Lower bounds for maximal matchings and maximal independent sets

Jukka Suomela

Aalto University, Finland

arXiv:1901.02441

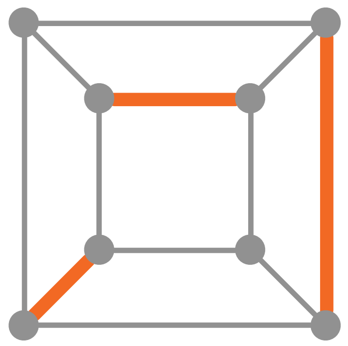
Joint work with

- **Alkida Balliu** · Aalto University
- **Sebastian Brandt** · ETH Zurich
- **Juho Hirvonen** · Aalto University
- **Dennis Olivetti** · Aalto University
- **Mikaël Rabie** · Aalto University and IRIF, University Paris Diderot

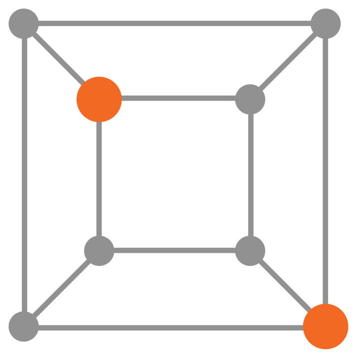
arXiv:1901.02441

Two classical graph problems

Maximal matching



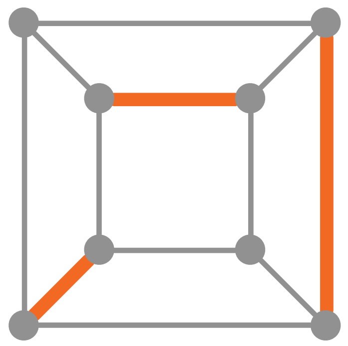
Maximal independent set



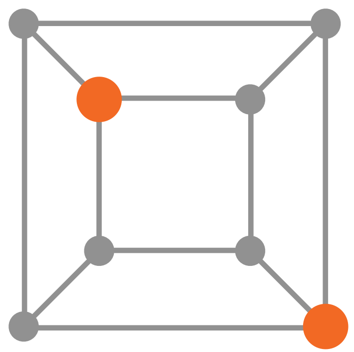
Trivial linear-time centralized, sequential algorithm:
add edges/nodes until stuck

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***?

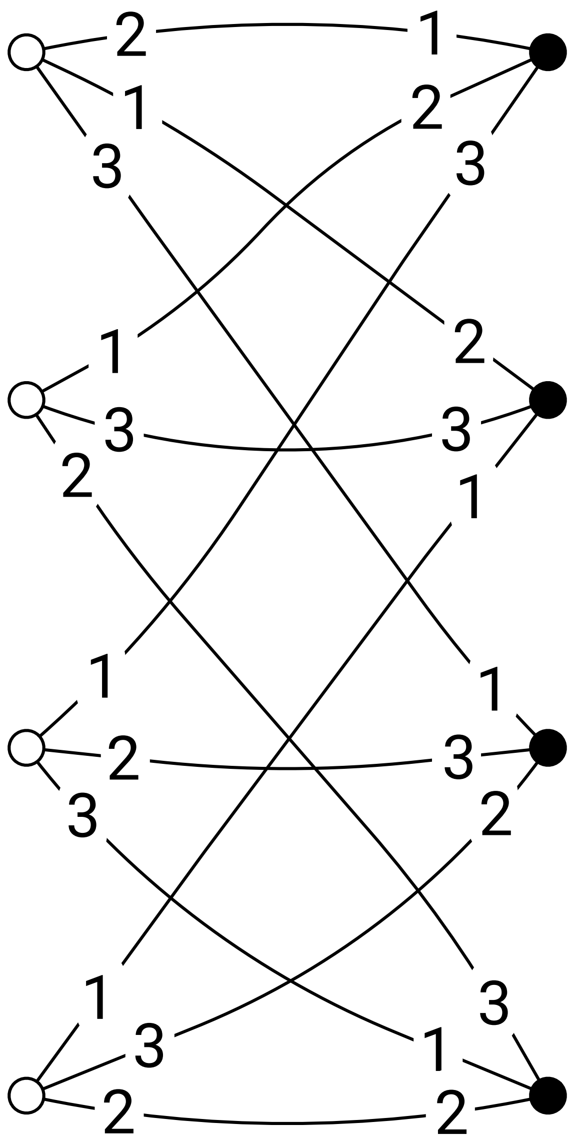
Warmup: toy example

Bipartite graphs & port-numbering model

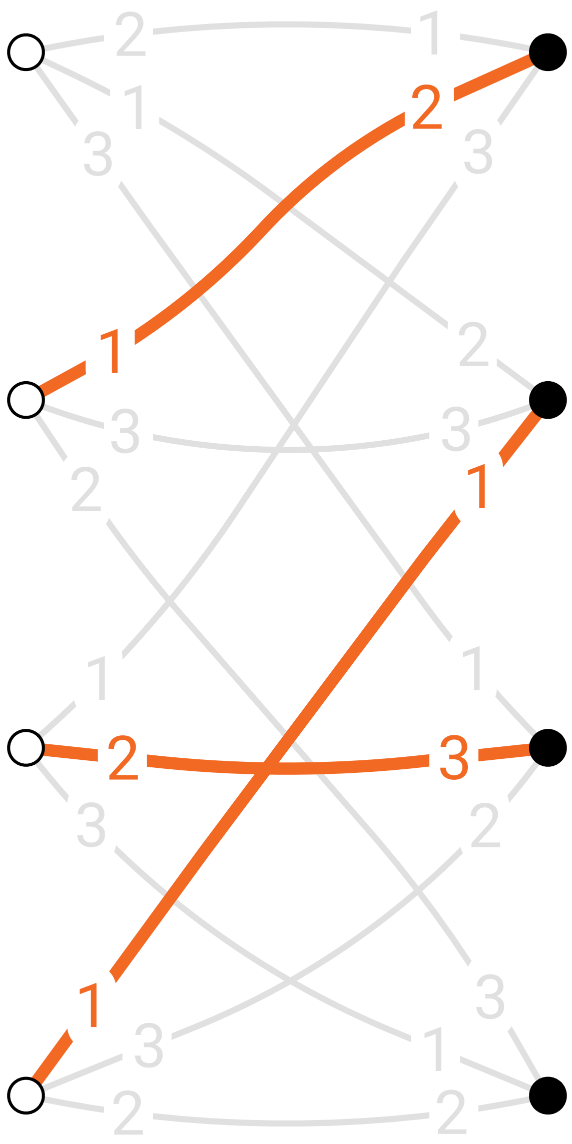
computer
network with
port numbering

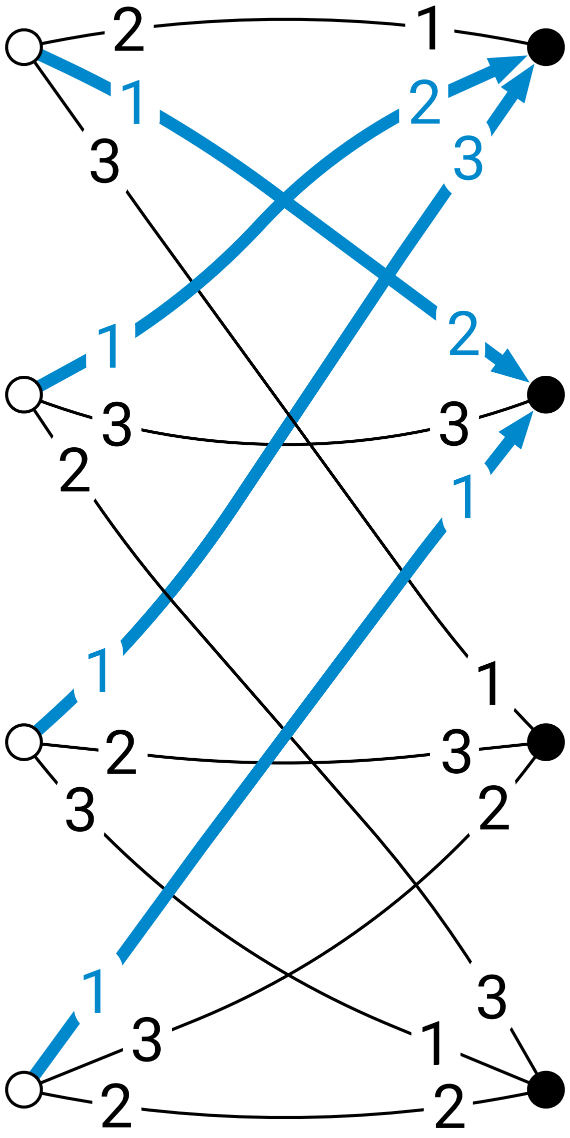
bipartite,
2-colored
graph

Δ-regular
(here $\Delta = 3$)



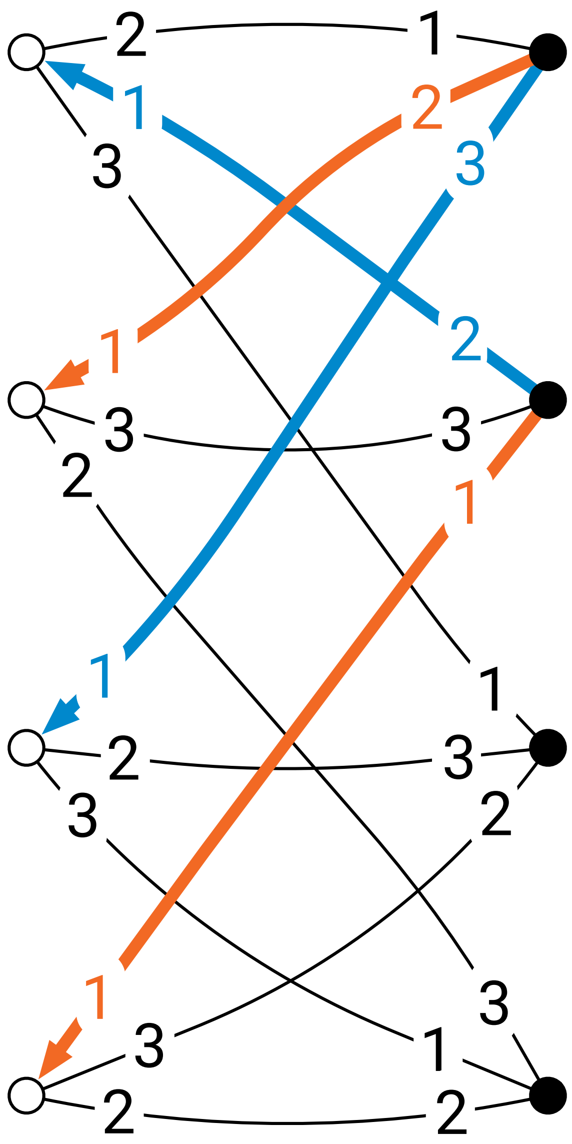
output:
maximal matching





Very simple algorithm

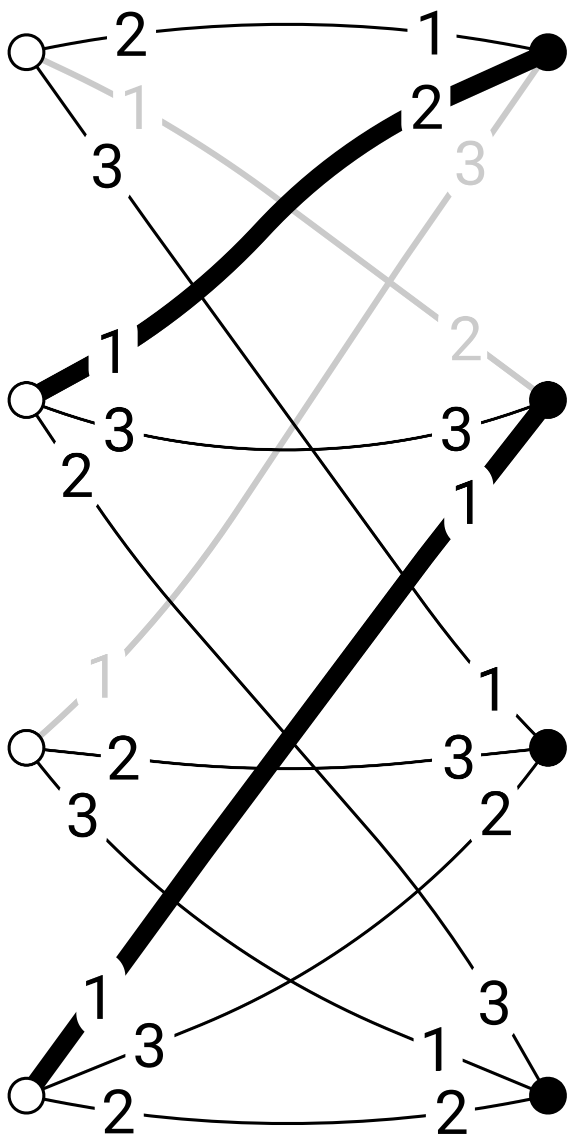
unmatched white nodes:
send ***proposal*** to port 1



Very simple algorithm

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

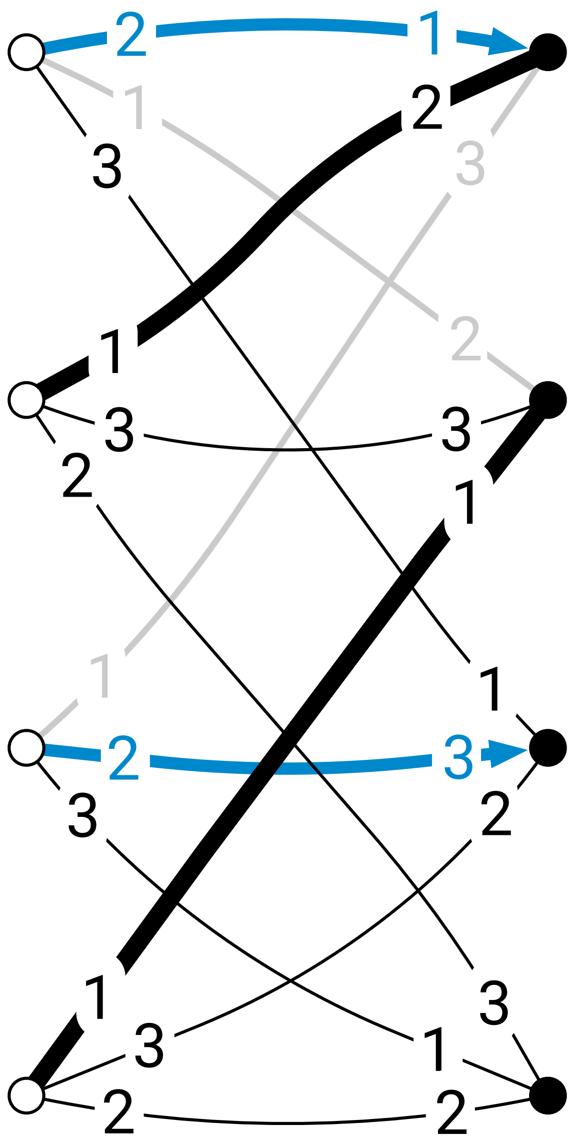
black nodes:
accept the first proposal you get, ***reject*** everything else
(break ties with port numbers)



Very simple algorithm

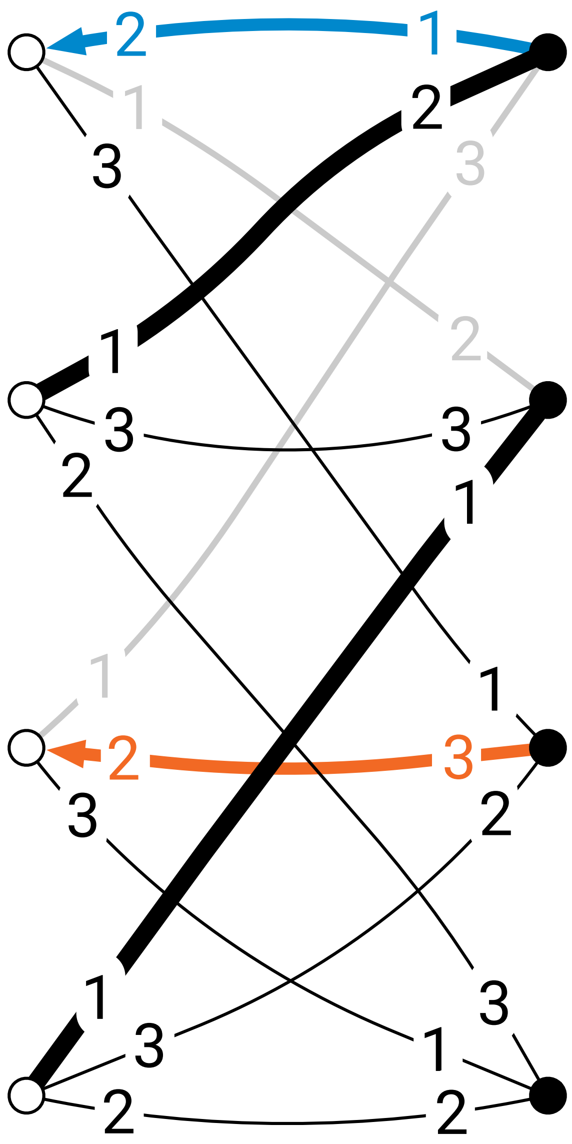
unmatched white nodes:
send ***proposal*** to port 1

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



Very simple algorithm

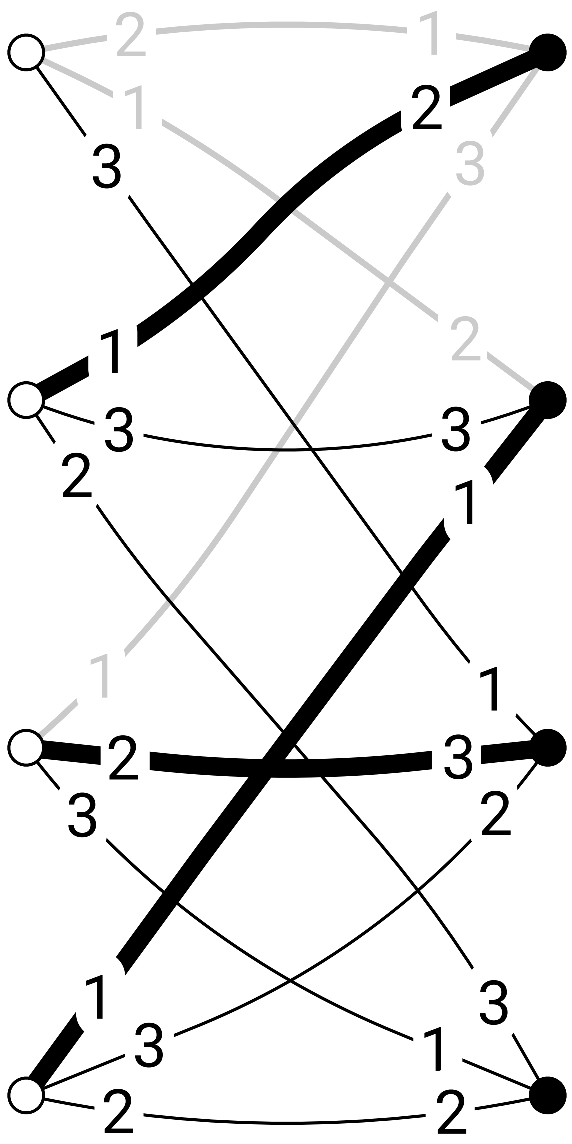
unmatched white nodes:
send ***proposal*** to port 2



Very simple algorithm

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

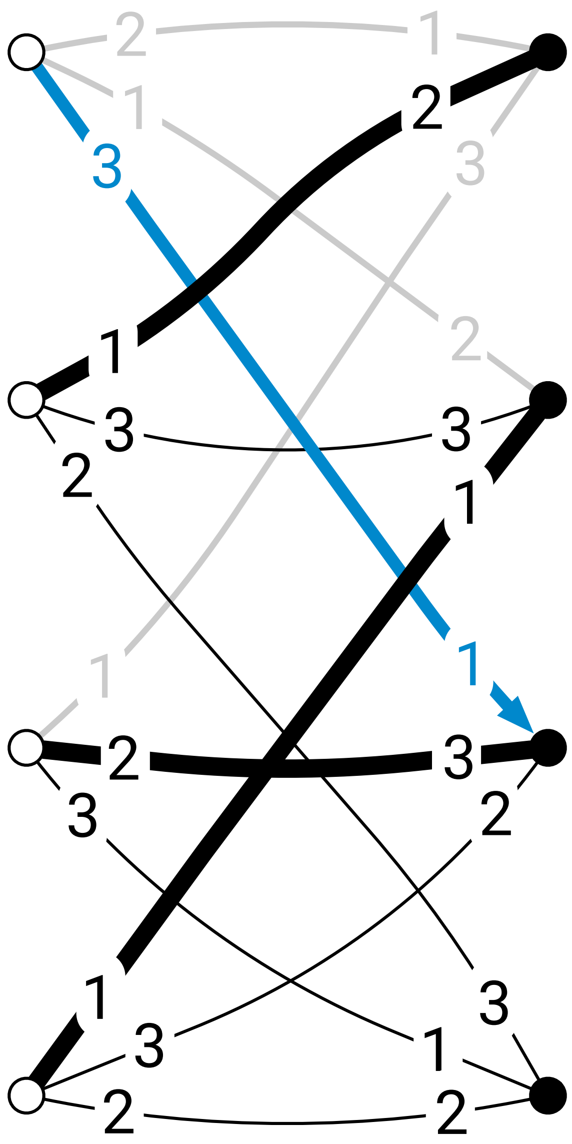
black nodes:
accept the first proposal you get, ***reject*** everything else
(break ties with port numbers)



Very simple algorithm

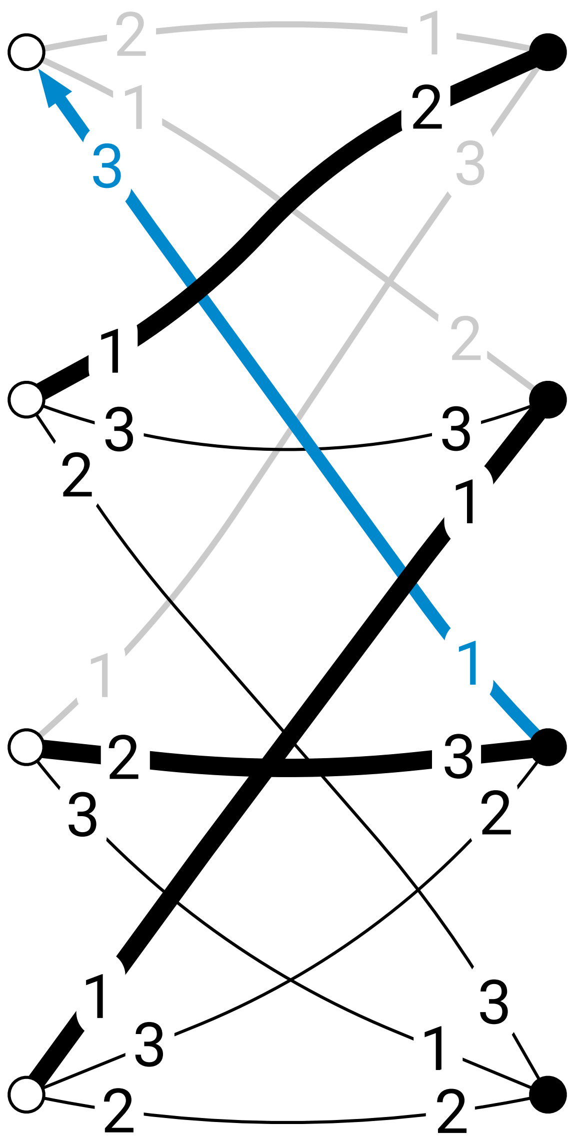
unmatched white nodes:
send ***proposal*** to port 2

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



Very simple algorithm

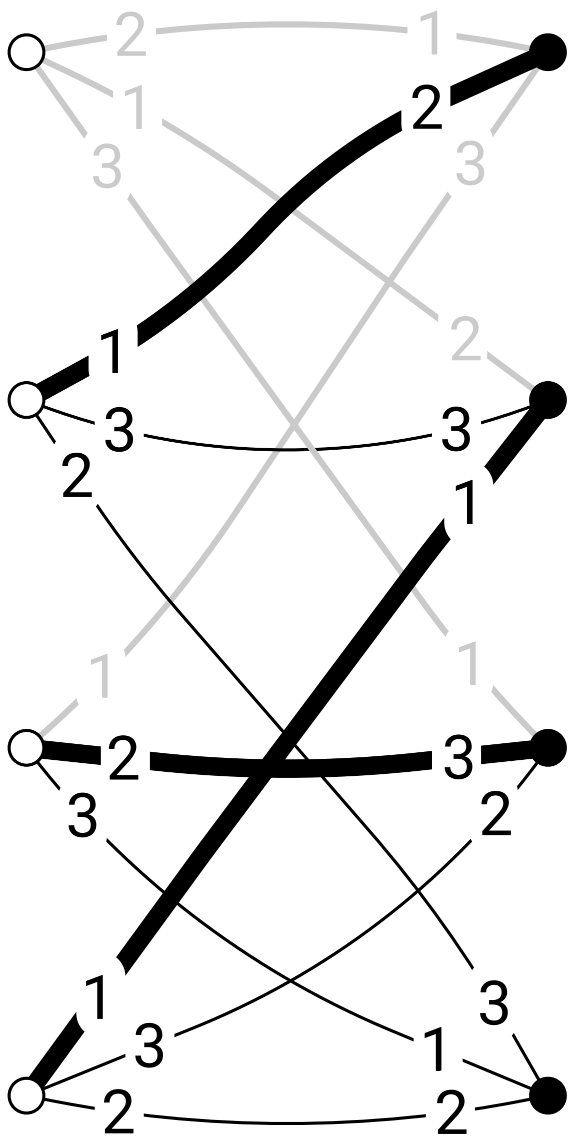
unmatched white nodes:
send ***proposal*** to port 3



Very simple algorithm

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

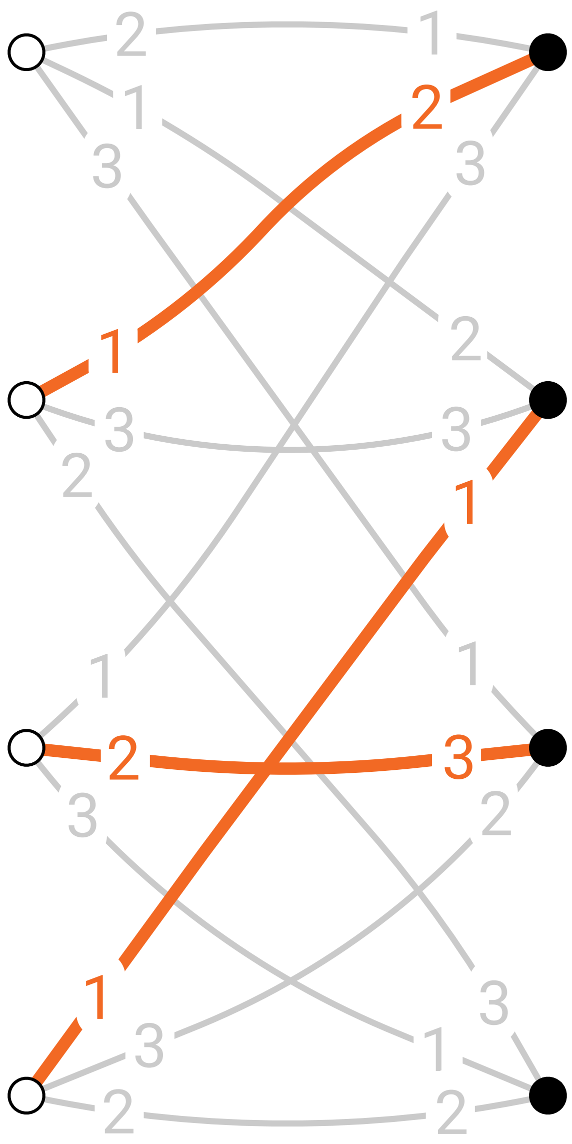
black nodes:
accept the first proposal you get, ***reject*** everything else (break ties with port numbers)



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

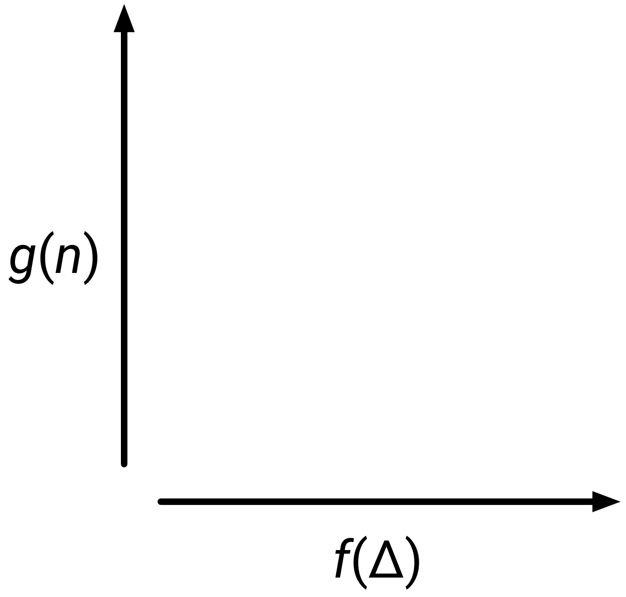
- Maximal matching in very large 2-colored Δ-regular graphs
- Simple algorithm: ***$O(\Delta)$ rounds***, independently of n
- ***Is this optimal?***
  - $o(\Delta)$ rounds?
  - $O(\log \Delta)$ rounds?
  - 4 rounds??

Big picture

Bounded-degree graphs & LOCAL model

Distributed graph algorithms for maximal matching

- Maximal matching in general graphs
  - ***n*** = number of nodes
  - **Δ** = maximum degree
- LOCAL model of distributed computing
  - ***"time"*** = number of synchronous communication rounds
    = ***how far*** do you need to see to choose your own part of solution
  - nodes are labeled with unique identifiers from { 1, 2, …, poly(n) }
  - ***O*(*n*)** = trivial, ***O*(diameter)** = trivial
- Strong model — lower bounds widely applicable



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

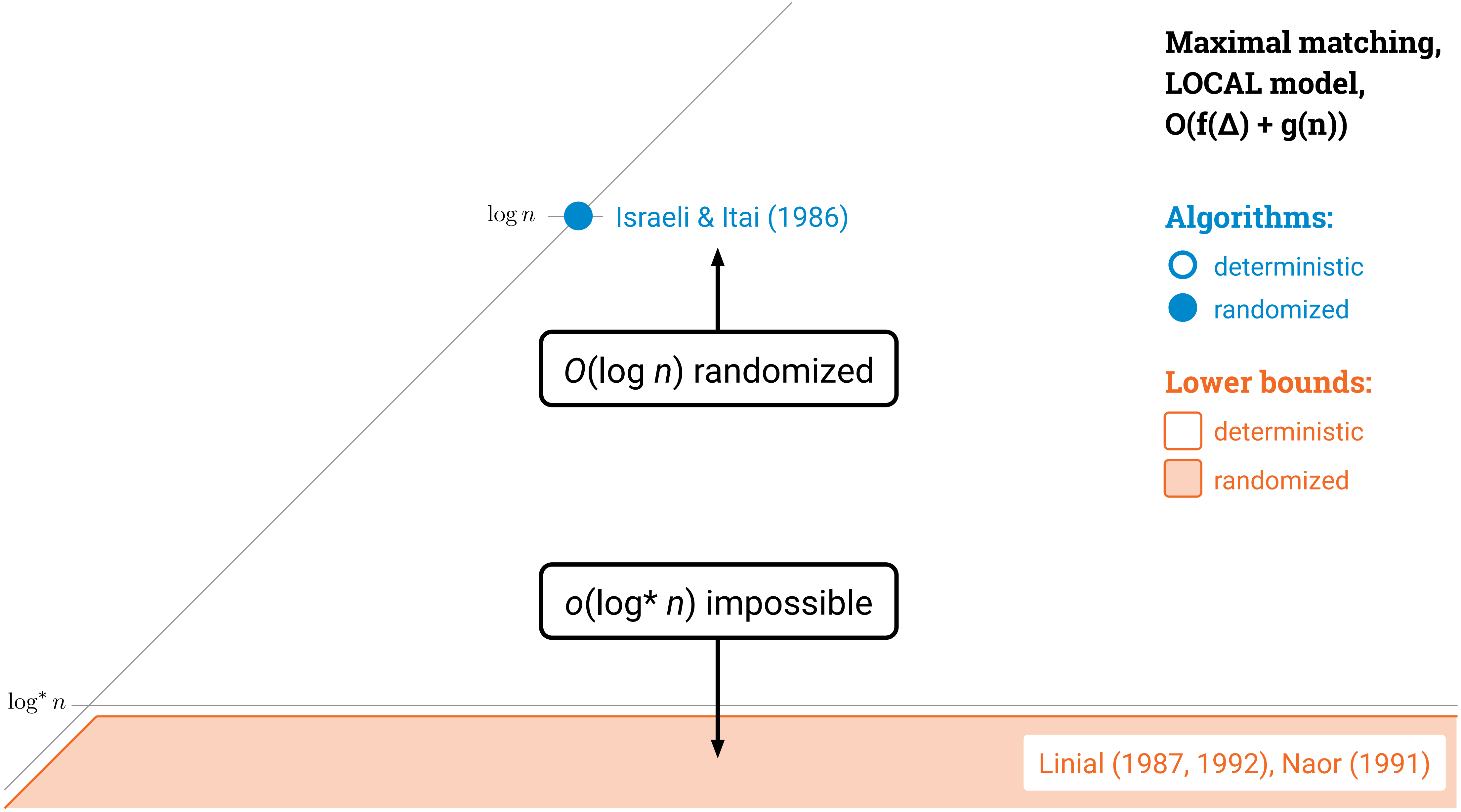
Algorithms:

- deterministic
- randomized

Lower bounds:

- deterministic
- randomized

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

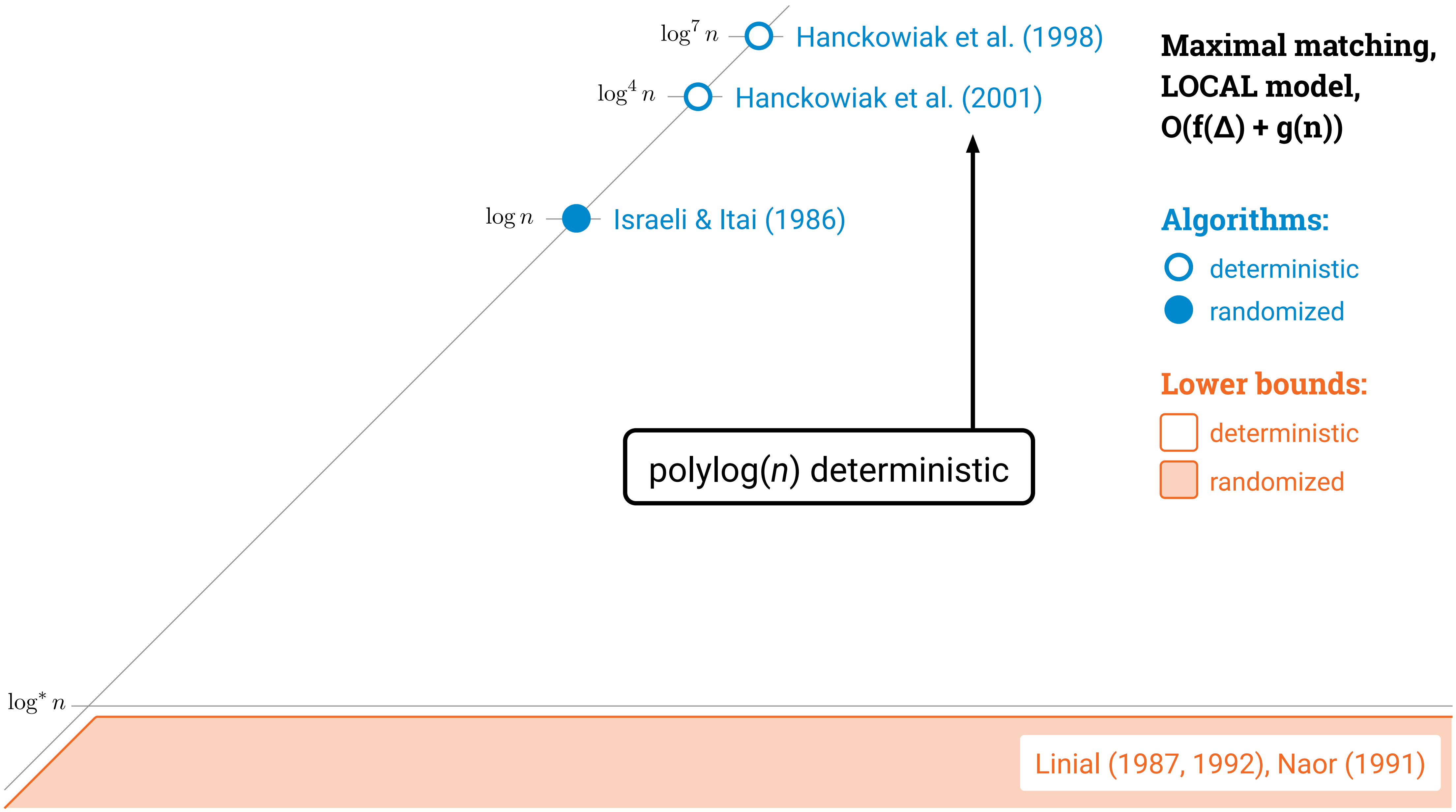


Algorithms:

- deterministic
- randomized

Lower bounds:

- deterministic
- randomized



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

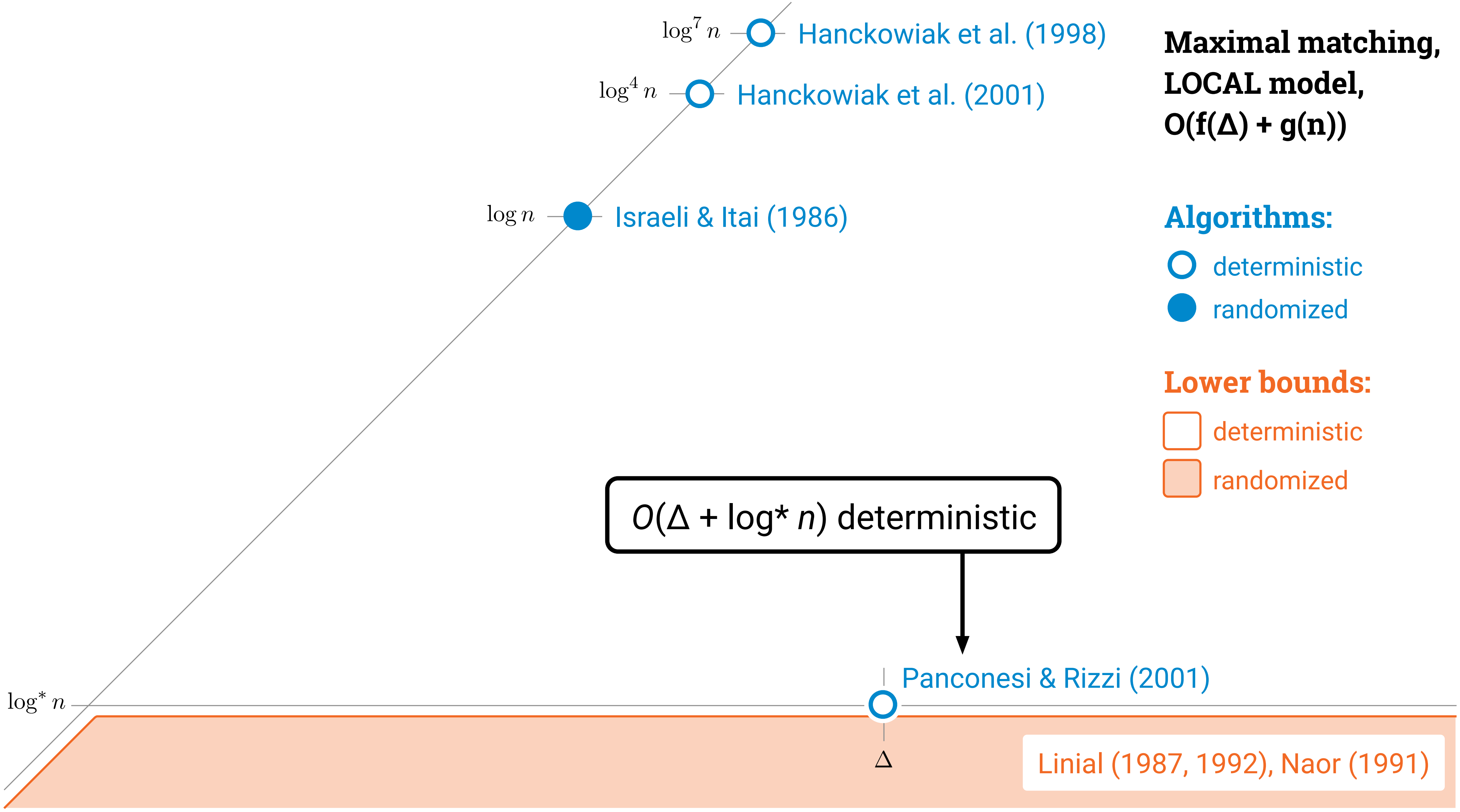
Algorithms:

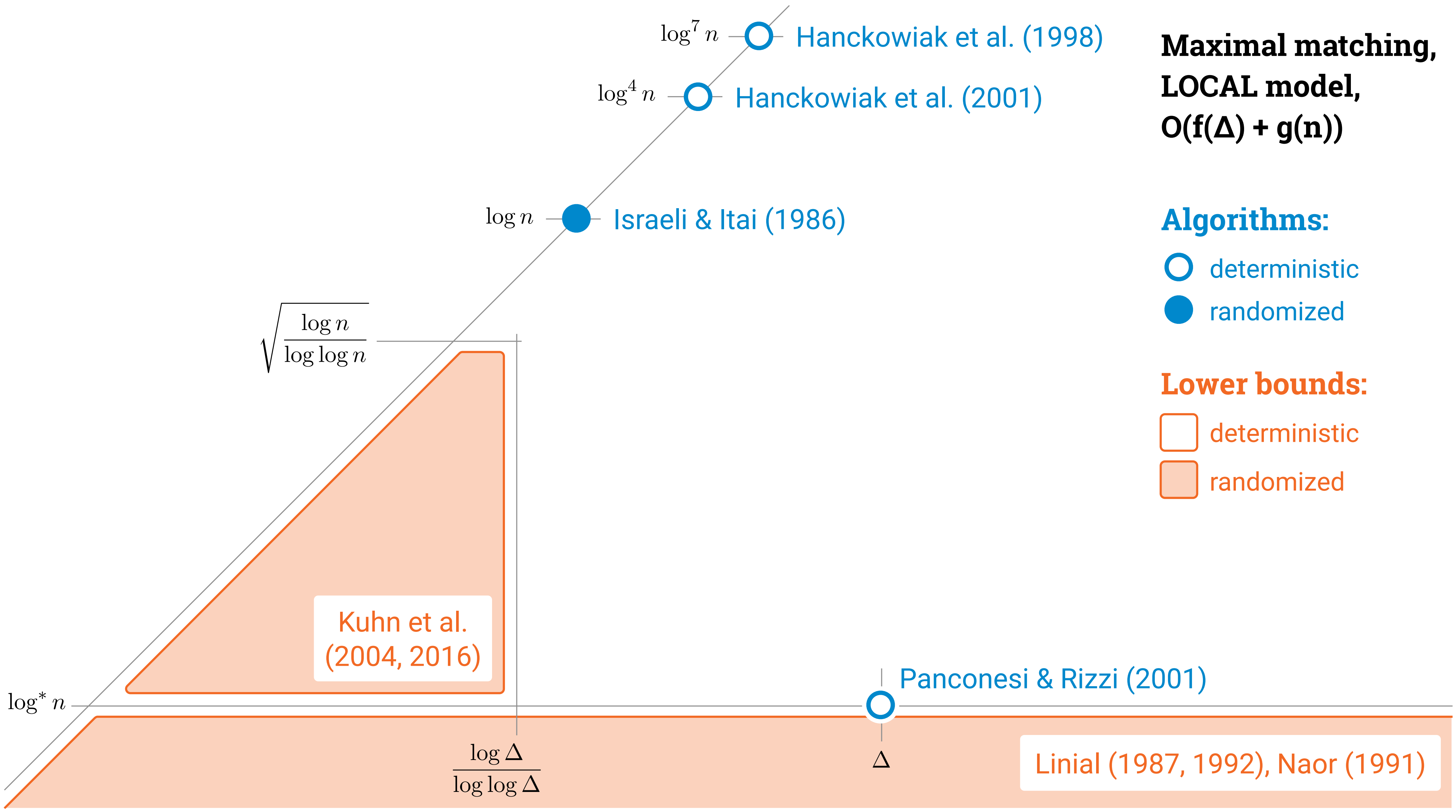
- deterministic
- randomized

Lower bounds:

- deterministic
- randomized

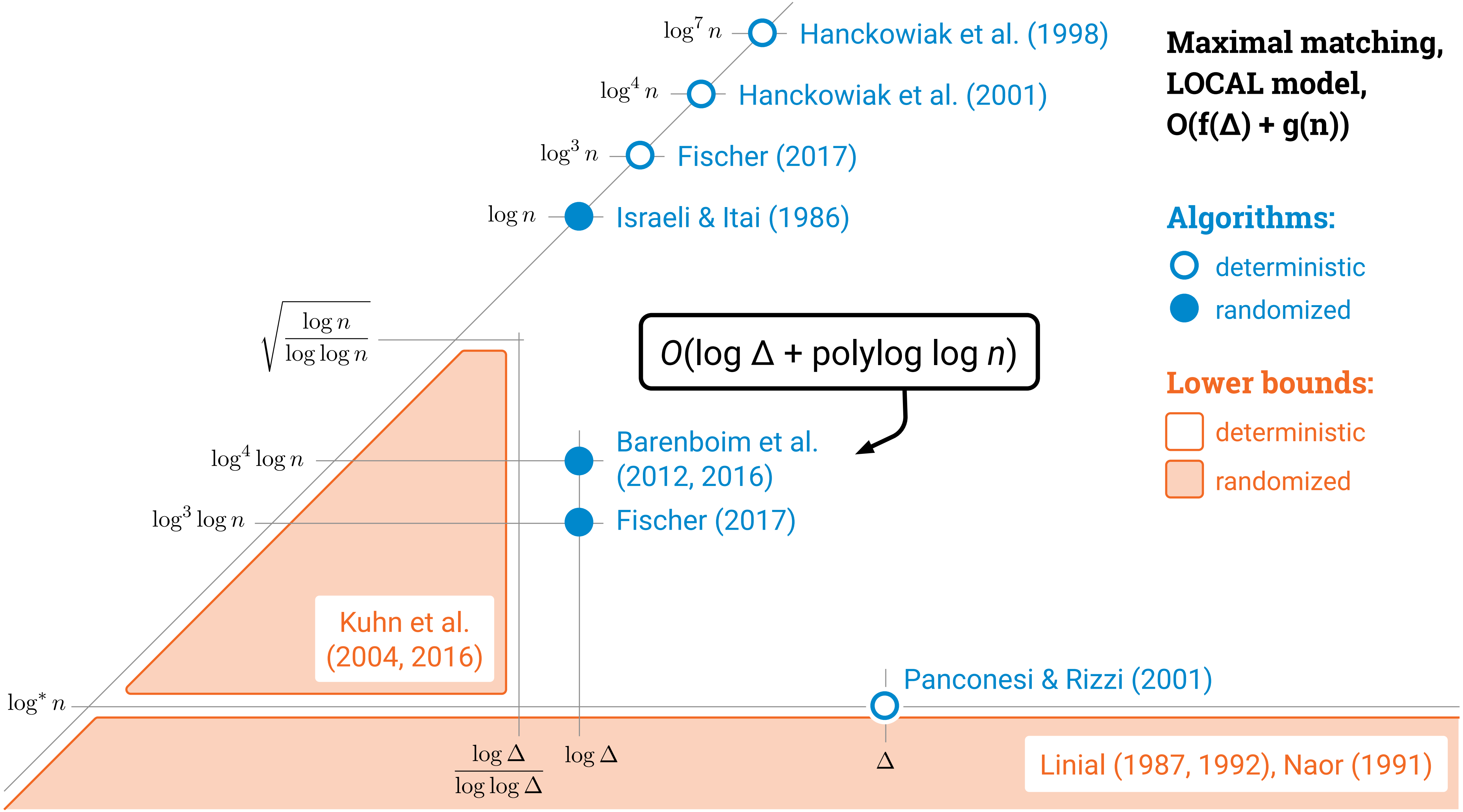
Maximal matching, LOCAL model, O(f(Δ) + g(n))

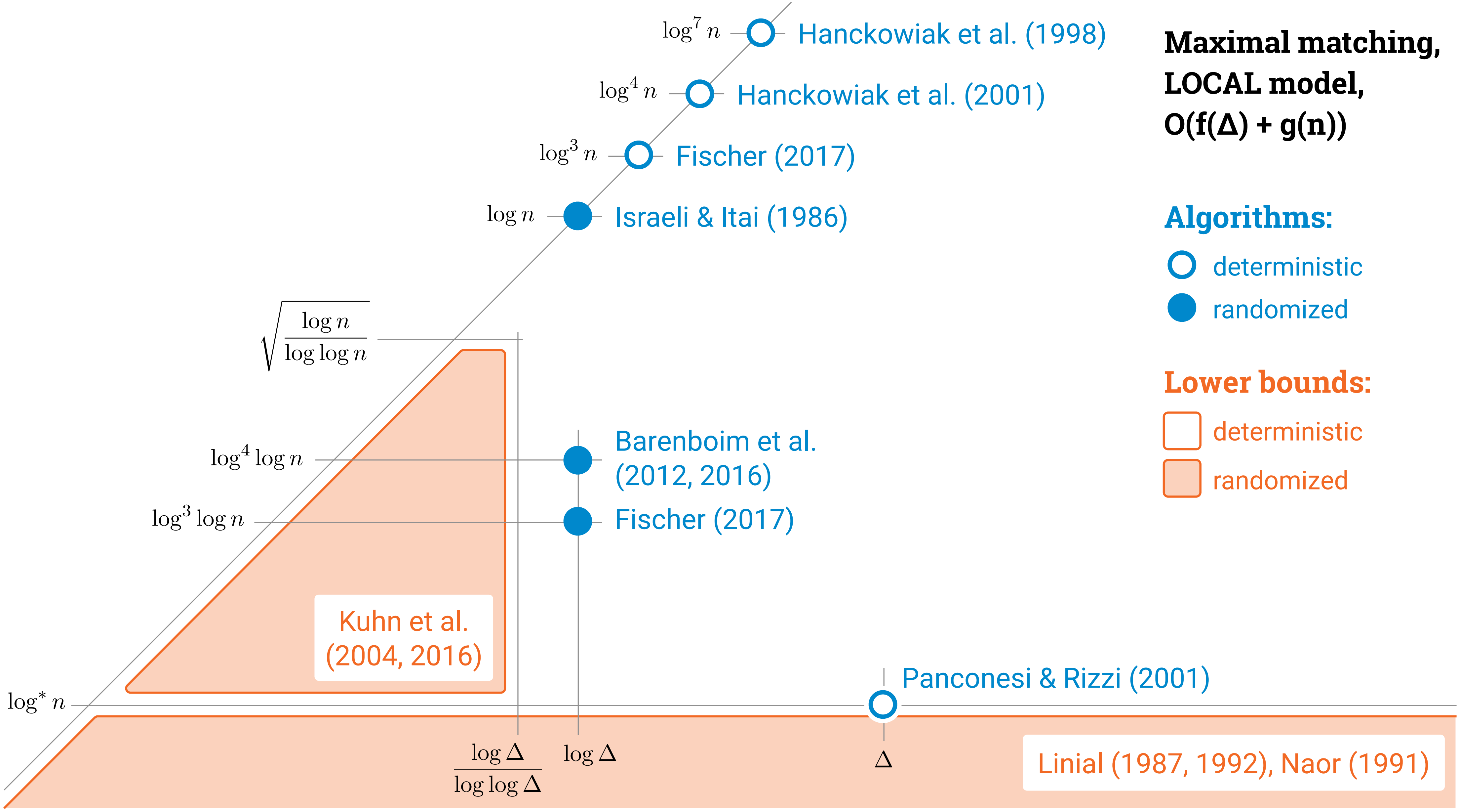




$\log^7 n$
Hanckowiak et al. (1998)
$\log^4 n$
Hanckowiak et al. (2001)
Maximal matching,
LOCAL model,
O(f(Δ) + g(n))
$\log n$
Israeli & Itai (1986)
Algorithms:
deterministic
randomized
$\sqrt{\frac{\log n}{\log \log n}}$
Lower bounds:
deterministic
randomized
Kuhn et al.
(2004, 2016)
Panconesi & Rizzi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

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





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

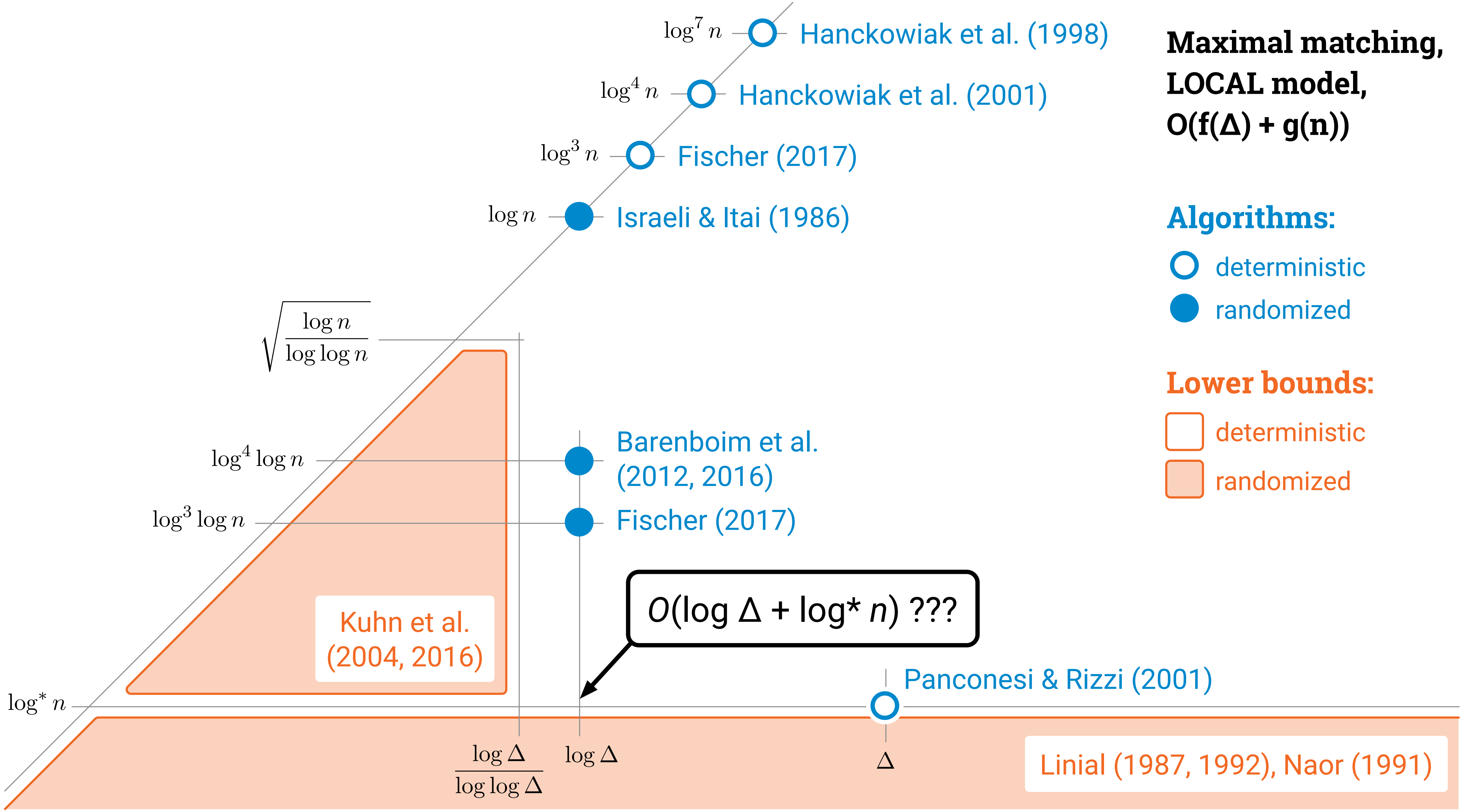
Algorithms:

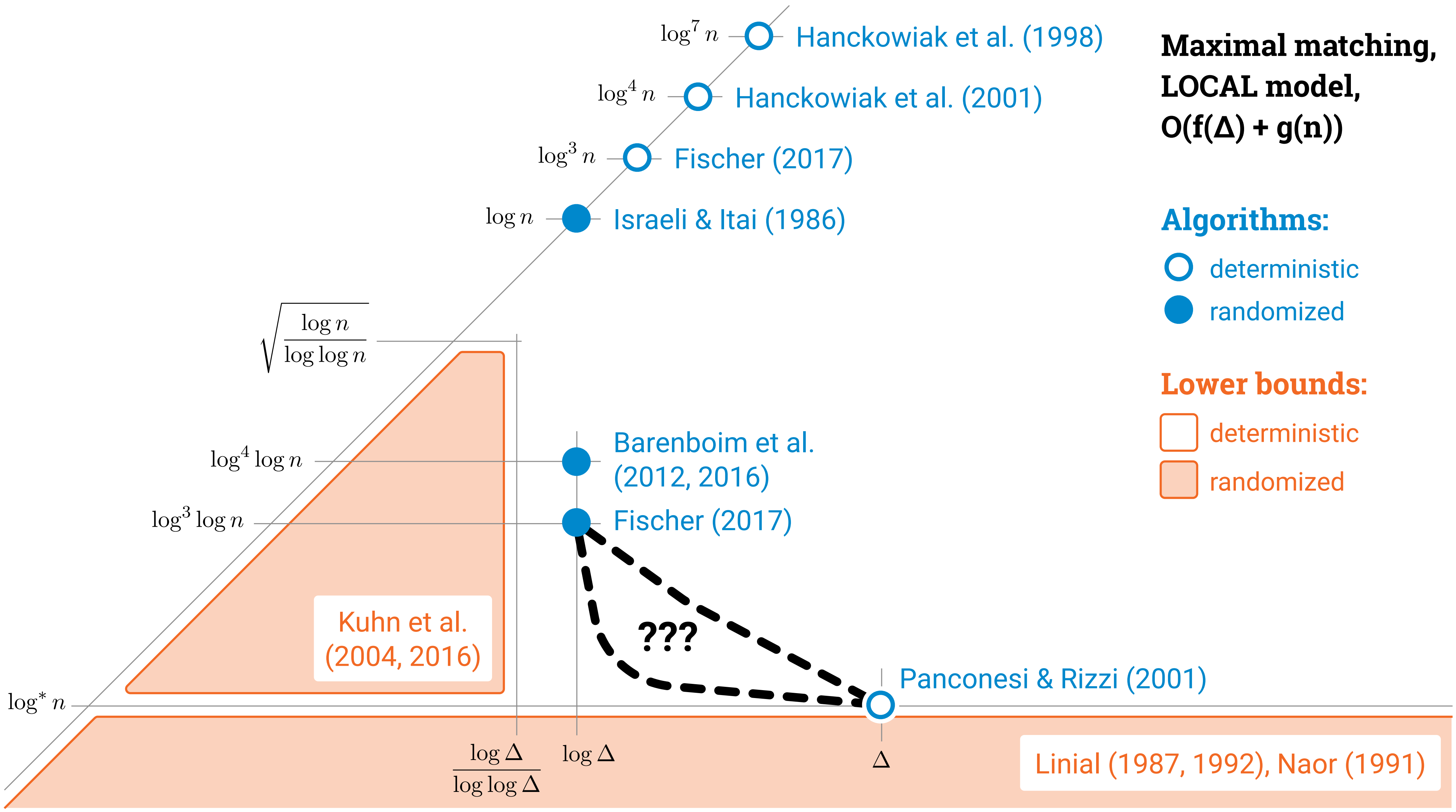
- deterministic
- randomized

Lower bounds:

- deterministic
- randomized

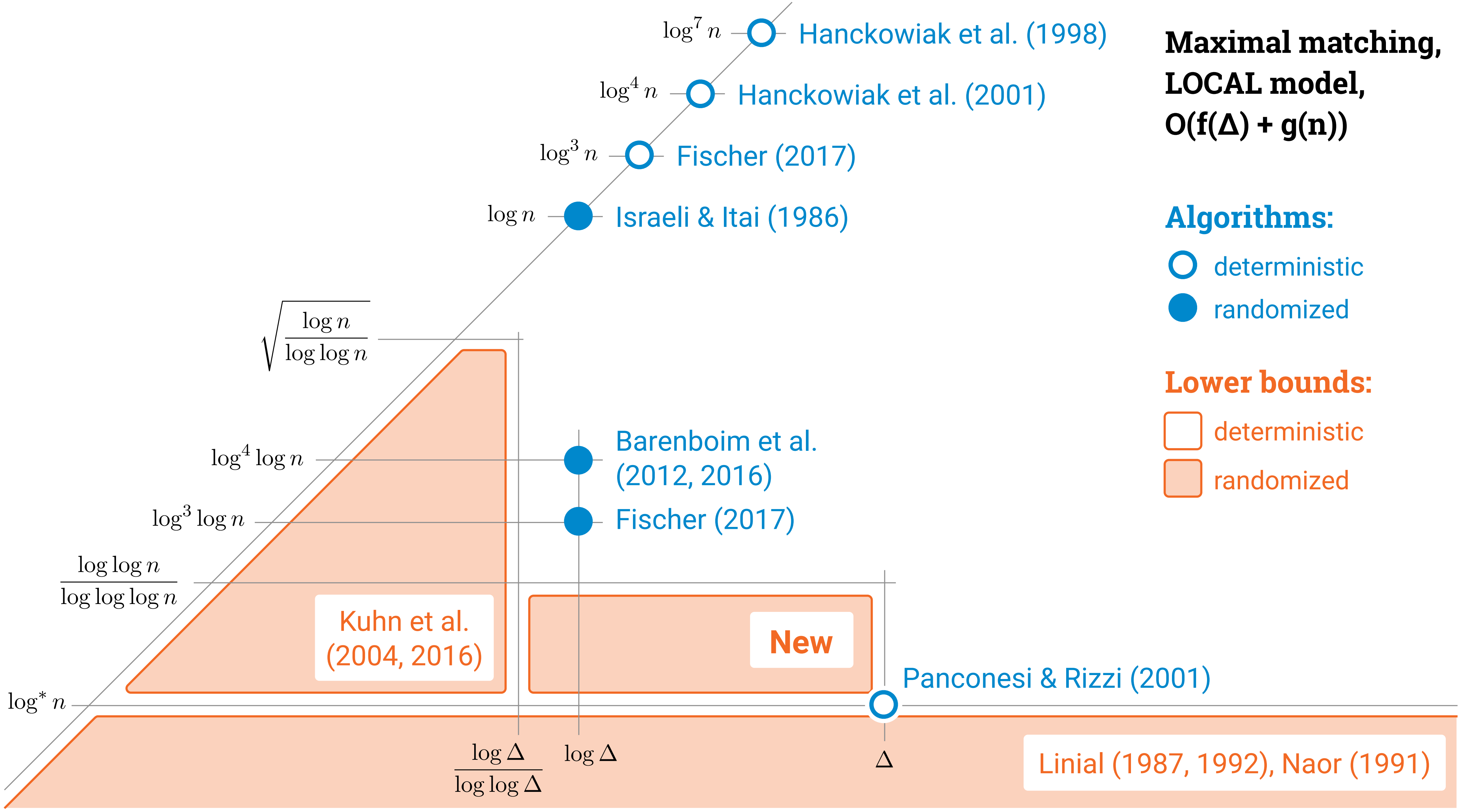
Maximal matching, LOCAL model, O(f(Δ) + g(n))



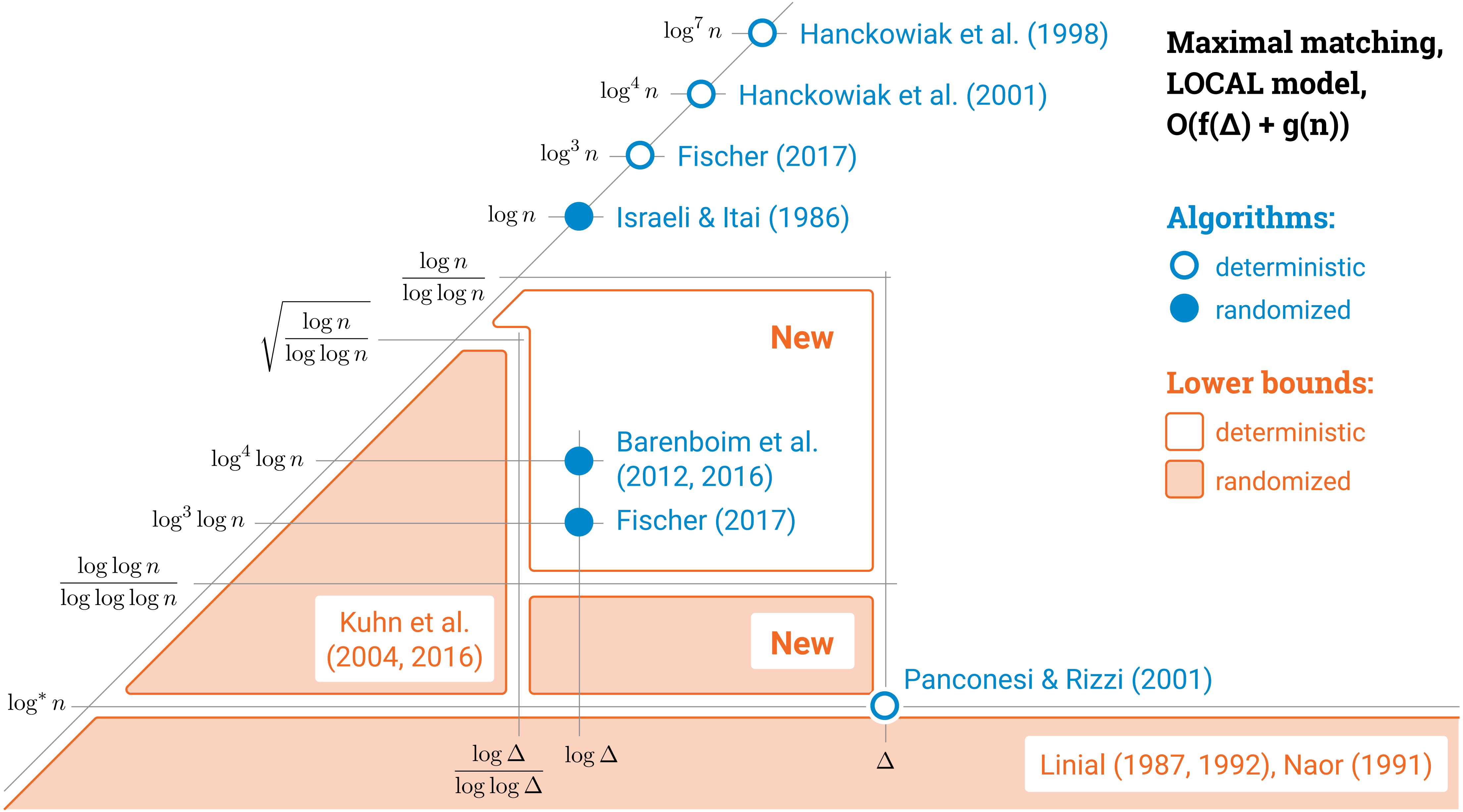


Maximal matching,
LOCAL model,
O(f(Δ) + g(n))
Algorithms:
deterministic
randomized
Lower bounds:
deterministic
randomized
$\log^7 n$
Hanckowiak et al. (1998)
$\log^4 n$
Hanckowiak et al. (2001)
$\log^3 n$
Fischer (2017)
$\log n$
Israeli & Itai (1986)
$\sqrt{\frac{\log n}{\log \log n}}$
$\log^4 \log n$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



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



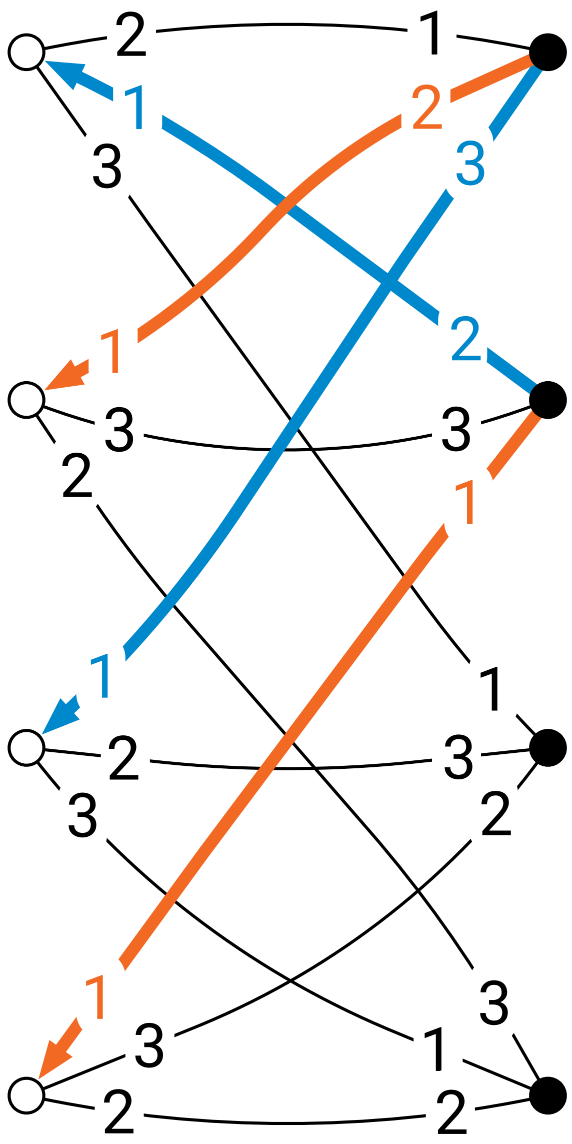
Main results

Maximal matching and **maximal independent set** cannot be solved in

- $o(\Delta + \log \log n / \log \log \log n)$ rounds with randomized algorithms
- $o(\Delta + \log n / \log \log n)$ rounds with deterministic algorithms

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)

Proof techniques

Speedup simulation

Speedup simulation 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

Brandt et al. (2016): sinkless orientation

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

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

Speedup simulation technique for maximal matching

- **Given:**
  - algorithm A_0 solves problem P_0 = **maximal matching** 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

What are the right problems P_i here?

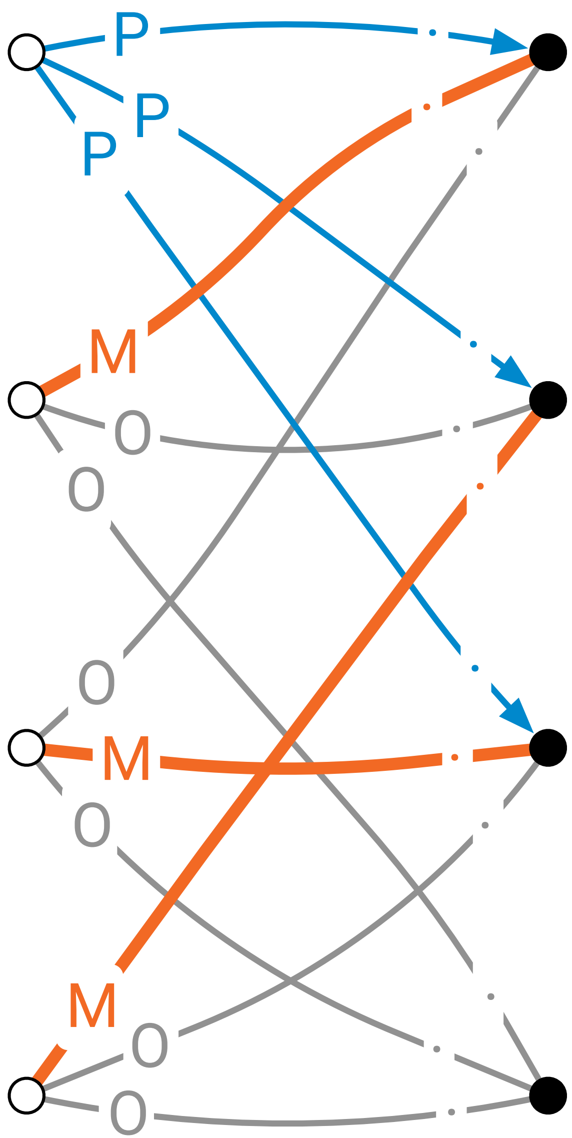
Speedup simulation technique for maximal matching

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

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

Let's start with P_0 …

Representation for maximal matchings



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

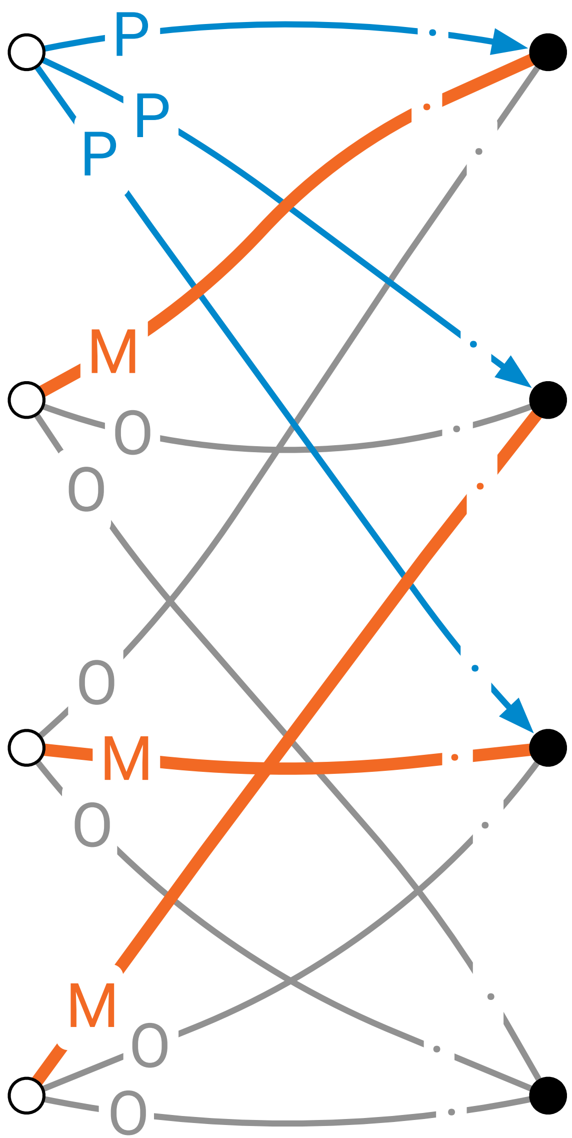
white nodes "active"

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

black nodes "passive"

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

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**

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

black nodes "passive"

accept one of these:

· **1 × M** and **(Δ−1) × {P, O}**
· **Δ × O**

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

Parameterized problem family

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

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

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

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

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

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

Putting things together

Maximal matching in ***o*(Δ)** 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*(Δ))** in **0** rounds

→ contradiction

What we really care about

k-matching: select at most k edges per node

Apply speedup simulation $o(\Delta^{1/2})$ times

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

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
- $o(\Delta + \log n / \log \log n)$ rounds with deterministic algorithms

Lower bound for MM implies a lower bound for MIS

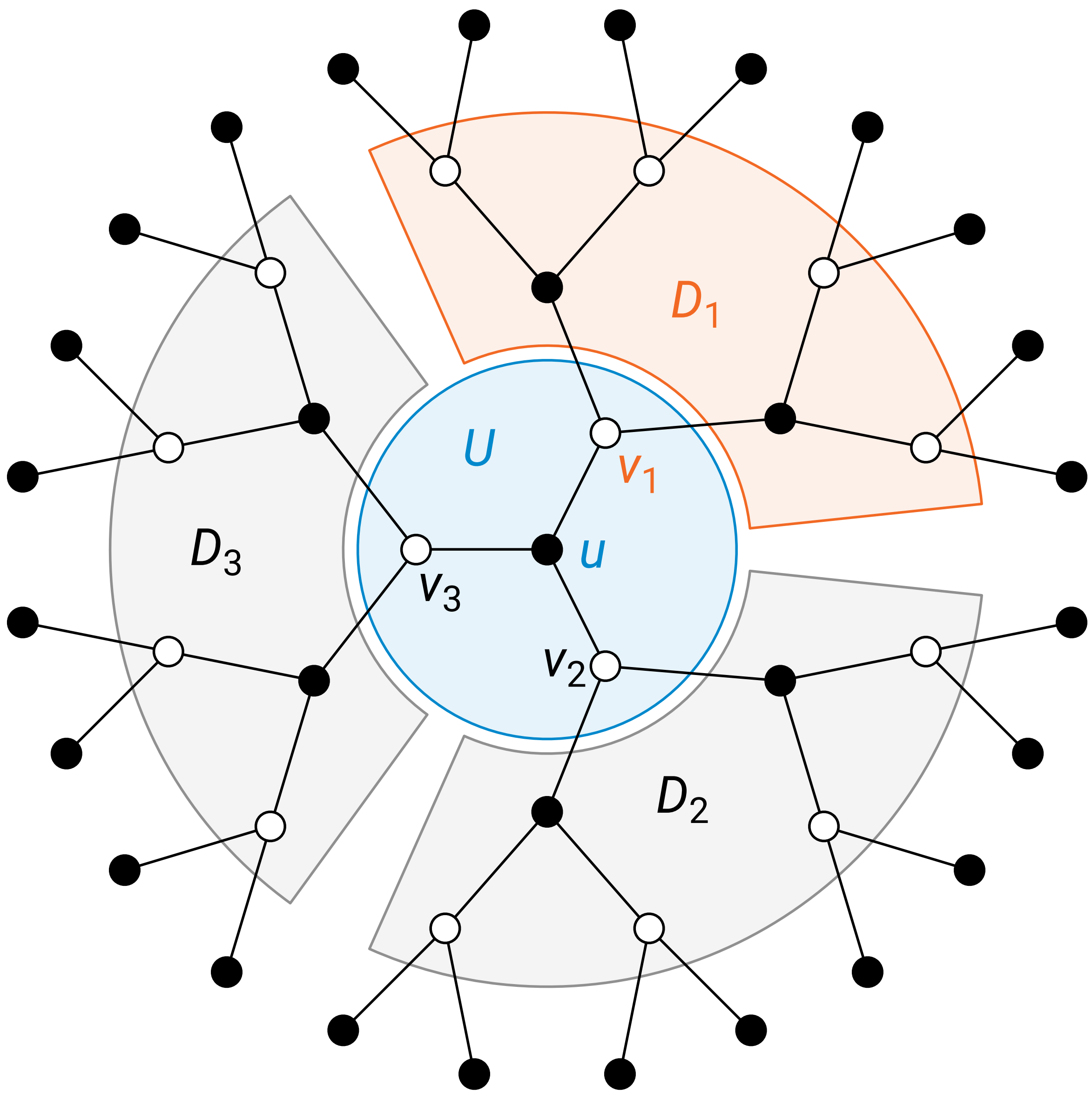
Some open questions

- **Δ << log log n:**
  - complexity of ***(Δ+1)-vertex coloring*** or ***(2Δ−1)-edge coloring***?
  - example: are these possible in $O(\log \Delta + \log^* n)$ time?
- **Δ >> log log n:**
  - complexity of ***maximal independent set***?
  - is it much harder than maximal matching in this region?
  - example: is it possible in deterministic polylog(n) time?

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: ***speedup simulation***

arXiv:1901.02441



Speedup simulation

Given: **white algorithm *A*** that runs in T = 2 rounds

- ***v_1*** in ***A*** sees ***U*** and ***D_1***

Construct: **black algorithm *A'*** that runs in T − 1 = 1 rounds

- ***u*** in ***A'*** only sees ***U***

A': what is the **set of possible outputs of *A*** for edge {***u***, ***v_1***} over all possible inputs in ***D_1***?