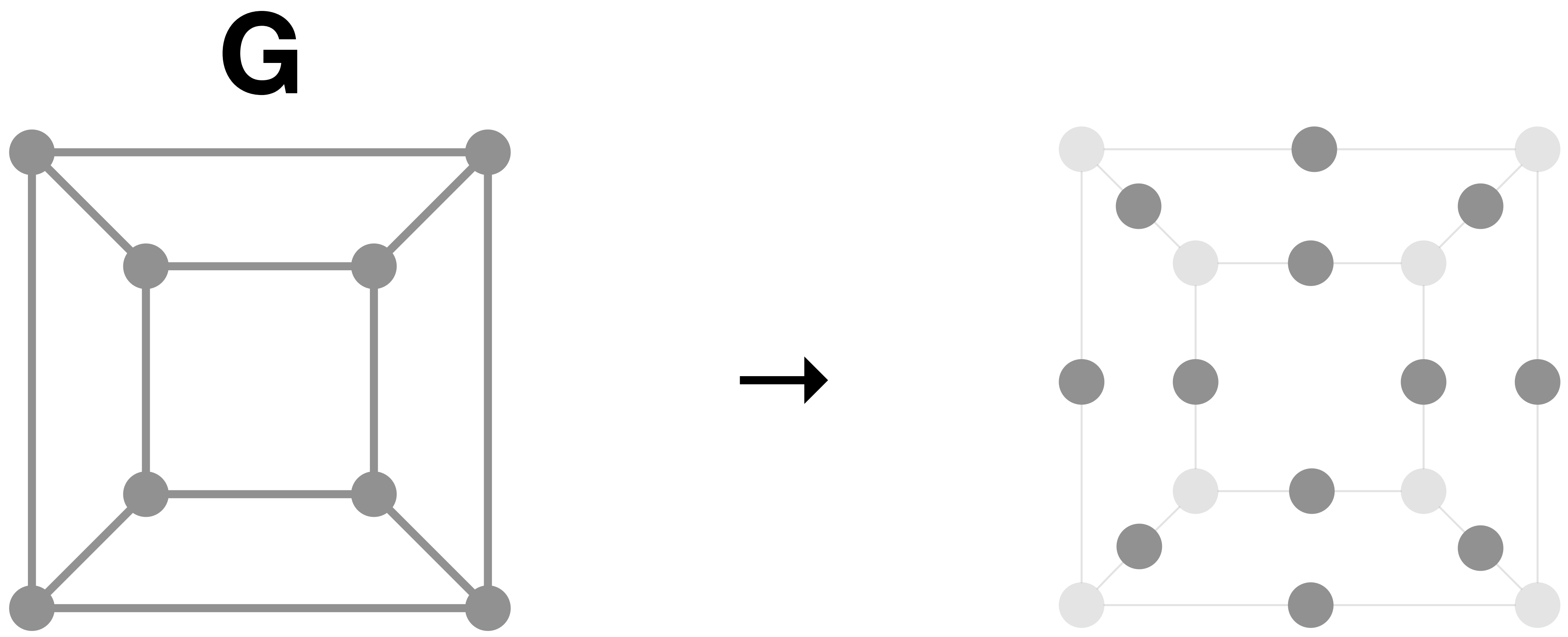
Lower bound for MM implies lower bound for MIS

An algorithm for **MIS** implies an algorithm for **MM**



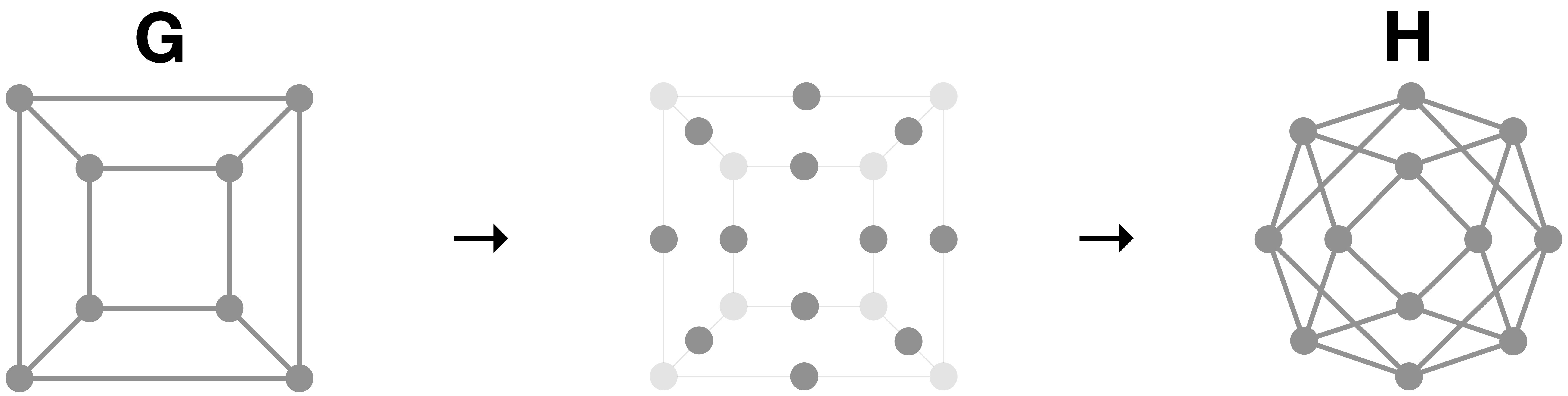
Lower bound for MM implies lower bound for MIS

An algorithm for **MIS** implies an algorithm for **MM**



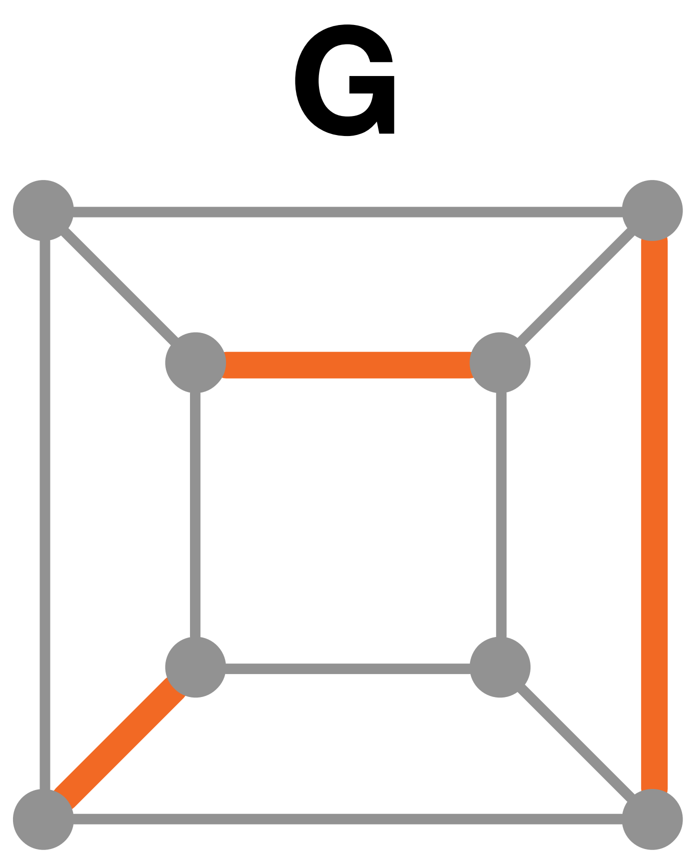
Lower bound for MM implies lower bound for MIS

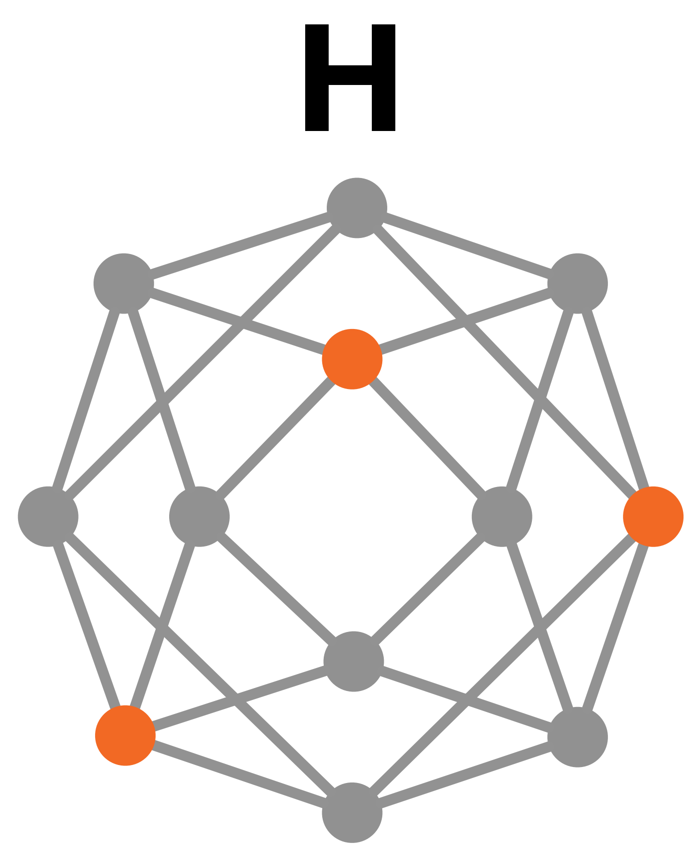
An algorithm for **MIS** implies an algorithm for **MM**



Lower bound for MM implies lower bound for MIS

An algorithm for **MIS** implies an algorithm for **MM**





If we **cannot solve MM in $o(\Delta)$**, then we **cannot solve MIS in $o(\Delta)$**

Proof techniques

Round elimination

Round elimination technique

- **Given:**
  - algorithm A_0 solves problem P_0 in T rounds
- **We construct:**
  - algorithm A_1 solves problem P_1 in $T - 1$ rounds
  - algorithm A_2 solves problem P_2 in $T - 2$ rounds
  - algorithm A_3 solves problem P_3 in $T - 3$ rounds

    …
  - algorithm A_T solves problem P_T in 0 rounds
- But P_T is nontrivial, so A_0 cannot exist

Linial (1987, 1992): coloring cycles

- **Given:**
  - algorithm A_0 solves 3-coloring in $T = o(\log^* n)$ rounds
- **We construct:**
  - algorithm A_1 solves 2^3-coloring in $T - 1$ rounds
  - algorithm A_2 solves 2^{2^3}-coloring in $T - 2$ rounds
  - algorithm A_3 solves $2^{2^{2^3}}$-coloring in $T - 3$ rounds

  …
  - algorithm A_T solves $o(n)$-coloring in 0 rounds
- But $o(n)$-coloring is nontrivial, so A_0 cannot exist

Linial (1987, 1992): coloring cycles

- **Given:**
  - algorithm A_0 solves 3-coloring in $T = o(\log^* n)$ rounds
- **We construct:**
  - algorithm A_1 solves 2^3-coloring in $T - 1$ rounds
  - algorithm A_2 solves 2^{2^3}-coloring in $T - 2$ rounds
  - algorithm A_3 solves $2^{2^{2^3}}$-coloring in $T - 3$ rounds

  …
  - algorithm A_T solves $o(n)$-coloring in 0 rounds
- But $o(n)$-coloring is nontrivial, so A_0 cannot exist

Challenge: discover P_i

Round elimination technique

- **Given:**
  - algorithm A_0 solves problem P_0 in T rounds
- **We construct:**
  - algorithm A_1 solves problem P_1 in $T - 1$ rounds
  - algorithm A_2 solves problem P_2 in $T - 2$ rounds
  - algorithm A_3 solves problem P_3 in $T - 3$ rounds

    …
  - algorithm A_T solves problem P_T in 0 rounds
- But P_T is nontrivial, so A_0 cannot exist

P_i can be found automatically

[Brandt, 2019]

Round elimination technique

- **Given:**
  - algorithm A_0 solves problem P_0 in T rounds
- **We construct:**
  - algorithm A_1 solves problem P_1 in $T - 1$ rounds
  - algorithm A_2 solves problem P_2 in $T - 2$ rounds
  - algorithm A_3 solves problem P_3 in $T - 3$ rounds

    …
  - algorithm A_T solves problem P_T in 0 rounds
- But P_T is nontrivial, so A_0 cannot exist

Challenge:
keep P_i small

Round elimination technique for MM

- **Given:**
  - algorithm $\boldsymbol{A_0}$ solves problem $\boldsymbol{P_0}$ = **maximal matching** in $\boldsymbol{T}$ rounds
- **We construct:**
  - algorithm $\boldsymbol{A_1}$ solves problem $\boldsymbol{P_1}$ in $\boldsymbol{T - 1}$ rounds
  - algorithm $\boldsymbol{A_2}$ solves problem $\boldsymbol{P_2}$ in $\boldsymbol{T - 2}$ rounds
  - algorithm $\boldsymbol{A_3}$ solves problem $\boldsymbol{P_3}$ in $\boldsymbol{T - 3}$ rounds

    …
  - algorithm $\boldsymbol{A_T}$ solves problem $\boldsymbol{P_T}$ in **0** rounds
- But $\boldsymbol{P_T}$ is nontrivial, so $\boldsymbol{A_0}$ cannot exist

What are these problems P_i here?

General approach

What we really care about

Maximal matching in $o(\Delta)$ rounds

General approach

Maximal matching in $o(\Delta)$ rounds

→ **"$\Delta^{1/2}$ matching"** in $o(\Delta^{1/2})$ rounds

What we really care about

k-matching: select at most k edges per node

General approach

Maximal matching in $o(\Delta)$ rounds

→ **"$\Delta^{1/2}$ matching"** in $o(\Delta^{1/2})$ rounds

→ $P(\Delta^{1/2}, 0)$ in $o(\Delta^{1/2})$ rounds

What we really care about

k-matching: select at most k edges per node

General approach

Maximal matching in $o(\Delta)$ rounds

→ **"$\Delta^{1/2}$ matching"** in $o(\Delta^{1/2})$ rounds

→ $P(\Delta^{1/2}, 0)$ in $o(\Delta^{1/2})$ rounds

What we really care about

k-matching: select at most k edges per node

Apply round elimination $o(\Delta^{1/2})$ times

General approach

Maximal matching in $o(\Delta)$ rounds

→ "$\Delta^{1/2}$ matching" in $o(\Delta^{1/2})$ rounds

→ $P(\Delta^{1/2}, 0)$ in $o(\Delta^{1/2})$ rounds

→ $P(O(\Delta^{1/2}), o(\Delta))$ in 0 rounds

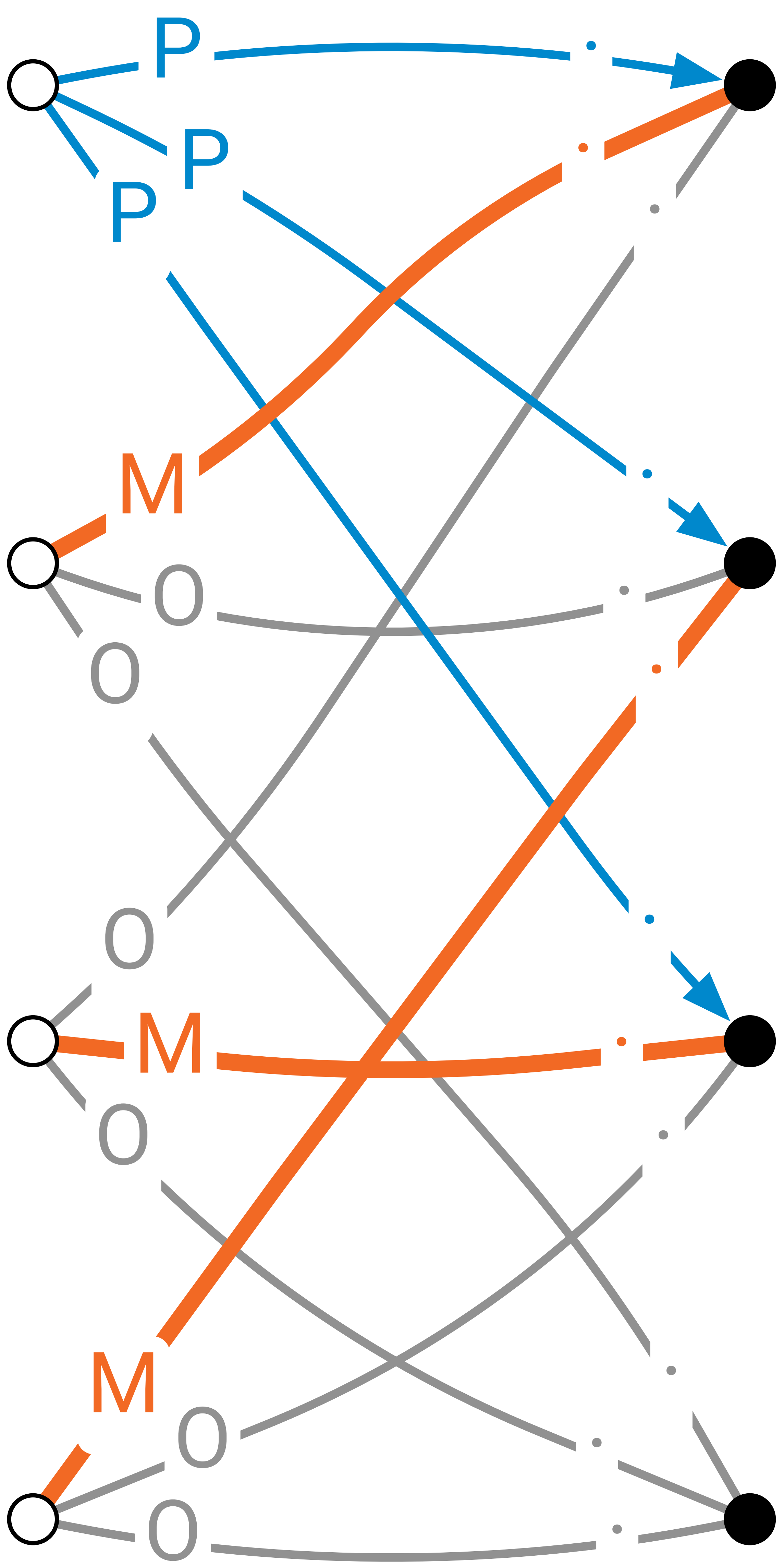
→ contradiction

What we really care about

k-matching: select at most k edges per node

Apply round elimination $o(\Delta^{1/2})$ times

Representation for maximal matchings



M = "matched"
P = "pointer to matched"
O = "other"

white nodes "active"

output one of these:
· 1 × M and (Δ−1) × O
· Δ × P

$$W = \mathsf{MO}^{\Delta-1} \mid \mathsf{P}^{\Delta}$$

black nodes "passive"

accept one of these:
· 1 × M and **(Δ−1) × {P, O}**
· **Δ × O**

$$B = \mathsf{M[PO]}^{\Delta-1} \mid \mathsf{O}^{\Delta}$$

Parametrized problem family

$$W = \mathsf{MO}^{\Delta-1} \mid \mathsf{P}^{\Delta},$$
$$B = \mathsf{M[PO]}^{\Delta-1} \mid \mathsf{O}^{\Delta}$$

maximal matching

$$W_{\Delta}(x,y) = \left(\mathsf{MO}^{d-1} \,\middle|\, \mathsf{P}^{d}\right)\mathsf{O}^{y}\mathsf{X}^{x},$$
$$B_{\Delta}(x,y) = \left([\mathsf{MX}][\mathsf{POX}]^{d-1} \,\middle|\, [\mathsf{OX}]^{d}\right)[\mathsf{POX}]^{y}[\mathsf{MPOX}]^{x},$$
$$d = \Delta - x - y$$

“weak” matching

Parametrized problem family

$$W = \mathsf{MO}^{\Delta-1} \mid \mathsf{P}^{\Delta},$$

$$B = \mathsf{M[PO]}^{\Delta-1} \mid \mathsf{O}^{\Delta}$$

maximal matching

$$W_{\Delta}(x, y) = \left(\mathsf{MO}^{d-1} \mid \mathsf{P}^{d}\right)\mathsf{O}^{y}\mathsf{X}^{x},$$

“weak” matching

A node v can be matched with at most x neighbours
If v is not matched, at most y neighbours can be unmatched

Main Lemma

- Given: ***A*** solves ***P*(*x*, *y*)** in ***T*** rounds
- We can construct: ***A'*** solves ***P*(*x* + 1, *y* + *x*)** in ***T* − 1** rounds

$$W_\Delta(x,y) = \left(\mathsf{MO}^{d-1} \,\middle|\, \mathsf{P}^{d}\right)\mathsf{O}^{y}\mathsf{X}^{x},$$

$$B_\Delta(x,y) = \left([\mathsf{MX}][\mathsf{POX}]^{d-1} \,\middle|\, [\mathsf{OX}]^{d}\right)[\mathsf{POX}]^{y}[\mathsf{MPOX}]^{x},$$

$$d = \Delta - x - y$$

Putting things together

Proof technique does not work directly with unique IDs

- Basic version:
  - deterministic lower bound, ***port-numbering model***
- Analyze what happens to local failure probability:
  - ***randomized*** lower bound, port-numbering model
- With randomness you can construct unique identifiers w.h.p.:
  - randomized lower bound, ***LOCAL model***
- Fast deterministic → very fast randomized
  - stronger ***deterministic*** lower bound, LOCAL model

Summary

- ***Linear-in-Δ lower bounds*** for **maximal matchings** and **maximal independent sets**
- Old: can be solved in $O(\Delta + \log^* n)$ rounds
- New: cannot be solved in
  - $o(\Delta + \log \log n / \log \log \log n)$ rounds with randomized algorithms
  - $o(\Delta + \log n / \log \log n)$ rounds with deterministic algorithms
- Technique: ***round elimination***

Round eliminator: example MM

- Round eliminator program link:

  https://users.aalto.fi/~olivetd1/round-eliminator

- An example of maximal matching on round eliminator:

  http://alkida.net/wp-content/uploads/2019/11/Round-Eliminator-.mov

Some open questions

- Complexity of **$(\Delta+1)$*-vertex coloring***?
  - can be solved in $\tilde{O}(\Delta^{1/2}) + O(\log^* n)$ rounds [Fraigniaud et al., 2016]
  - cannot be solved in $o(\log^* n)$ rounds [Linial, 1987]
  - example: is it solvable in $O(\log \Delta + \log^* n)$ time?
- Better understanding of the **round elimination technique**

Some open questions

- Complexity of **(Δ+1)-*vertex coloring***?
  - can be solved in $\tilde{O}(\Delta^{1/2}) + O(\log^* n)$ rounds [Fraigniaud et al., 2016]
  - cannot be solved in $o(\log^* n)$ rounds [Linial, 1987]
  - example: is it solvable in $O(\log \Delta + \log^* n)$ time?
- Better understanding of the **round elimination technique**

Thank you!