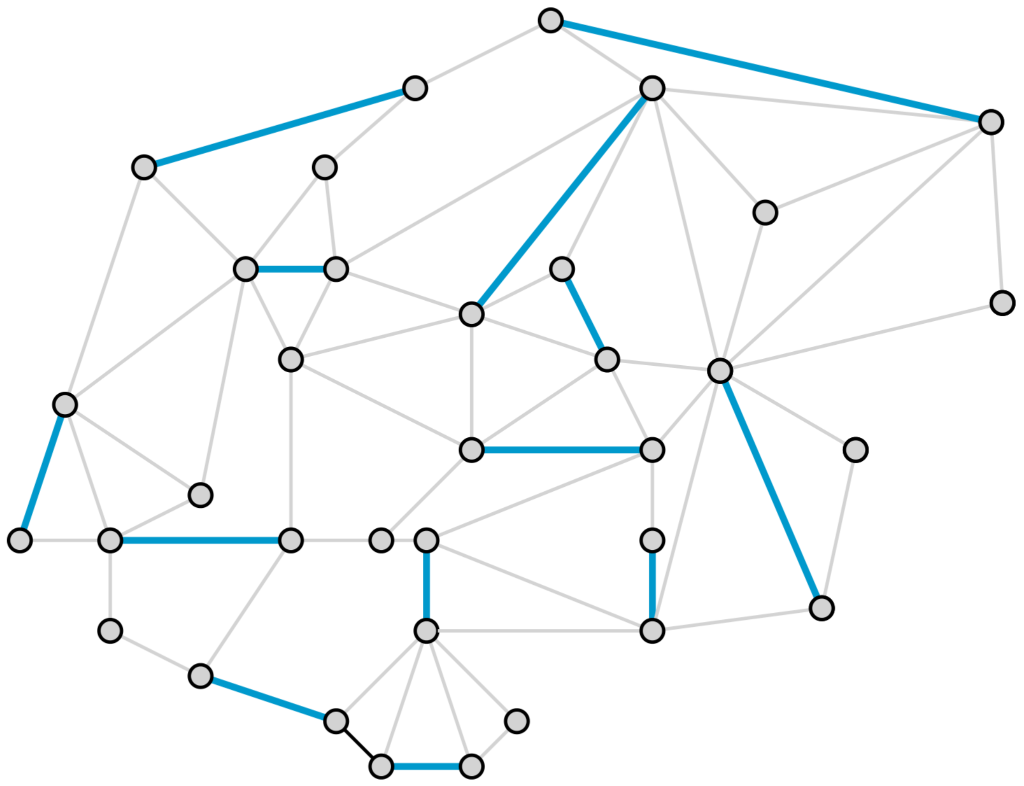
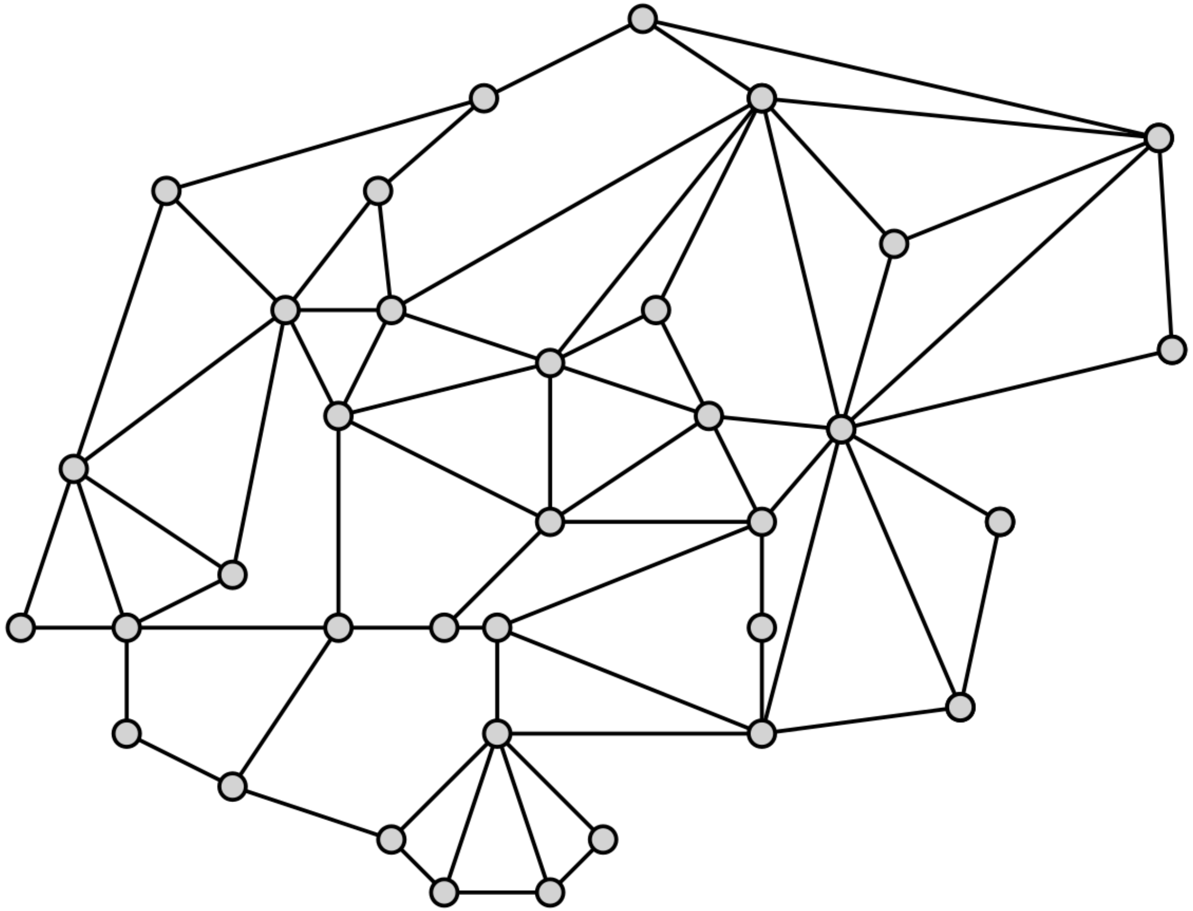


The Locality of Maximal Matching

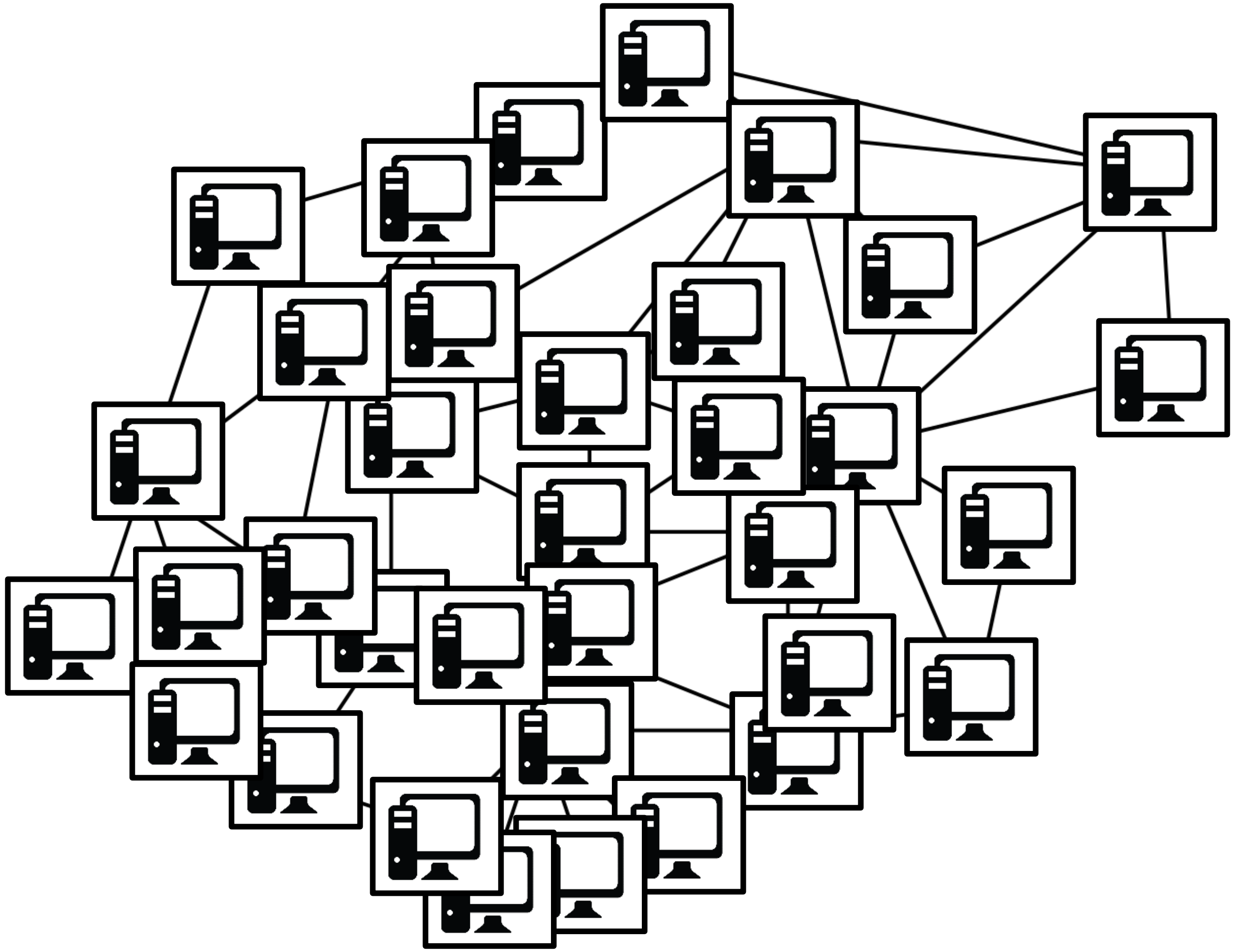
Manuela Fischer
ETH Zurich

Locality

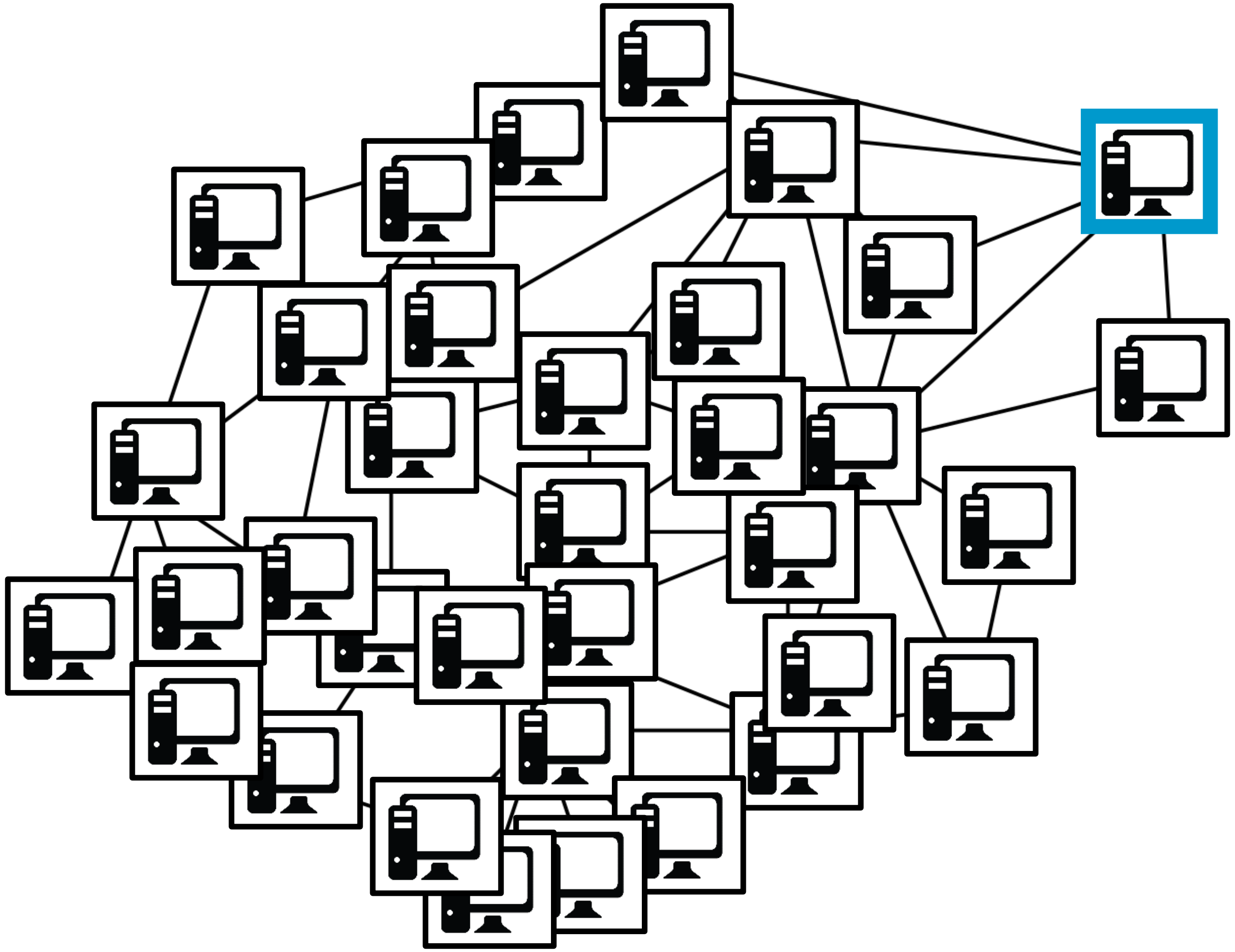
Locality



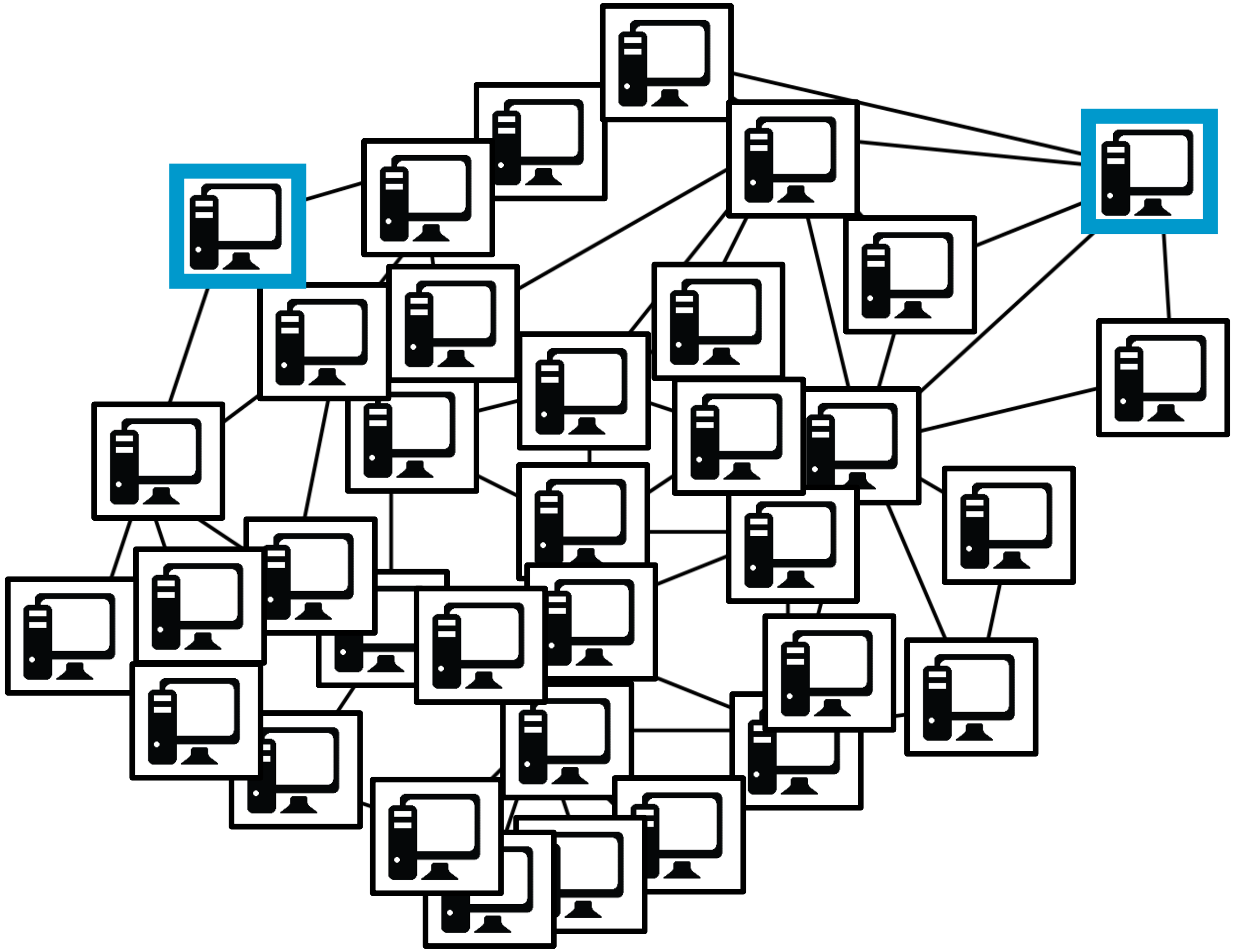
Locality



Locality



Locality



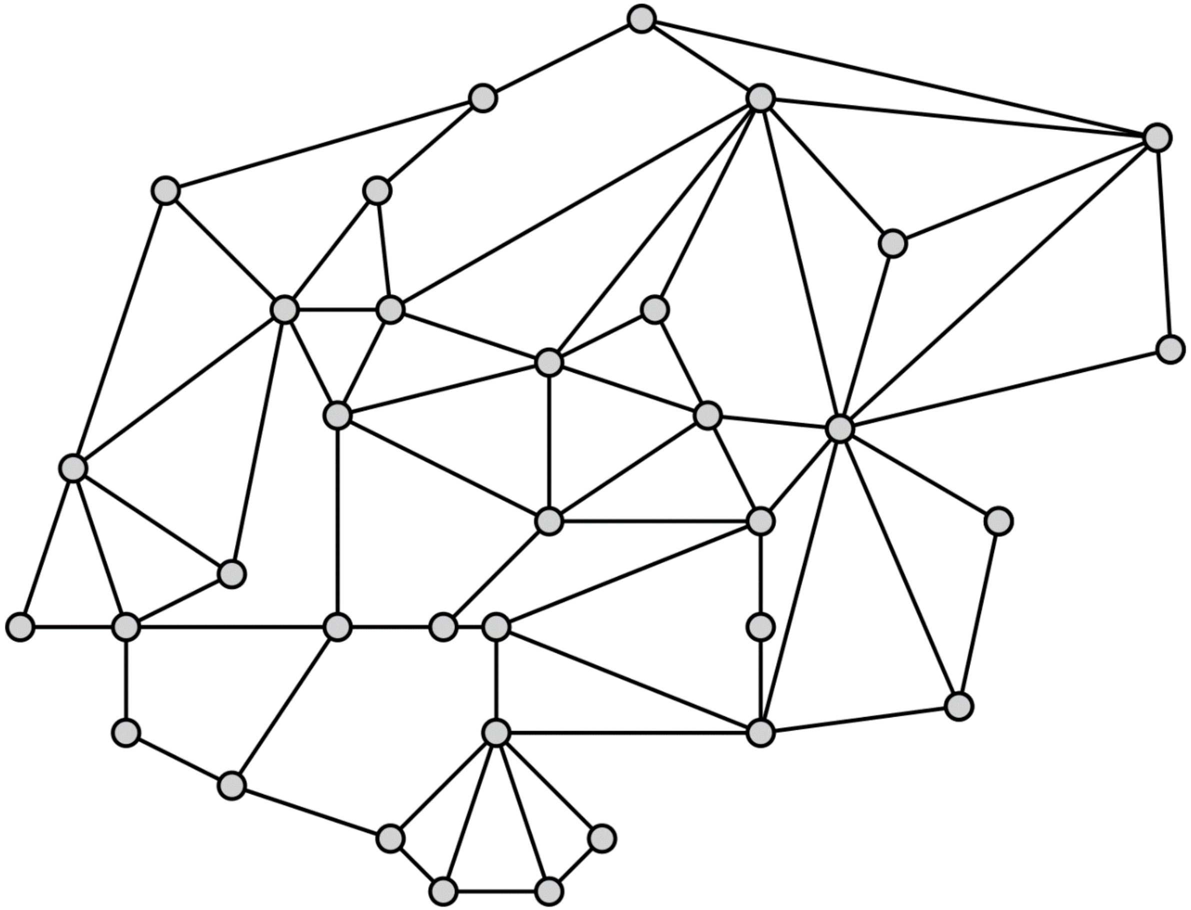
LOCAL Model *Linial* [FOCS'87]

LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

LOCAL Model *Linial* [FOCS'87]

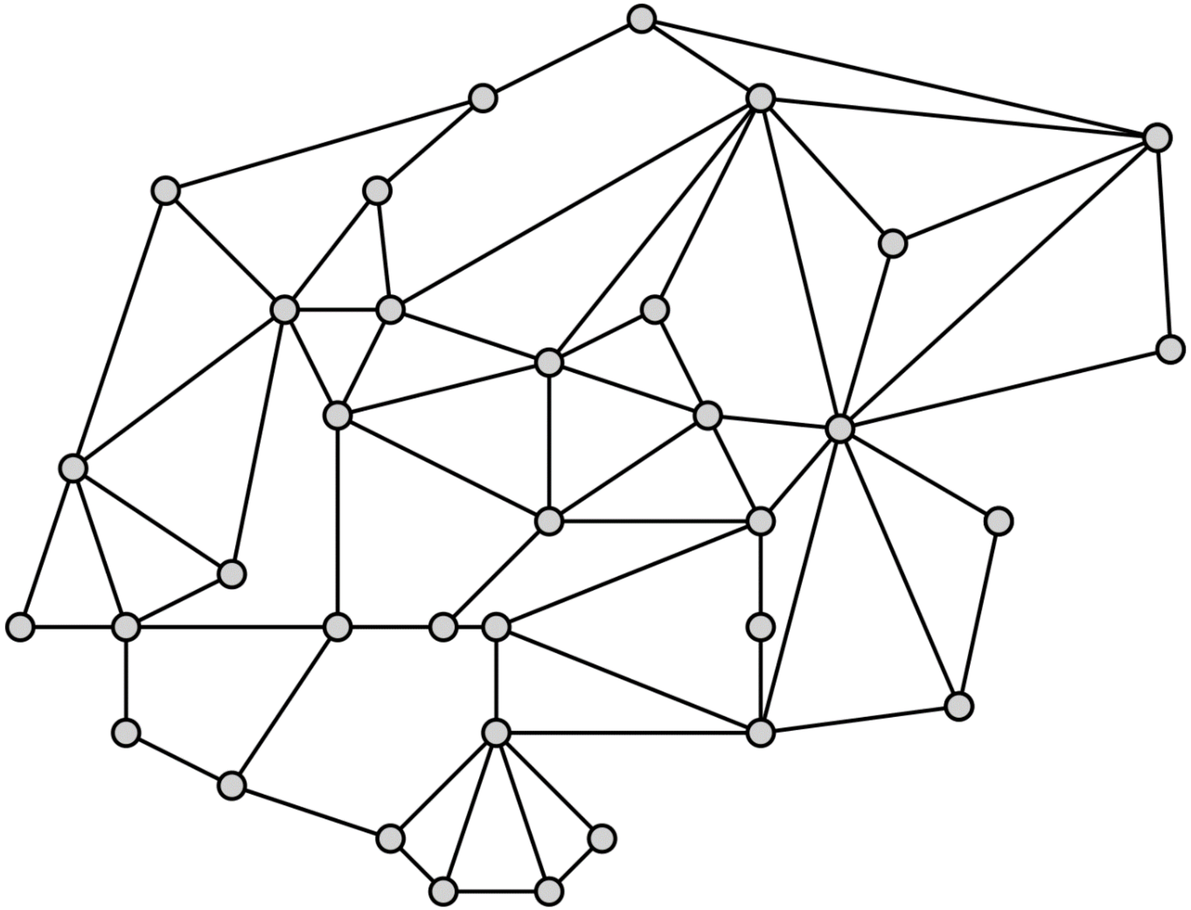
standard synchronous message-passing model of distributed computing



LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

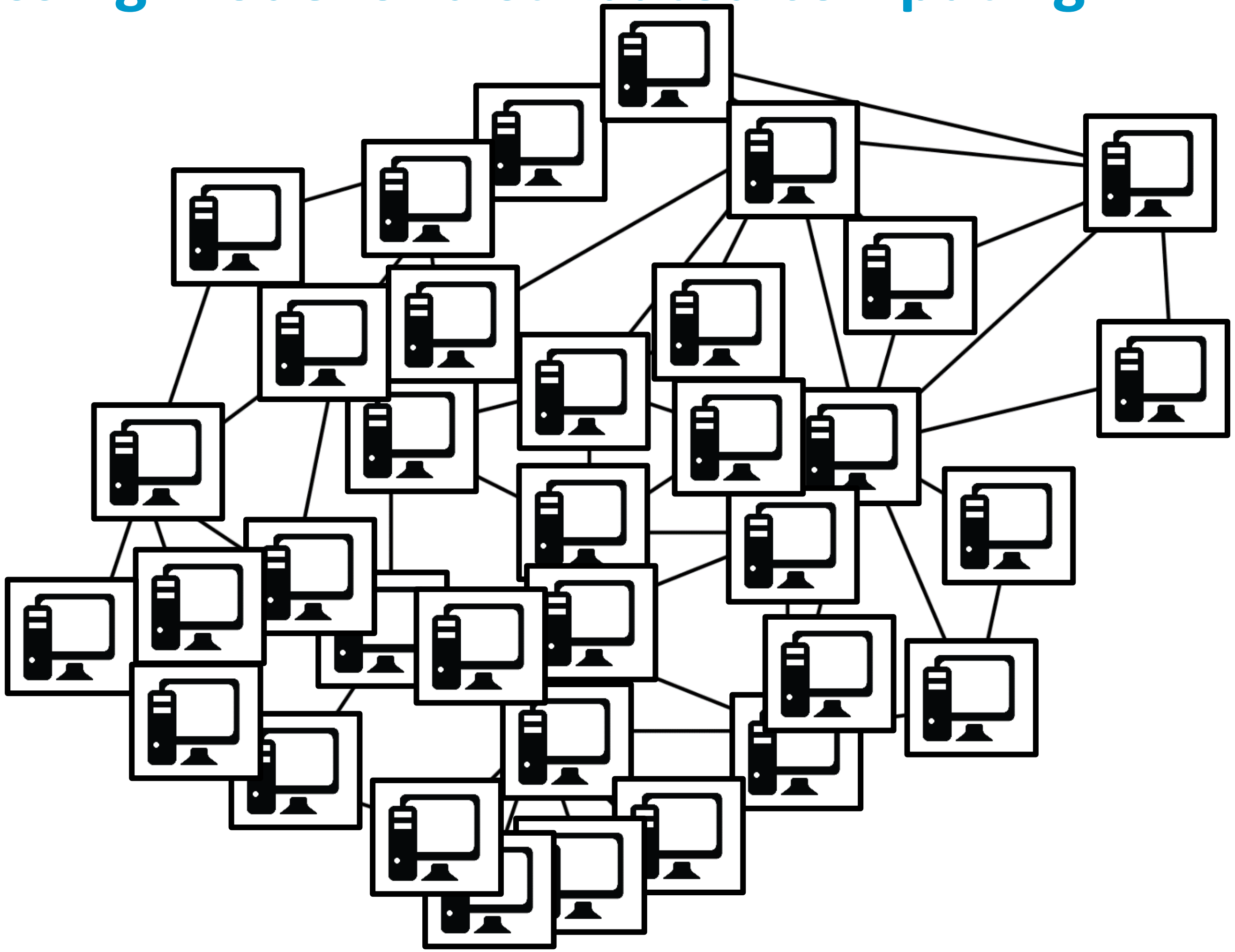
- undirected graph $G = (V, E)$, n nodes, maximum degree Δ



LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

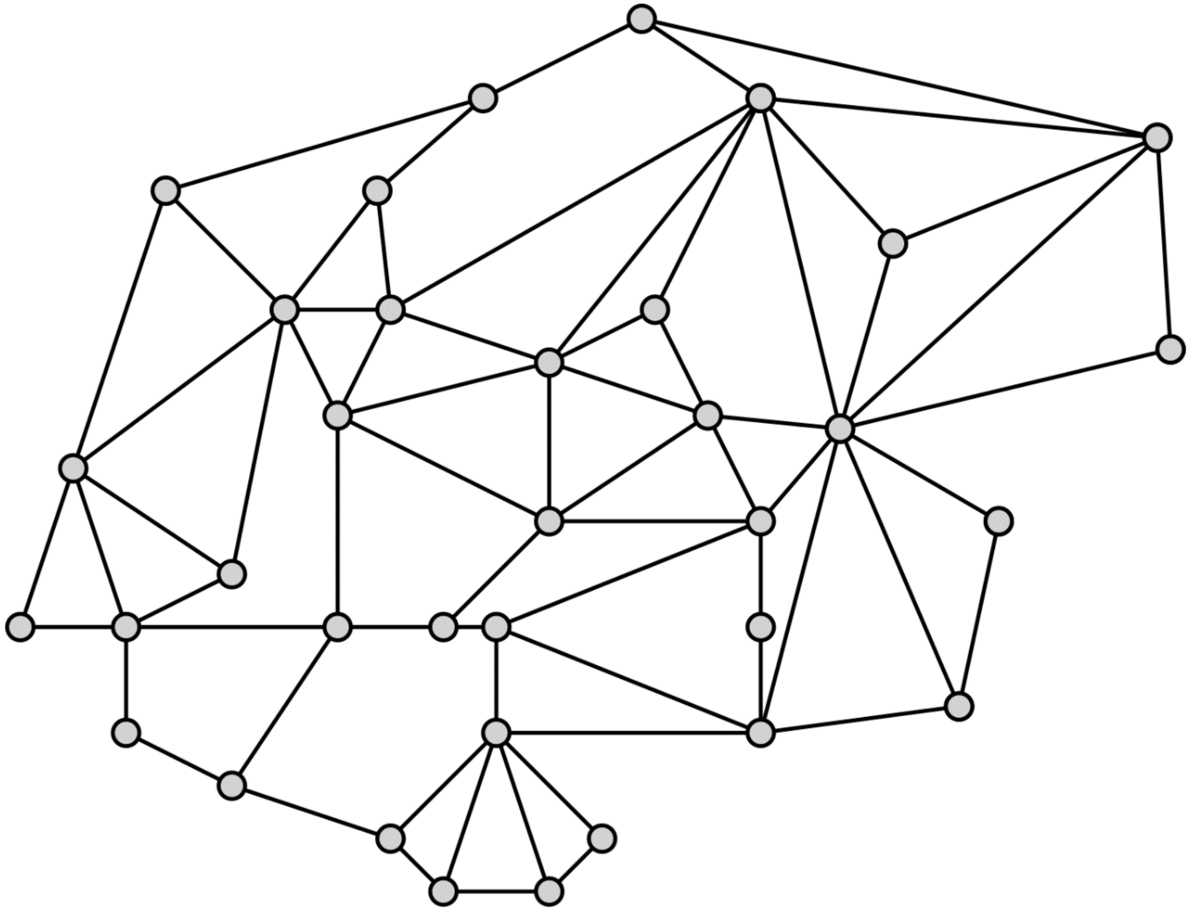
- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ



LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

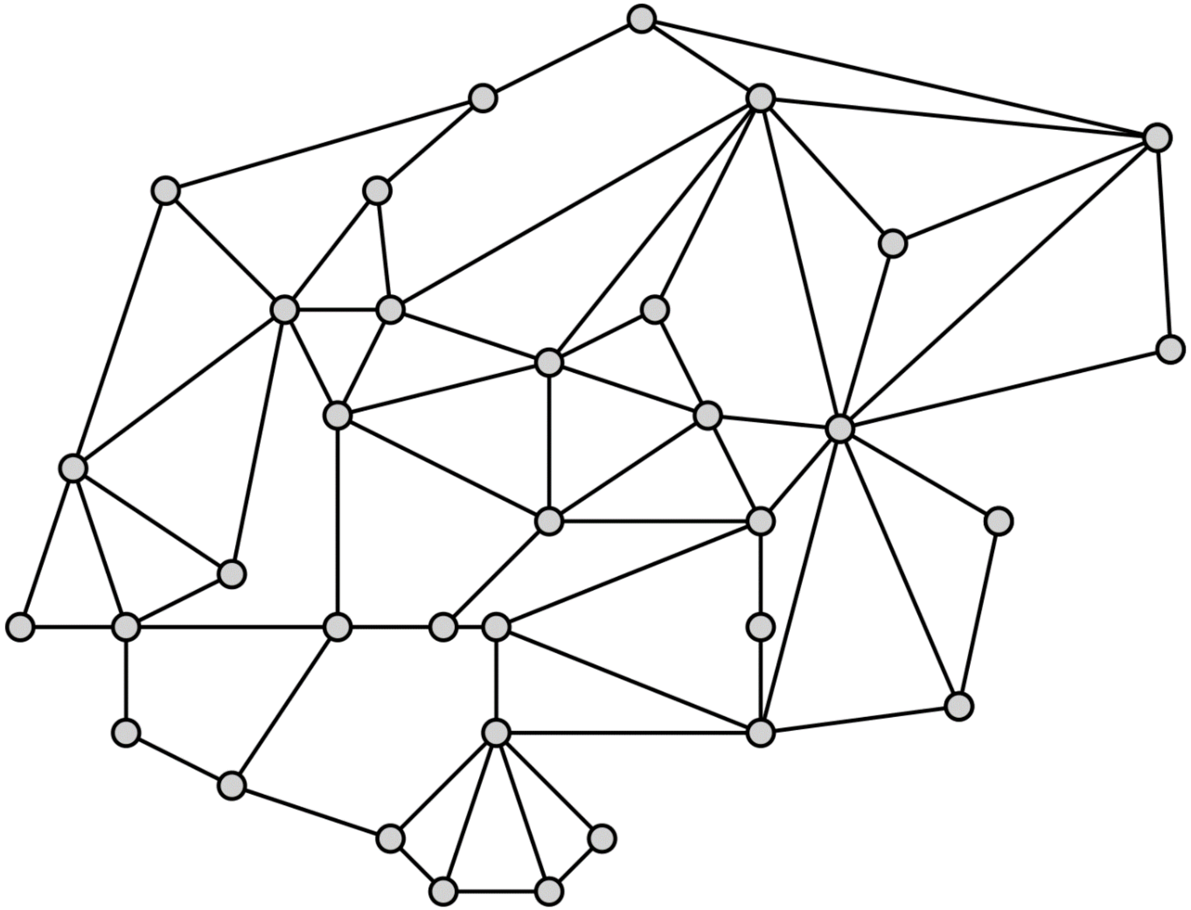
- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ



LOCAL Model *Linial* [FOCS’87]

standard synchronous message-passing model of distributed computing

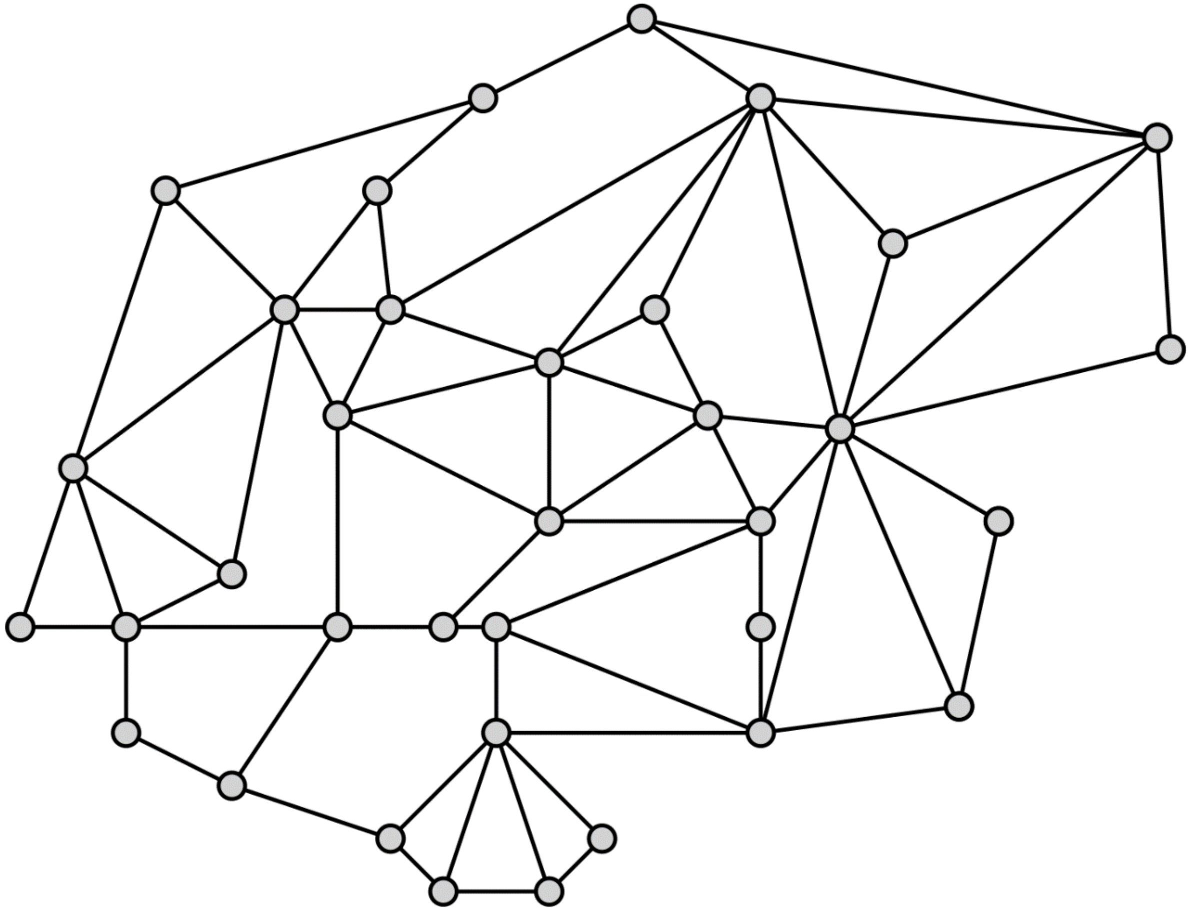
- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors



LOCAL Model *Linial* [FOCS’87]

standard synchronous message-passing model of distributed computing

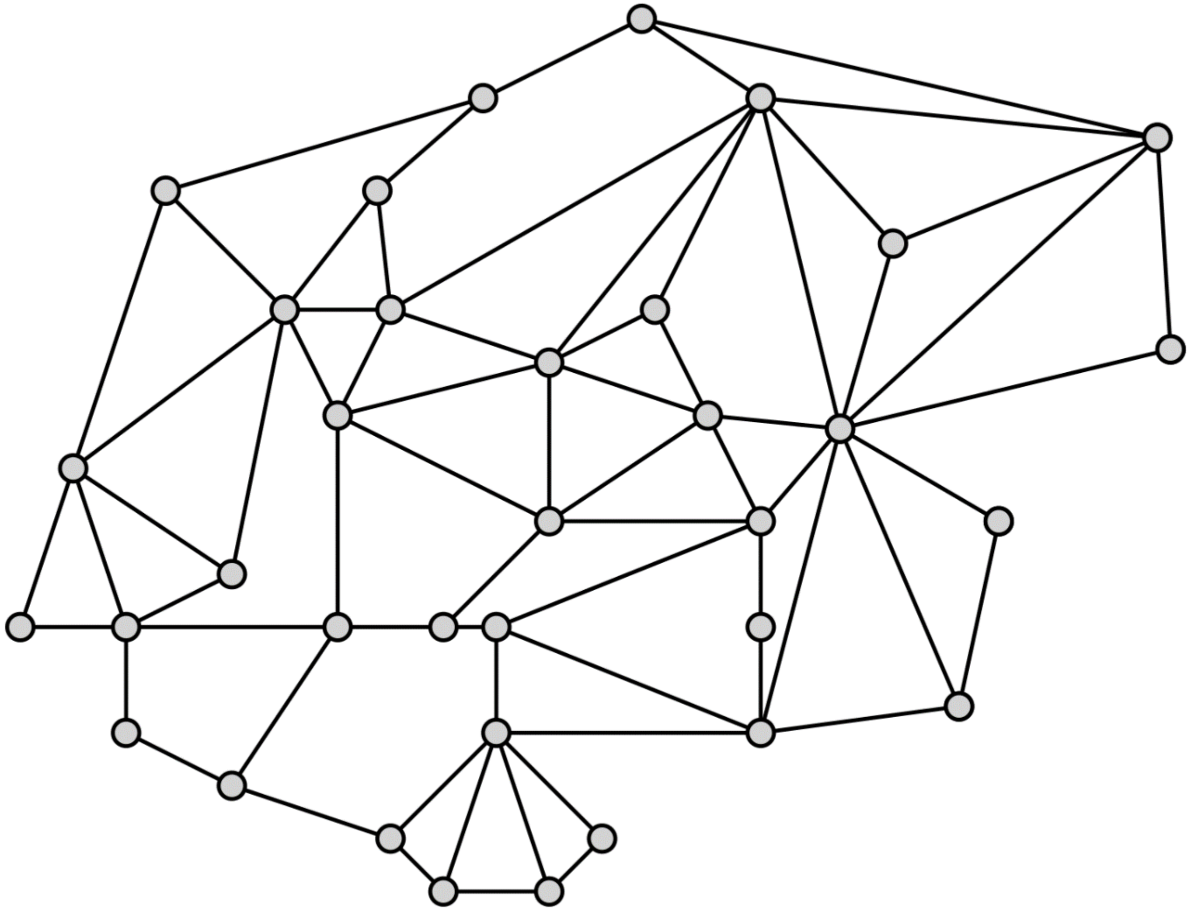
- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size



LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size
- unbounded computation

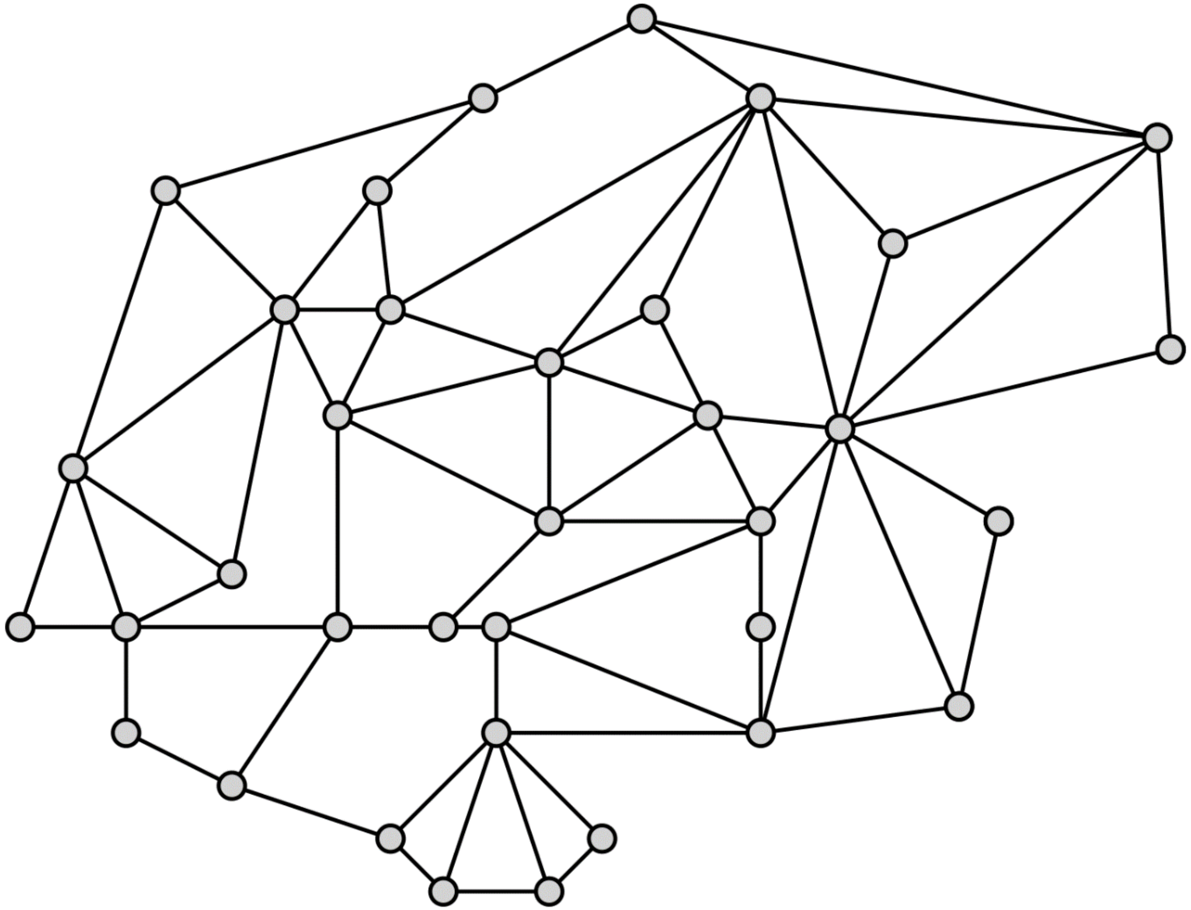


LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size
- unbounded computation

- **Round Complexity**:
  number of rounds to solve the problem

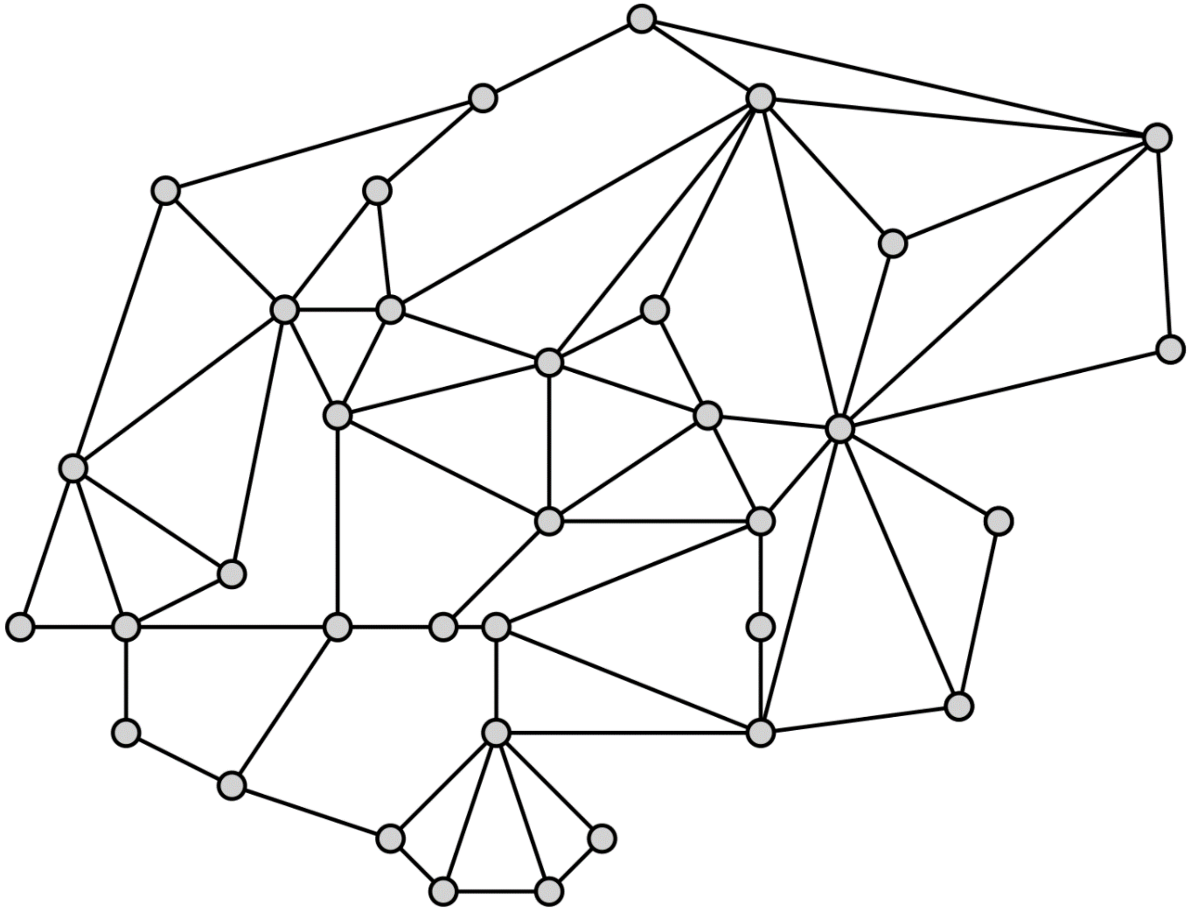


LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size
- unbounded computation

- **Round Complexity**:
  number of rounds to solve the problem



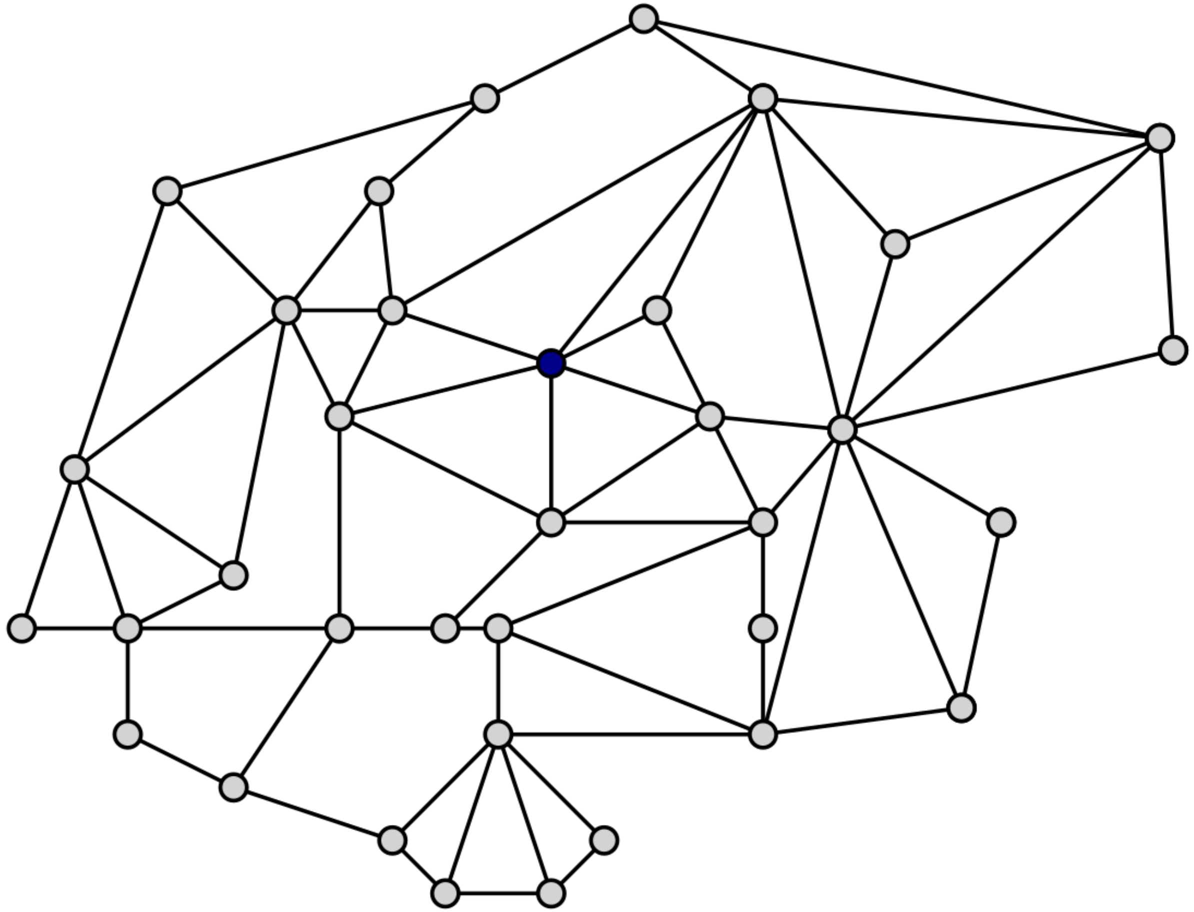
round complexity of a problem in the LOCAL model characterizes its locality

LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size
- unbounded computation

- **Round Complexity**:
  number of rounds to solve the problem

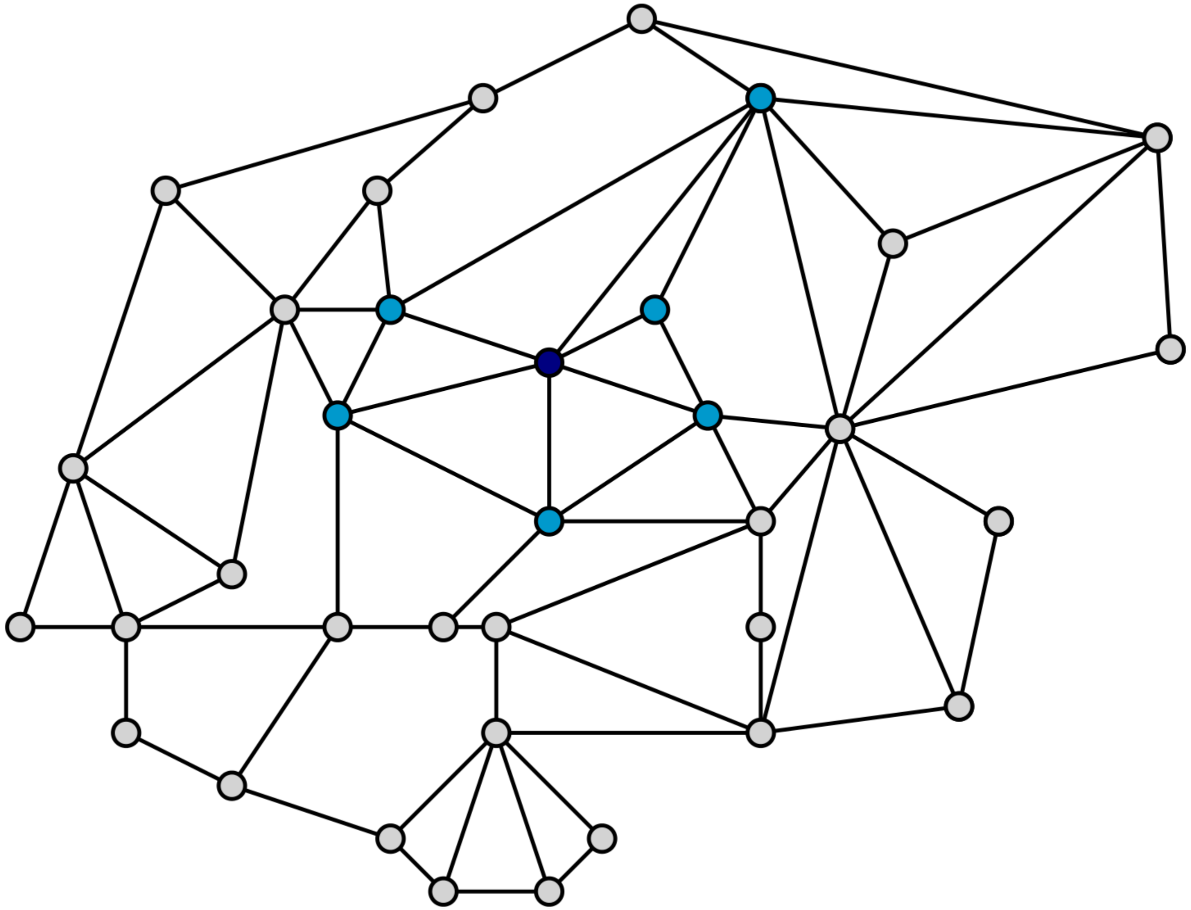


round complexity of a problem in the LOCAL model characterizes its locality

LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size
- unbounded computation



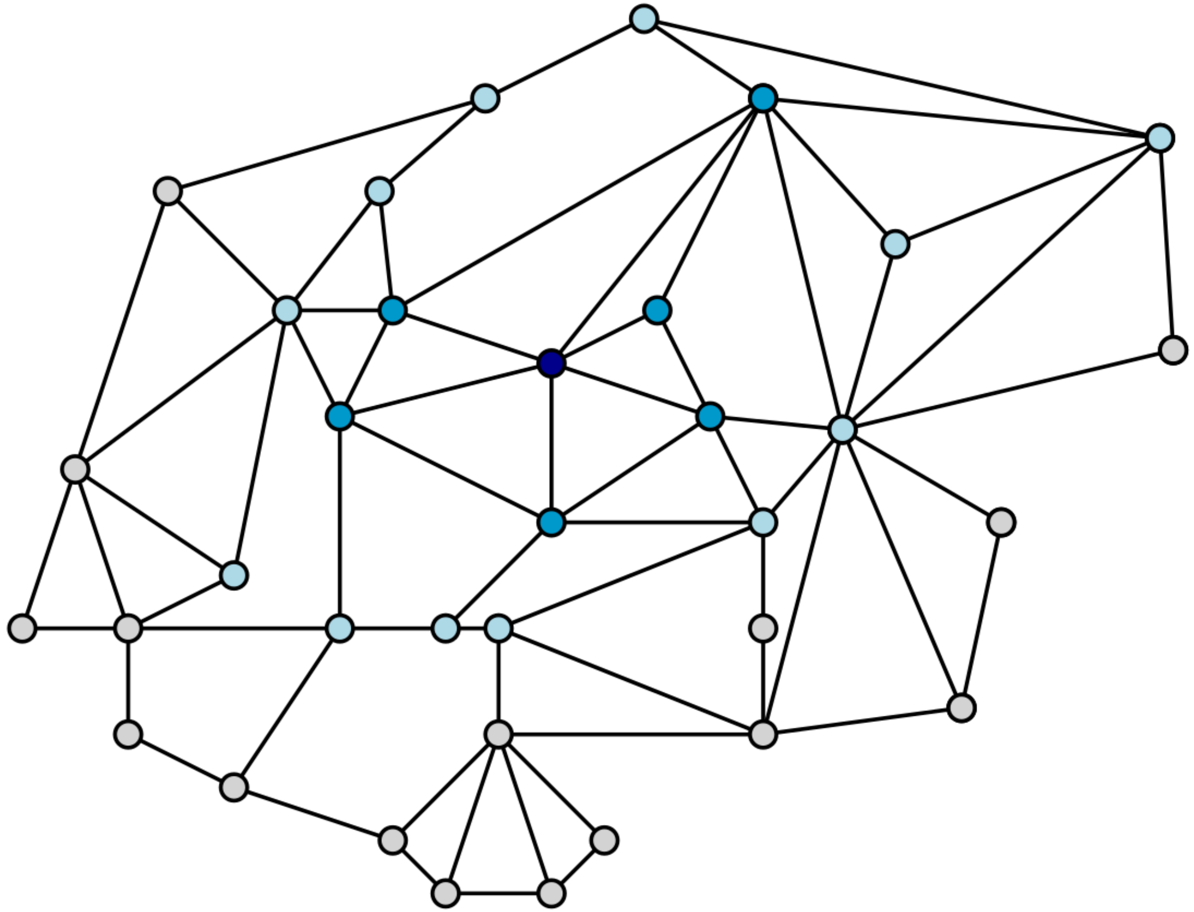
- **Round Complexity**:
  number of rounds to solve the problem

round complexity of a problem in the LOCAL model characterizes its locality

LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size
- unbounded computation



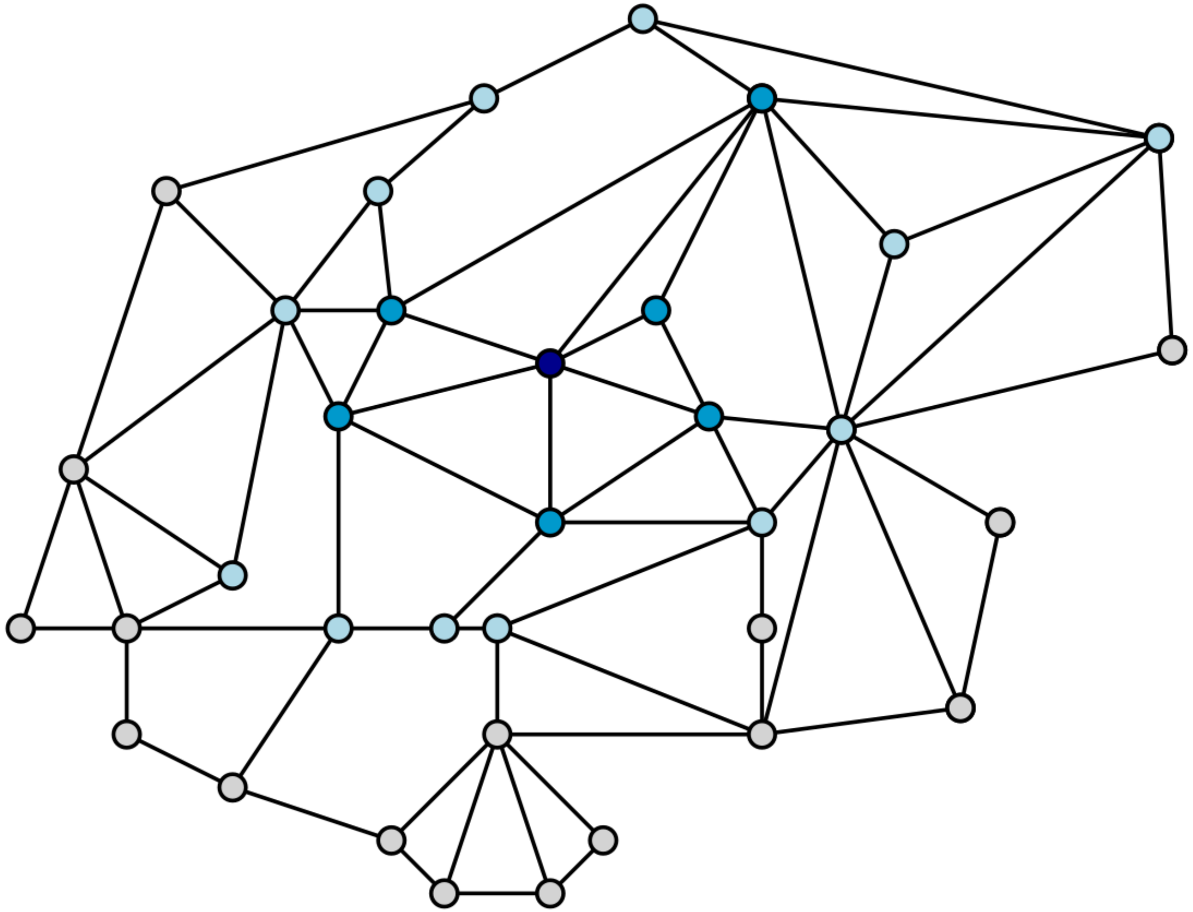
- **Round Complexity**:
  number of rounds to solve the problem

round complexity of a problem in the LOCAL model characterizes its locality

LOCAL Model *Linial* [FOCS'87]

standard synchronous message-passing model of distributed computing

- undirected graph $G = (V, E)$,
  n nodes, maximum degree Δ
- each round, every node
  - receives messages (sent in previous round)
  - performs some computation
  - sends message to all its neighbors
- unbounded message size
- unbounded computation



- **Round Complexity**:
  number of rounds to solve the problem

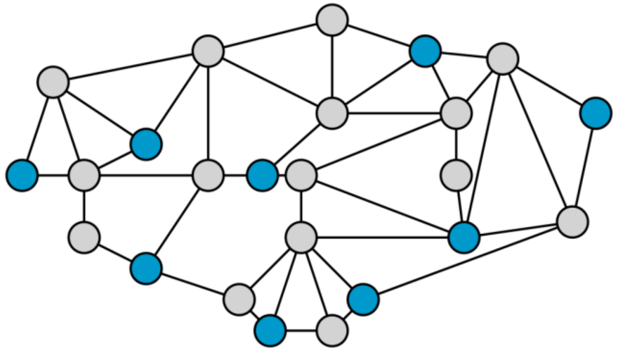
round complexity of a problem in the LOCAL model characterizes its locality

every problem is trivially solvable in $O(\mathbf{diameter})$ rounds

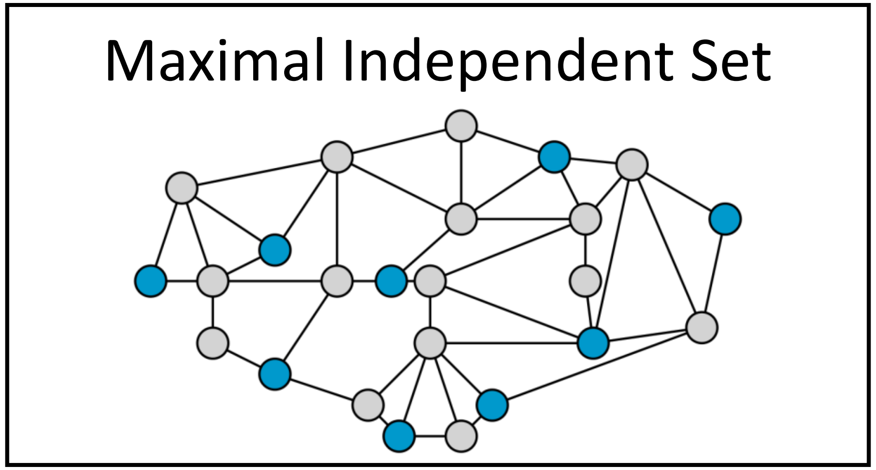
Classic LOCAL Graph Problems

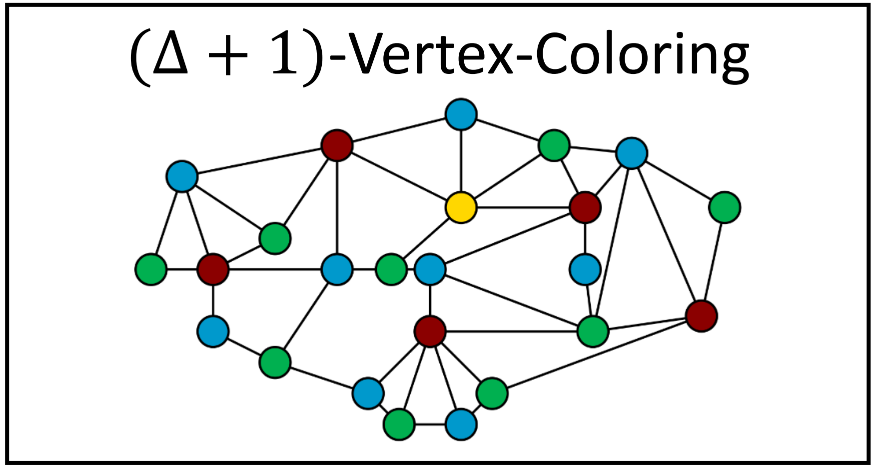
Classic LOCAL Graph Problems

Maximal Independent Set



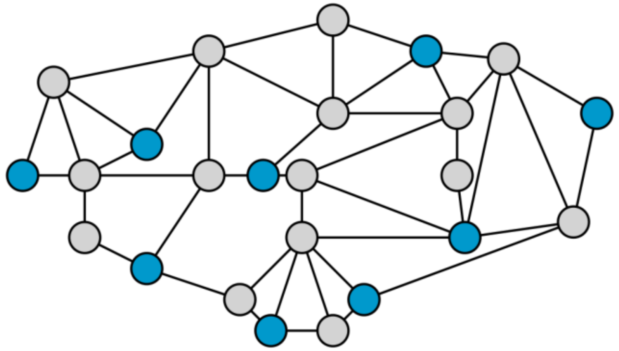
Classic LOCAL Graph Problems



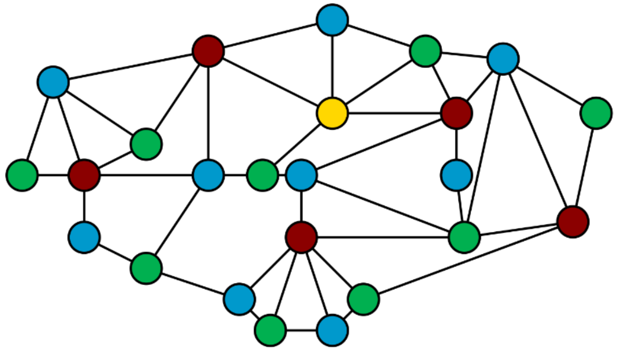


Classic LOCAL Graph Problems

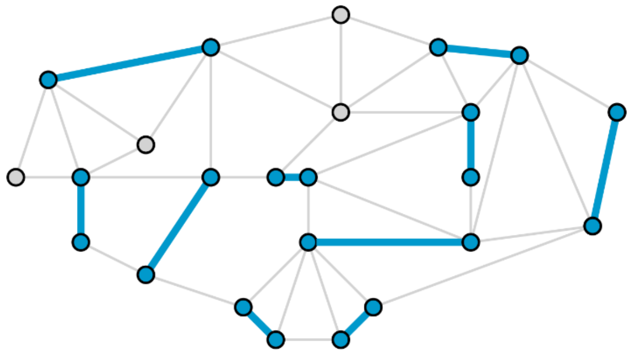
Maximal Independent Set



$(\Delta + 1)$-Vertex-Coloring

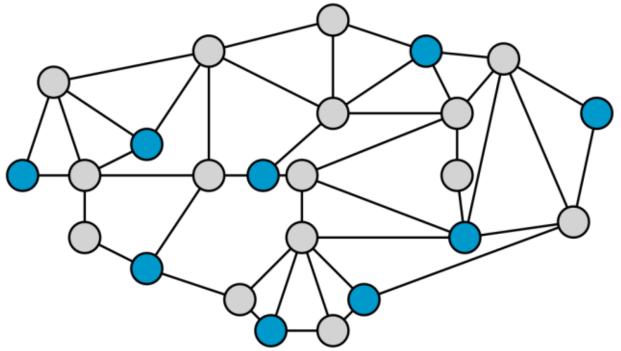


Maximal Matching

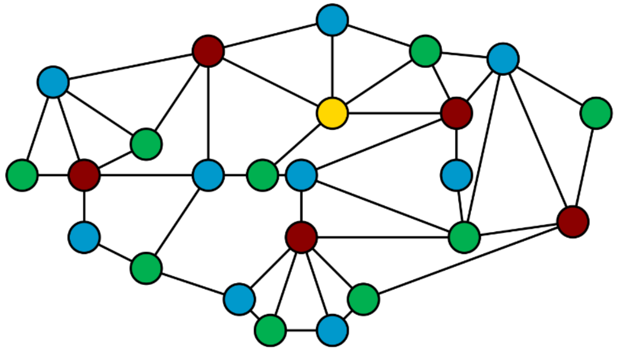


Classic LOCAL Graph Problems

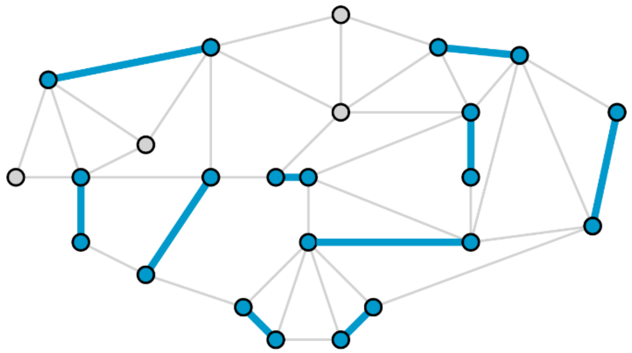
Maximal Independent Set



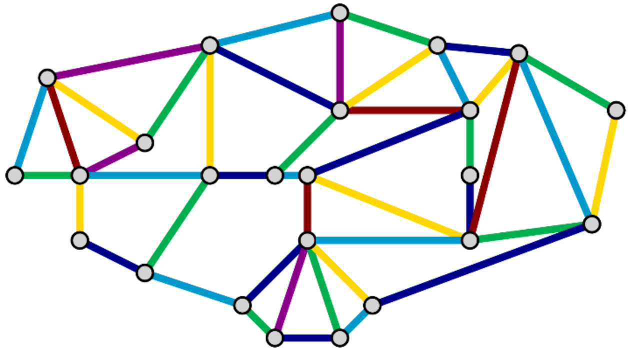
$(\Delta + 1)$-Vertex-Coloring



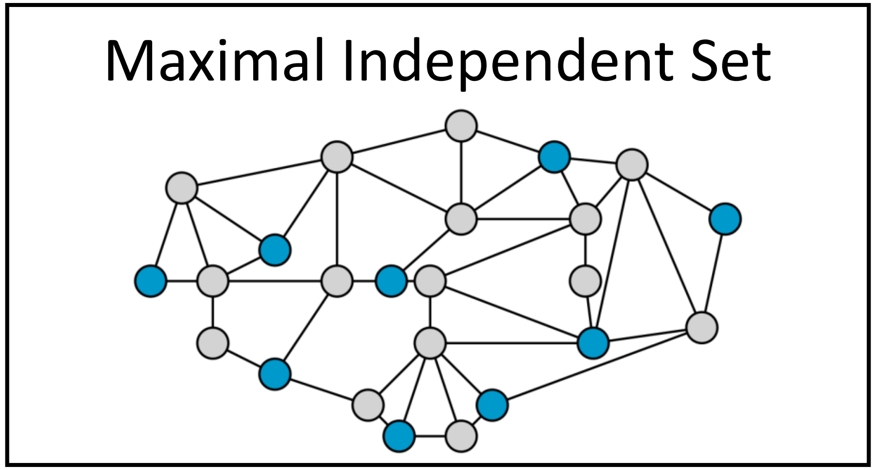
Maximal Matching

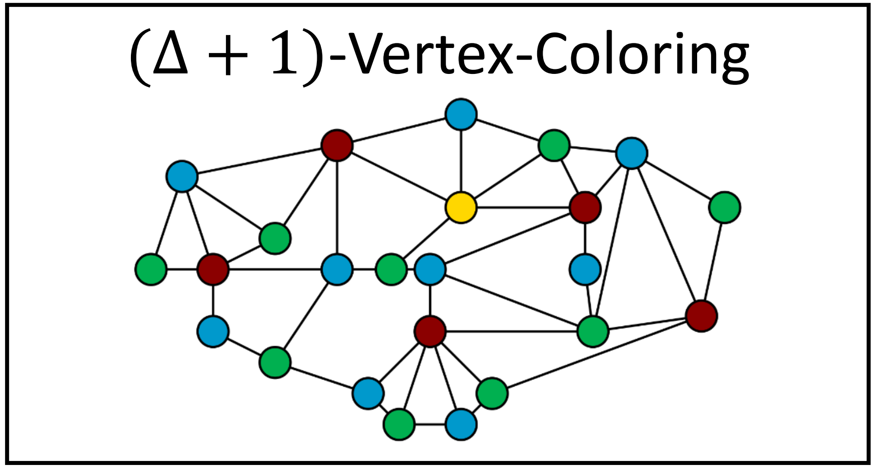


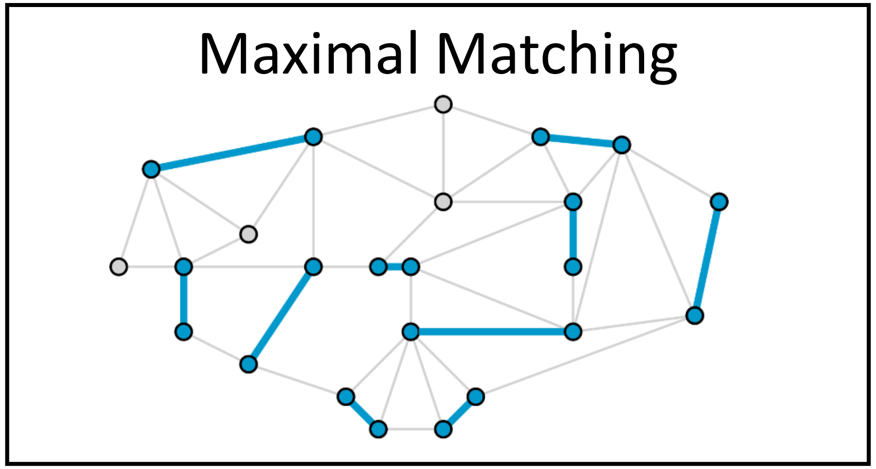
$(2\Delta - 1)$-Edge-Coloring

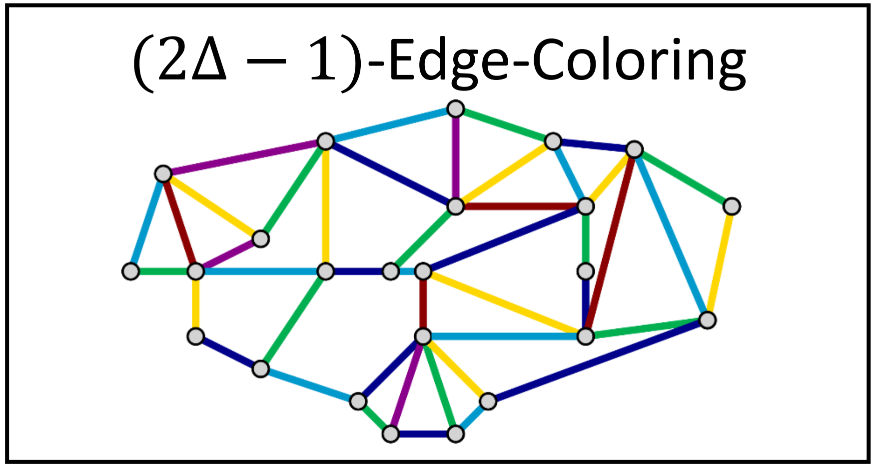


Classic LOCAL Graph Problems





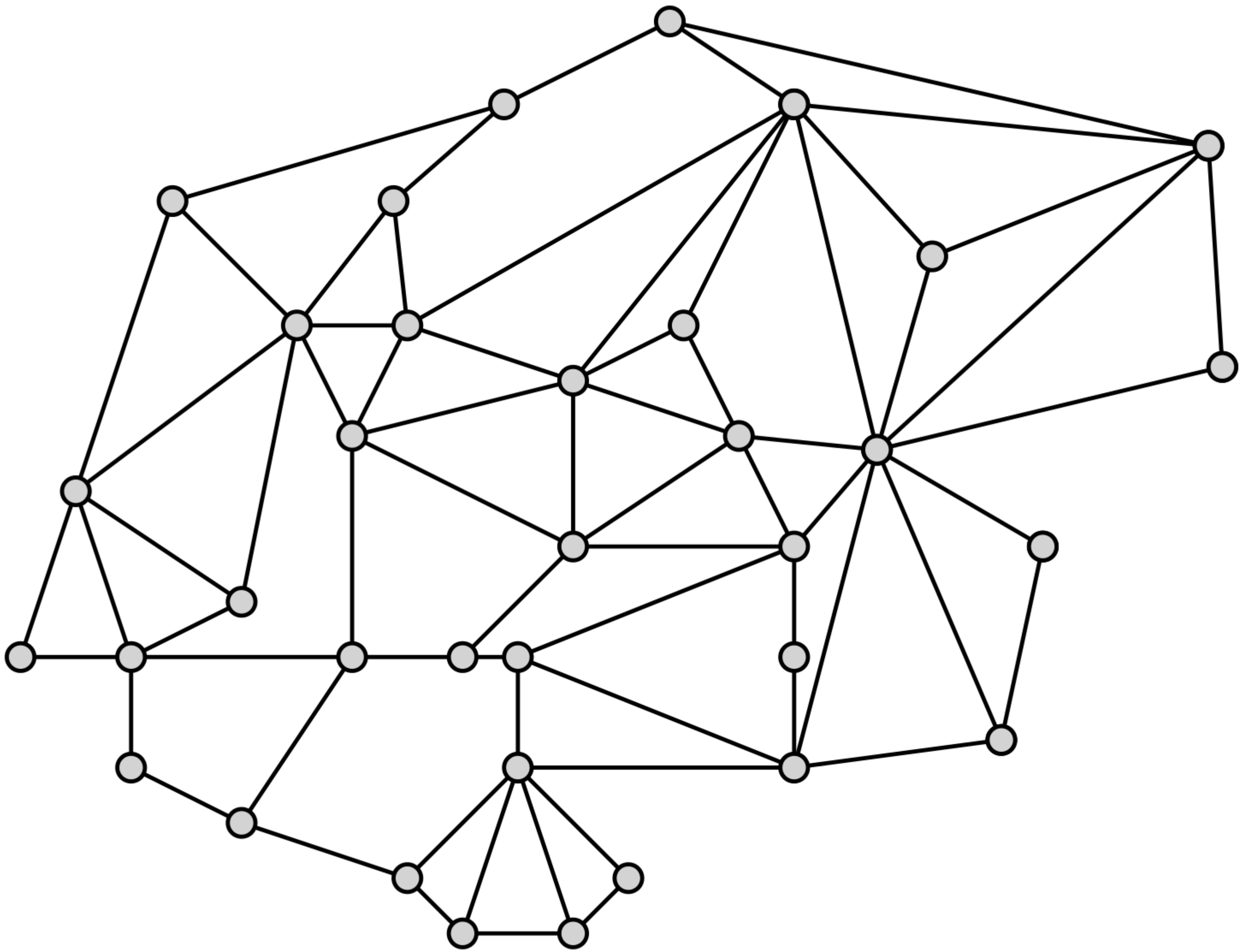




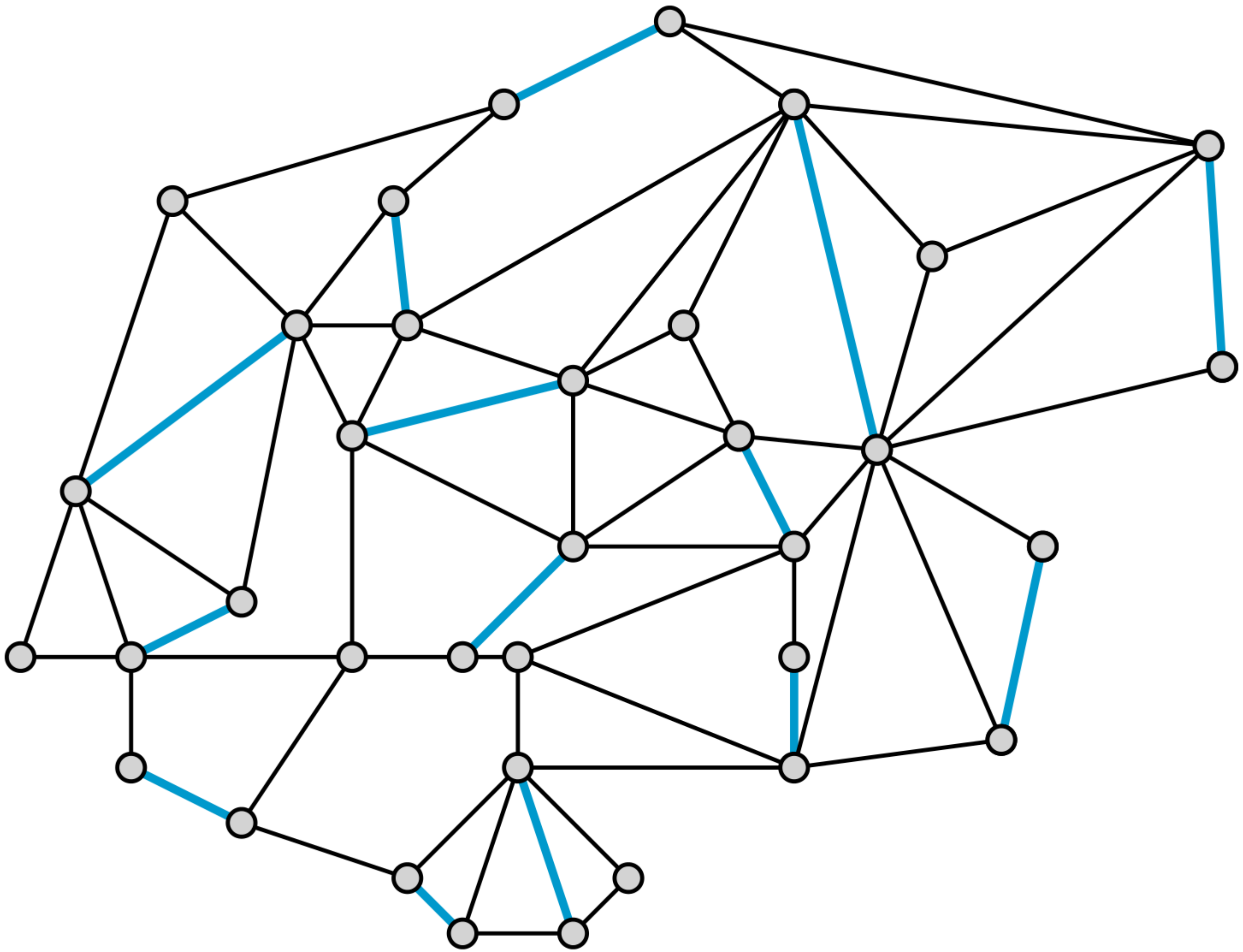
Easy centralized problems: greedy solutions.

Maximal Matching

Maximal Matching

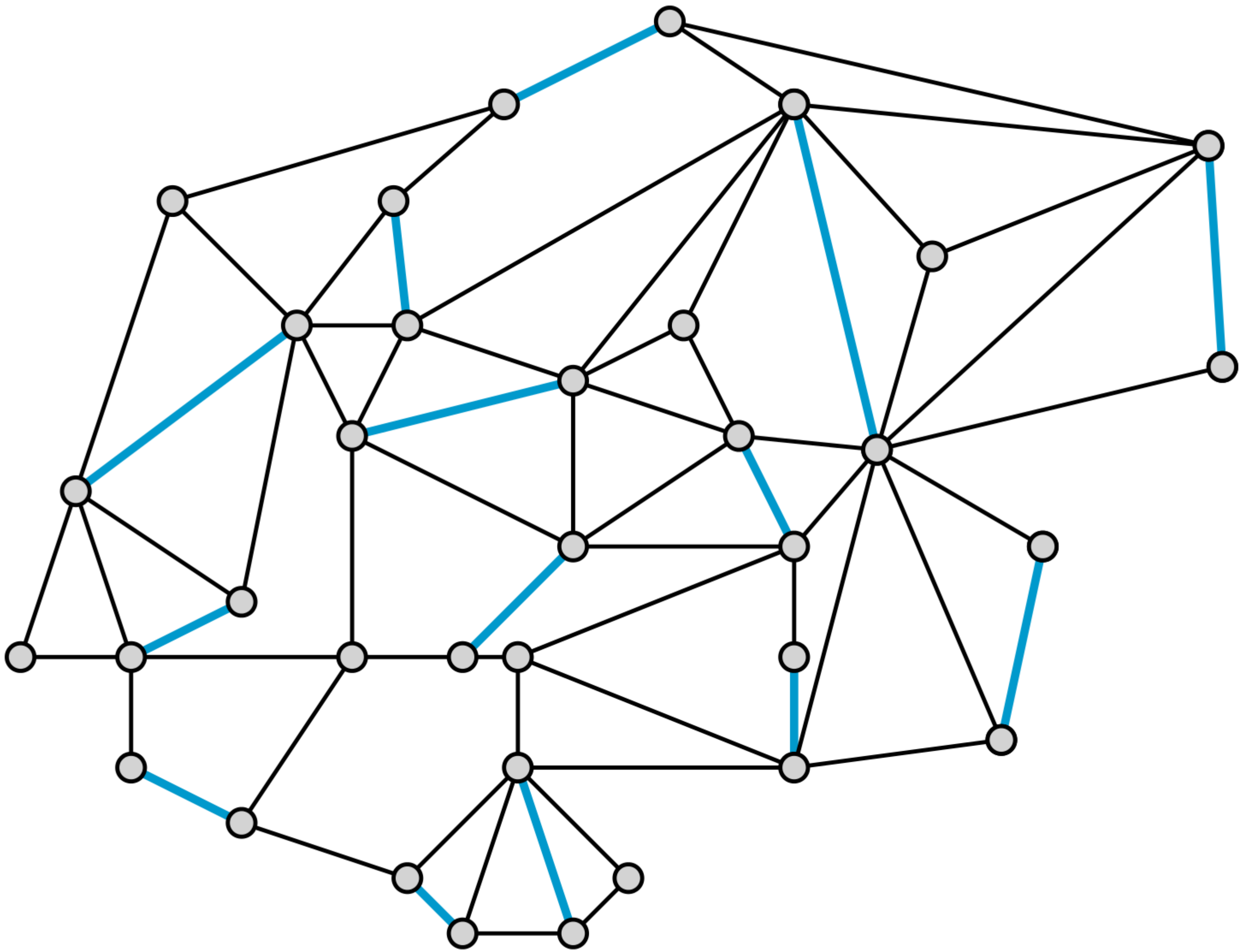


Maximal Matching



Maximal Matching

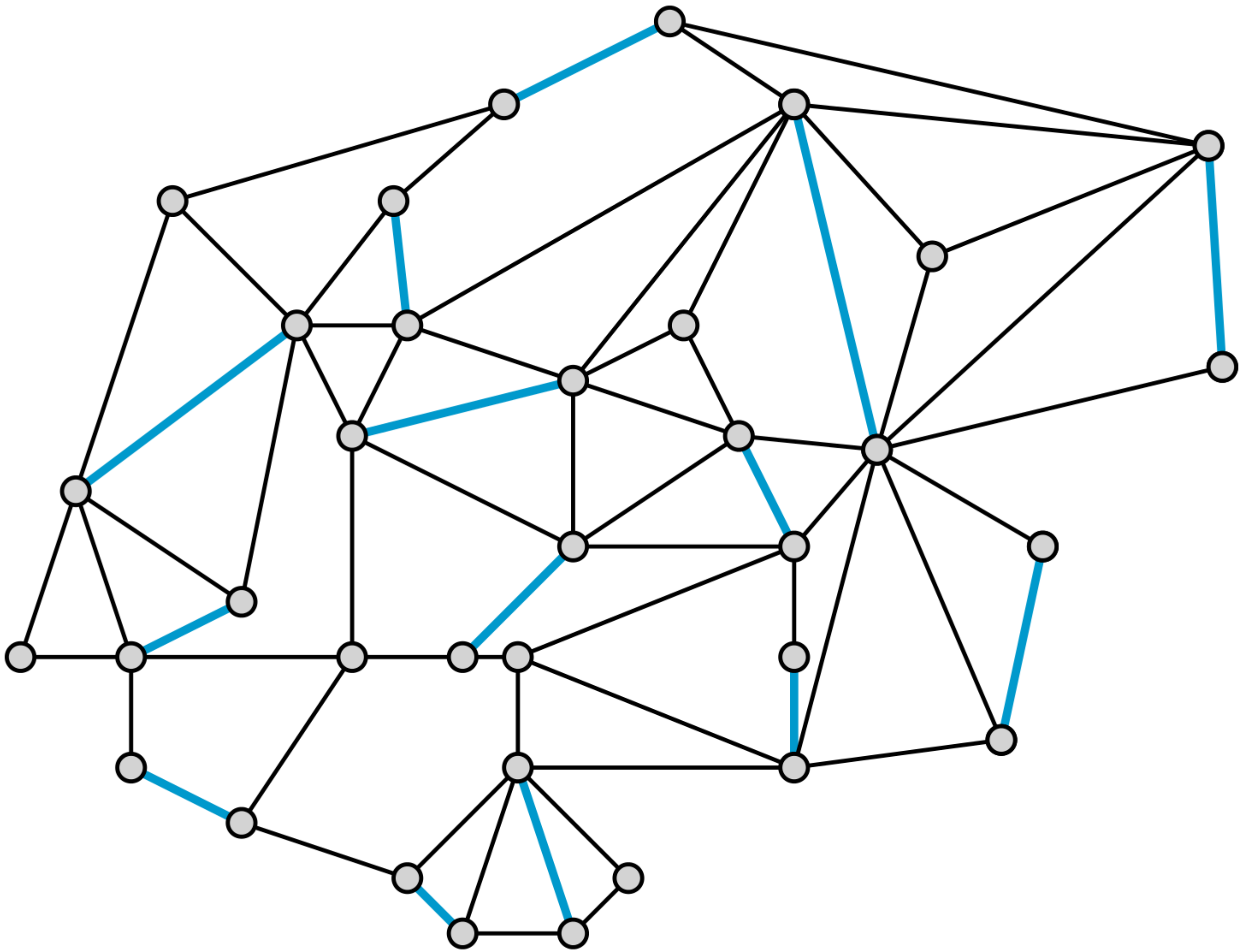
Matching:
set of non-incident edges



Maximal Matching

Matching:
set of non-incident edges

Maximal:
no edge can be added

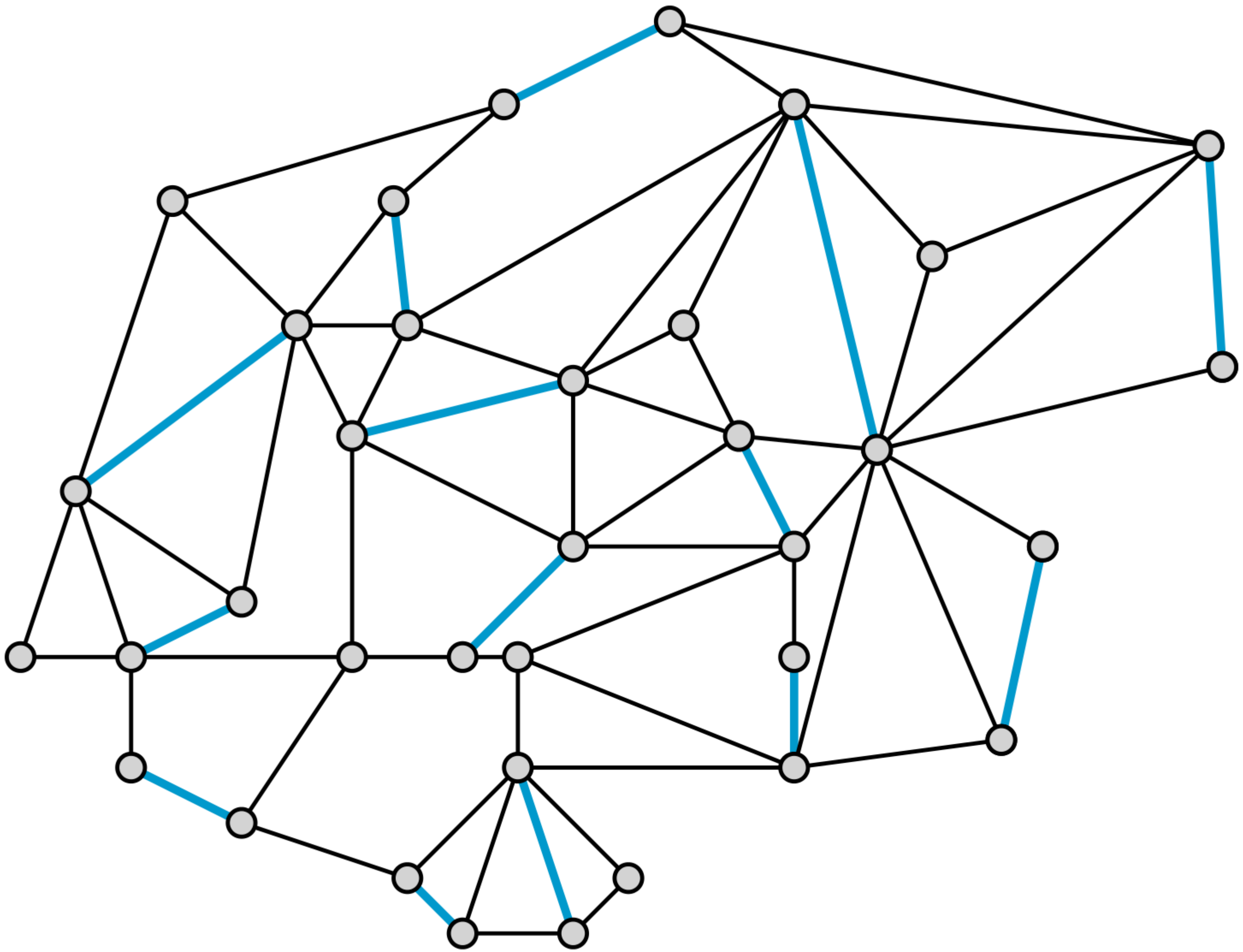


Maximal Matching

Matching:
set of non-incident edges

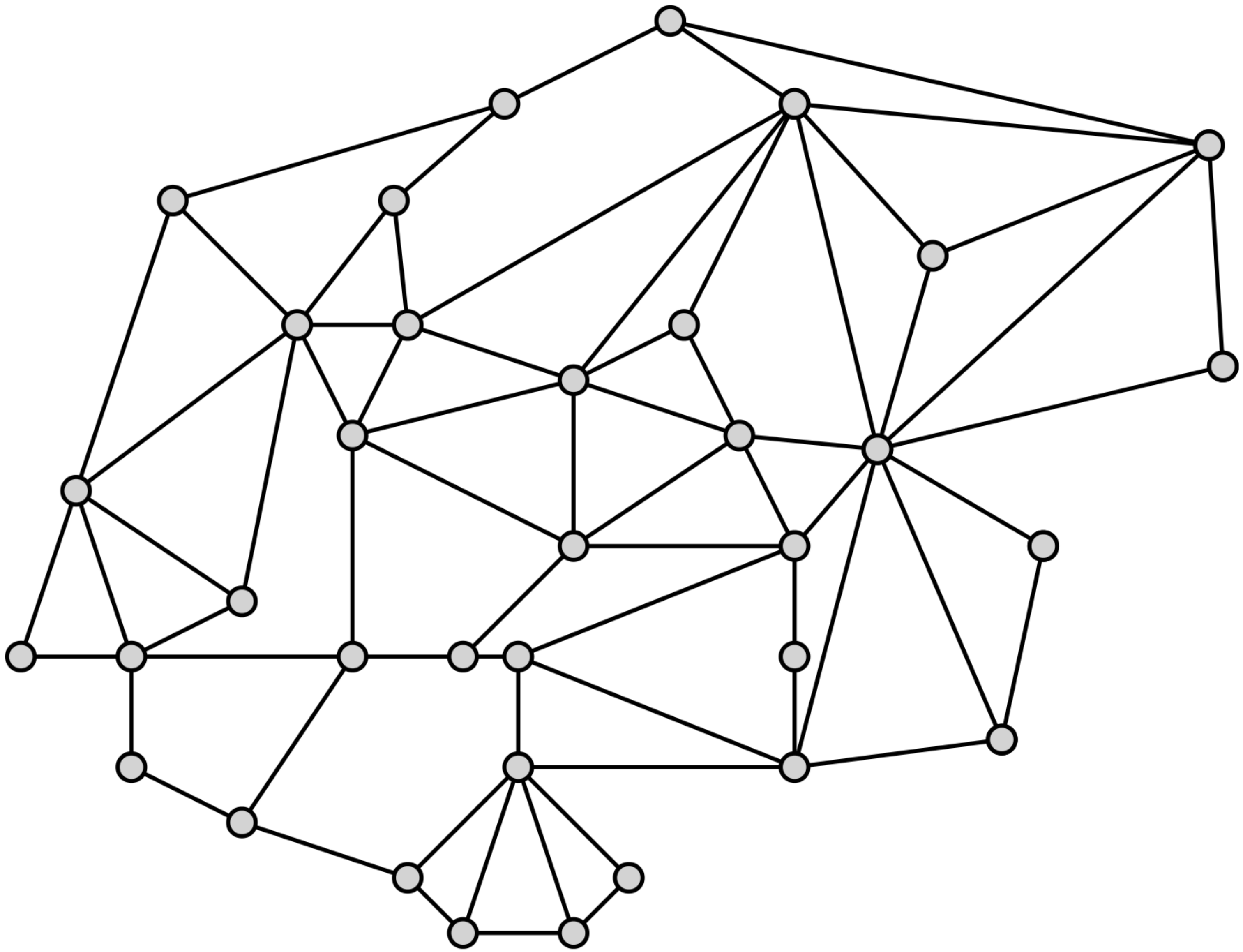
Maximal:
no edge can be added

greedy property!

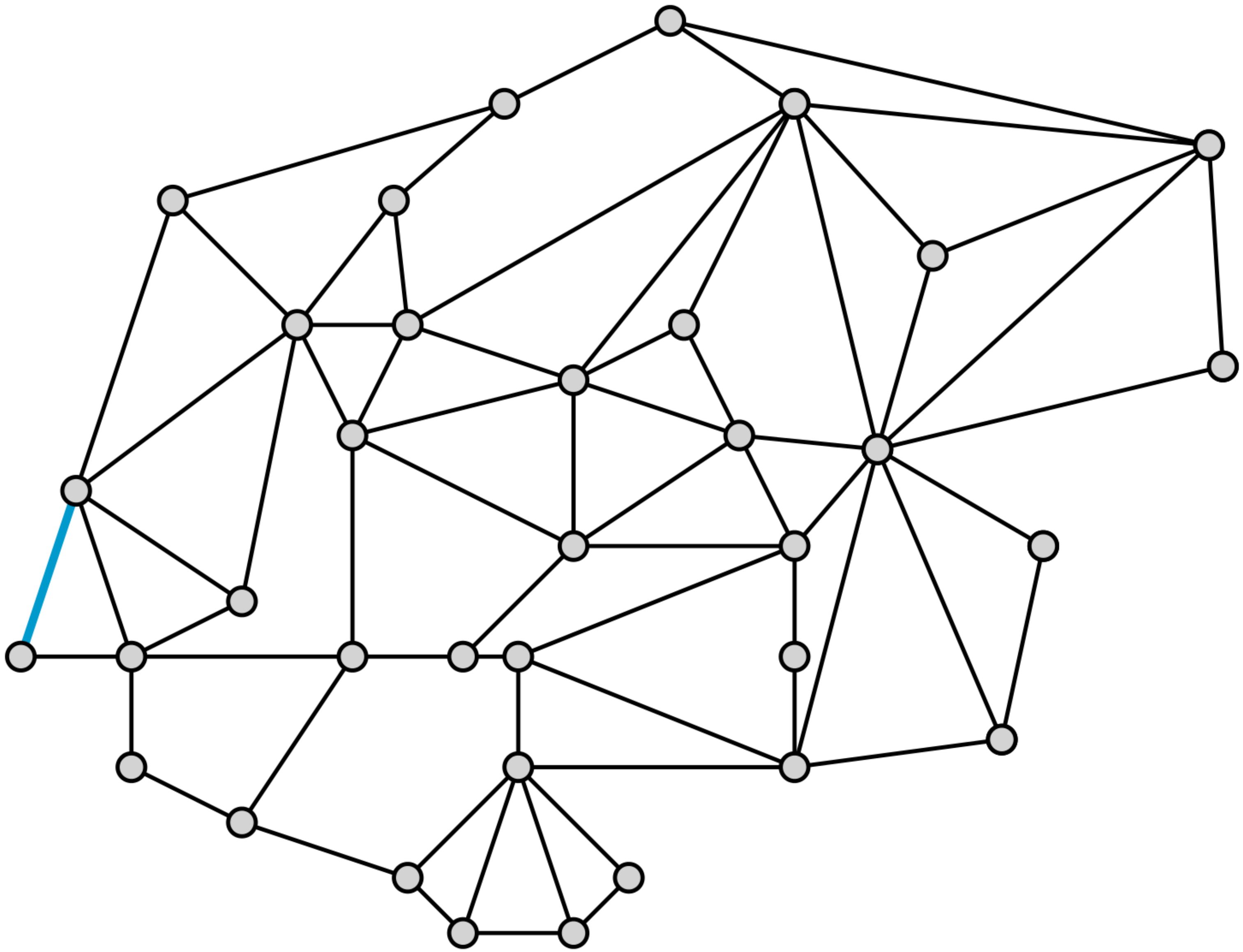


Centralized (Sequential) Algorithm

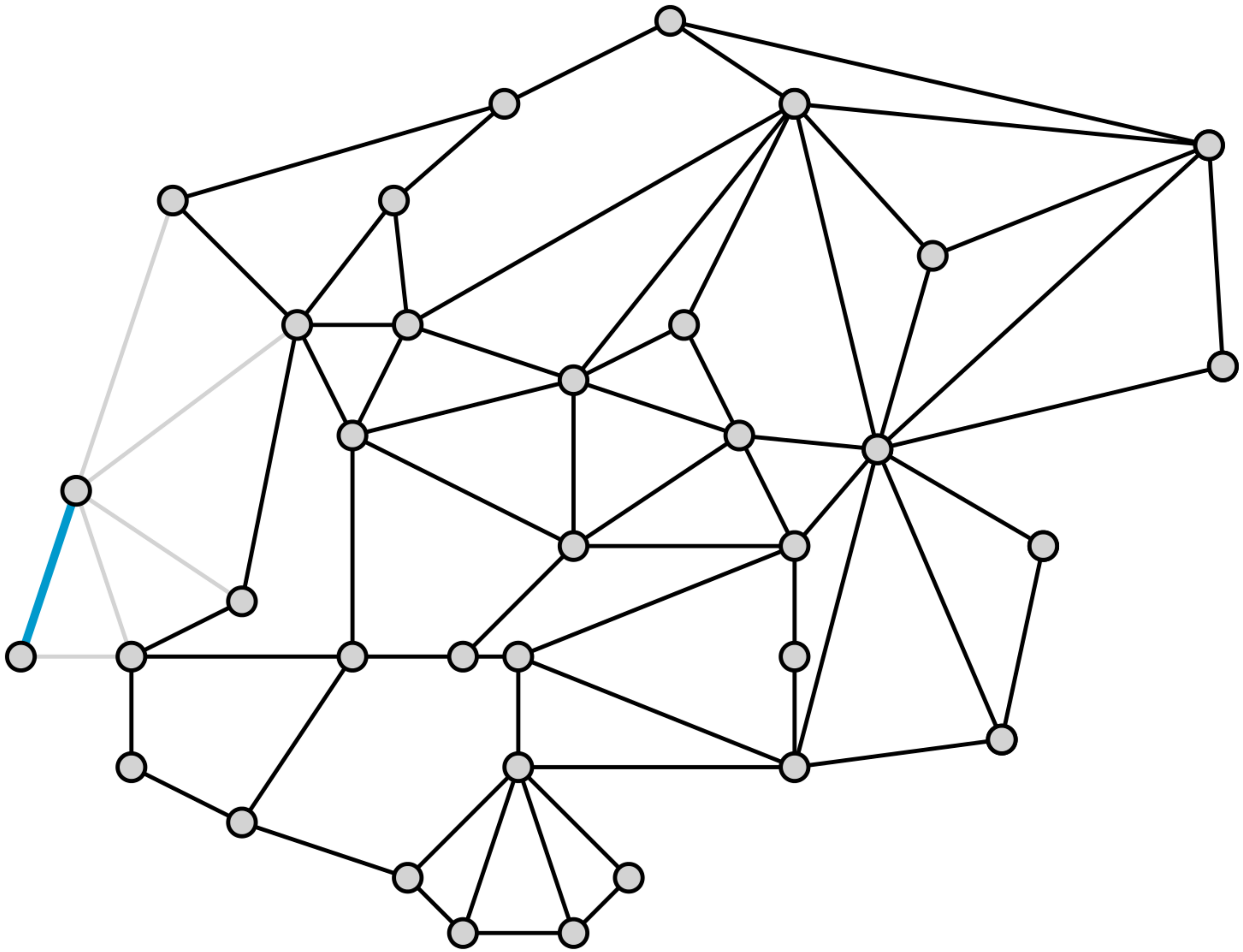
Centralized (Sequential) Algorithm



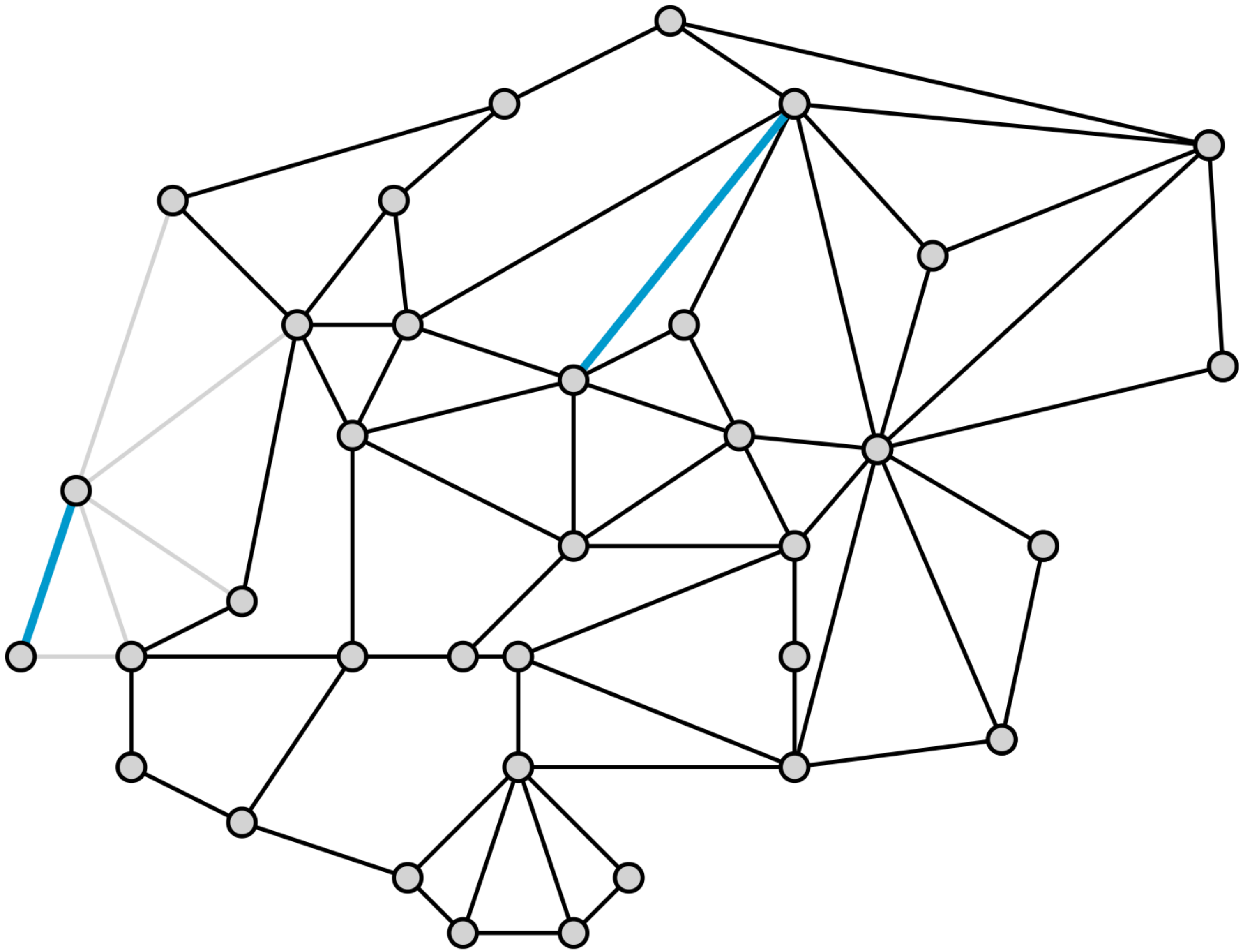
Centralized (Sequential) Algorithm



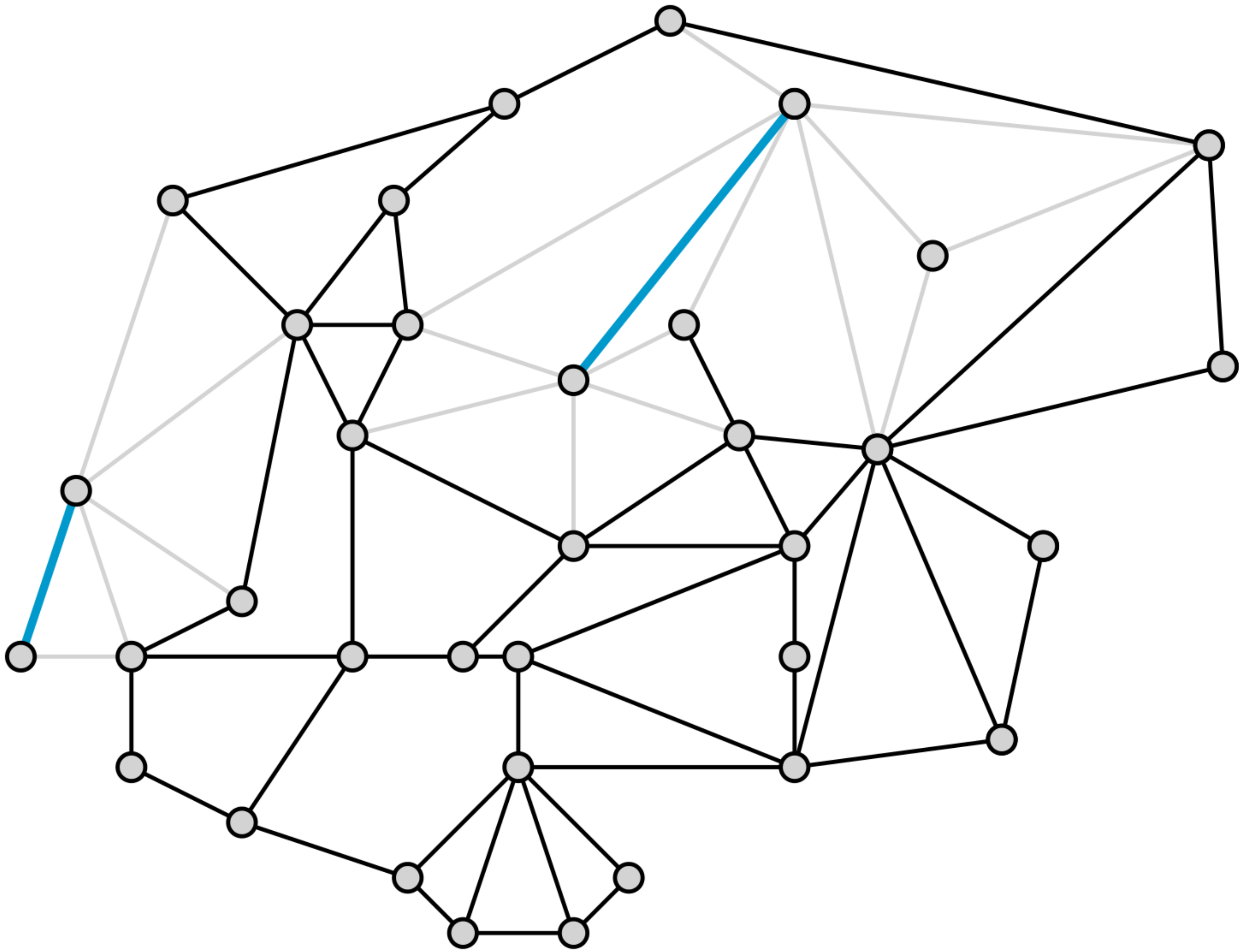
Centralized (Sequential) Algorithm



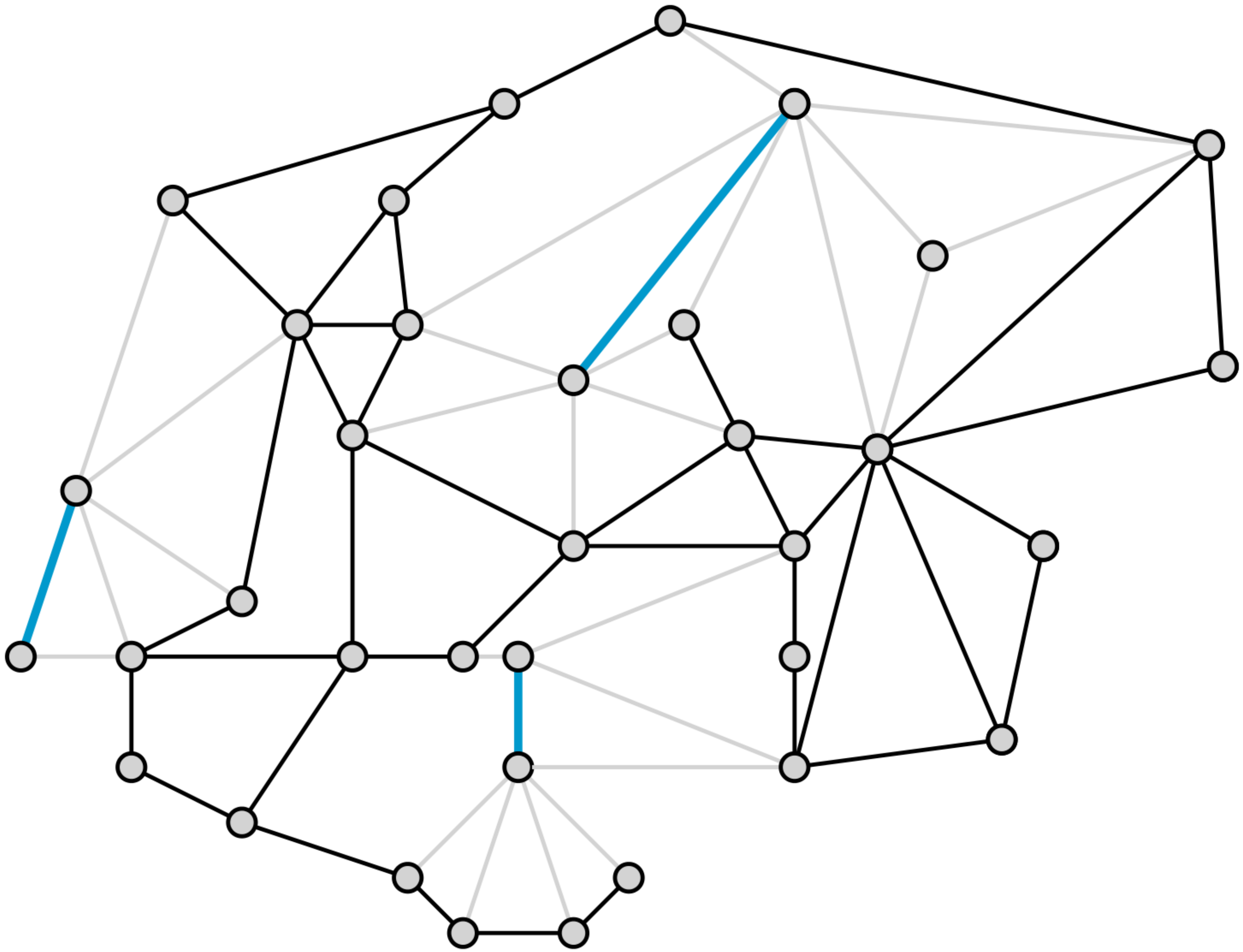
Centralized (Sequential) Algorithm



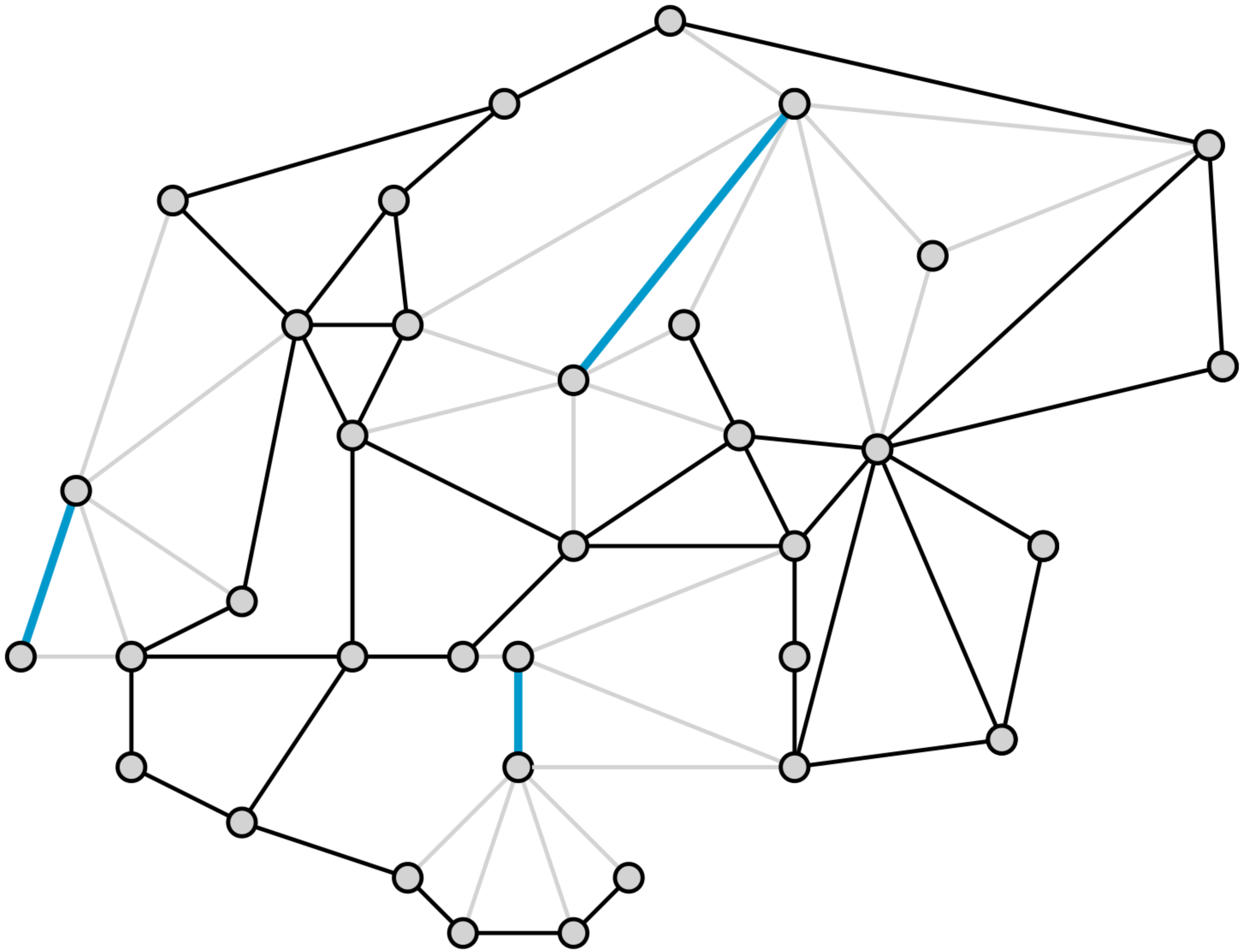
Centralized (Sequential) Algorithm



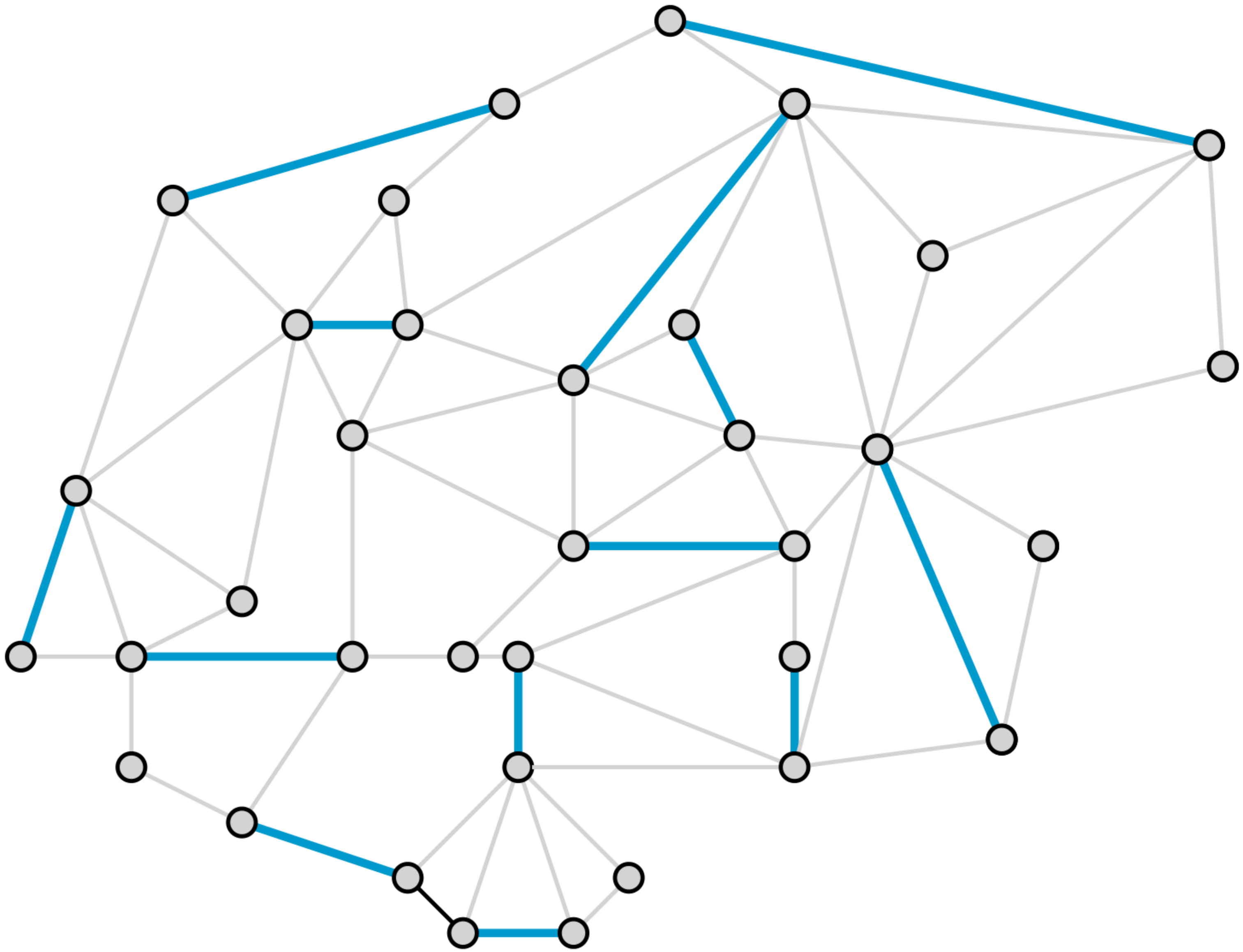
Centralized (Sequential) Algorithm



Centralized (Sequential) Algorithm

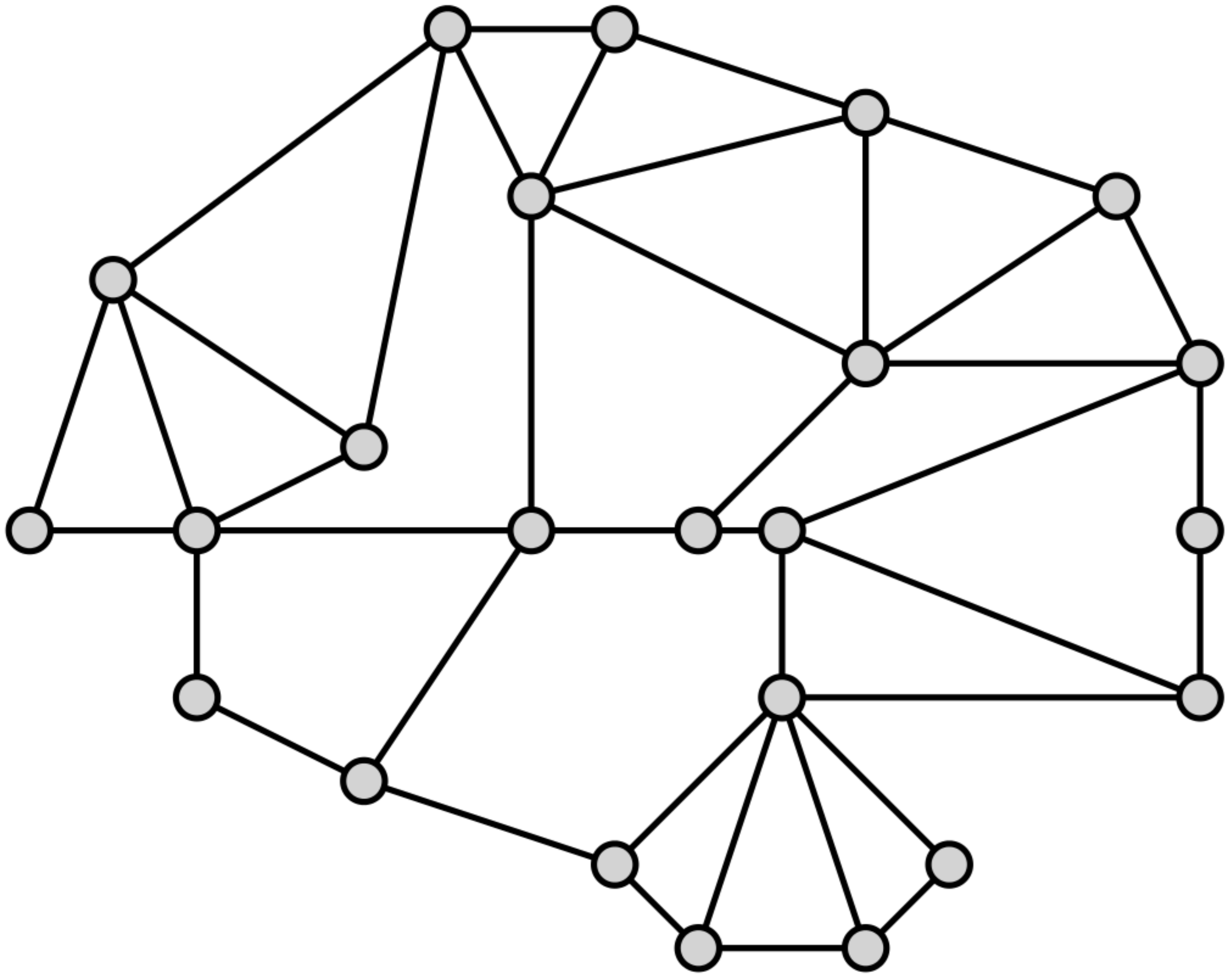


Centralized (Sequential) Algorithm

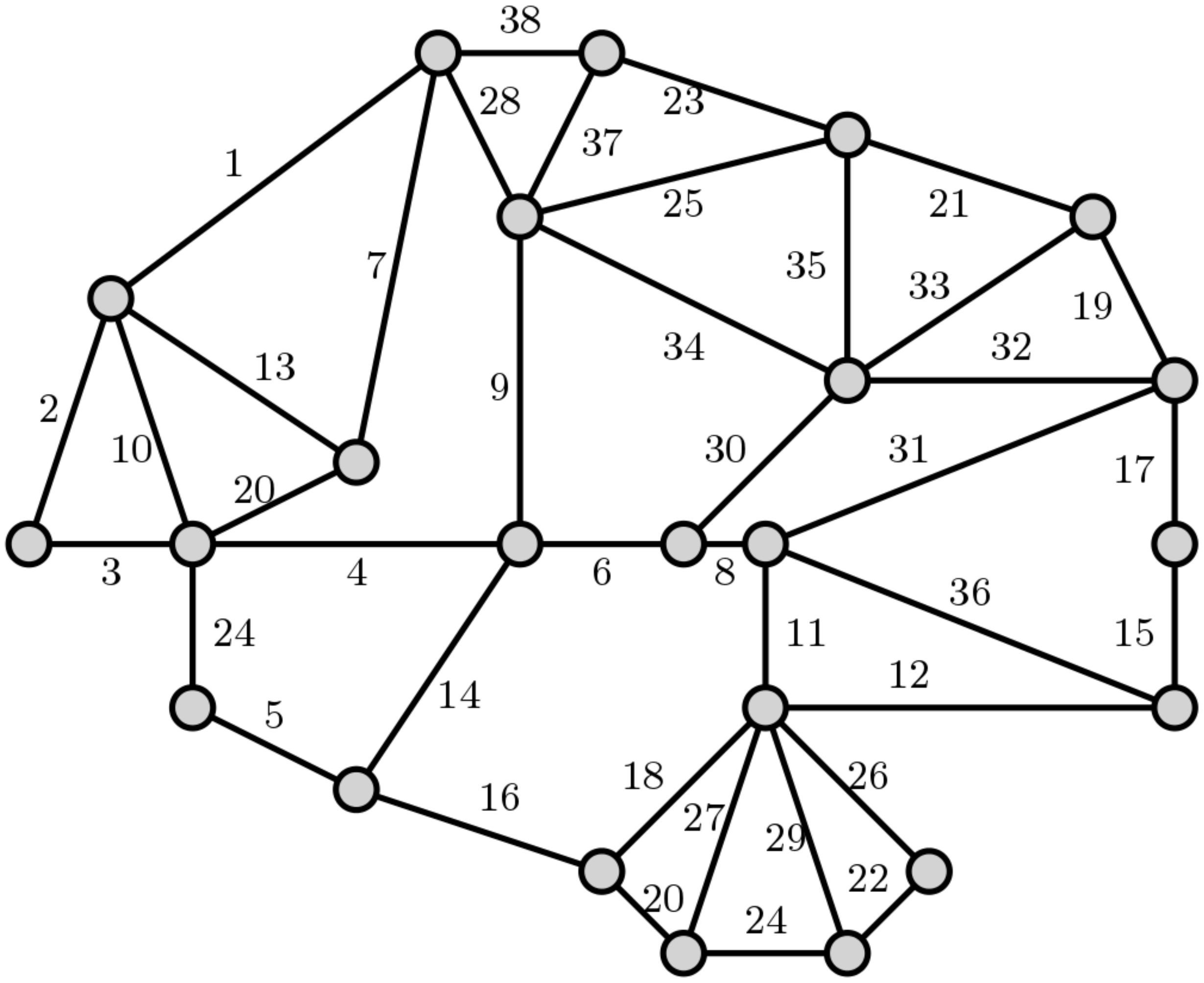


LOCAL Algorithm Mimicking Sequential Algorithm

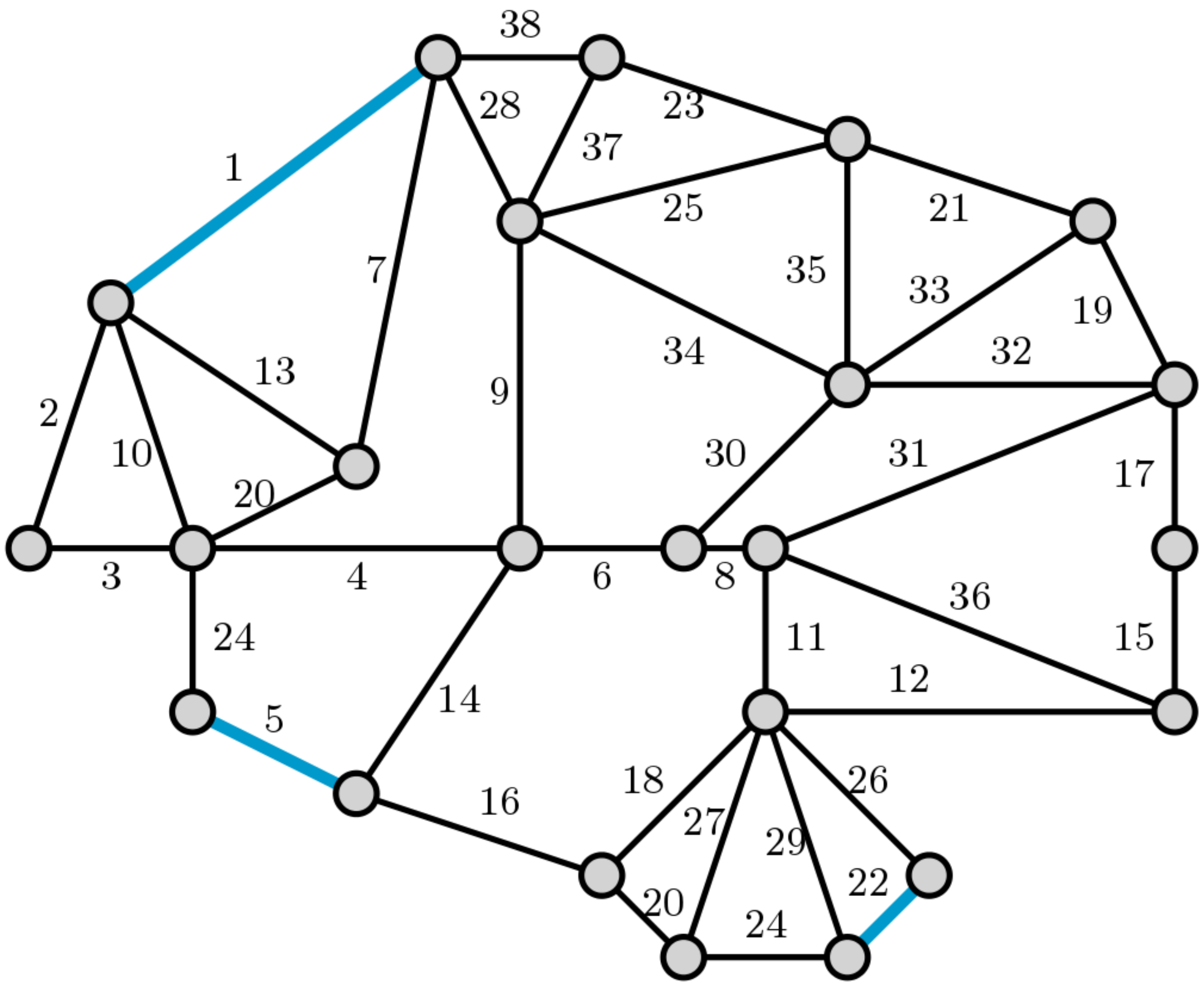
LOCAL Algorithm Mimicking Sequential Algorithm



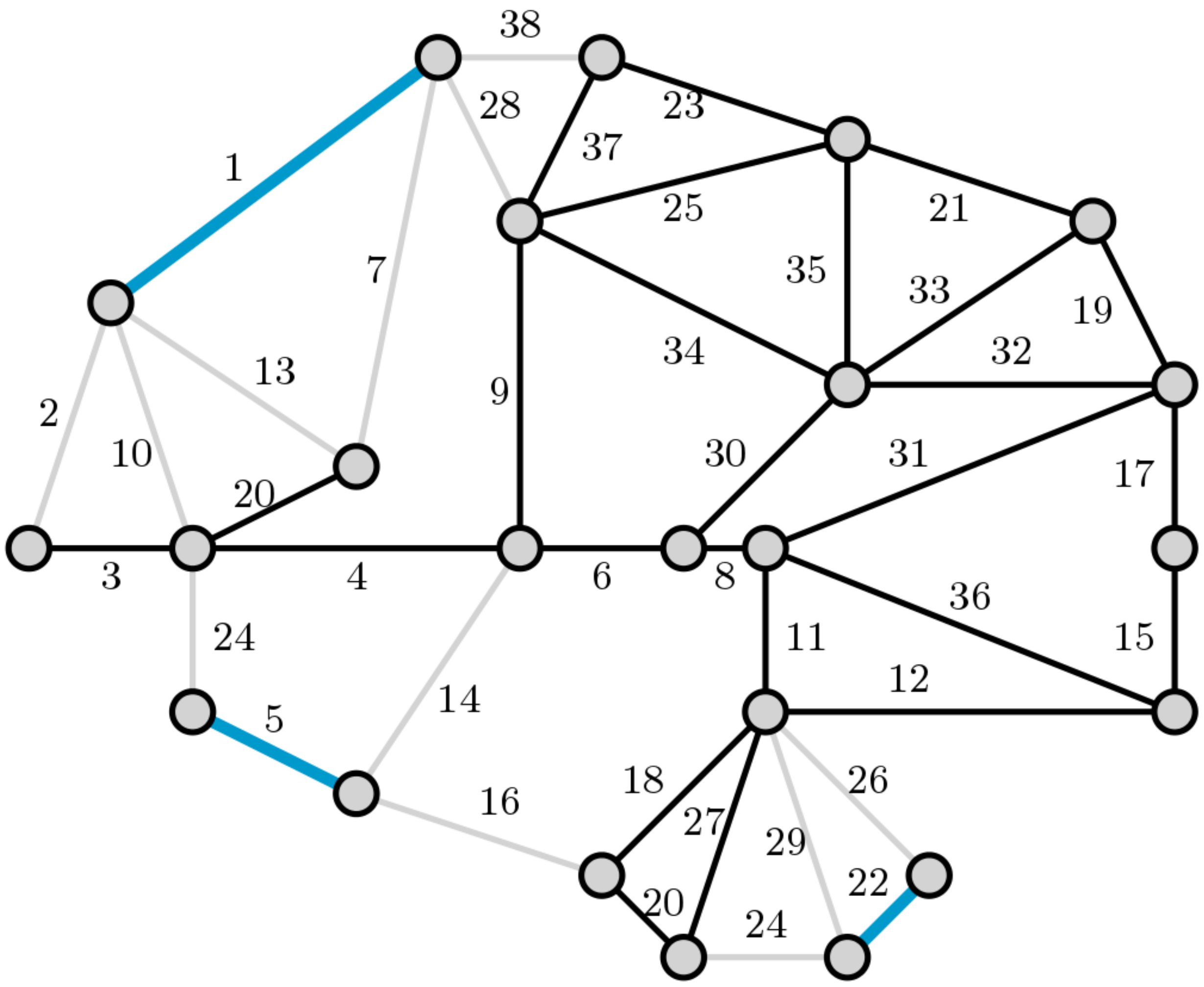
LOCAL Algorithm Mimicking Sequential Algorithm



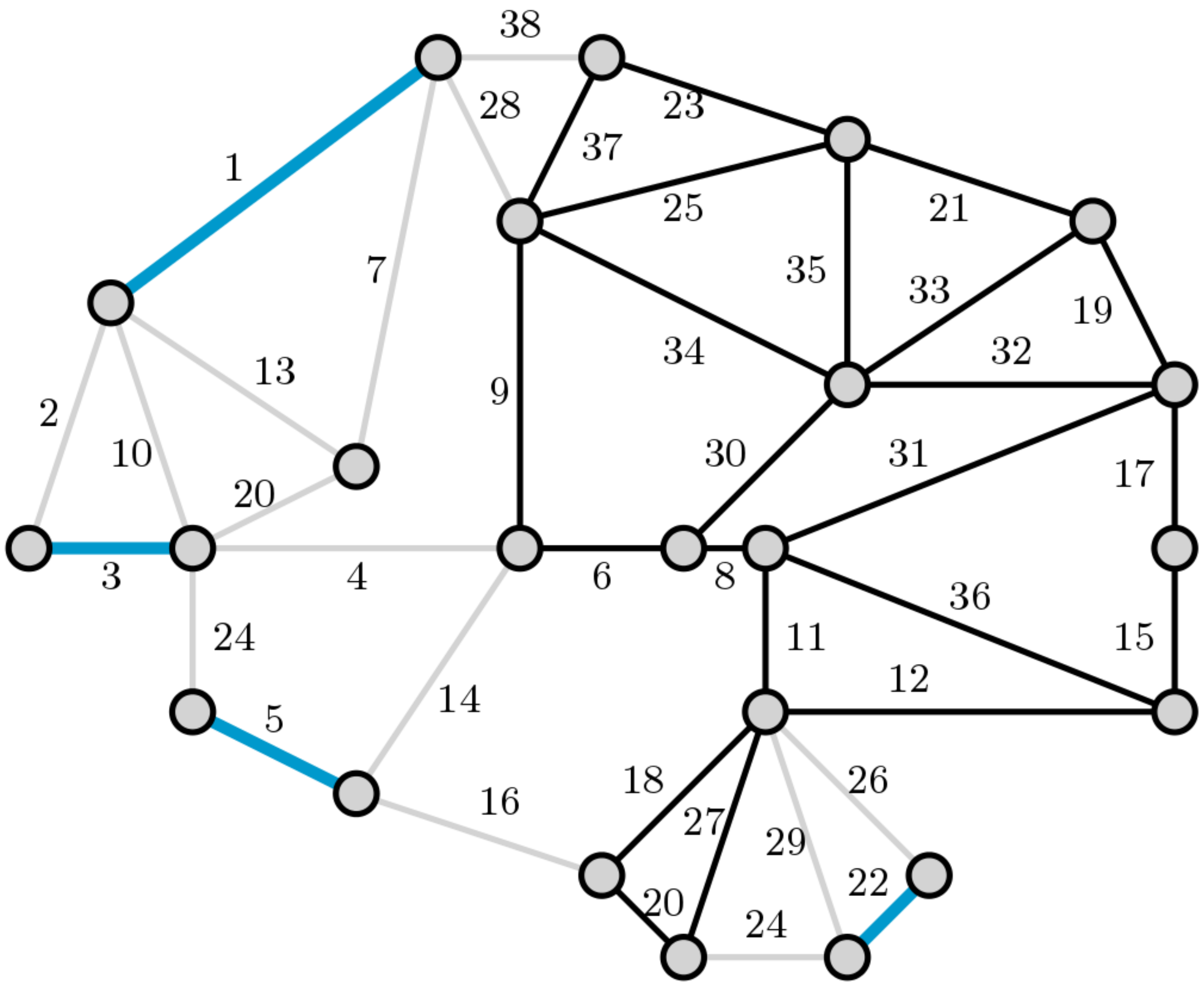
LOCAL Algorithm Mimicking Sequential Algorithm



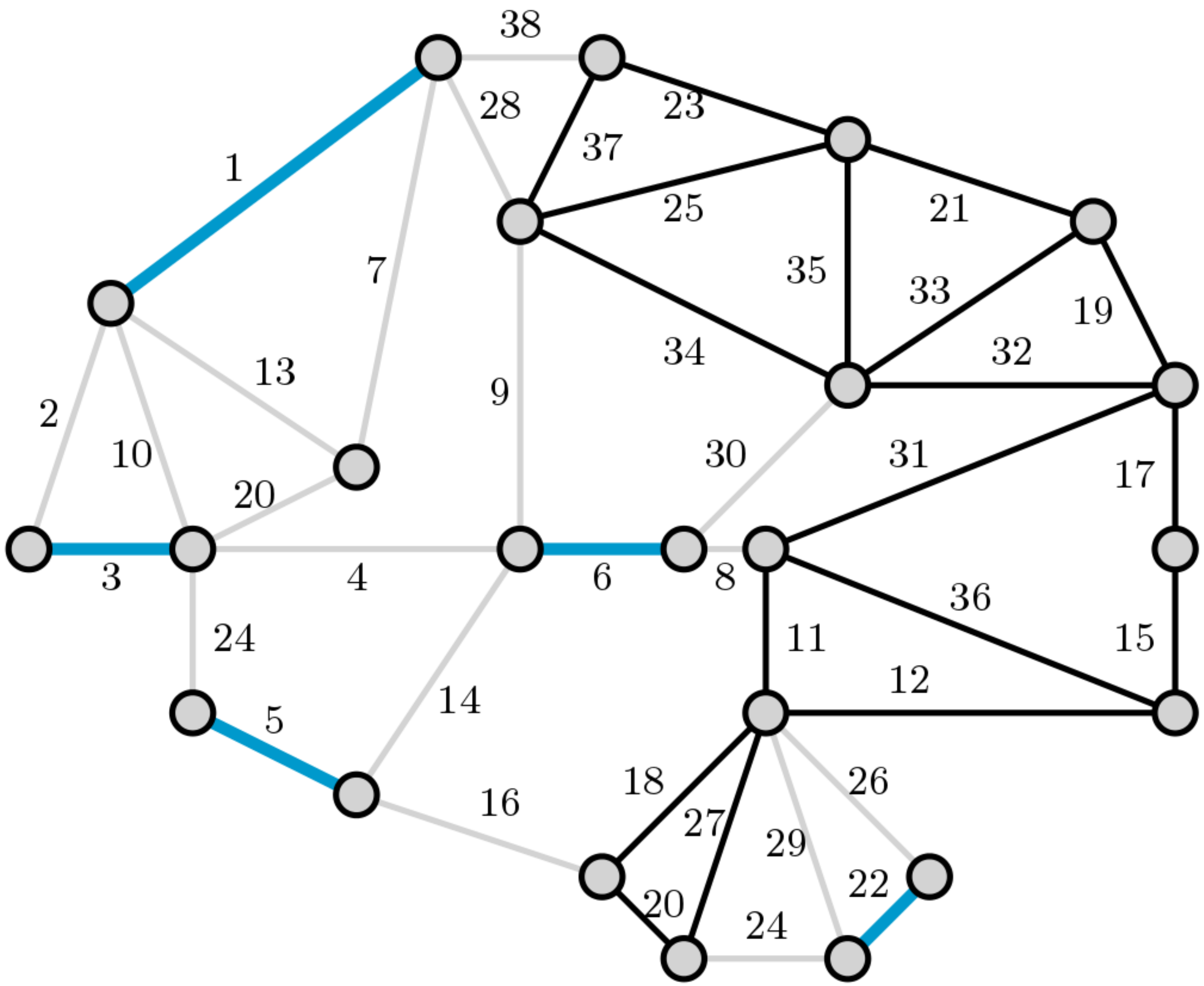
LOCAL Algorithm Mimicking Sequential Algorithm



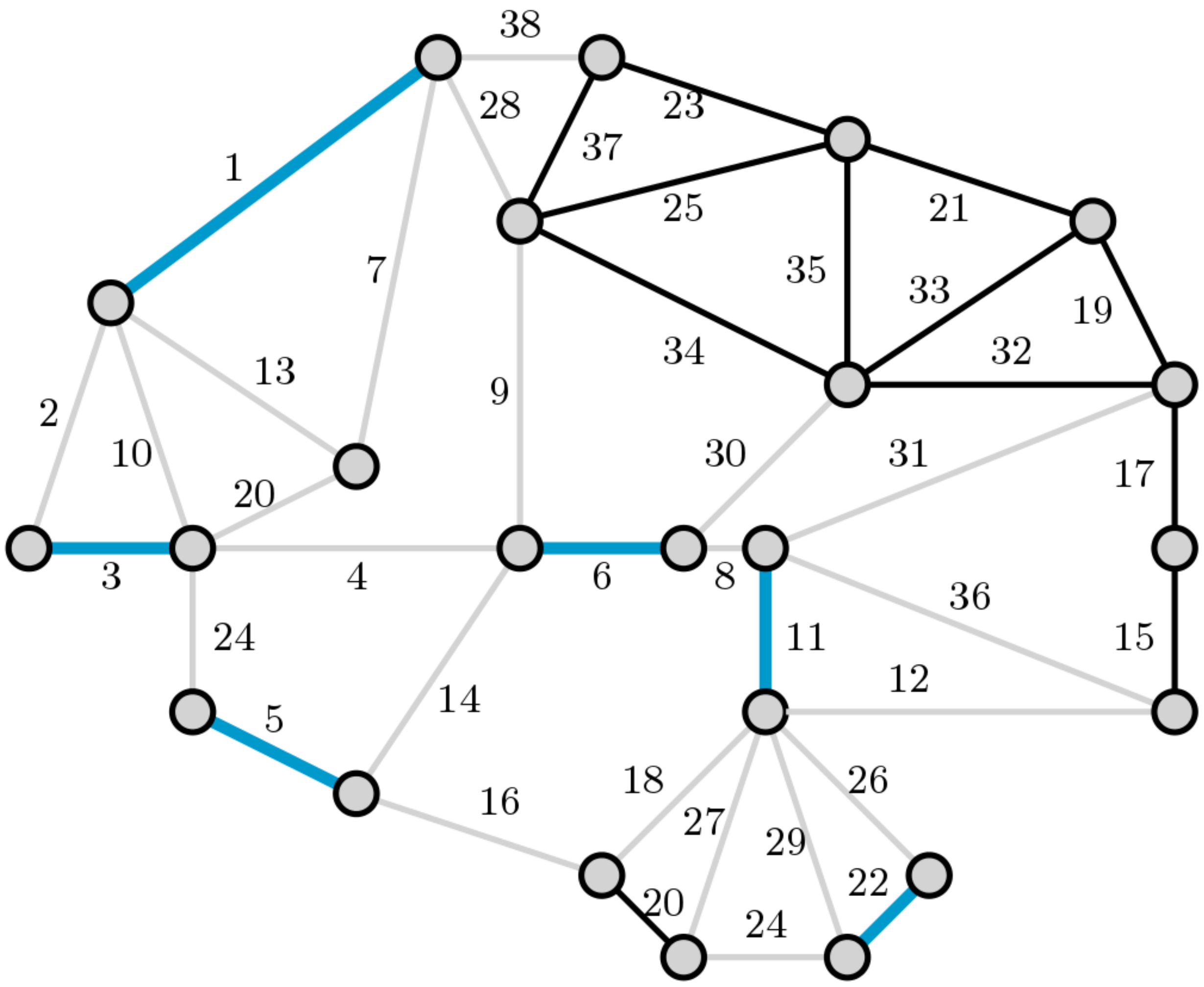
LOCAL Algorithm Mimicking Sequential Algorithm



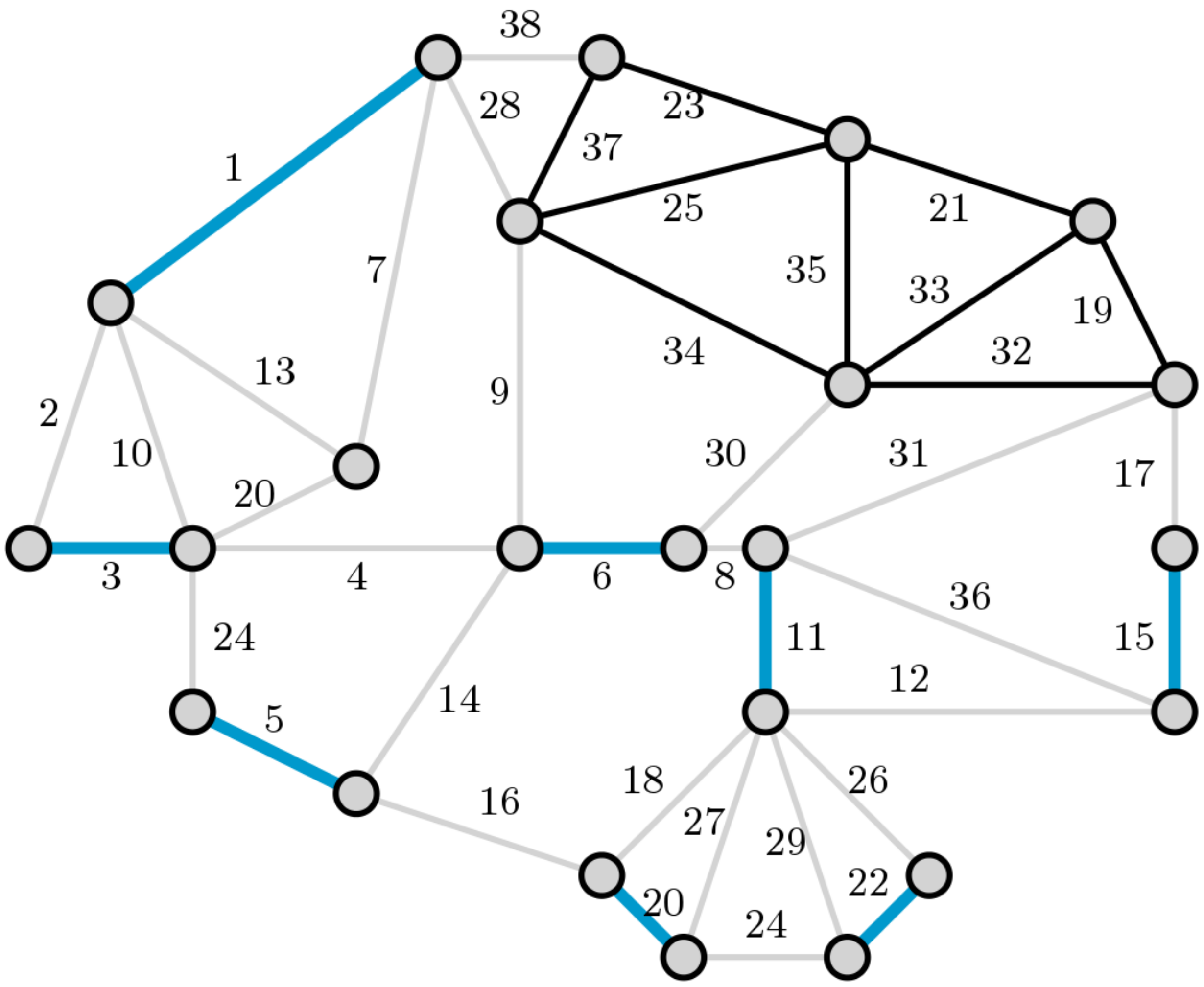
LOCAL Algorithm Mimicking Sequential Algorithm



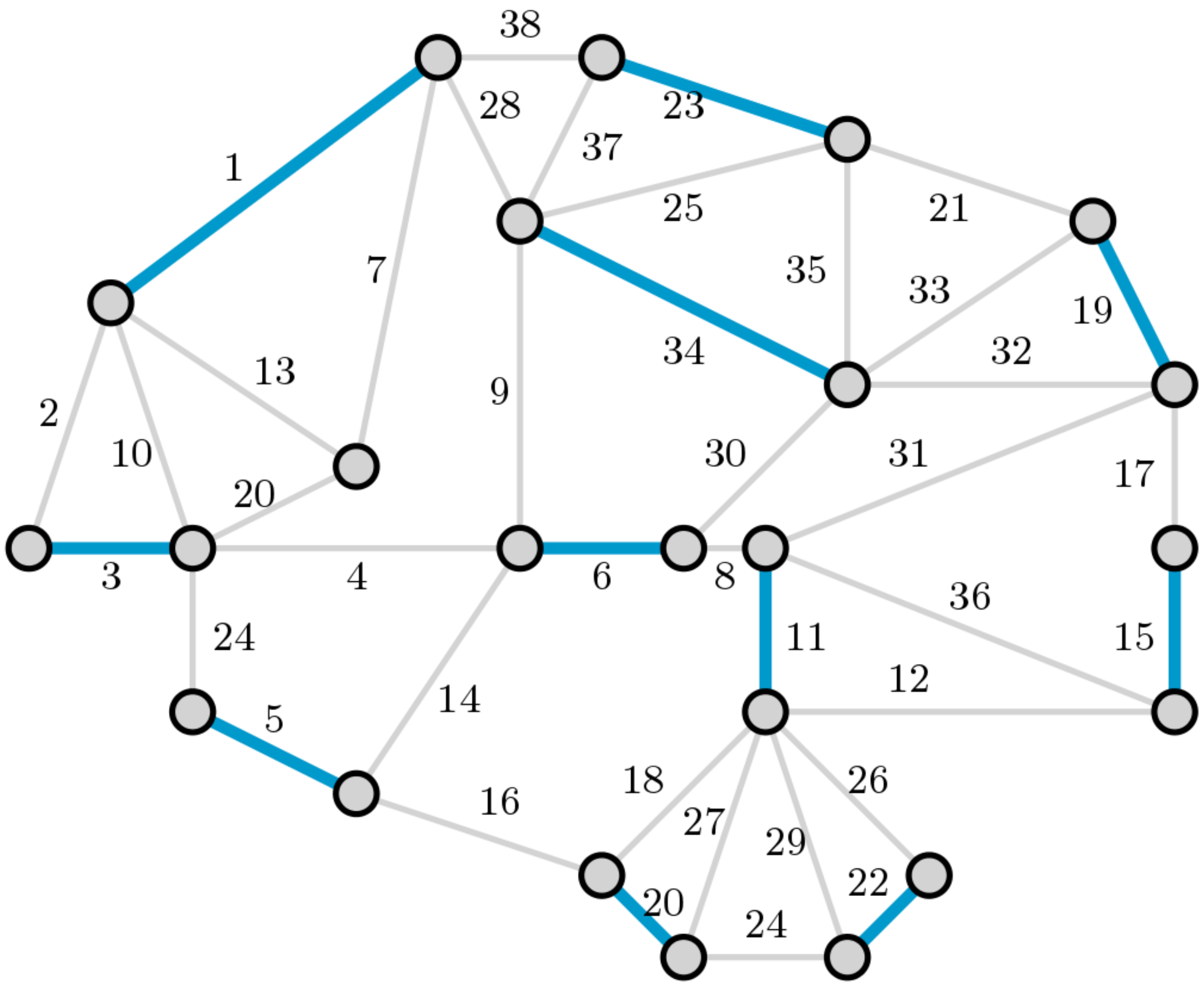
LOCAL Algorithm Mimicking Sequential Algorithm



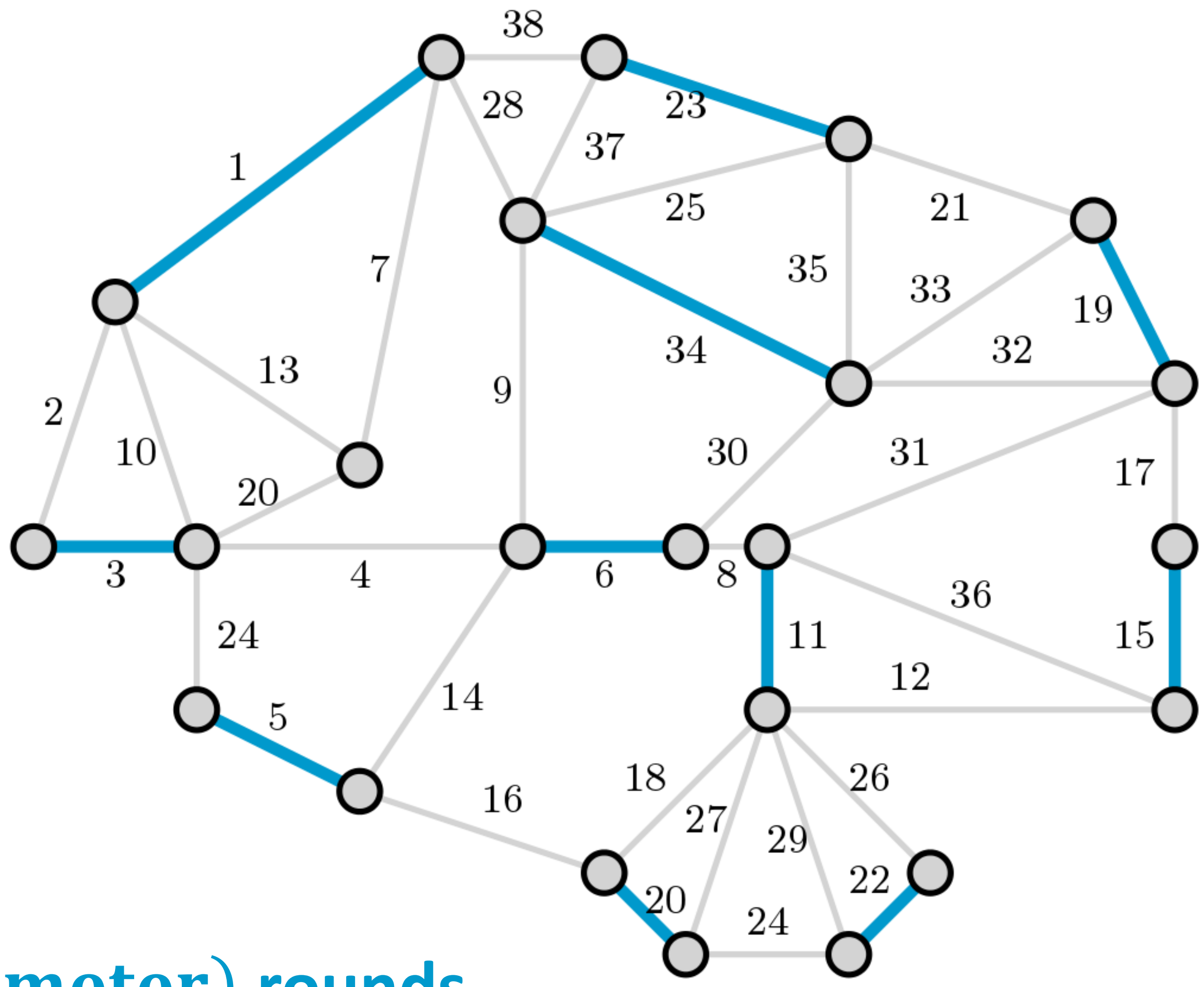
LOCAL Algorithm Mimicking Sequential Algorithm



LOCAL Algorithm Mimicking Sequential Algorithm

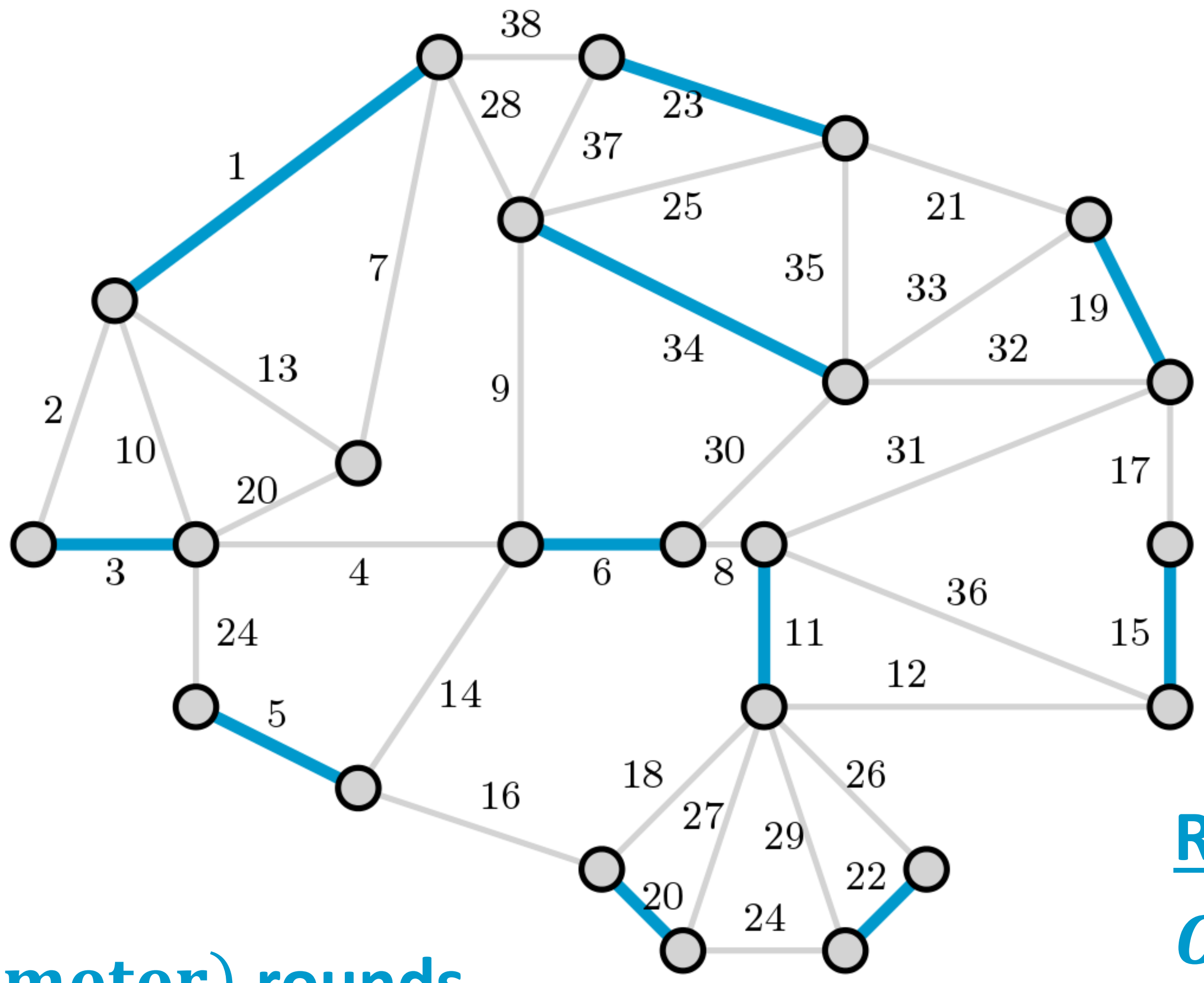


LOCAL Algorithm Mimicking Sequential Algorithm



can take $\Omega(\text{diameter})$ rounds in worst case

LOCAL Algorithm Mimicking Sequential Algorithm



can take $\Omega(\text{diameter})$ rounds in worst case

Random Numbers:

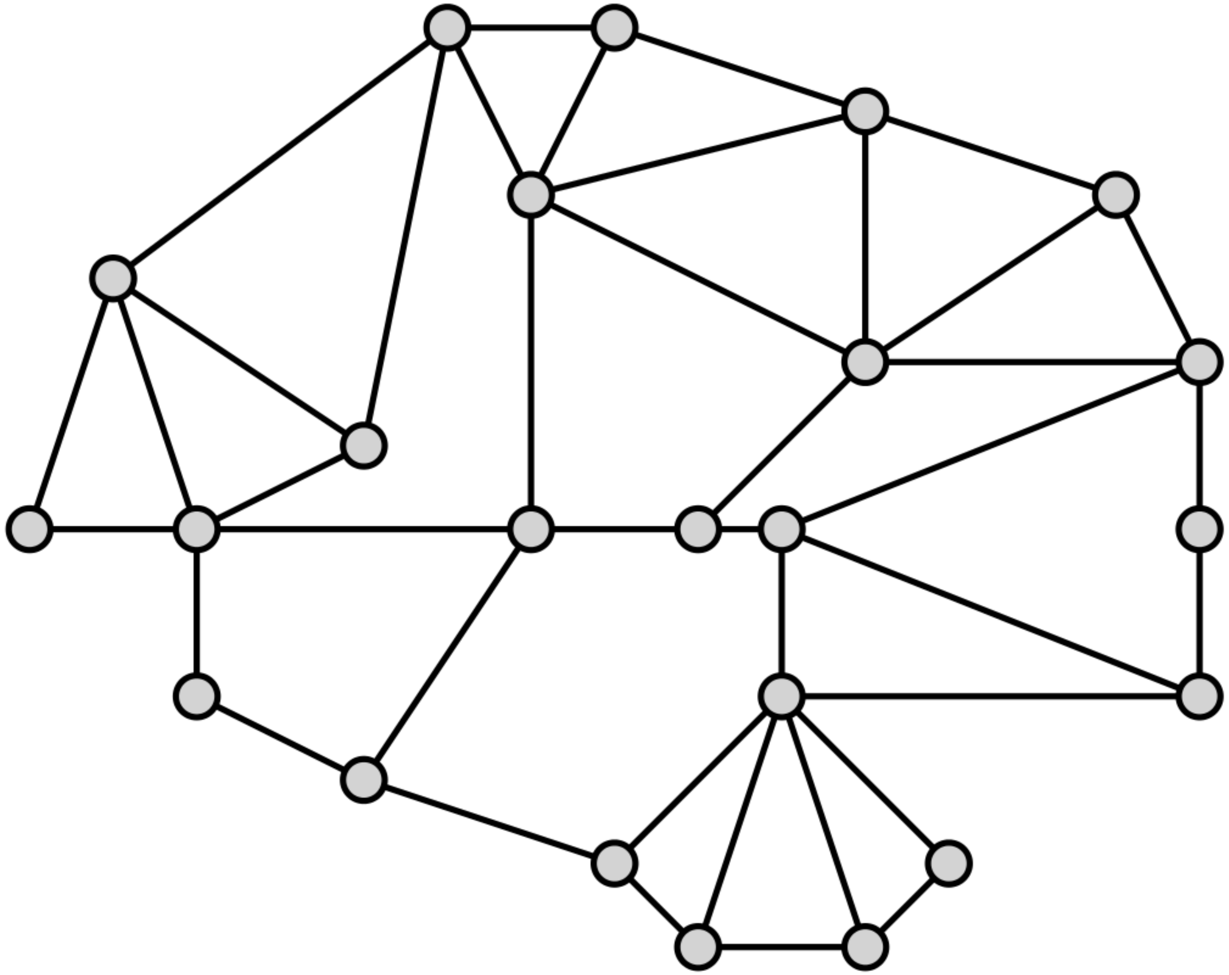
$O(\log n)$ rounds w.h.p.

Luby [STOC'85]

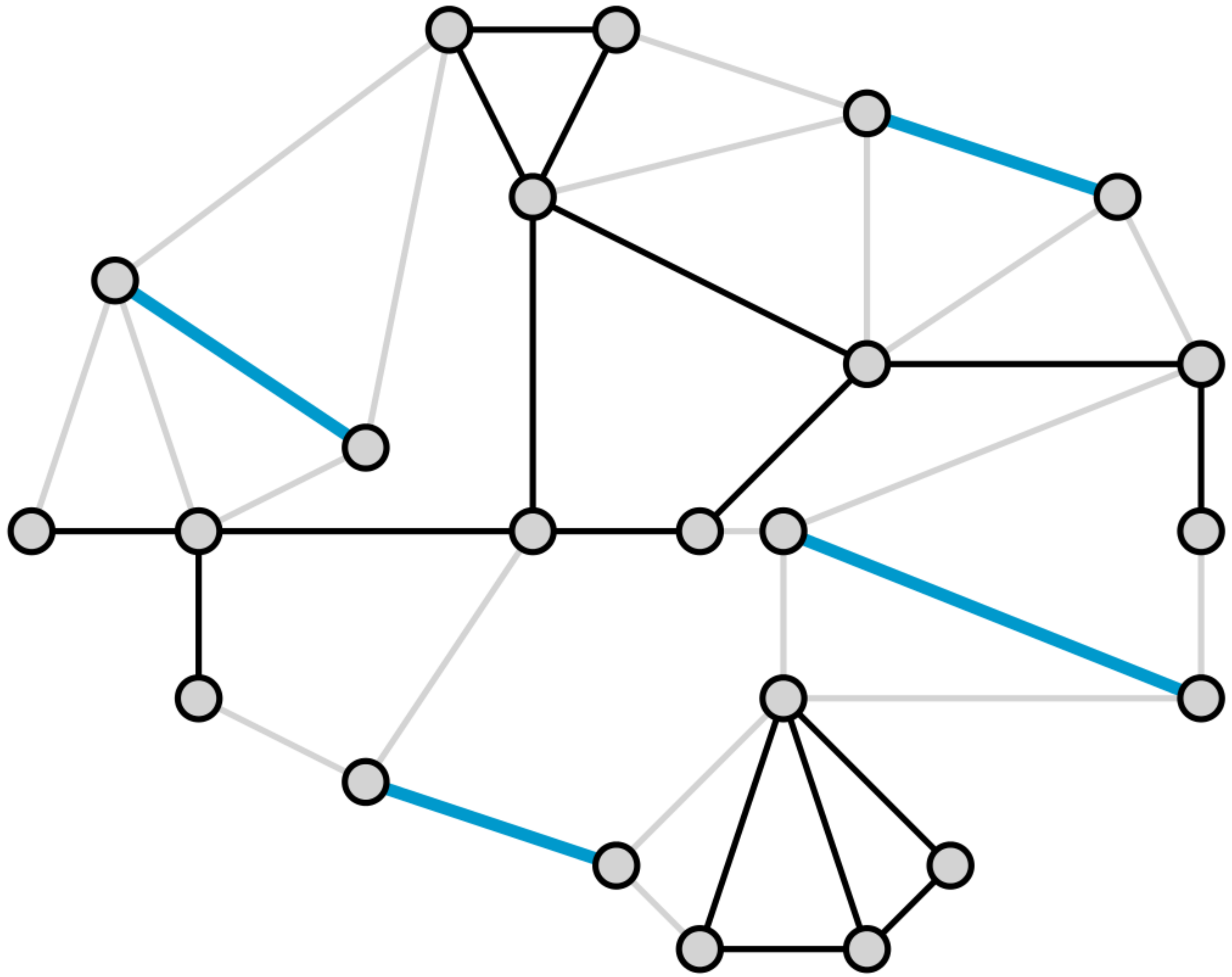
F., Noever [SODA'18]

LOCAL Algorithm: Luby’s Randomized Algorithm

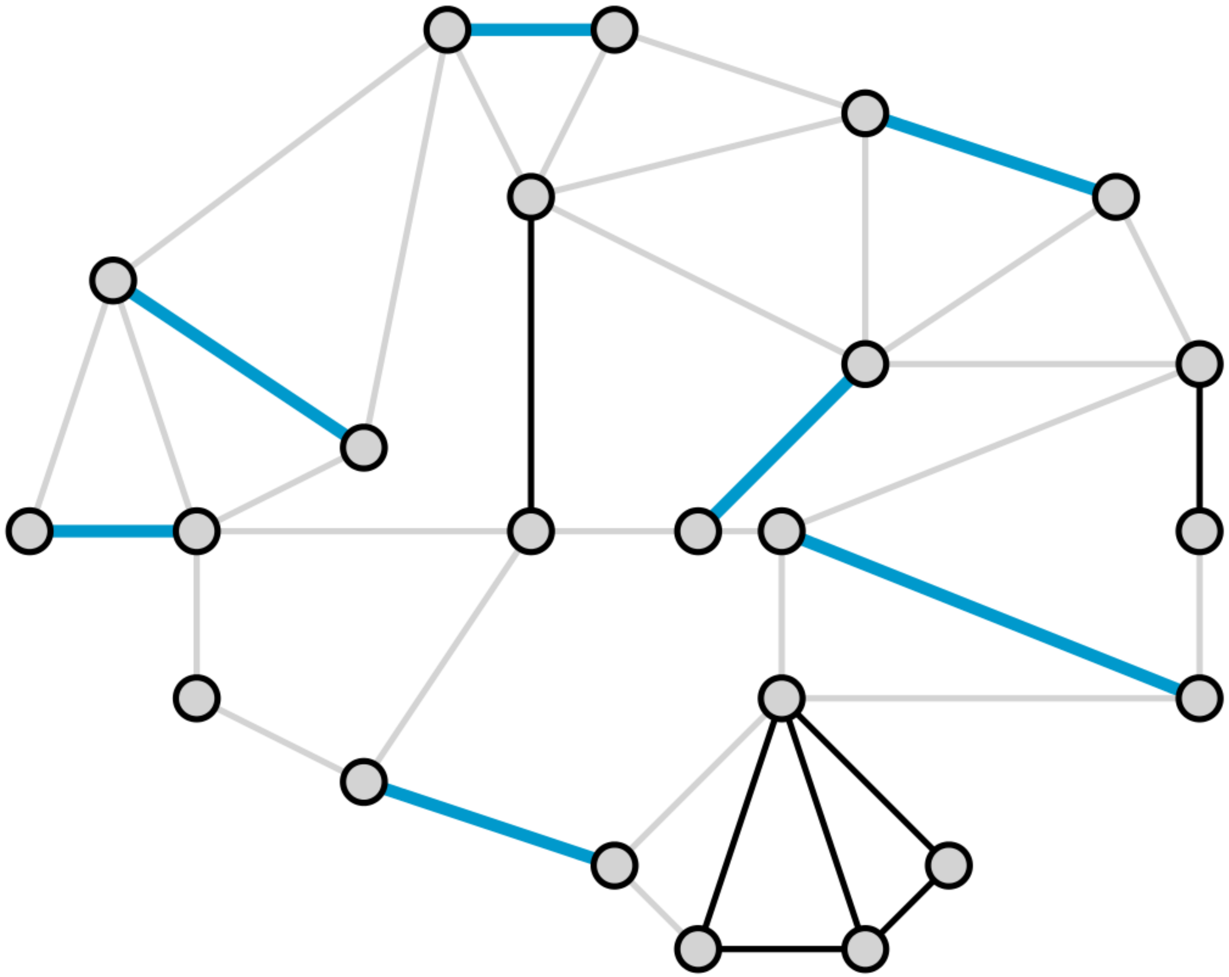
LOCAL Algorithm: Luby’s Randomized Algorithm



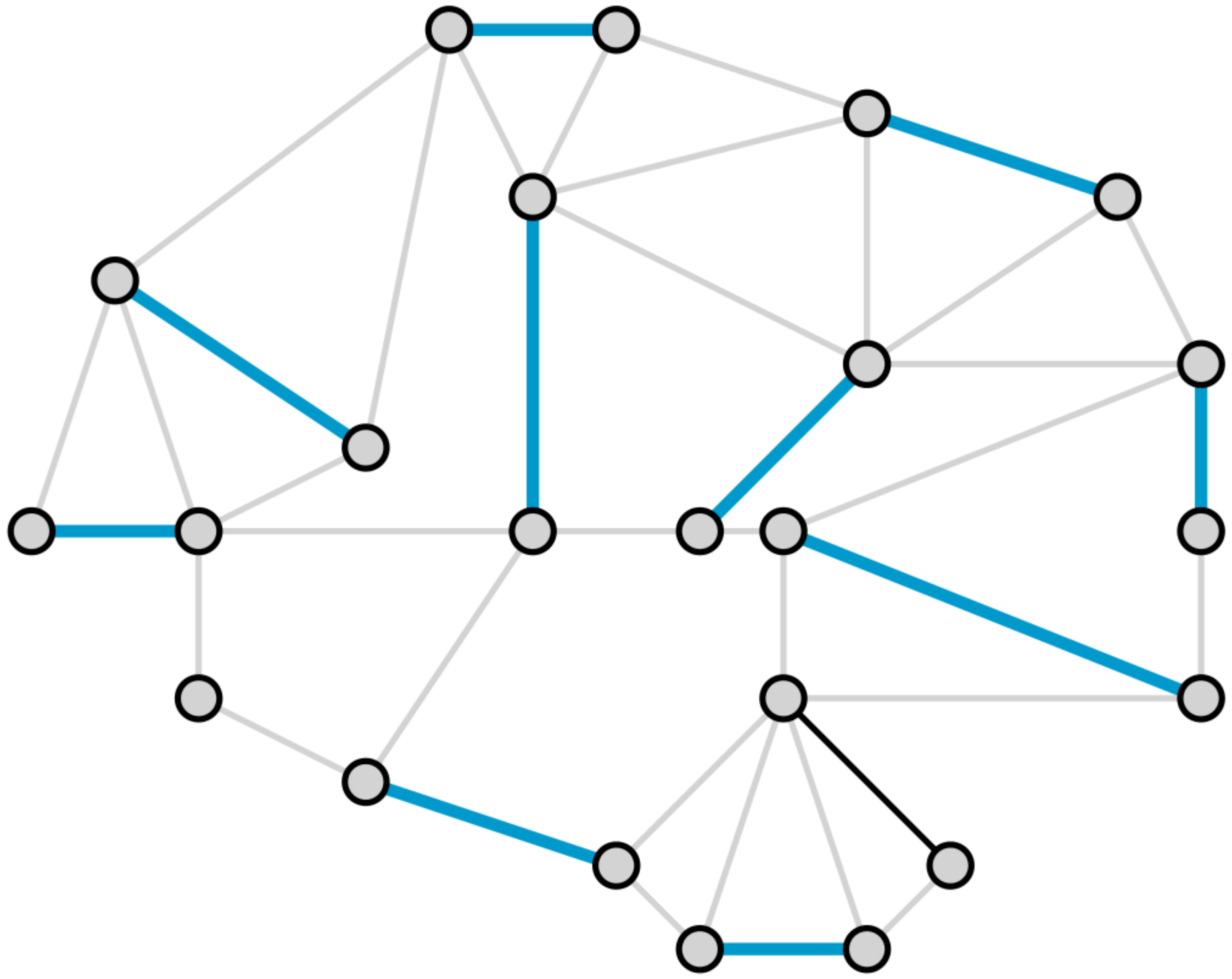
LOCAL Algorithm: Luby's Randomized Algorithm



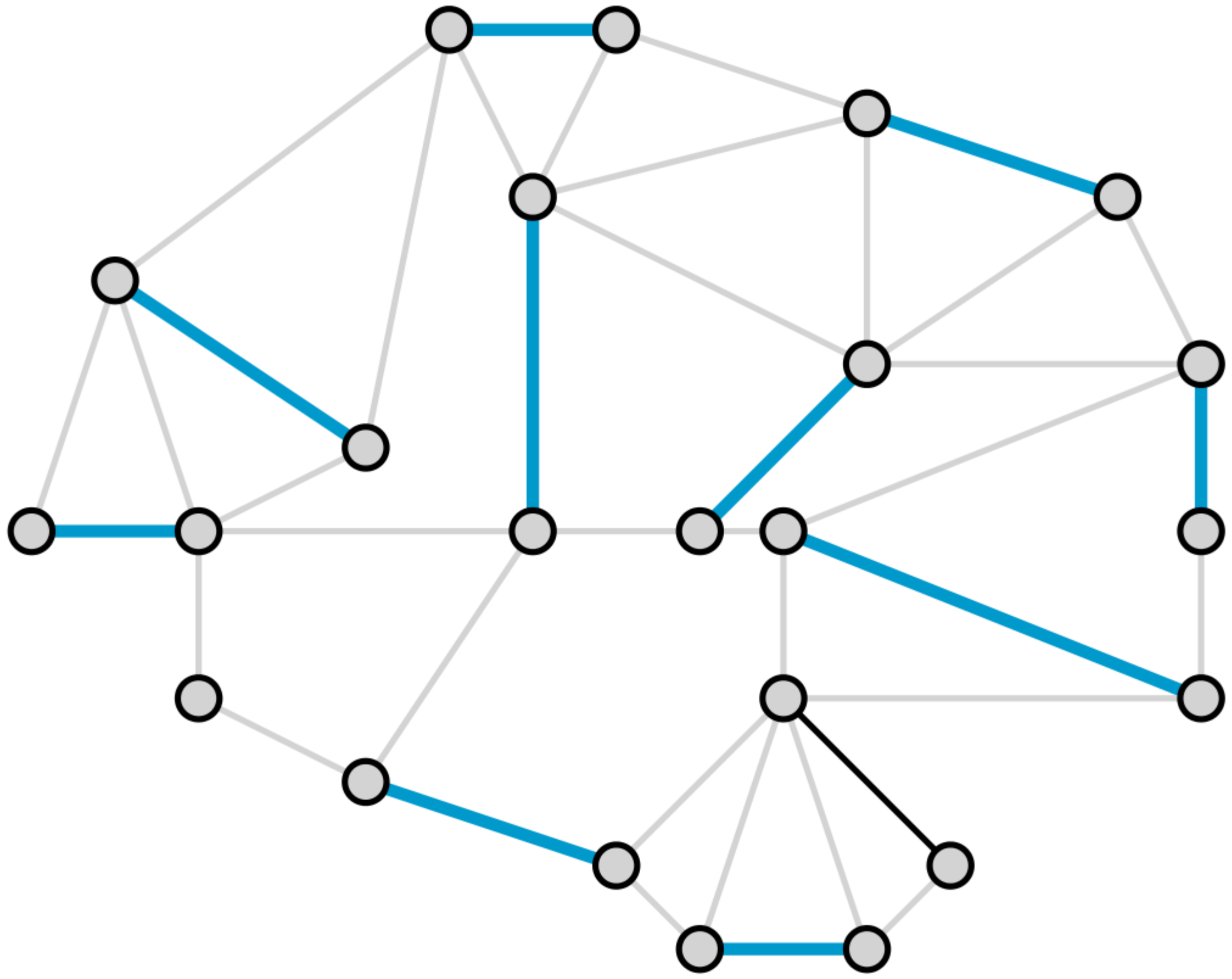
LOCAL Algorithm: Luby's Randomized Algorithm



LOCAL Algorithm: Luby's Randomized Algorithm

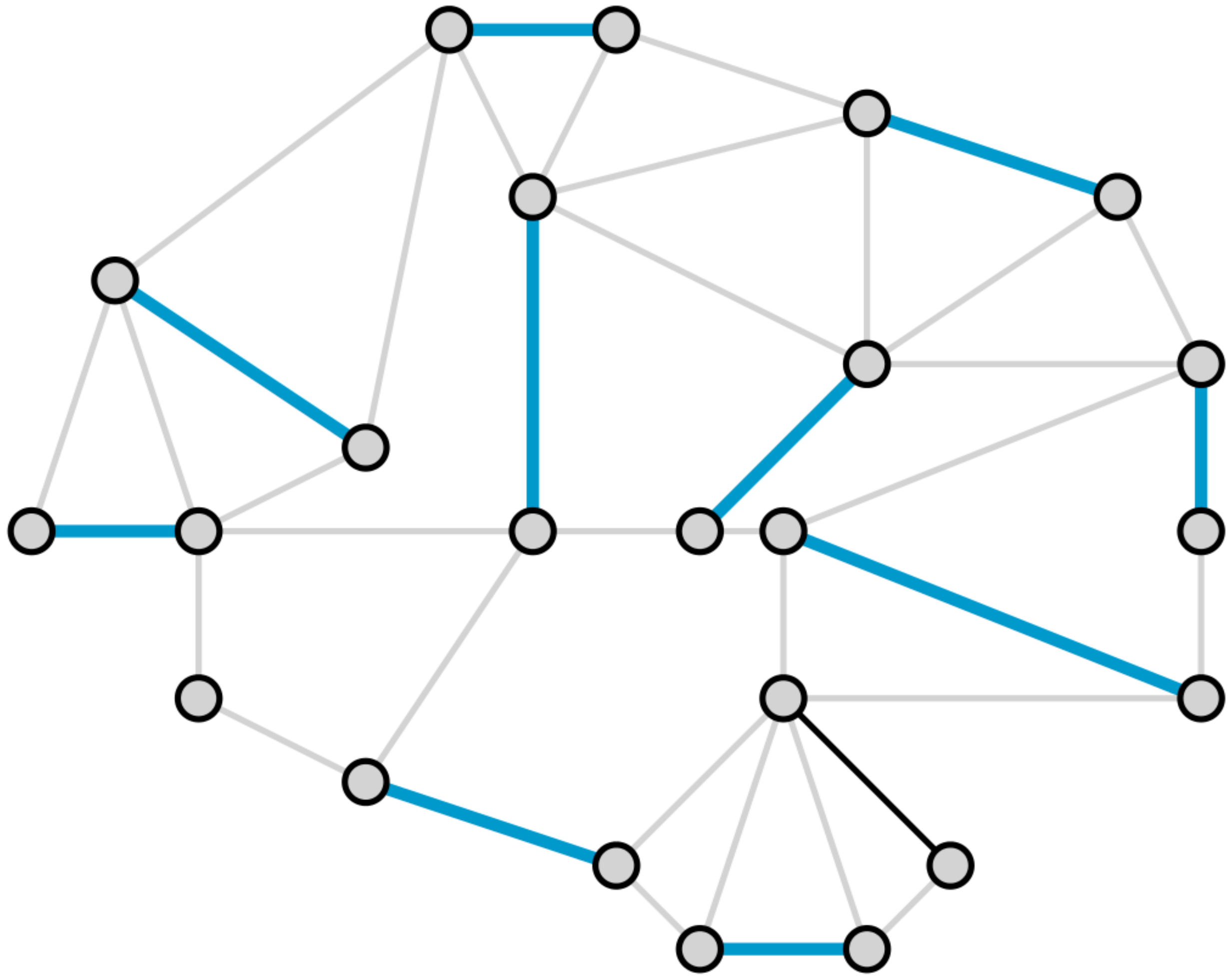


LOCAL Algorithm: Luby's Randomized Algorithm



$\mathbb{E}[\#\text{removed edges per round}] \geq c|E_i|$

LOCAL Algorithm: Luby's Randomized Algorithm



$\mathbb{E}[\#\text{removed edges per round}] \geq c|E_i|$

$O(\log n)$ rounds w.h.p.

Our Result

Our Result

deterministic $O(\log^2 \Delta \cdot \log n)$-round Maximal Matching

Our Result

deterministic $O(\log^2 \Delta \cdot \log n)$-round Maximal Matching

improving over

Our Result

deterministic $O(\log^2 \Delta \cdot \log n)$-round Maximal Matching

improving over

$O(\log^4 n)$
Hańćkowiak, Karoński, Panconesi [SODA'98, PODC'99]

Our Result

<u>deterministic</u> $O(\log^2 \Delta \cdot \log n)$-round Maximal Matching

improving over

$O(\log^4 n)$
Hańćkowiak, Karoński, Panconesi [SODA'98, PODC'99]

$O(\Delta + \log^* n)$
Panconesi, Rizzi [DIST'01]

Overview of Results

Maximal Matching

• Maximal Matching	$O(\log^2 \Delta \cdot \log n)$
• Randomized Maximal Matching	$O(\log^3 \log n + \log \Delta)$

Approximate Matching

• $(2+\varepsilon)$ - Approximate Maximum Matching	$O\left(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n\right)$
• $(2+\varepsilon)$ - Approximate Maximum Weighted Matching	$O\left(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n\right)$
• $(2+\varepsilon)$ - Approximate Maximum B-Matching	$O\left(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n\right)$
• $(2+\varepsilon)$ - Approximate Maximum Weighted B-Matching	$O\left(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n\right)$
• ε - Maximal Matching	$O\left(\log^2 \Delta \cdot \log \frac{1}{\varepsilon}\right)$
• $(2+\varepsilon)$ - Approximate Minimum Edge Dominating Set	$O\left(\log^2 \Delta \cdot \log \frac{1}{\varepsilon}\right)$

Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds

Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds

Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds

Constant - Approximate Bipartite Matching

$O(\log^2 \Delta)$ rounds

I) 4 - Approximate Fractional Matching

$O(\log \Delta)$ rounds

Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds

I) 4 - Approximate Fractional Matching

$O(\log \Delta)$ rounds

II) Rounding Fractional Bipartite Matching

$\mathrm{O}(\log^2 \Delta)$ rounds, $\mathrm{O}(1)$ loss

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds

Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds

Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds

Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

s.t. $\sum_{e \in E(v)} x_e \leq 1$ for all $v \in V$

$x_e \in [0,1]$ for all $e \in E$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

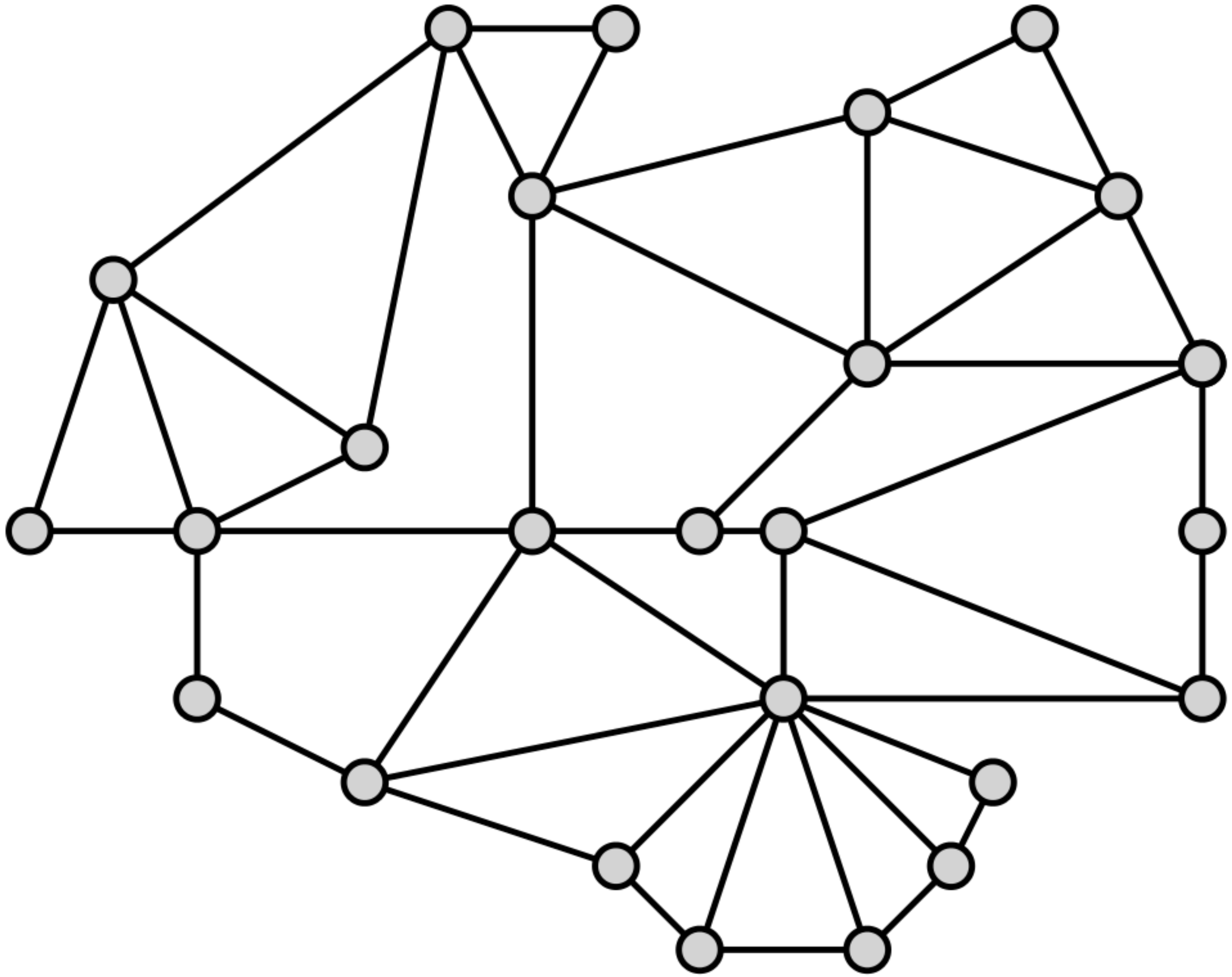
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

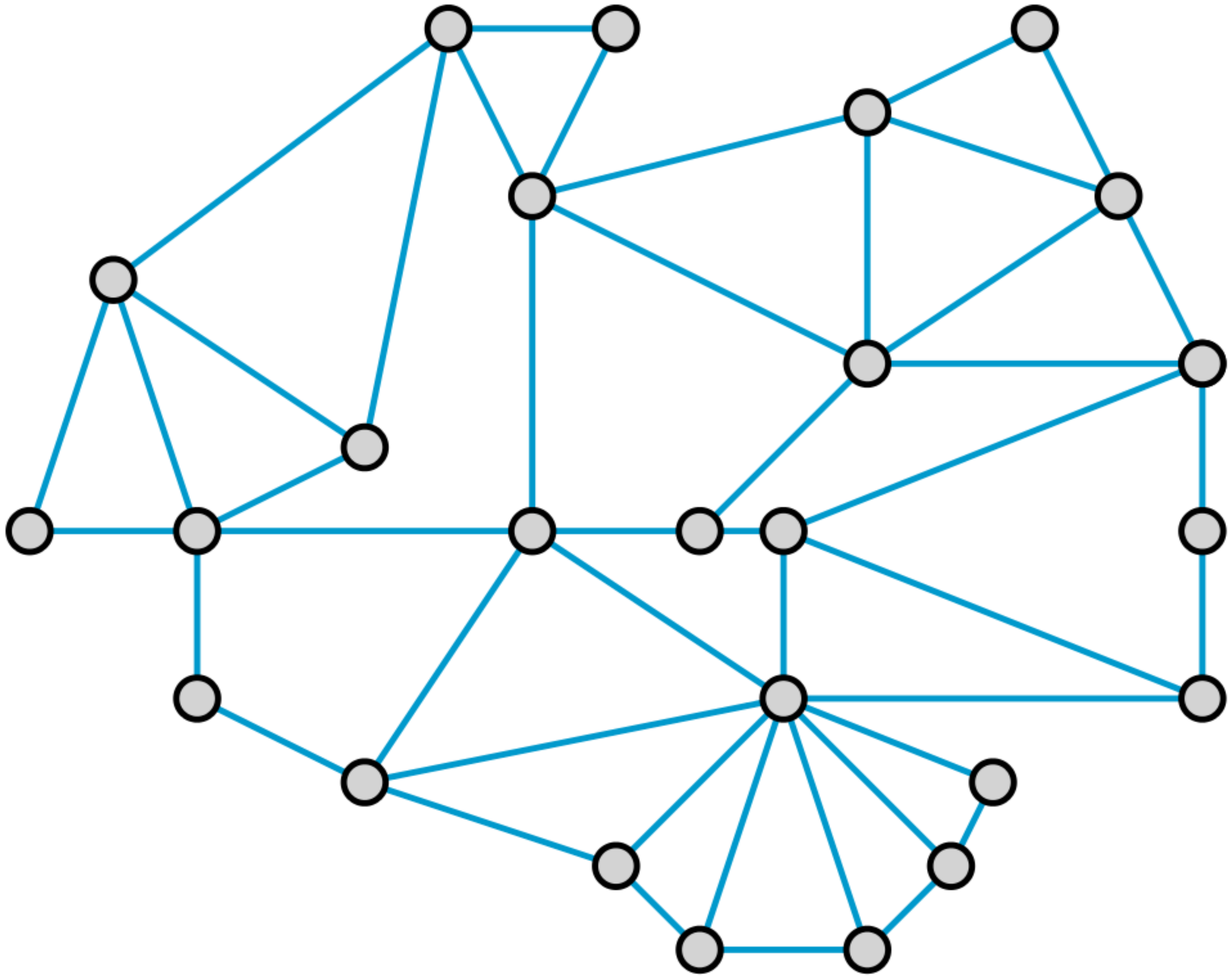
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

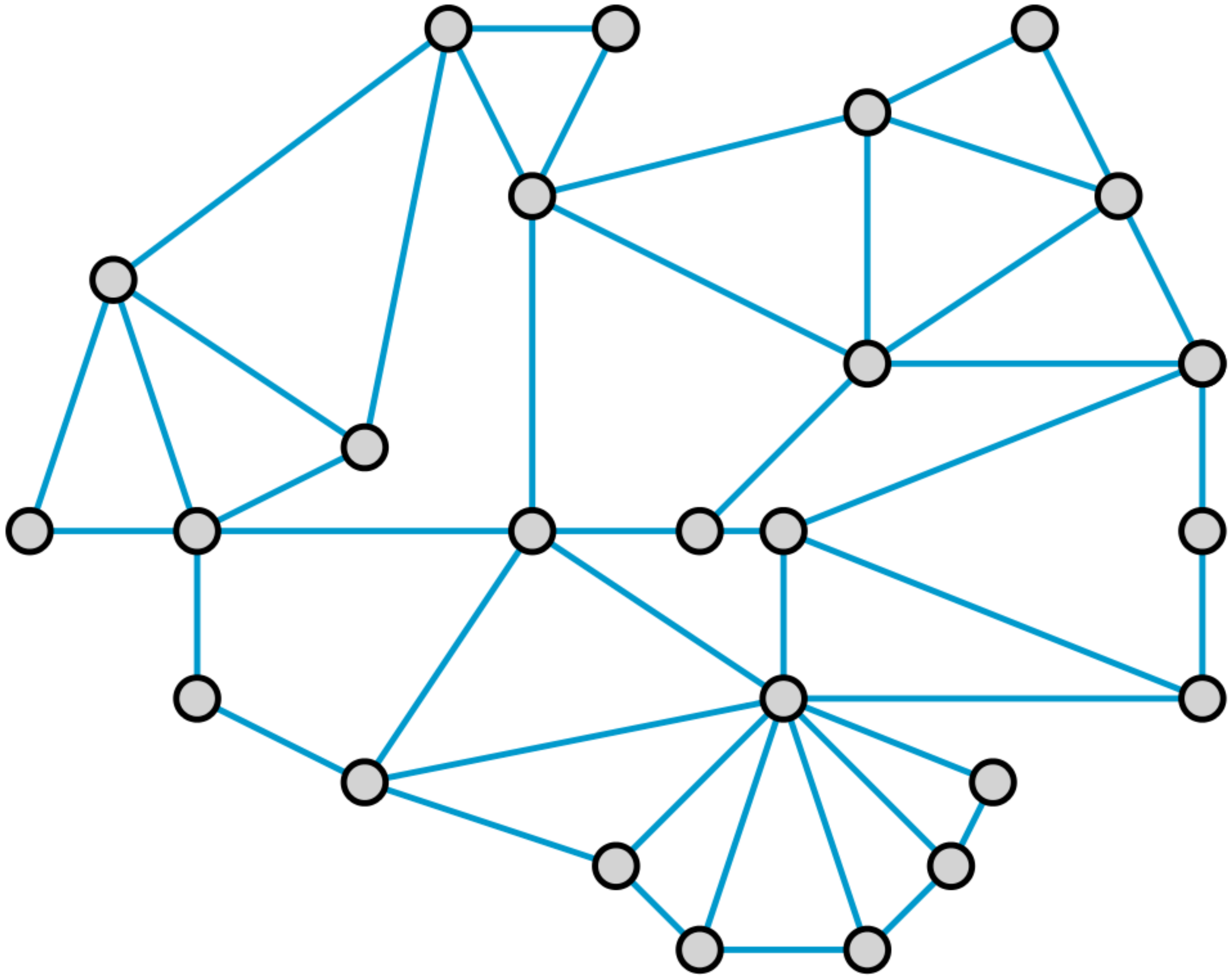
LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

repeat until all edges are blocked

- mark half-tight nodes
- block its edges
- double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds



$\frac{1}{16}$

Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

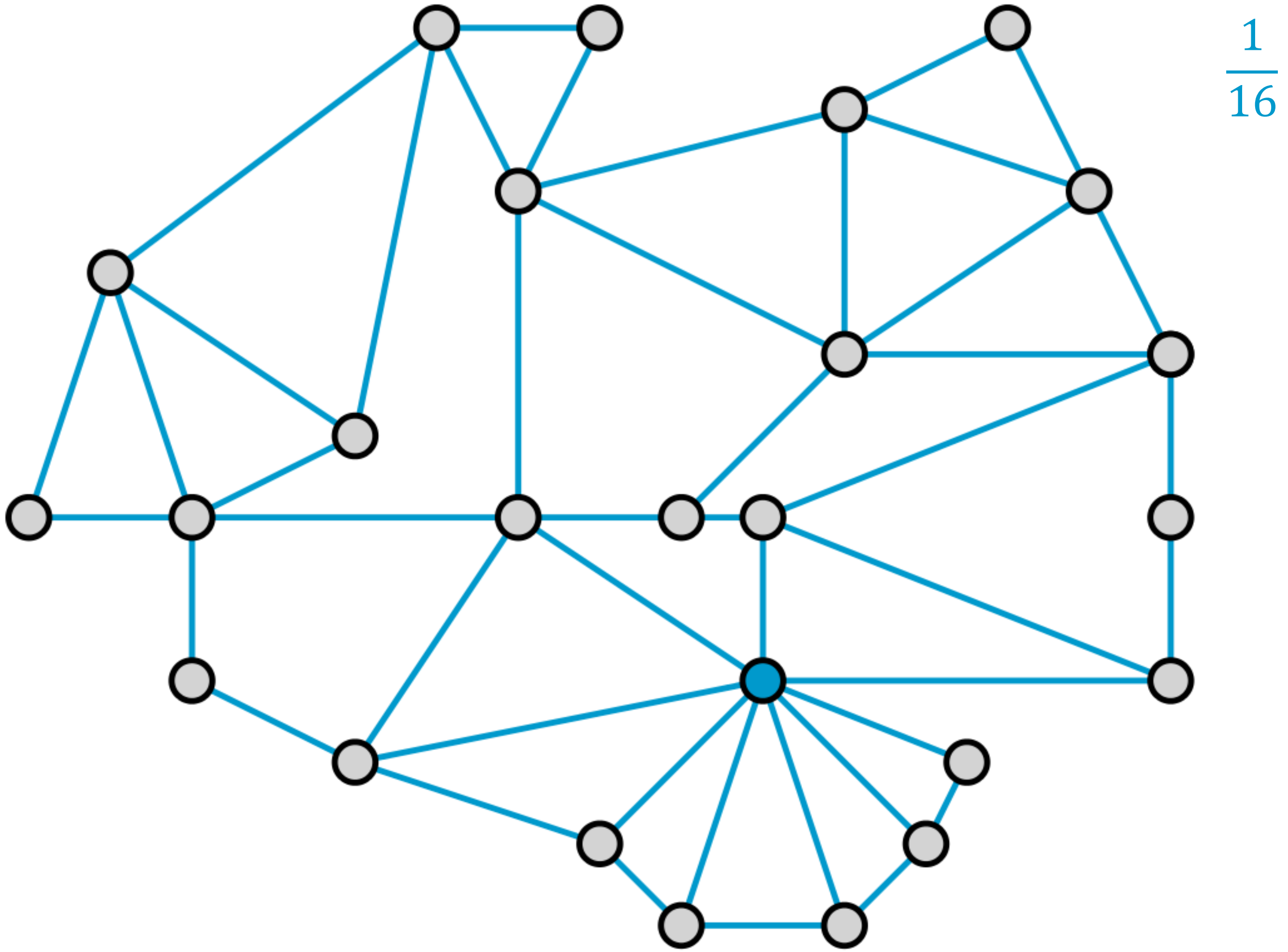
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

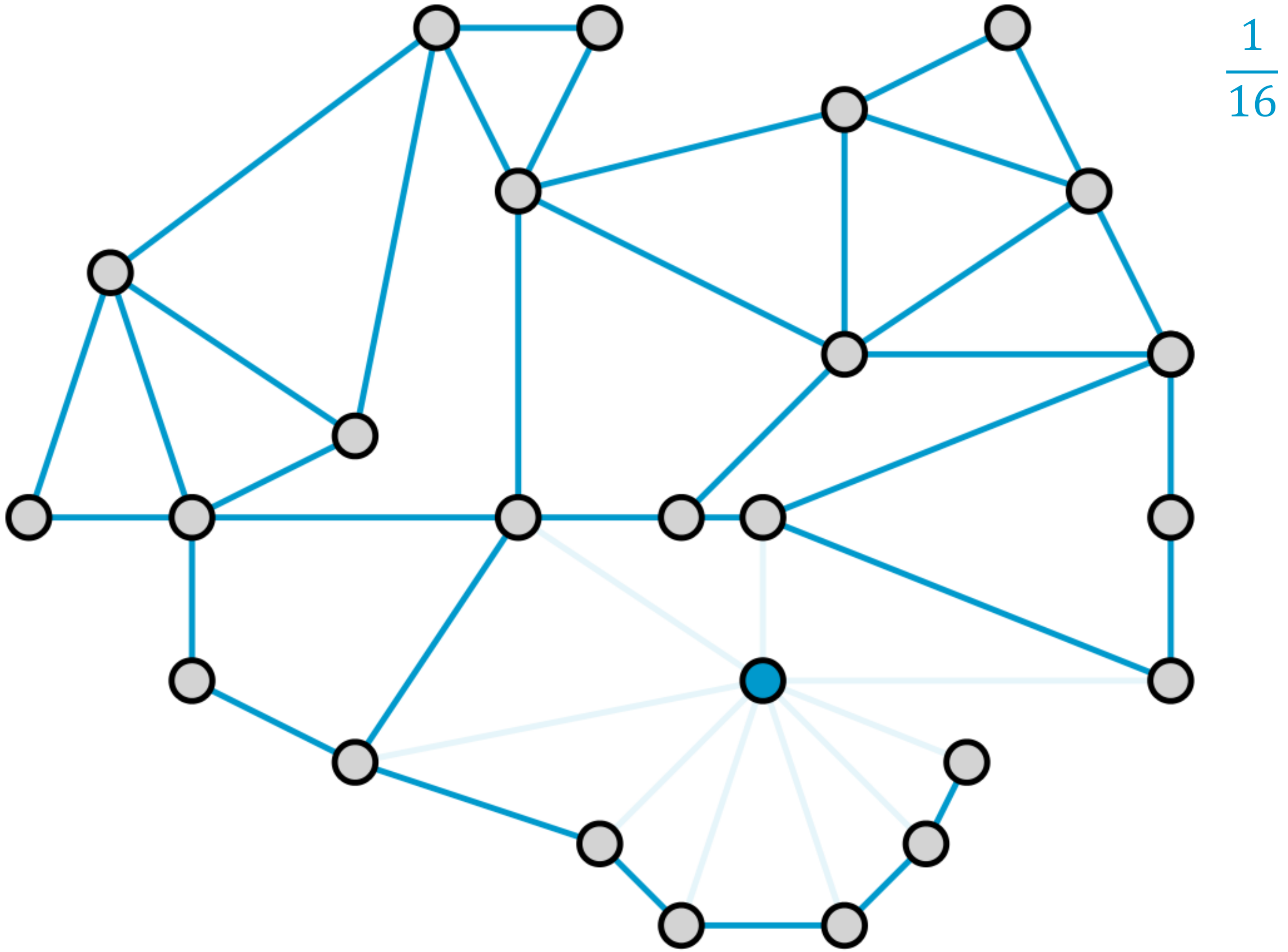
$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

repeat until all edges are blocked

- mark half-tight nodes
- block its edges
- double value of unblocked edges

I) 4-Approximate Fractional Matching

$O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \quad \sum_{e \in E(v)} x_e \leq 1 \qquad \text{for all } v \in V$$

$$x_e \in [0,1] \qquad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

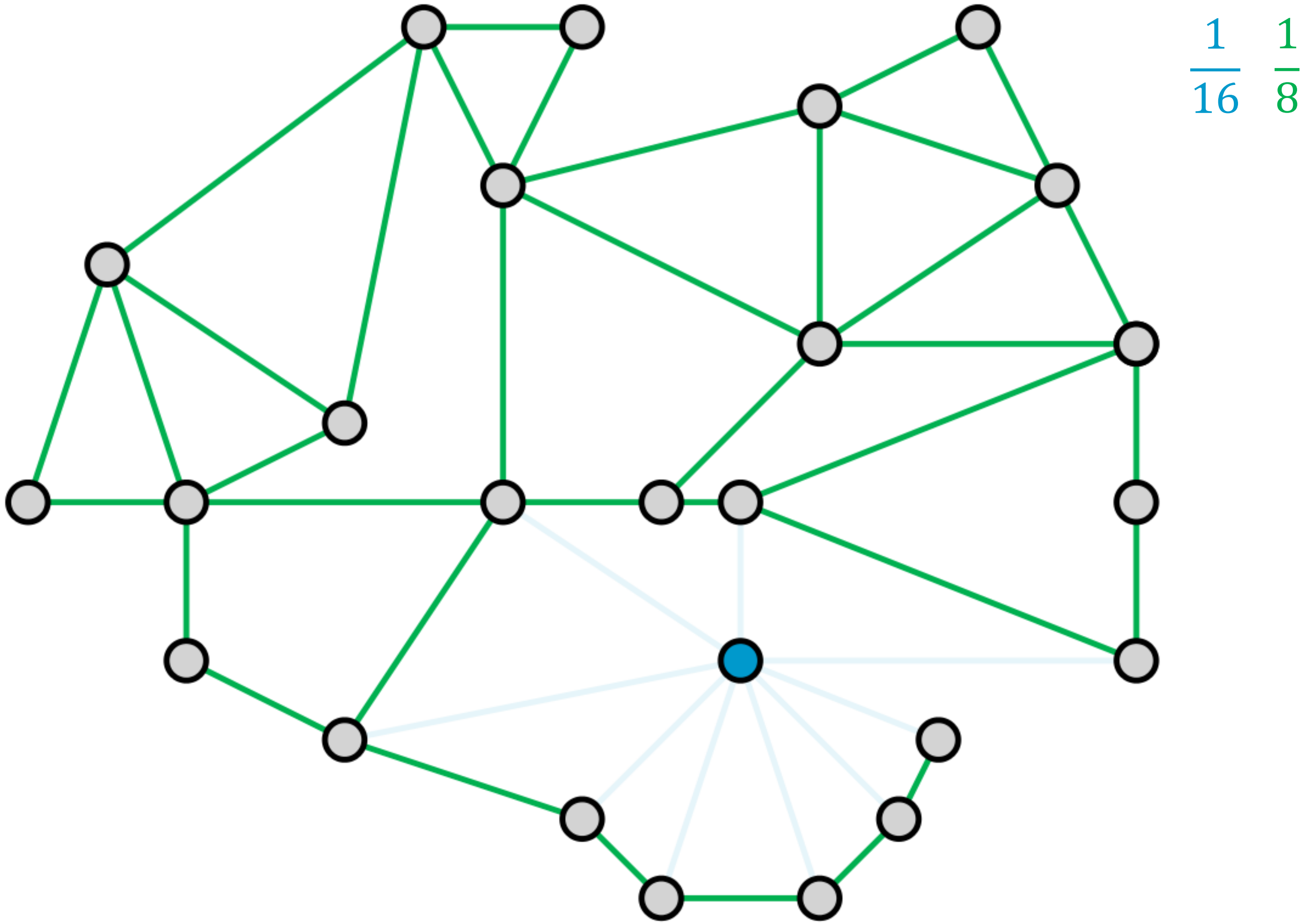
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

repeat until all edges are blocked

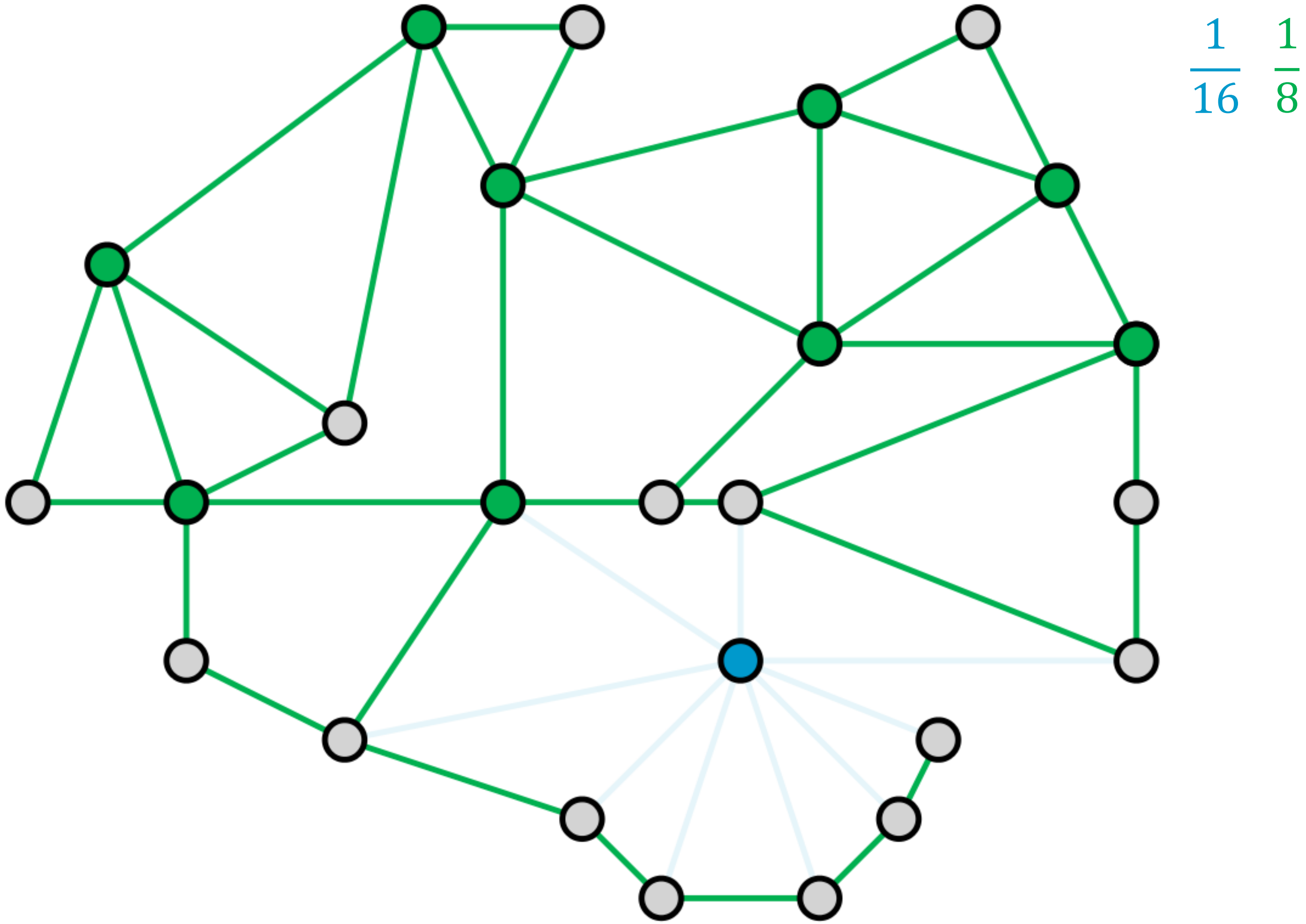
mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching

$O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \le 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\ge \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

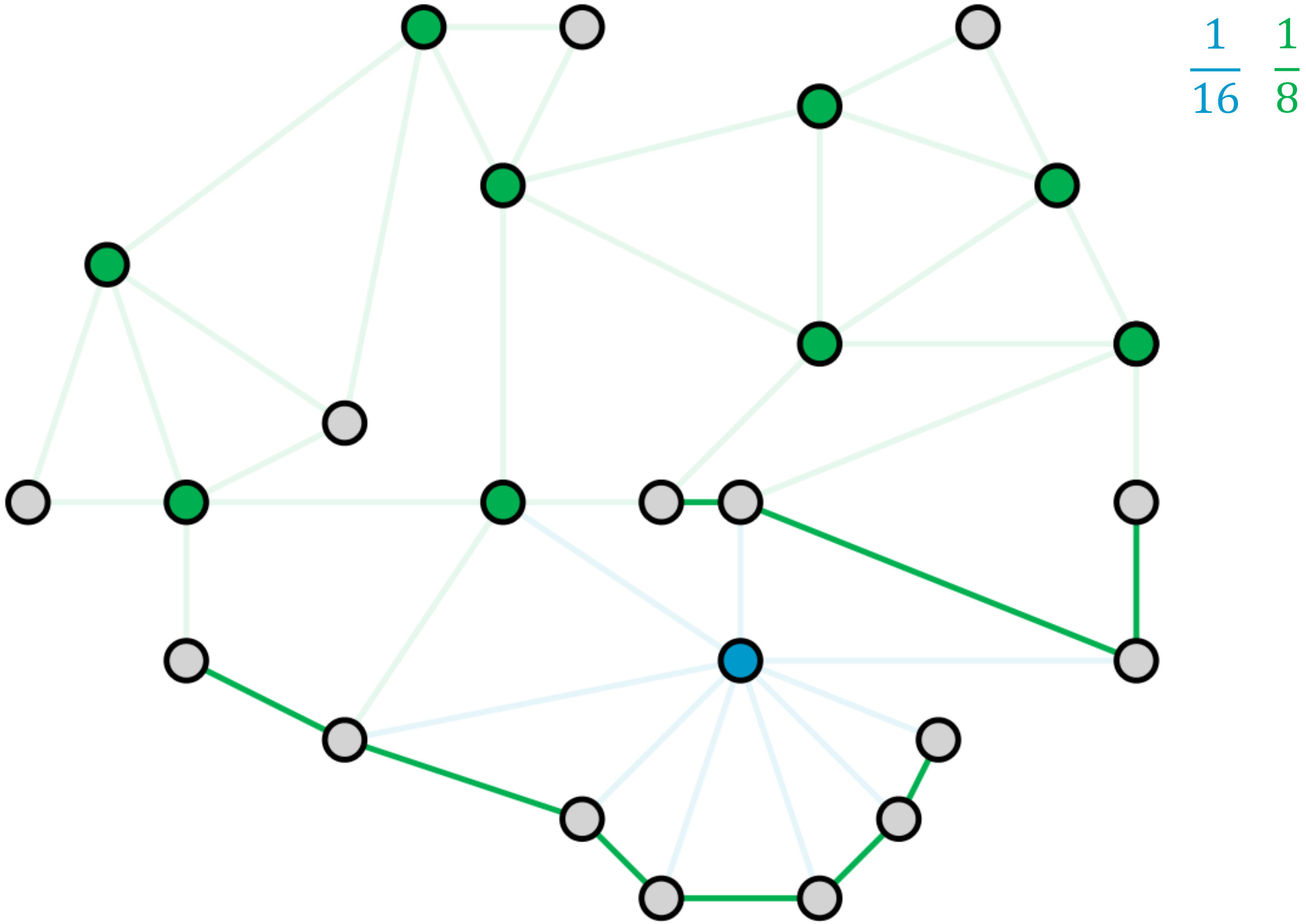
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log\Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e\in E} x_e$$

value of v

$$\text{s.t.} \sum_{e\in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

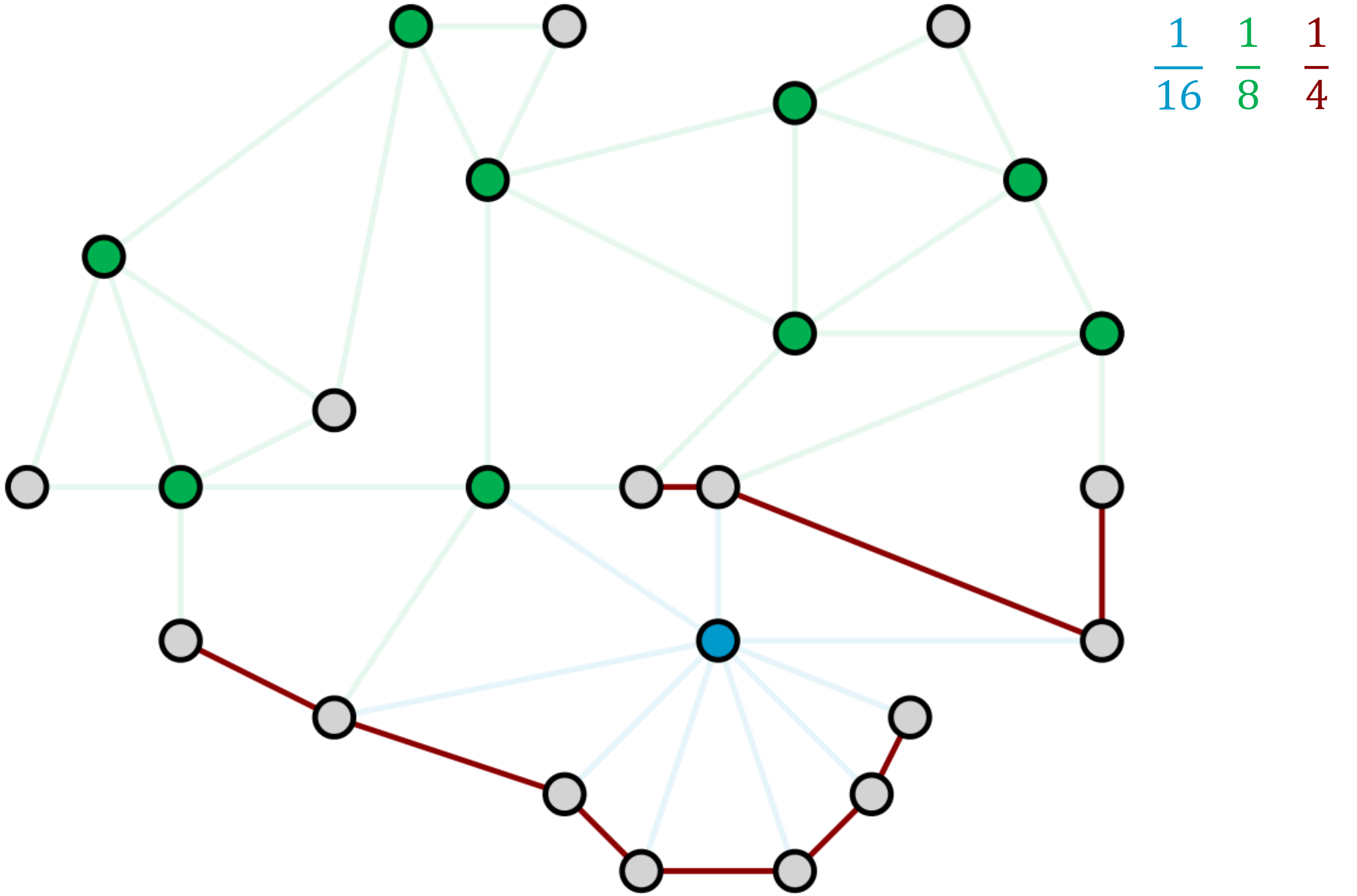
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

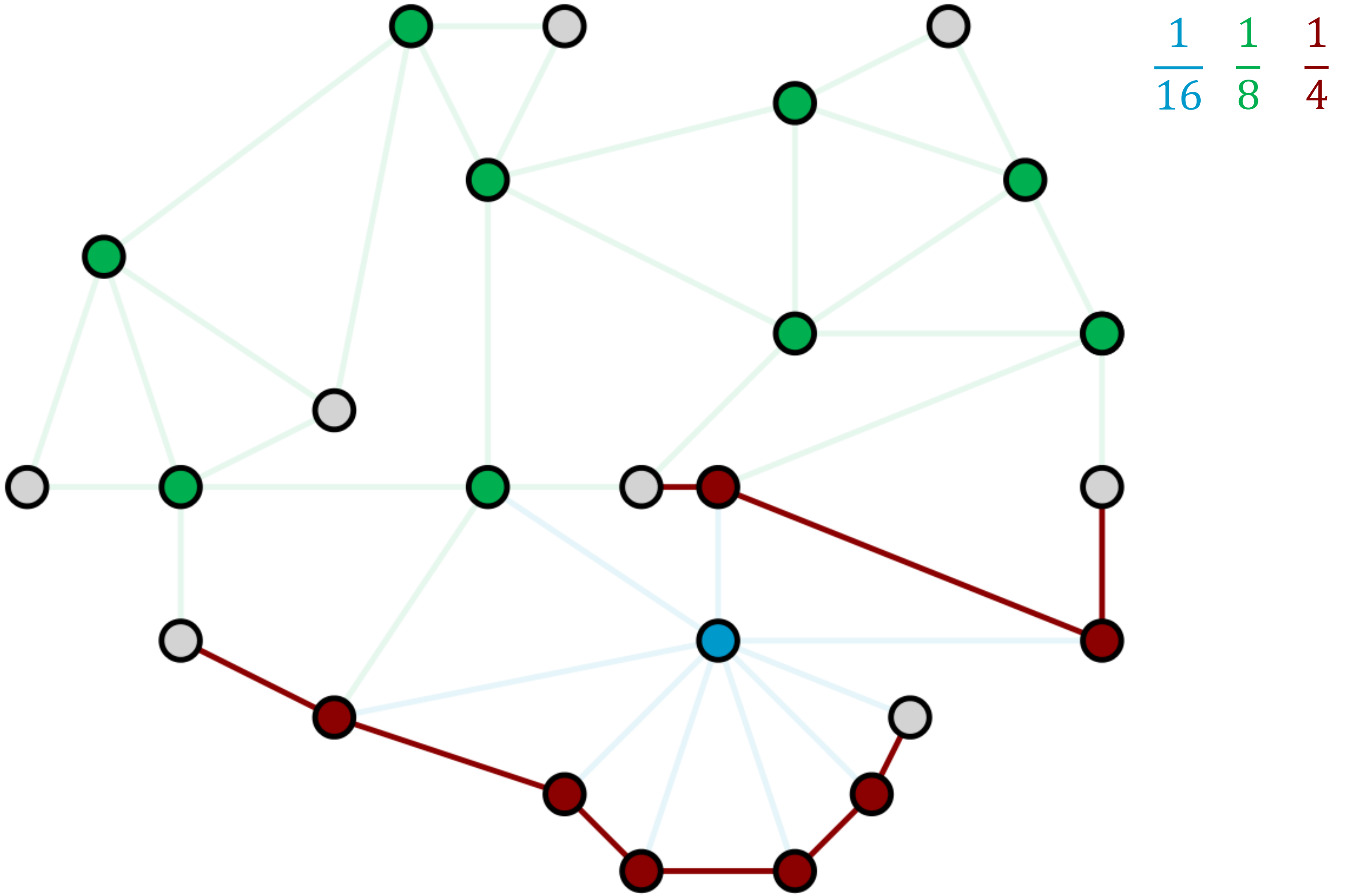
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching $O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

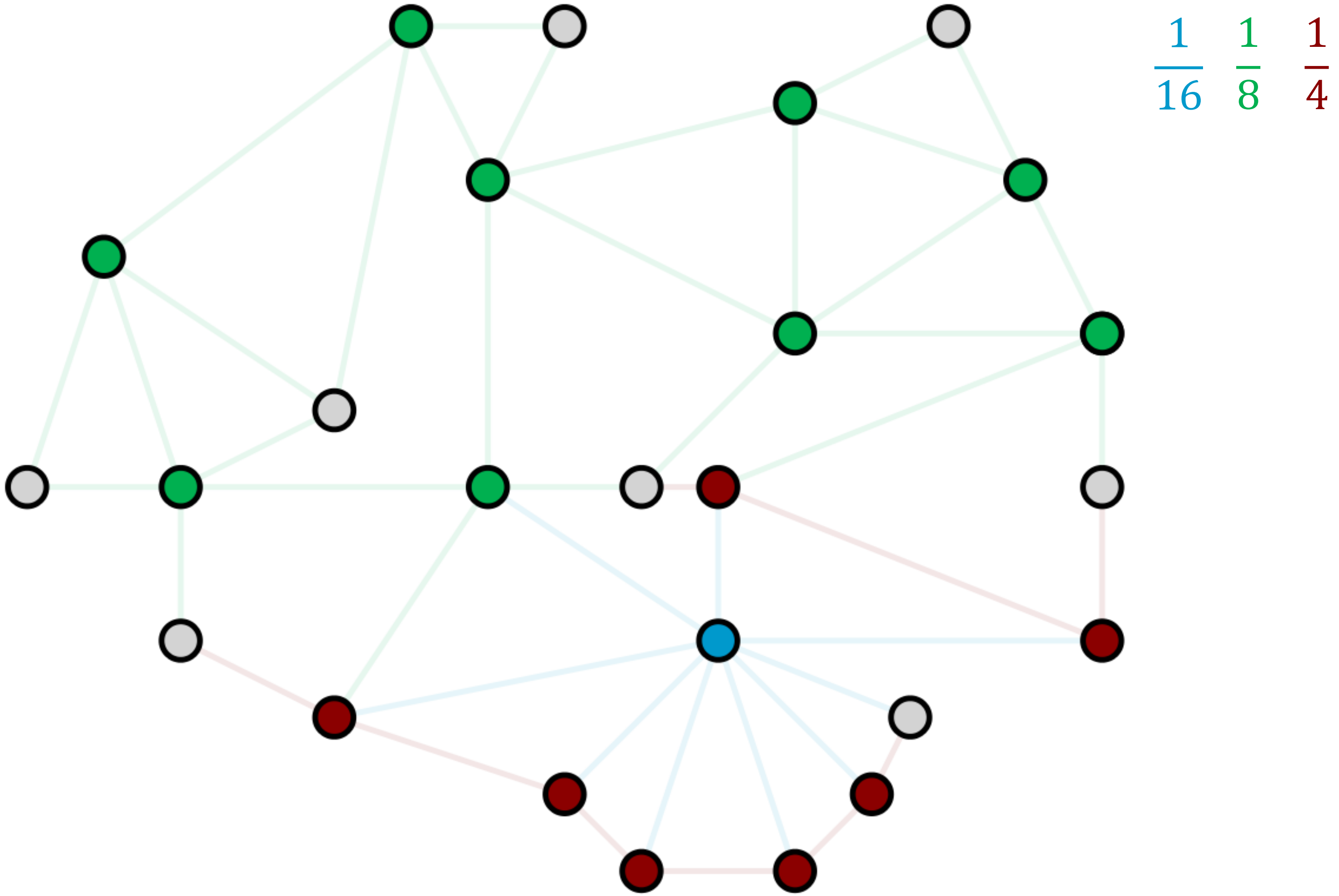
repeat until all edges are blocked

mark half-tight nodes

block its edges

double value of unblocked edges

I) 4-Approximate Fractional Matching — $O(\log \Delta)$ rounds



Fractional Maximum Matching

$$\max \sum_{e \in E} x_e$$

value of v

$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \quad \text{for all } v \in V$$

$$x_e \in [0,1] \quad \text{for all } e \in E$$

v is half-tight if its value is $\geq \frac{1}{2}$

LOCAL Greedy Algorithm

$x_e = 2^{-\lceil \log \Delta \rceil}$ for all $e \in E$

repeat until all edges are blocked

- mark half-tight nodes
- block its edges
- double value of unblocked edges

Constant - Approximate Bipartite Matching

$O(\log^2 \Delta)$ rounds

I) 4-Approximate Fractional Matching

$O(\log \Delta)$ rounds

II) Rounding Fractional Bipartite Matching

$\mathrm{O}(\log^2 \Delta)$ rounds, $\mathrm{O}(1)$ loss

II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

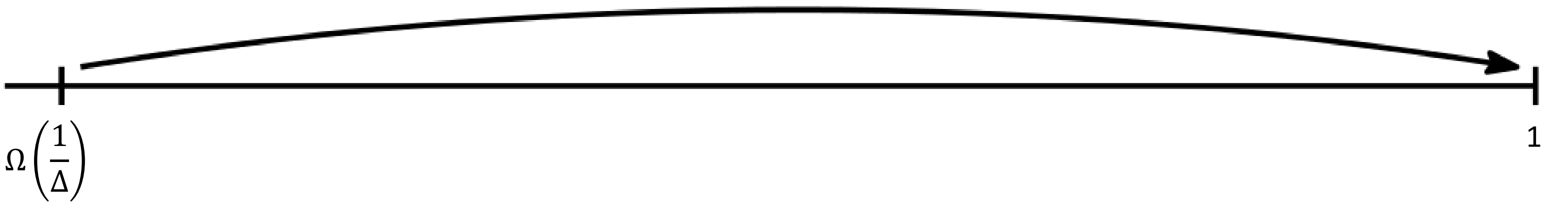
II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

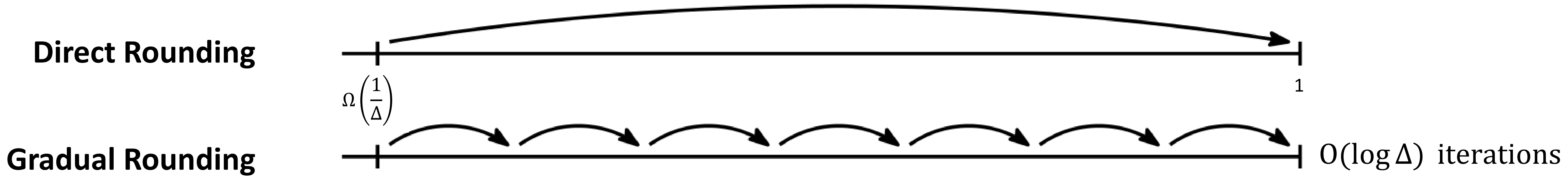
II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Direct Rounding



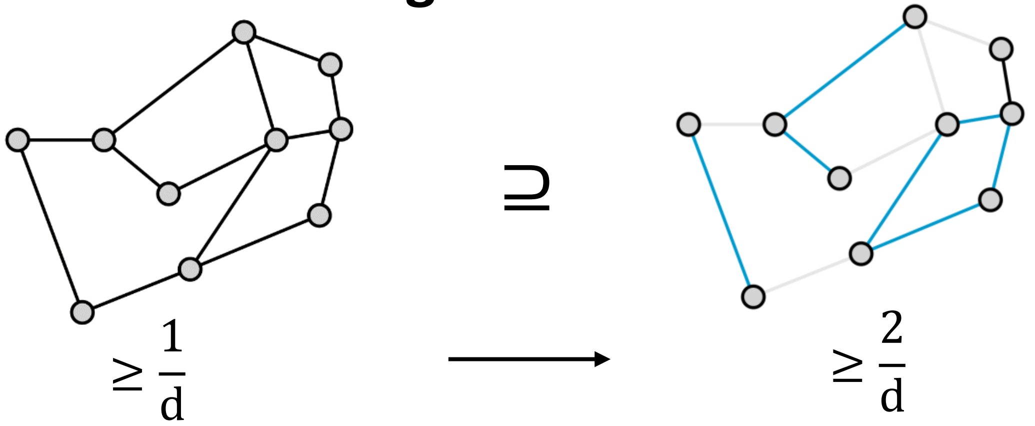
II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

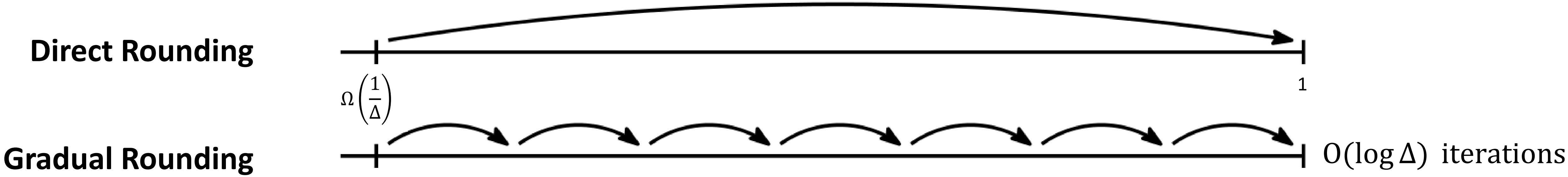


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

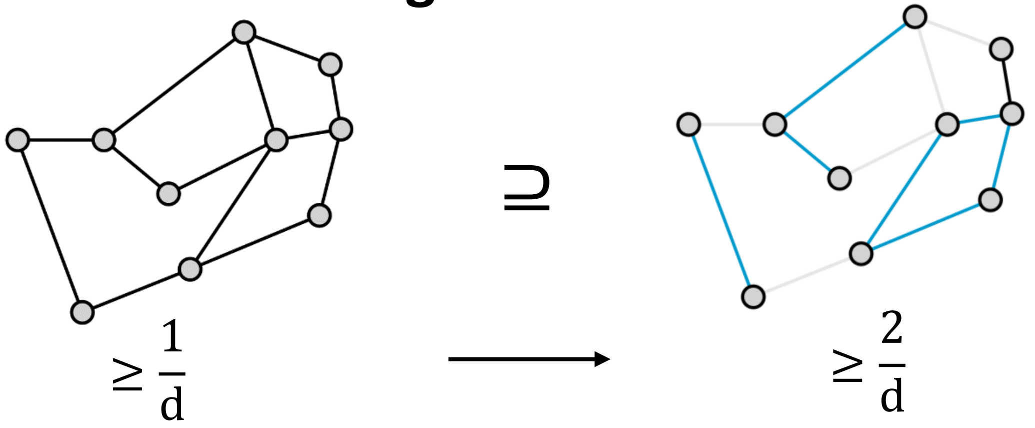
Factor-2-Rounding





II) Rounding Fractional Bipartite Matching — $O(\log^2 \Delta)$ rounds, $O(1)$ loss

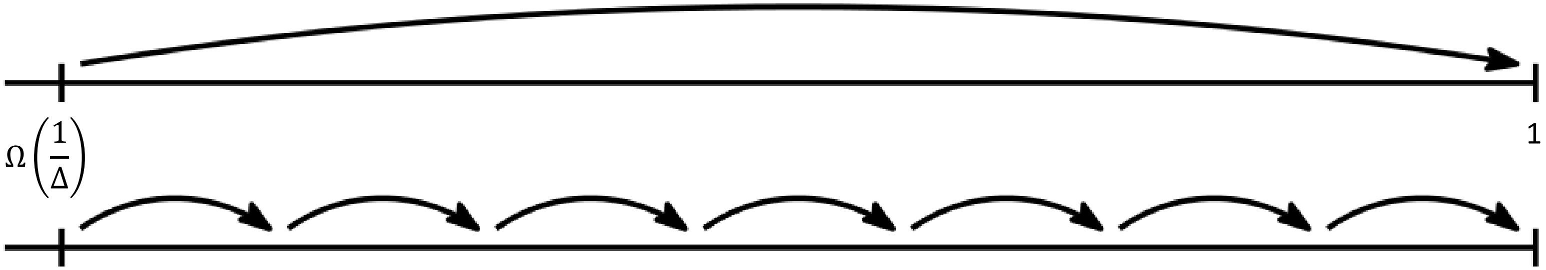
Factor-2-Rounding



using Locally Balanced Splitting,
inspired by
Hańćkowiak, Karoński, Panconesi [SODA'98,PODC'99]

Direct Rounding

Gradual Rounding



$O(\log \Delta)$ iterations

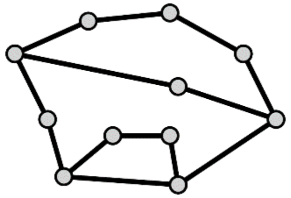
II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

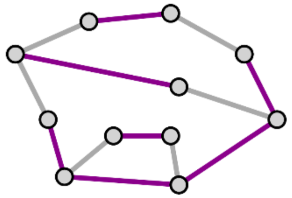
<u>Locally Balanced Splitting:</u>
2-edge-coloring so that
every node roughly balanced



II) Rounding Fractional Bipartite Matching — $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced

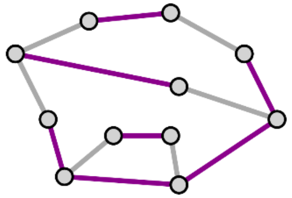


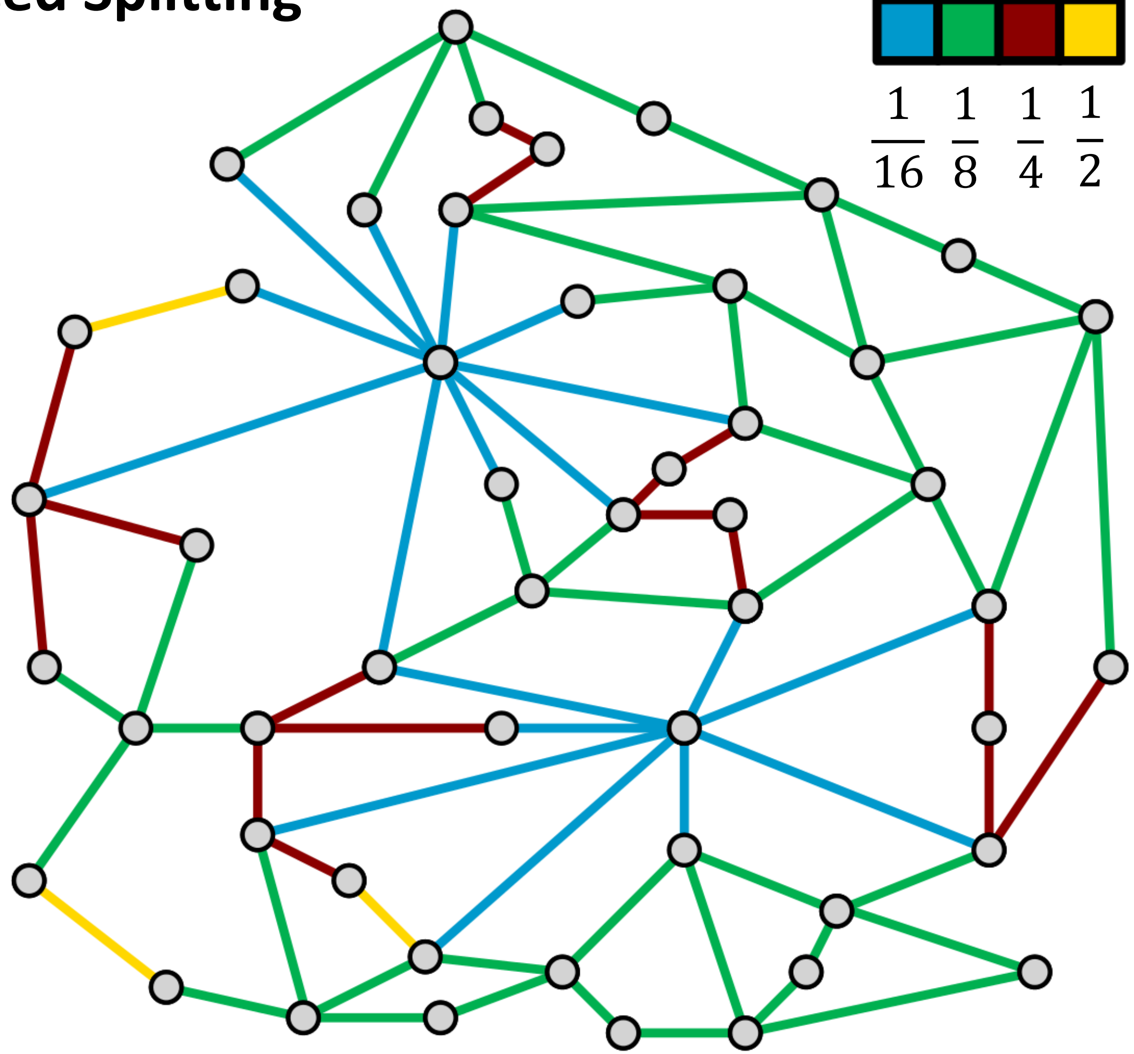
II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



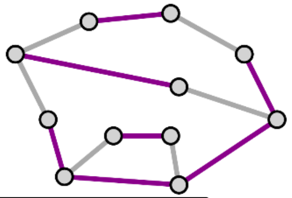


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

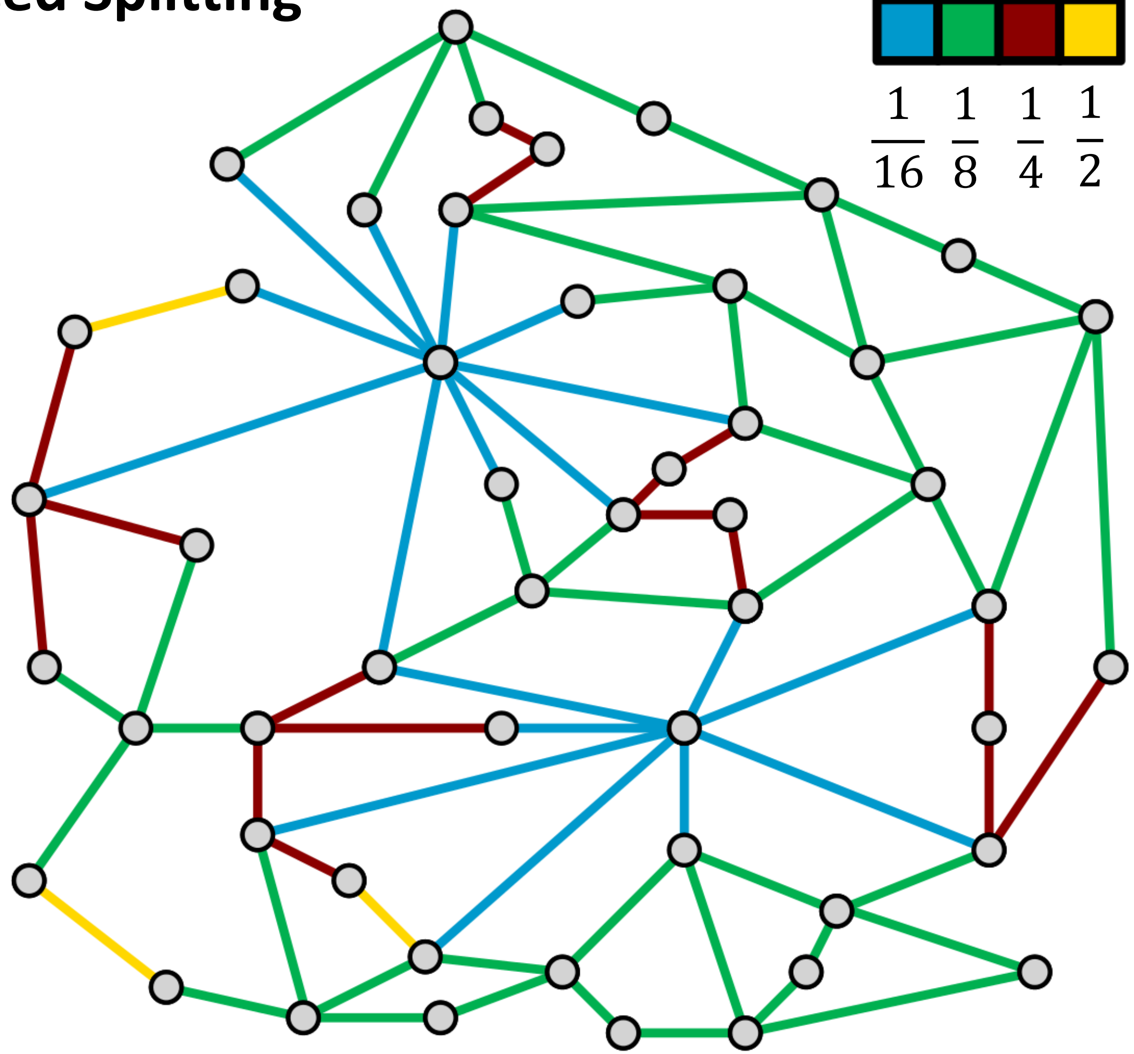
for $i = \lceil \log \Delta \rceil, \ldots, 1$

$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

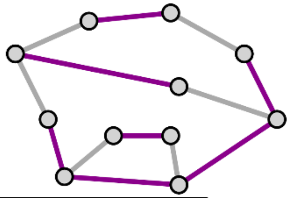
decrease ■ to 0



II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that
every node roughly balanced



Iterated Factor-2-Rounding

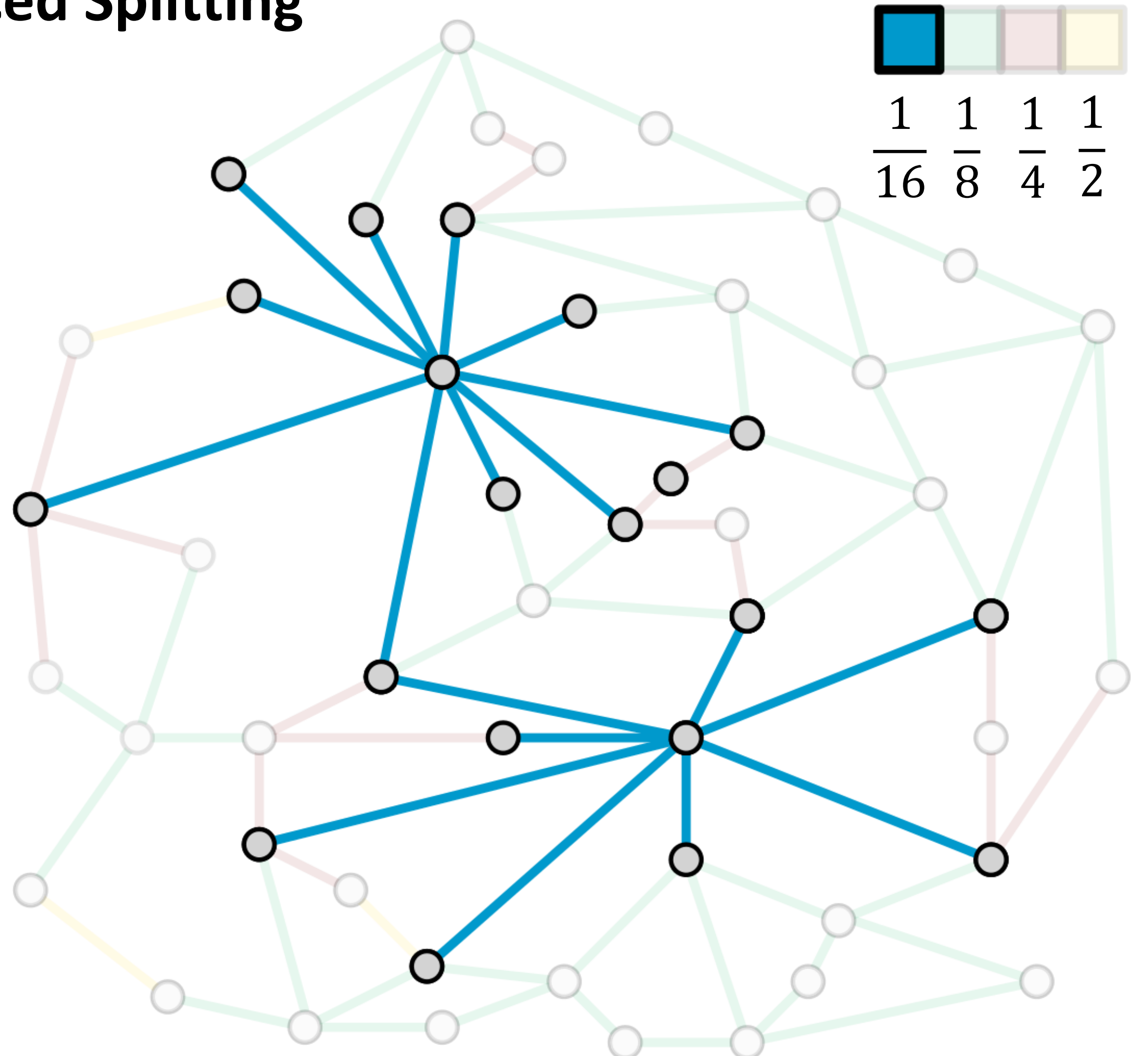
for $i = \lceil \log \Delta \rceil, \ldots, 1$

$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

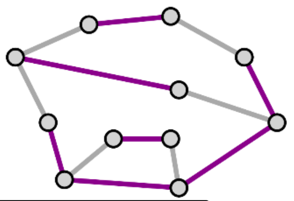


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that
every node roughly balanced



Iterated Factor-2-Rounding

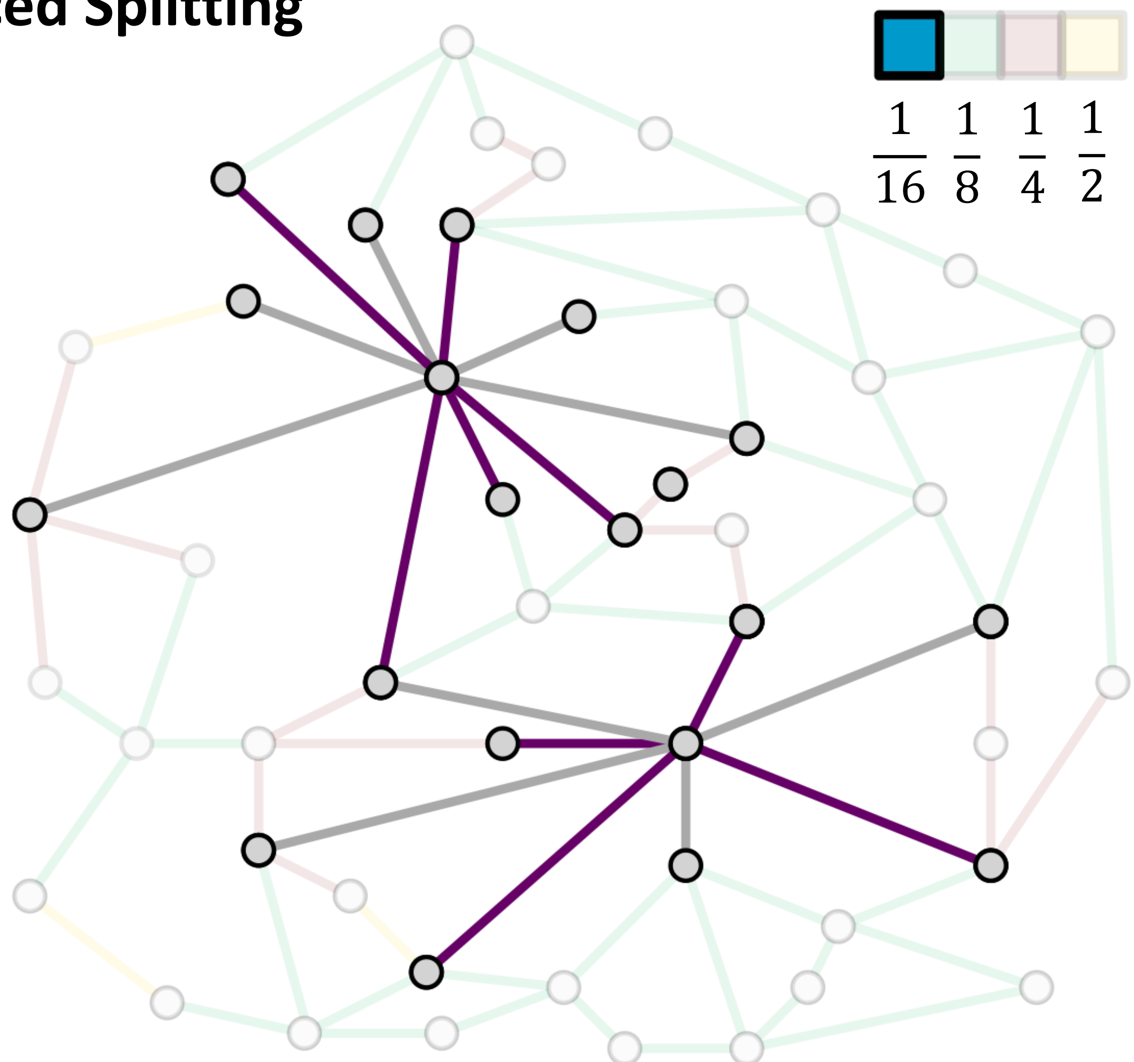
for i = $\lceil \log \Delta \rceil, \ldots, 1$

$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

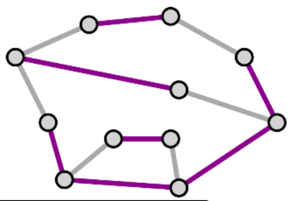


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for i $= \lceil \log \Delta \rceil, \ldots, 1$

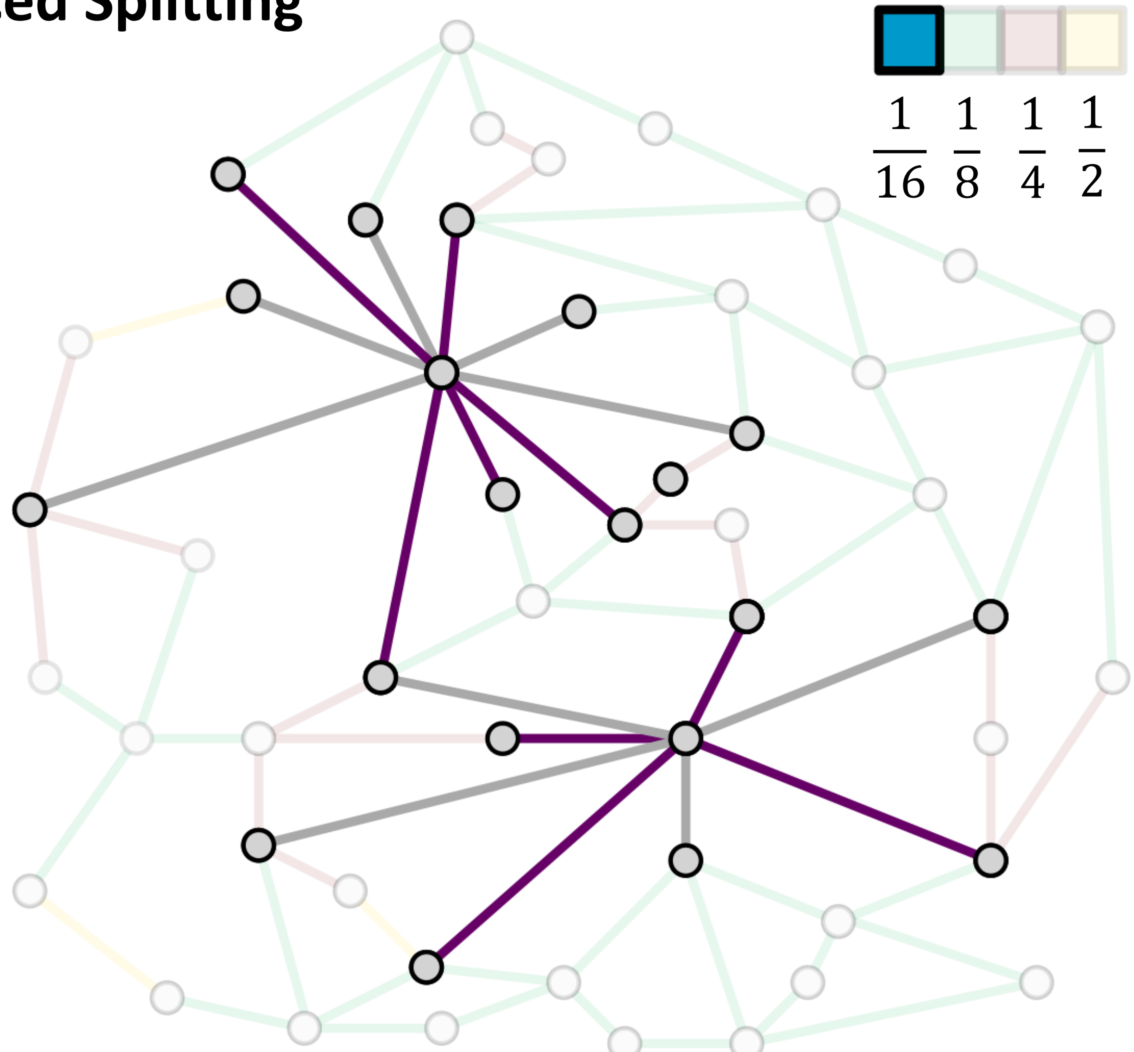
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

In case of **perfect** locally balanced **splitting**:
no constraint violated & no loss in total value
(i.e., **perfect rouding**)

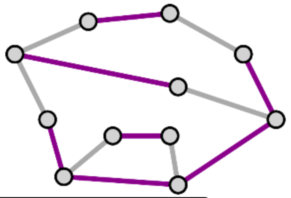


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that
every node roughly balanced



Iterated Factor-2-Rounding

for i = $\lceil \log \Delta \rceil, \ldots, 1$

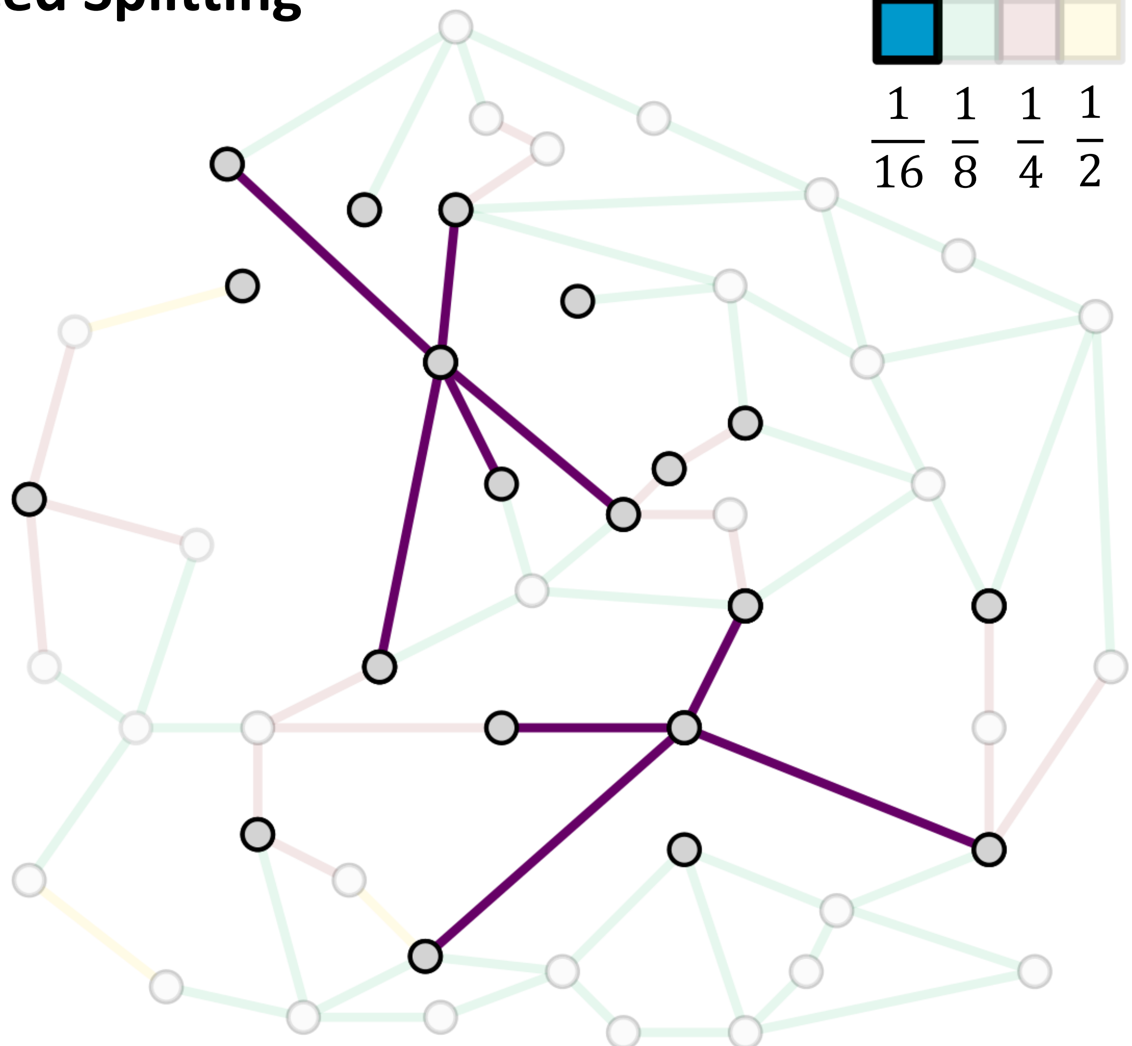
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

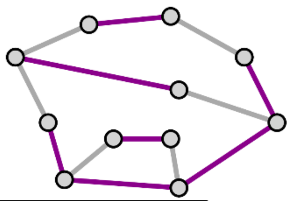
In case of **perfect** locally balanced **splitting**:
no constraint violated & no loss in total value
(i.e., **perfect rouding**)



II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

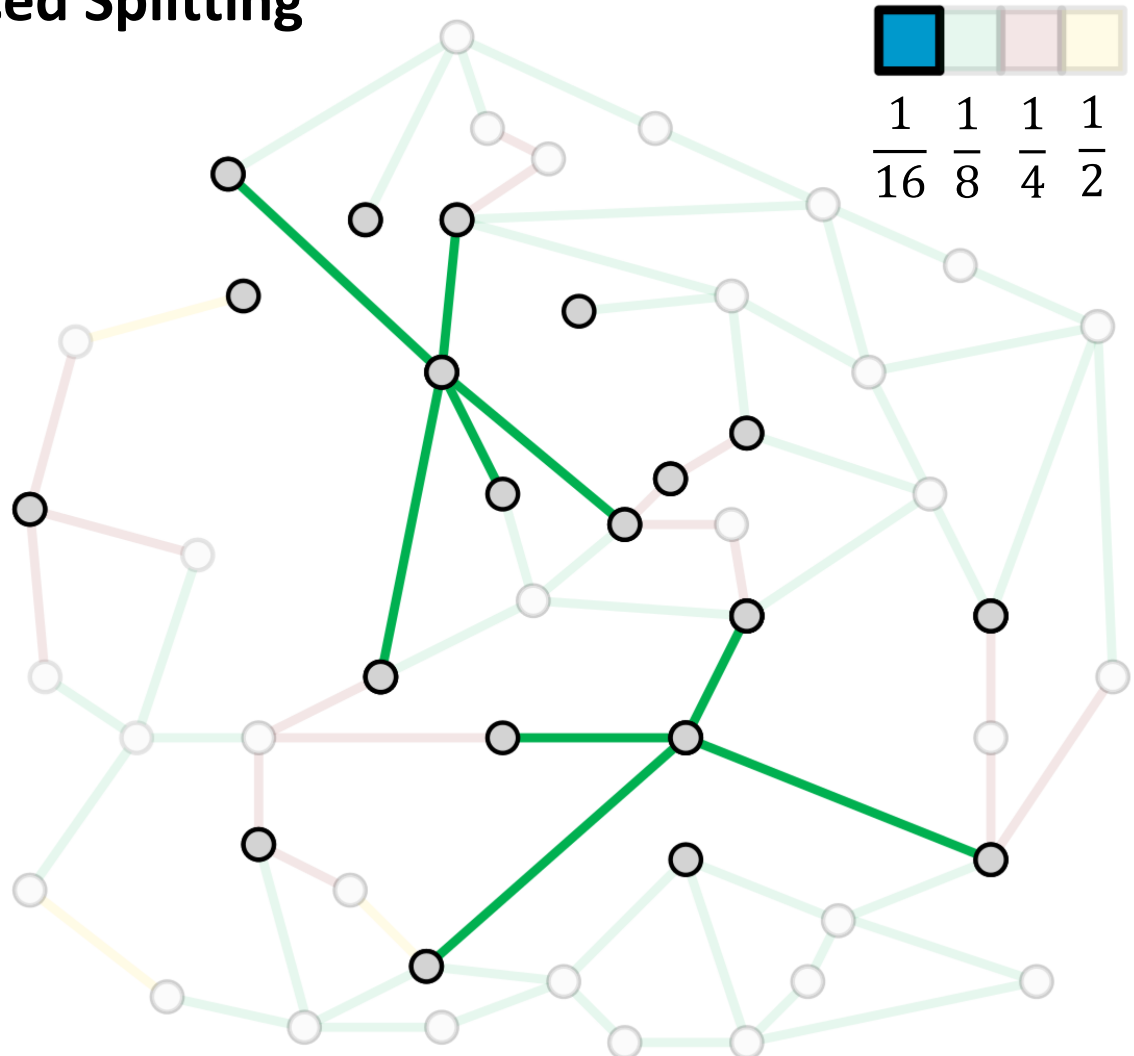
for i $= \lceil \log \Delta \rceil, \ldots, 1$

$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

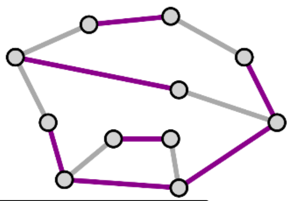


In case of **perfect** locally balanced **splitting**: no constraint violated & no loss in total value (i.e., **perfect rouding**)

II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for i = $\lceil \log \Delta \rceil, \ldots, 1$

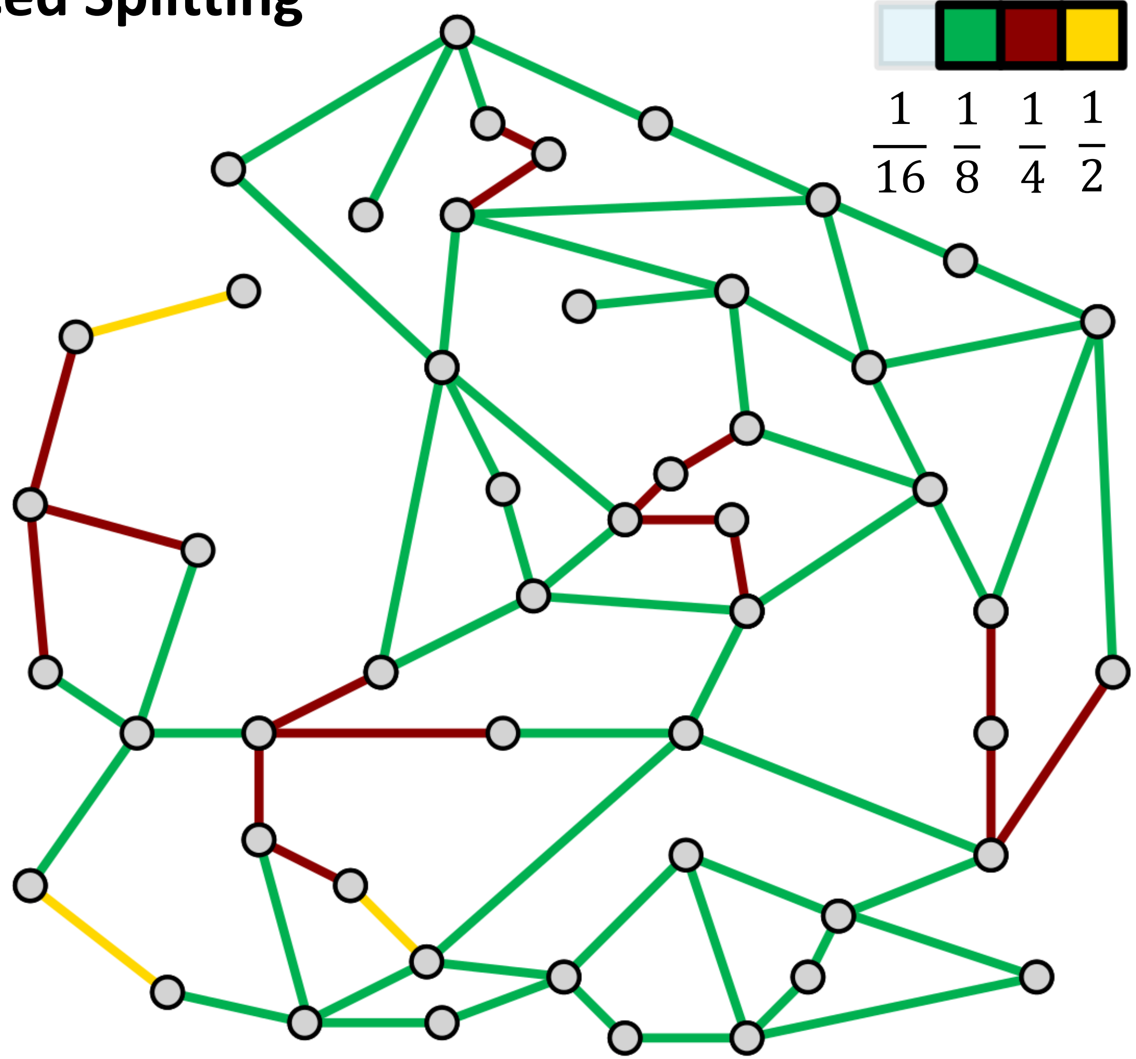
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■ ■

increase ■ to 2^{-i+1}

decrease ■ to 0

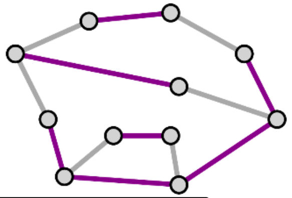
In case of **perfect** locally balanced **splitting**: no constraint violated & no loss in total value (i.e., **perfect rouding**)



II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for i = $\lceil \log \Delta \rceil, \ldots, 1$

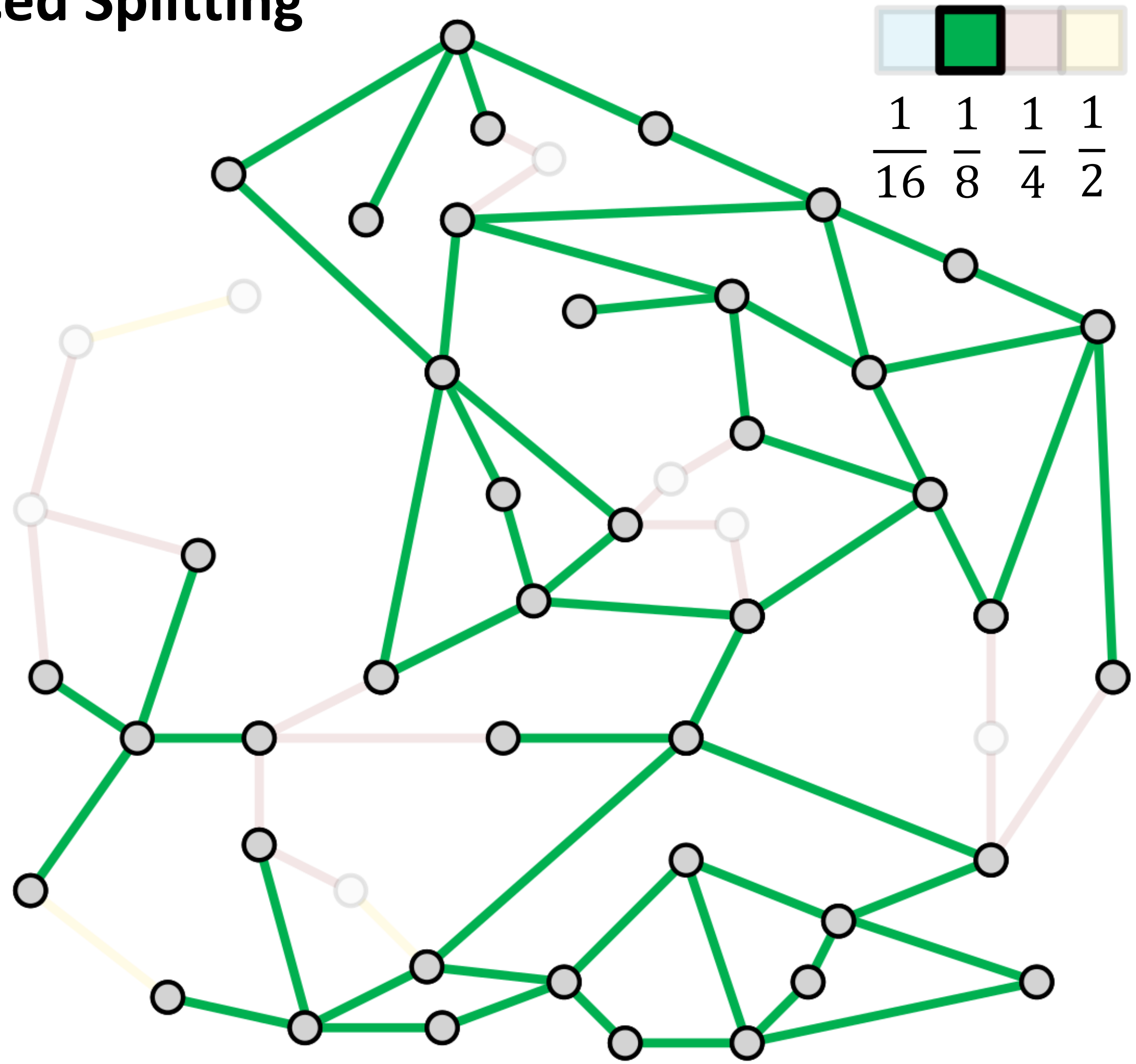
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

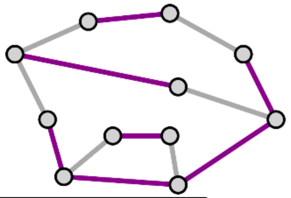
In case of **perfect** locally balanced **splitting**: no constraint violated & no loss in total value (i.e., **perfect rouding**)



II) Rounding Fractional Bipartite Matching — $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for i = $\lceil \log \Delta \rceil, \ldots, 1$

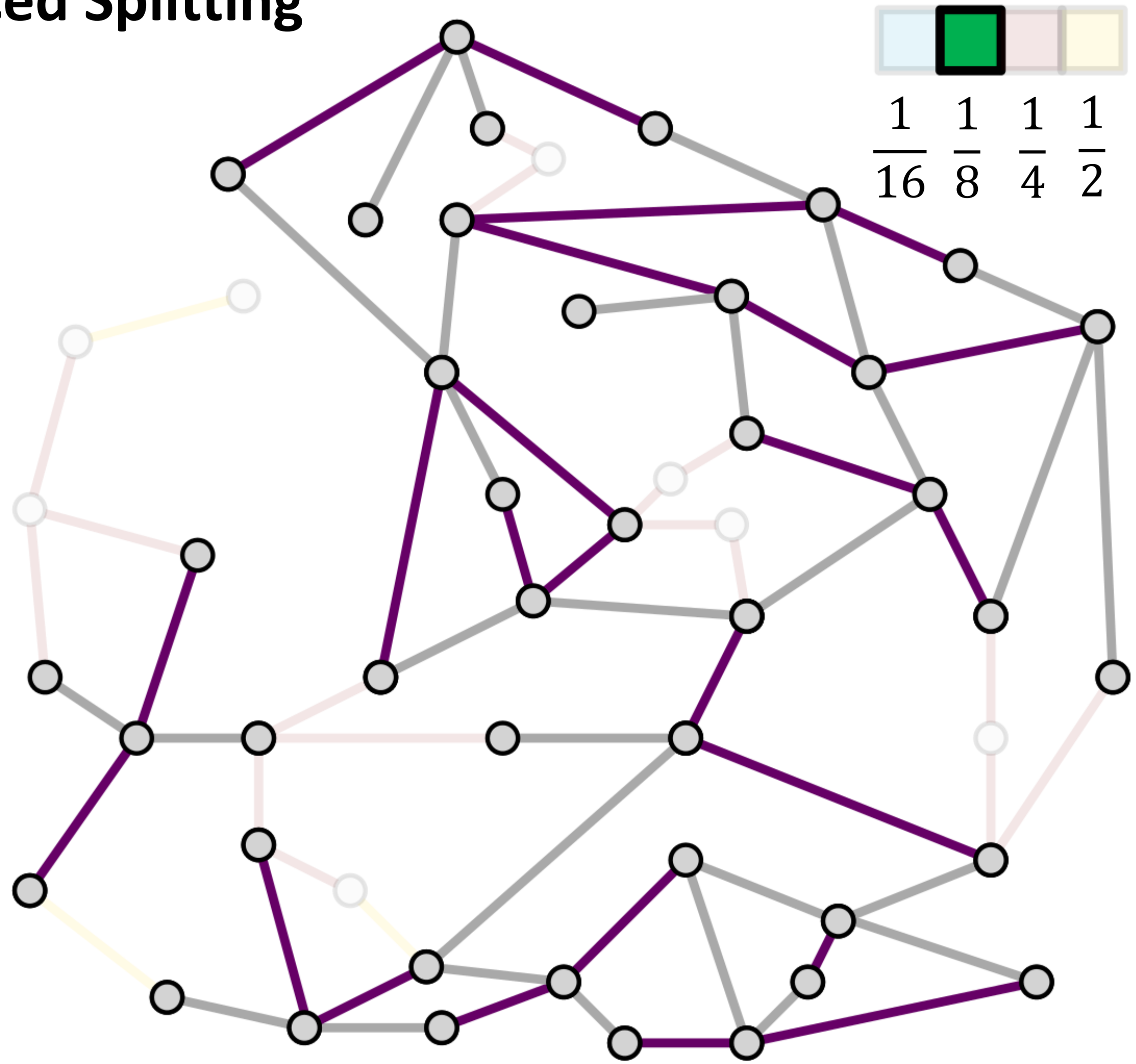
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

In case of **perfect** locally balanced **splitting**: no constraint violated & no loss in total value (i.e., **perfect rouding**)

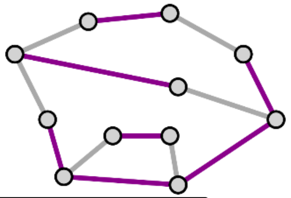


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for i $= \lceil \log \Delta \rceil, \ldots, 1$

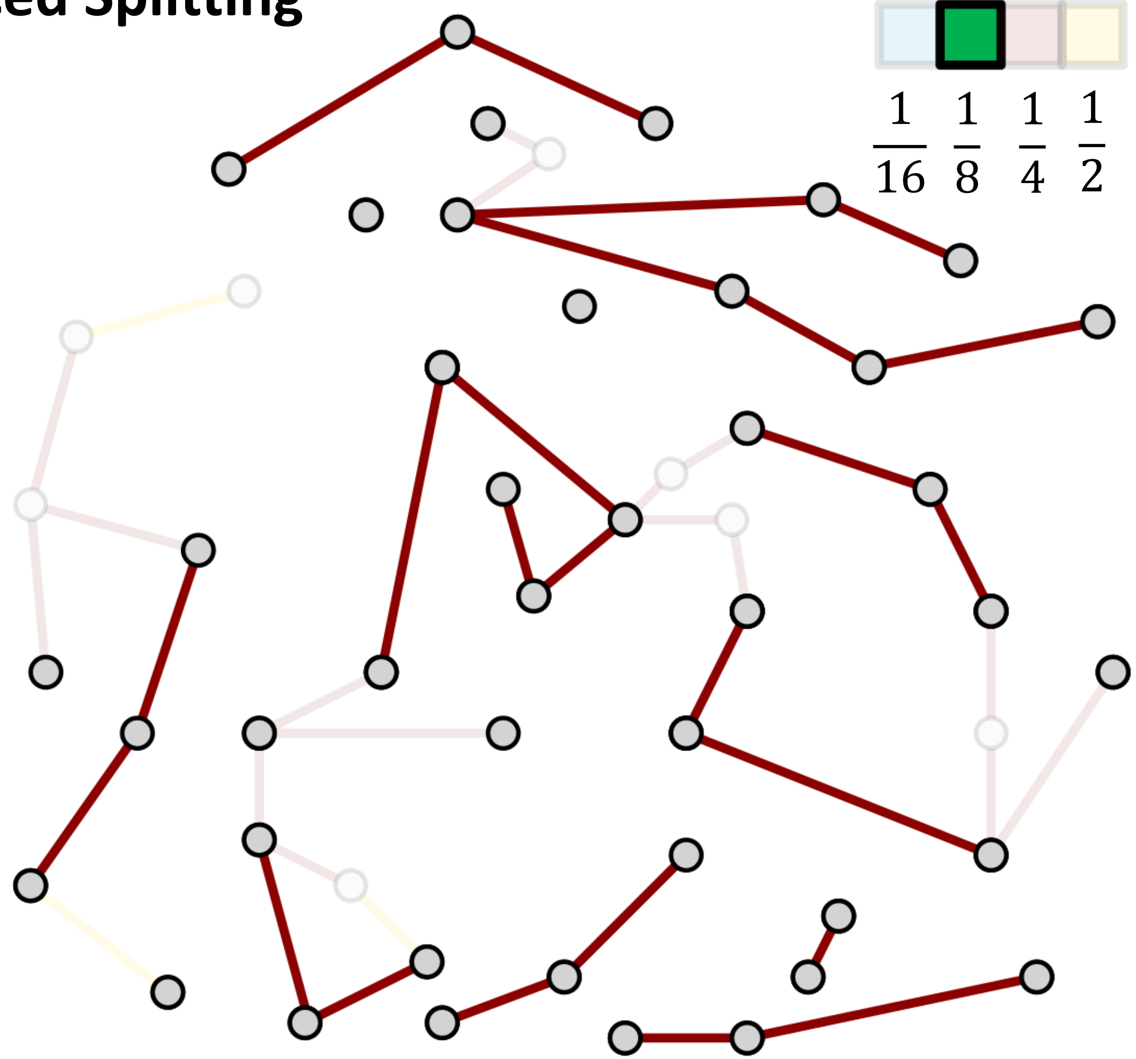
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■ □

increase ■ to 2^{-i+1}

decrease □ to 0

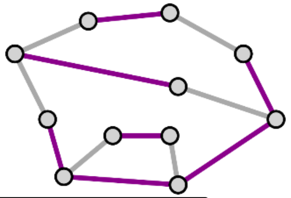
In case of **perfect** locally balanced **splitting**:
no constraint violated & no loss in total value
(i.e., **perfect rouding**)



II) Rounding Fractional Bipartite Matching — $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for $i = \lceil \log \Delta \rceil, \ldots, 1$

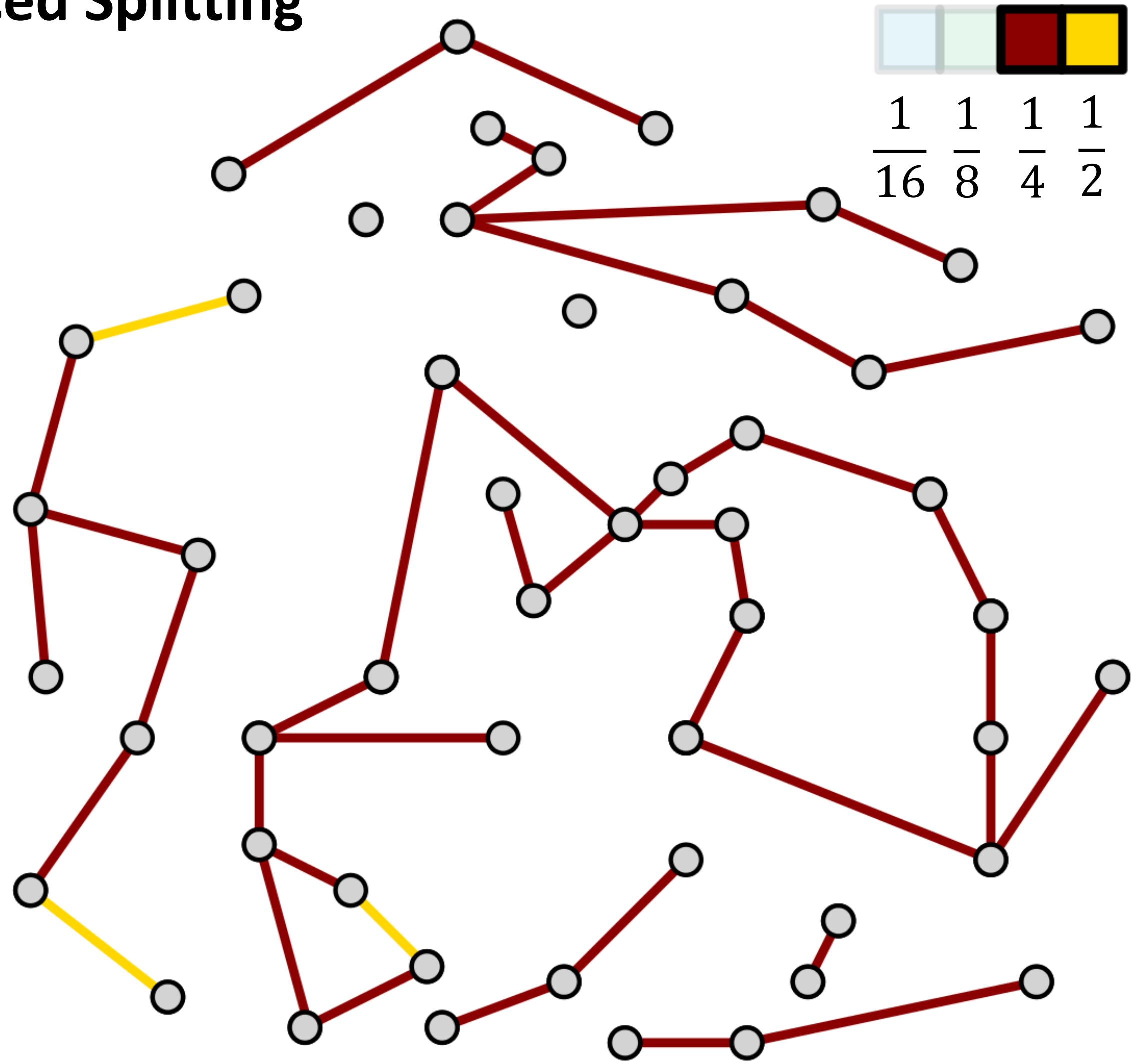
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

In case of **perfect** locally balanced **splitting**:
no constraint violated & no loss in total value
(i.e., **perfect rouding**)

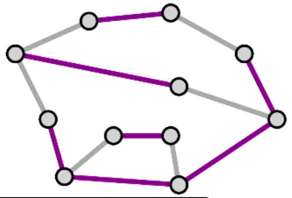


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that
every node roughly balanced



Iterated Factor-2-Rounding

for i = $\lceil \log \Delta \rceil, \ldots, 1$

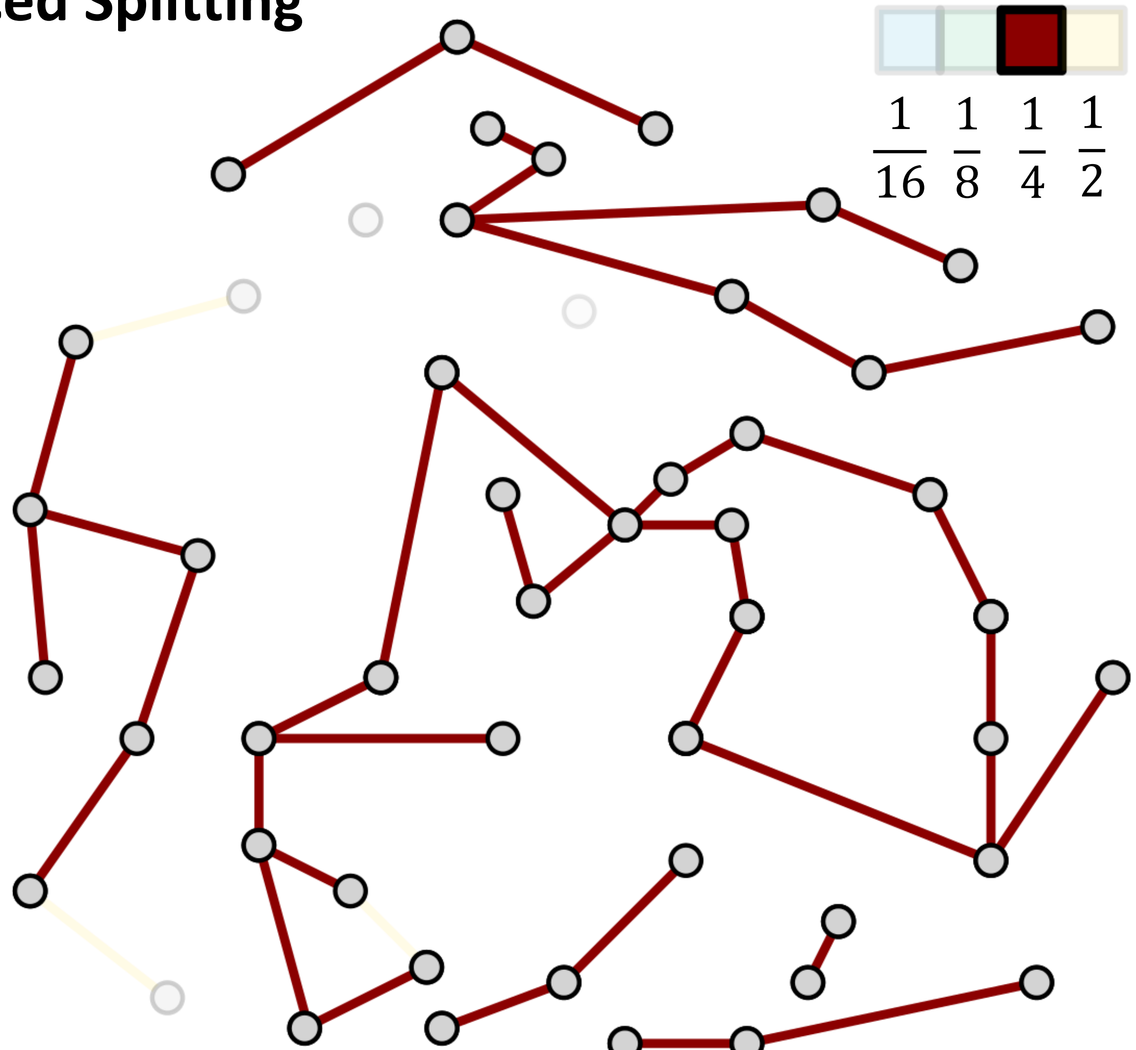
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0

In case of **perfect** locally balanced **splitting**:
no constraint violated & no loss in total value
(i.e., **perfect rouding**)

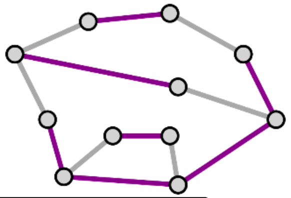


II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

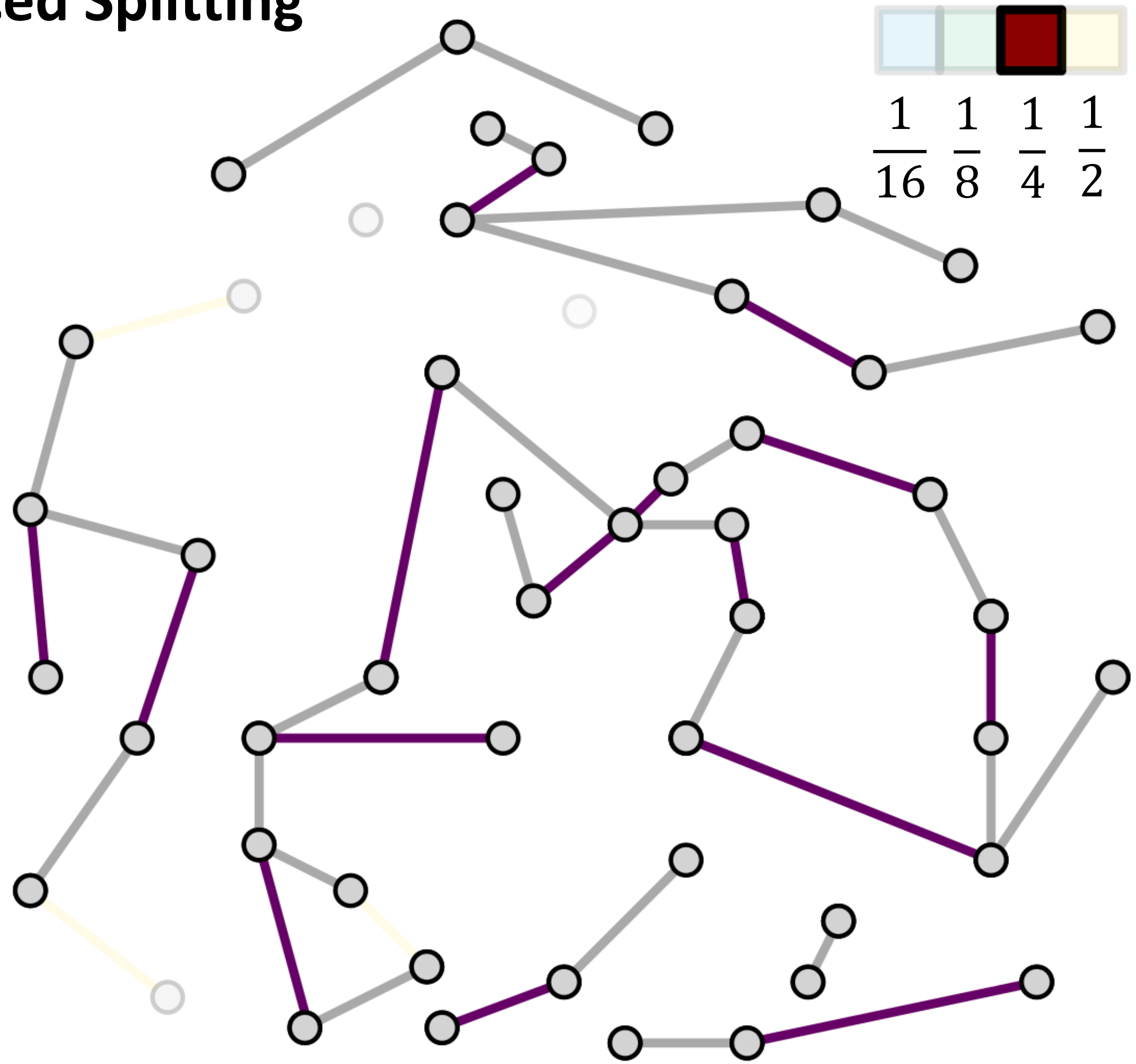
for i = $\lceil \log \Delta \rceil, \ldots, 1$

$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■■

increase ■ to 2^{-i+1}

decrease ■ to 0



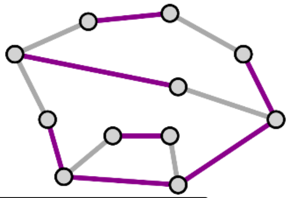
In case of **perfect** locally balanced **splitting**: no constraint violated & no loss in total value (i.e., **perfect rouding**)

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

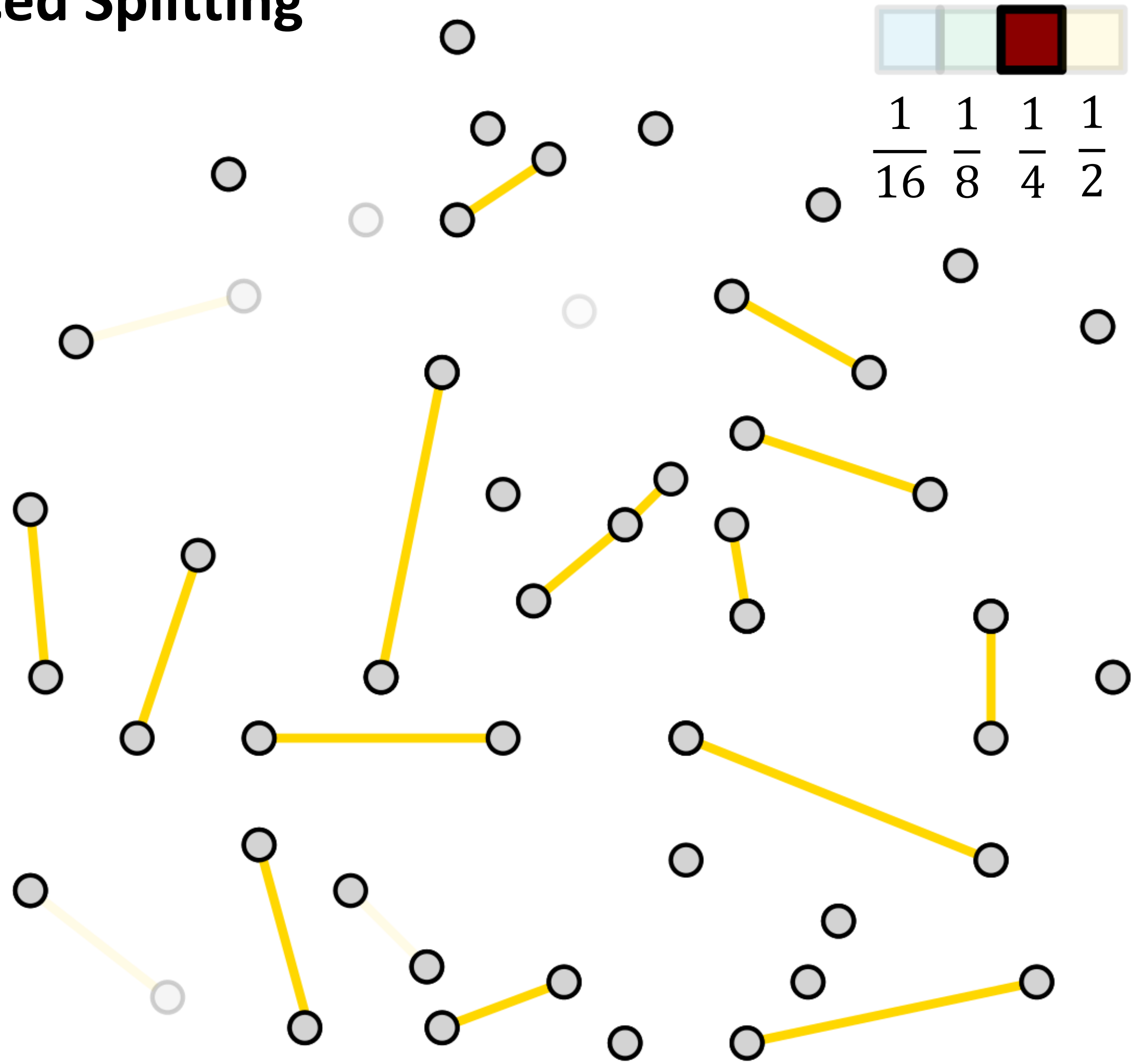
for $i = \lceil \log \Delta \rceil, \ldots, 1$

$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■ ■

increase ■ to 2^{-i+1}

decrease ■ to 0



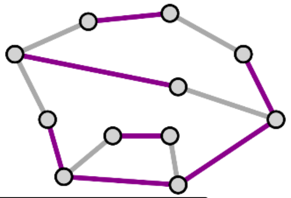
In case of **perfect** locally balanced **splitting**: no constraint violated & no loss in total value (i.e., **perfect rouding**)

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for i = $\lceil \log \Delta \rceil, \ldots, 1$

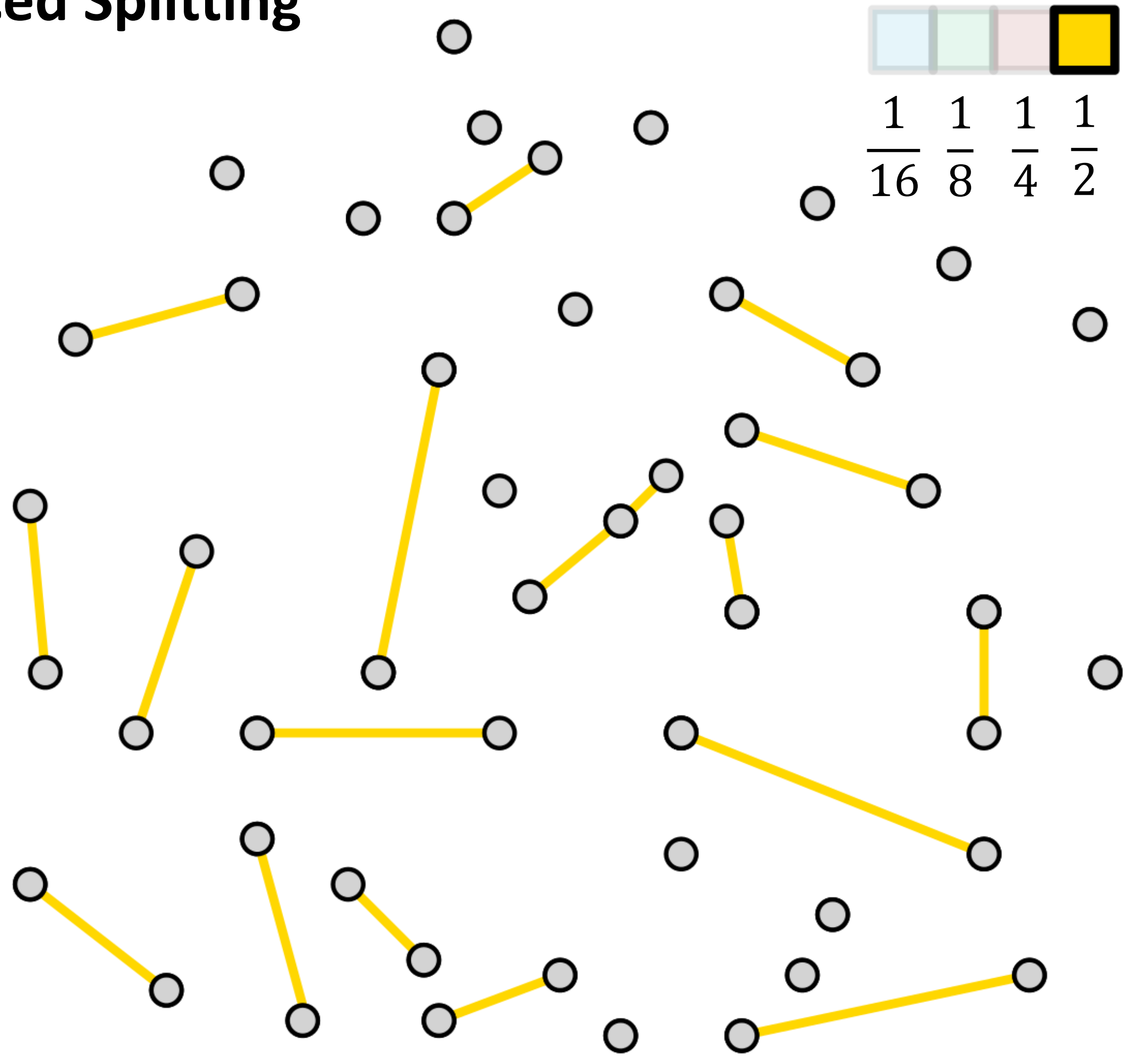
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into

increase to 2^{-i+1}

decrease to 0

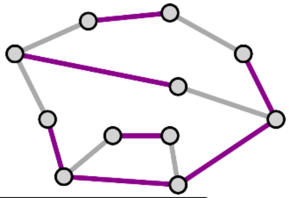
In case of **perfect** locally balanced **splitting**:
no constraint violated & no loss in total value
(i.e., **perfect rouding**)



II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Iterated Factor-2-Rounding using Locally Balanced Splitting

Locally Balanced Splitting:
2-edge-coloring so that every node roughly balanced



Iterated Factor-2-Rounding

for i $= \lceil \log \Delta \rceil, \dots, 1$

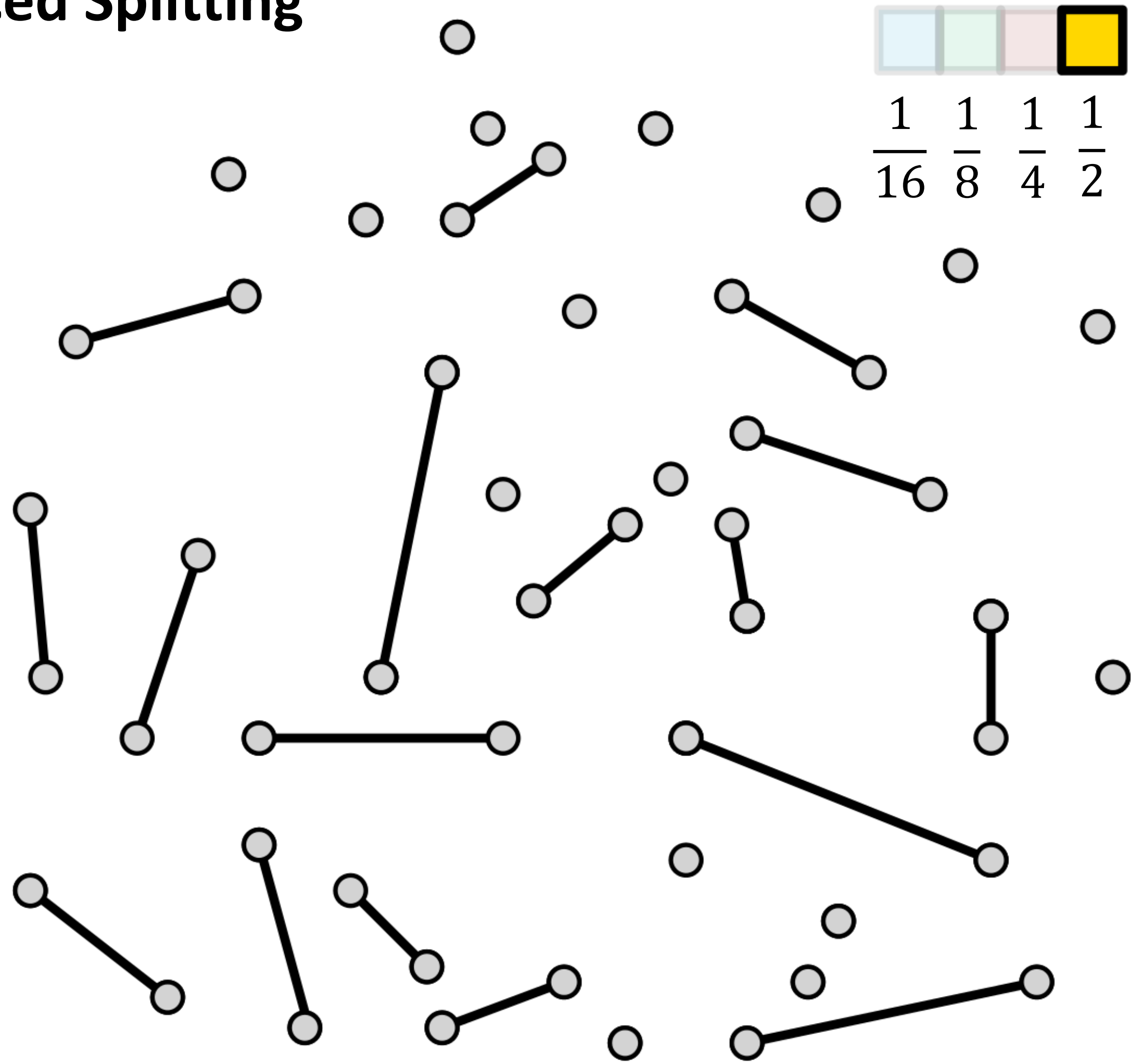
$E_i = \{e \in E : x_e = 2^{-i}\}$

splitting of E_i into ■□

increase ■ to 2^{-i+1}

decrease □ to 0

In case of **perfect** locally balanced **splitting**:
no constraint violated & no loss in total value
(i.e., **perfect rouding**)



II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

II) Rounding Fractional Bipartite Matching

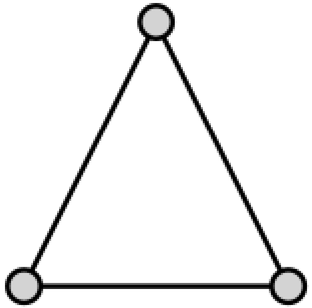
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Perfect Splitting not possible in case of…

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

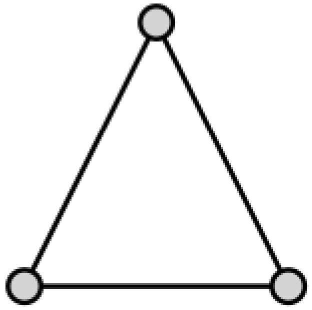
Perfect Splitting not possible in case of…



… odd cycles

II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Perfect Splitting not possible in case of…



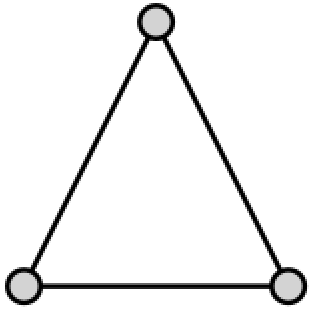
… odd cycles

bipartite graph!

II) Rounding Fractional Bipartite Matching

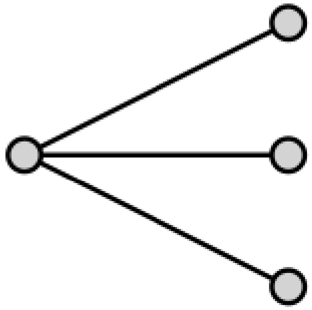
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Perfect Splitting not possible in case of…



… odd cycles

bipartite graph!

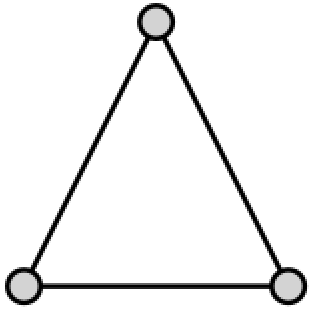


… odd-degree vertices

II) Rounding Fractional Bipartite Matching

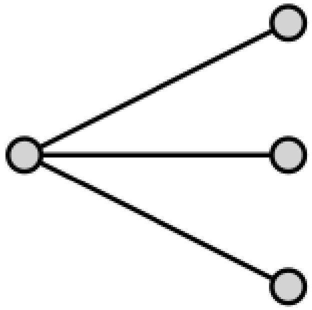
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Perfect Splitting not possible in case of…



… odd cycles

bipartite graph!

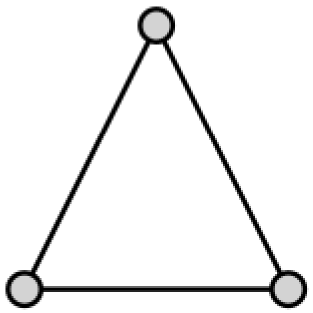


… odd-degree vertices

small technicality.

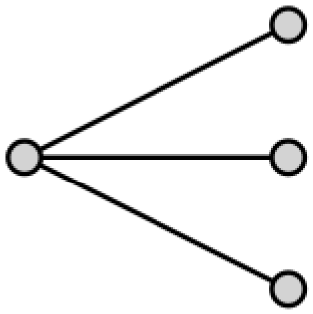
II) Rounding Fractional Bipartite Matching $O(\log^2 \Delta)$ rounds, $O(1)$ loss

Perfect Splitting not possible in case of…



… odd cycles

bipartite graph!



… odd-degree vertices

small technicality.

Suppose that bipartite and even degree!

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
   pick arbitrary cycle
   alternate ■□

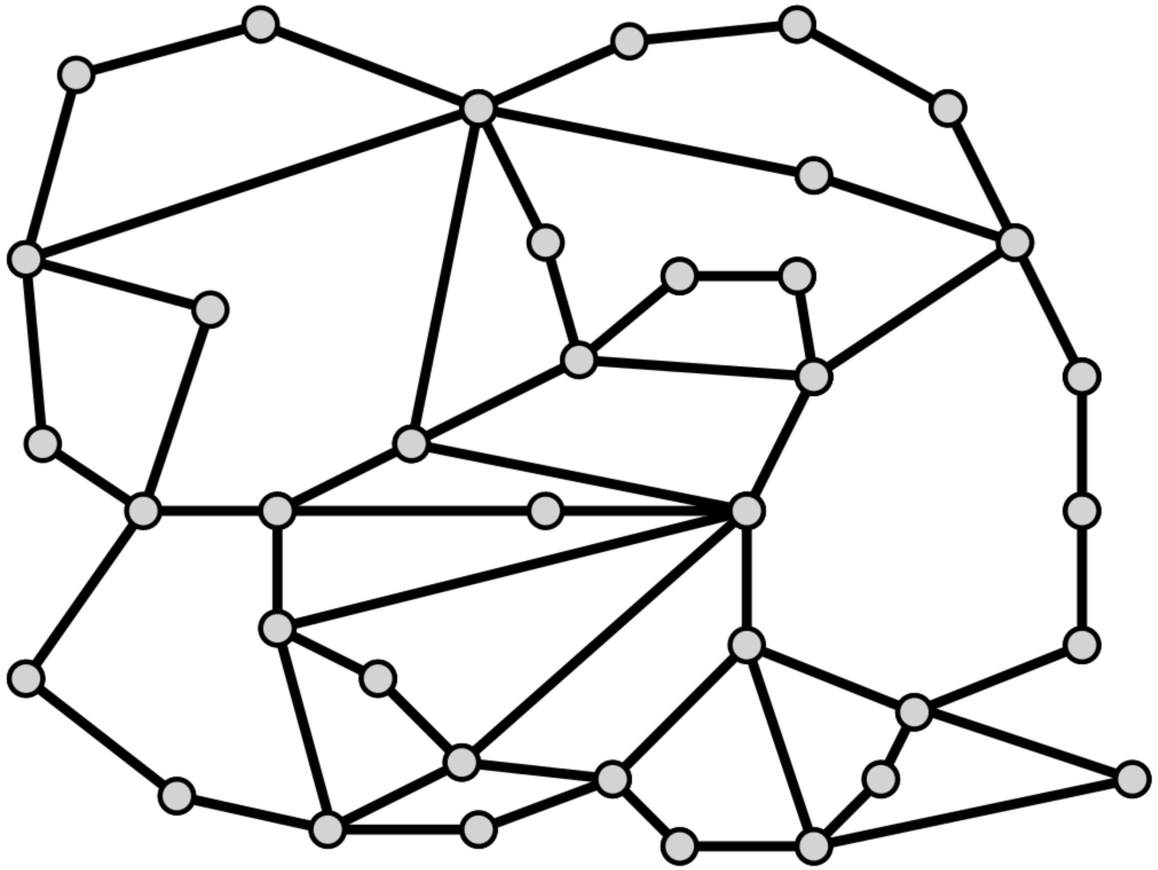
* bipartite and even degree!

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
    pick arbitrary cycle
    alternate



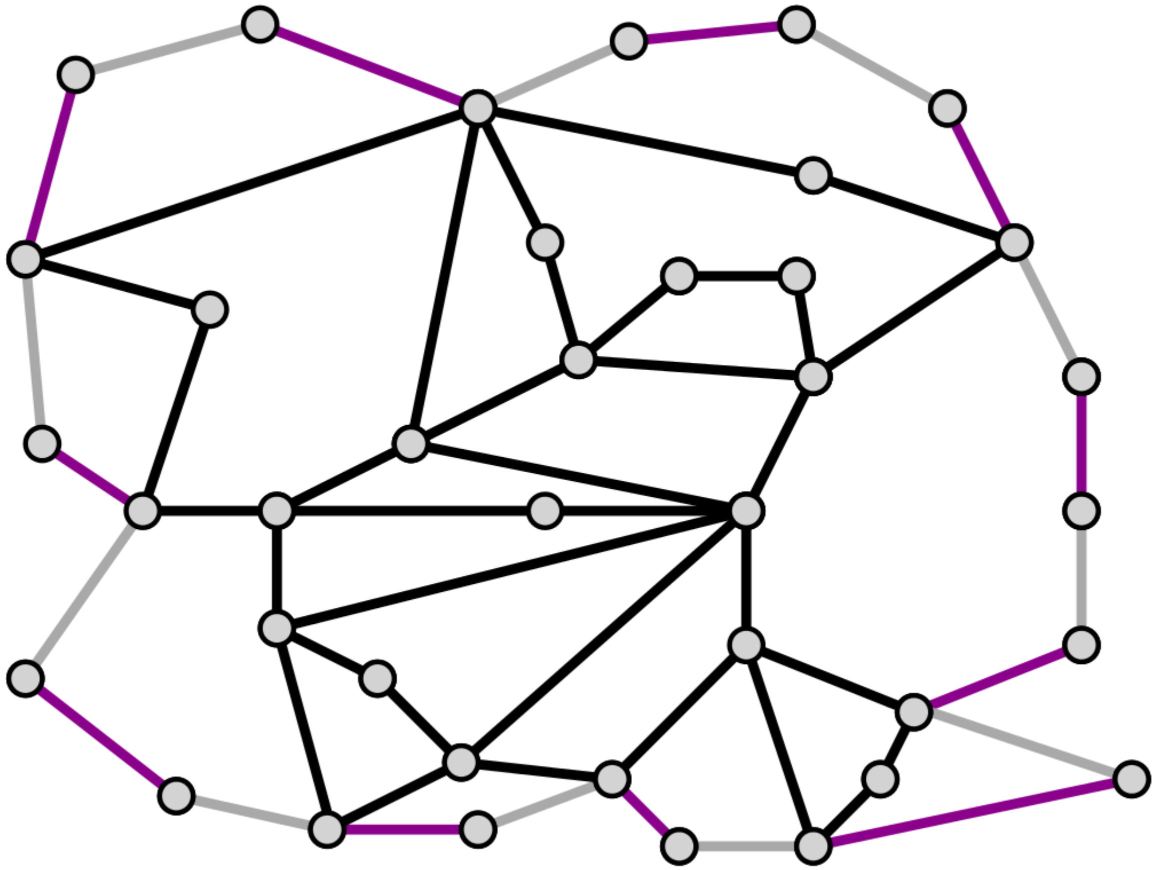
* bipartite and even degree!

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate



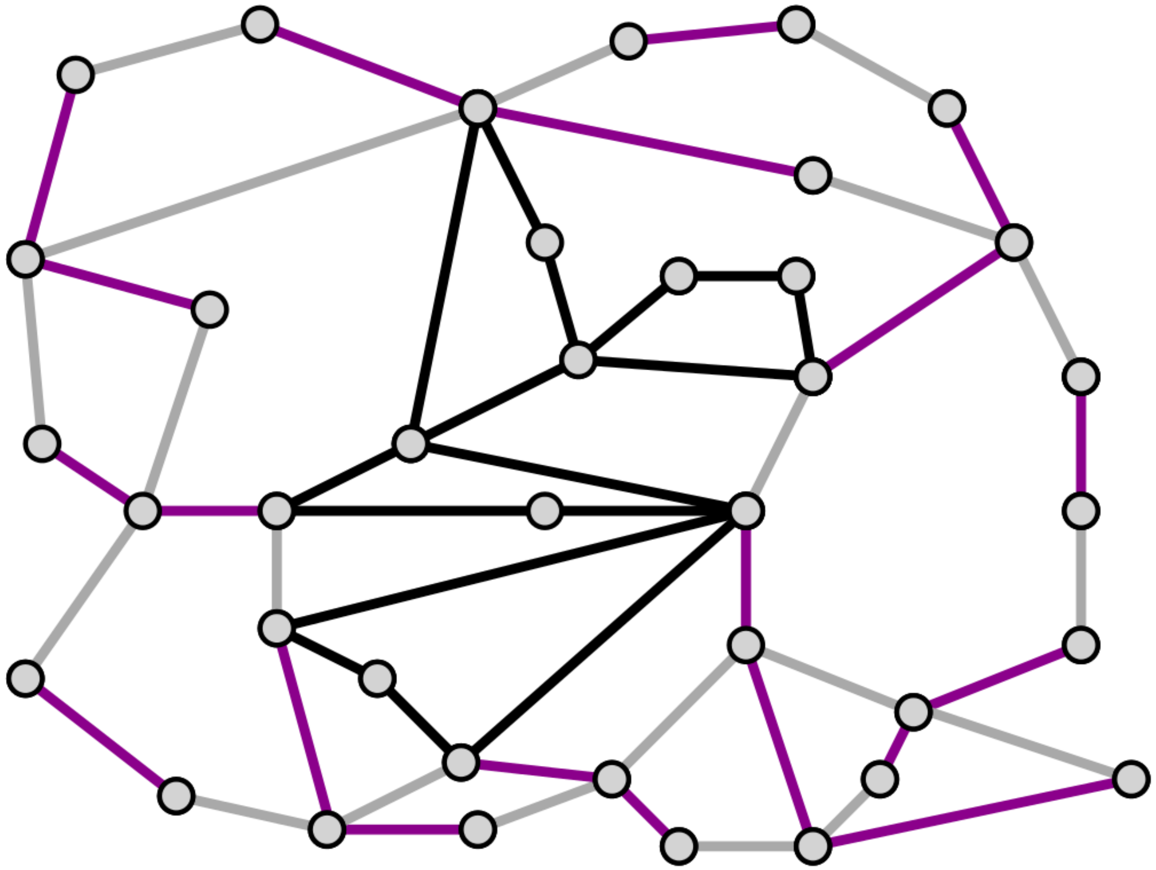
* bipartite and even degree!

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
pick arbitrary cycle
alternate



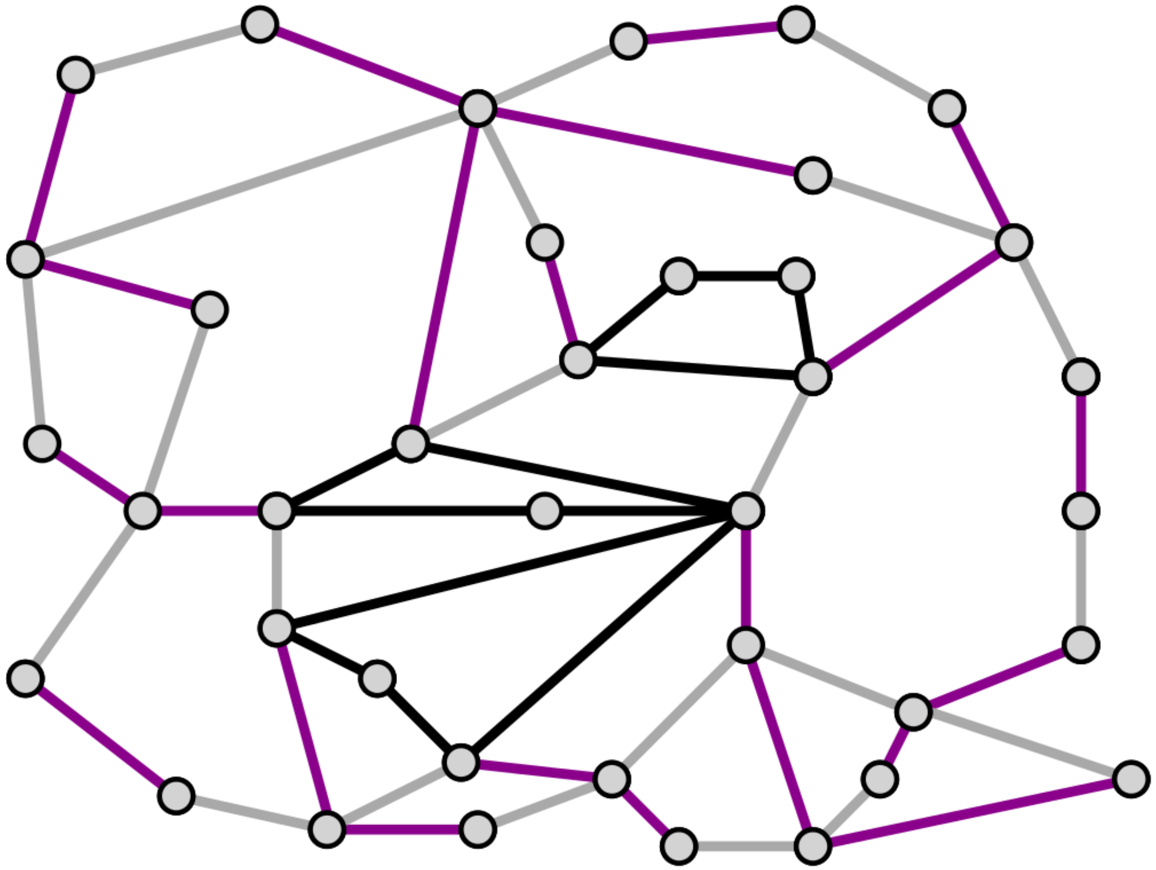
*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
    pick arbitrary cycle
    alternate ■□



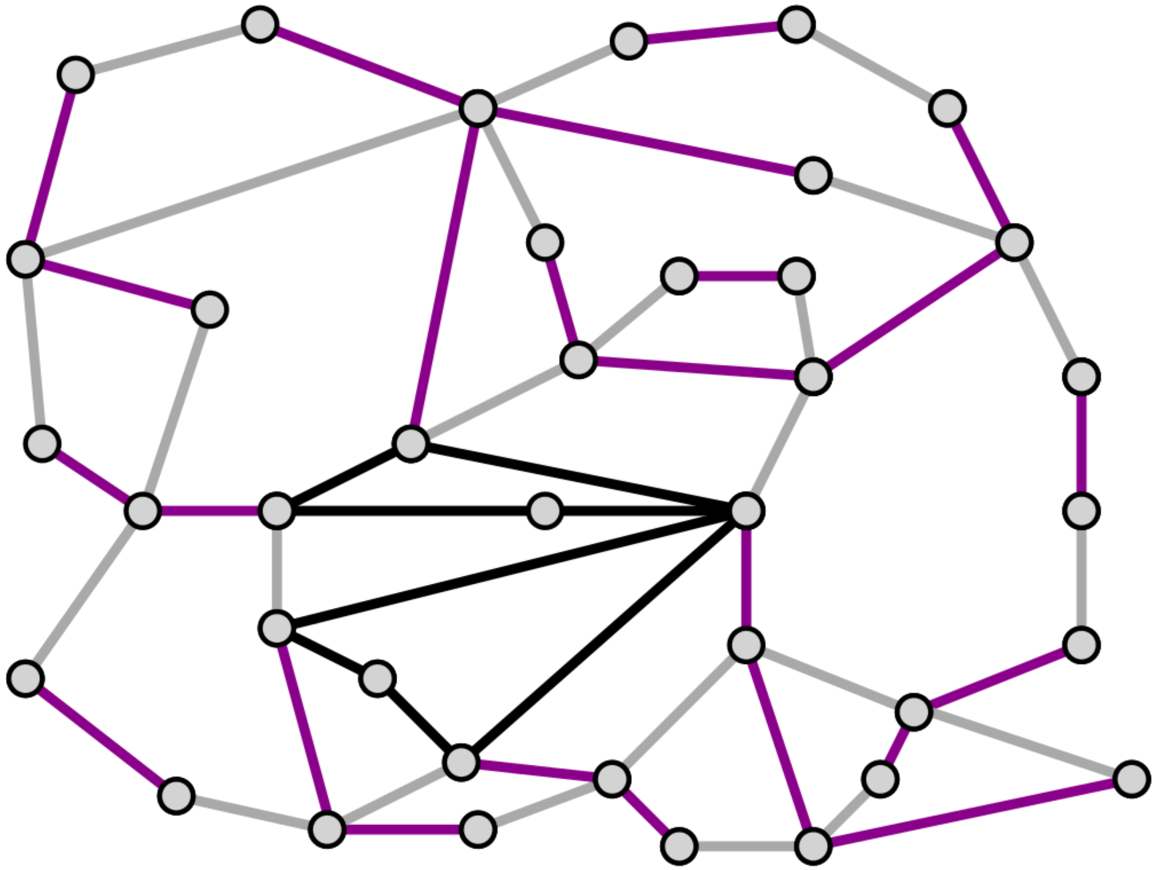
* bipartite and even degree!

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate



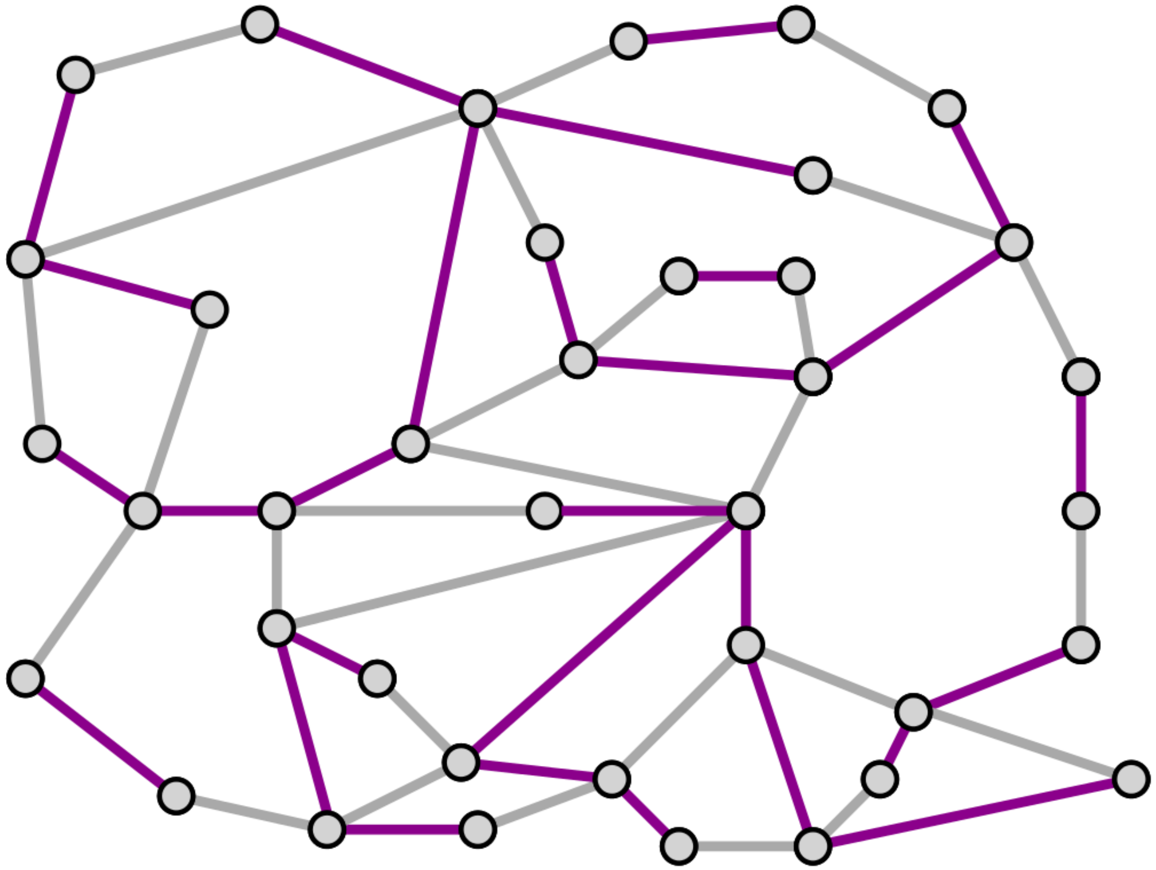
*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate



* bipartite and even degree!

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
  A) **Short cycles** of length $O(\log \Delta)$
    alternate
  B) **Long cycles**
    chop at length $\Theta(\log \Delta)$
    set boundary to 0
    alternate in between

*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

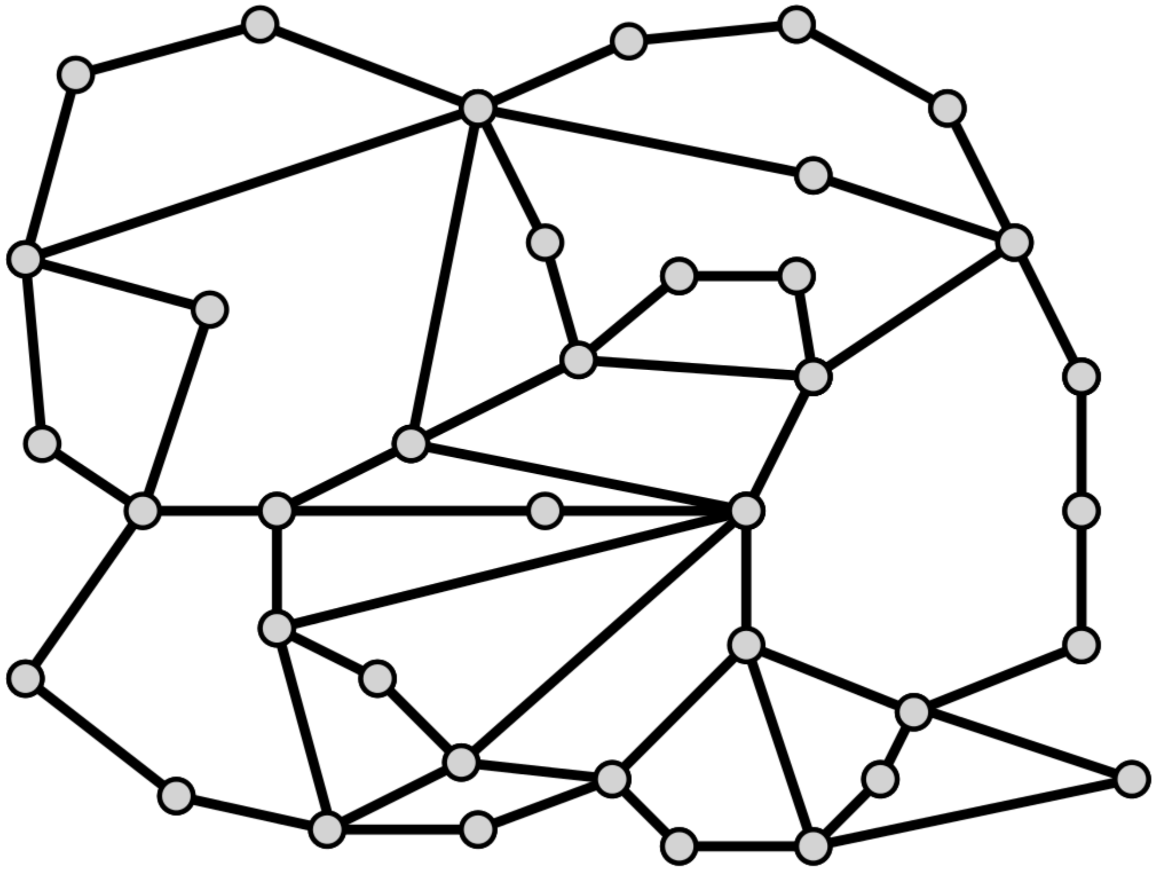
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate ■□

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
  A) **Short cycles** of length $O(\log \Delta)$
    alternate ■□
  B) **Long cycles**
    chop at length $\Theta(\log \Delta)$
    set boundary to 0
    alternate ■□ in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

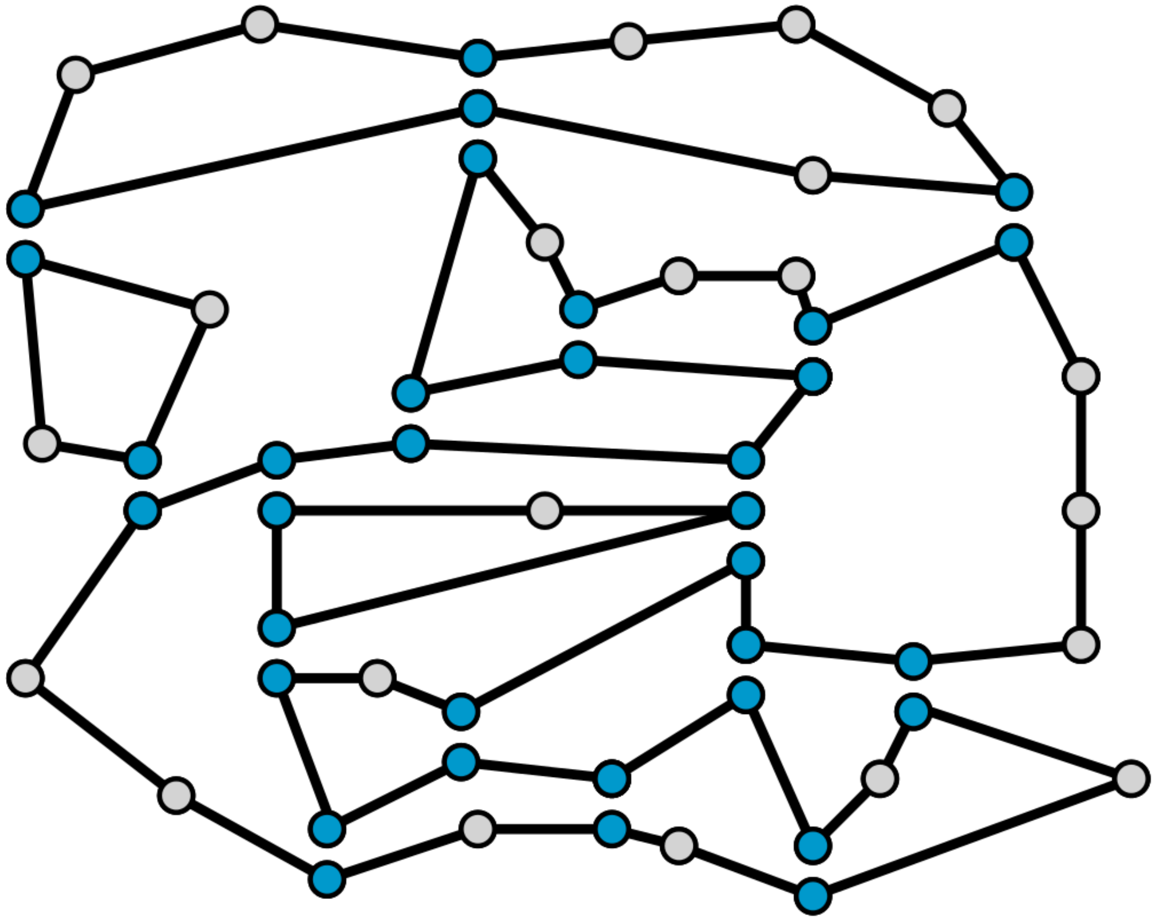
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
- pick arbitrary cycle
- alternate

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
- A) **<u>Short cycles</u>** of length $O(\log \Delta)$
  - alternate
- B) **<u>Long cycles</u>**
  - chop at length $\Theta(\log \Delta)$
  - set boundary to 0
  - alternate in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

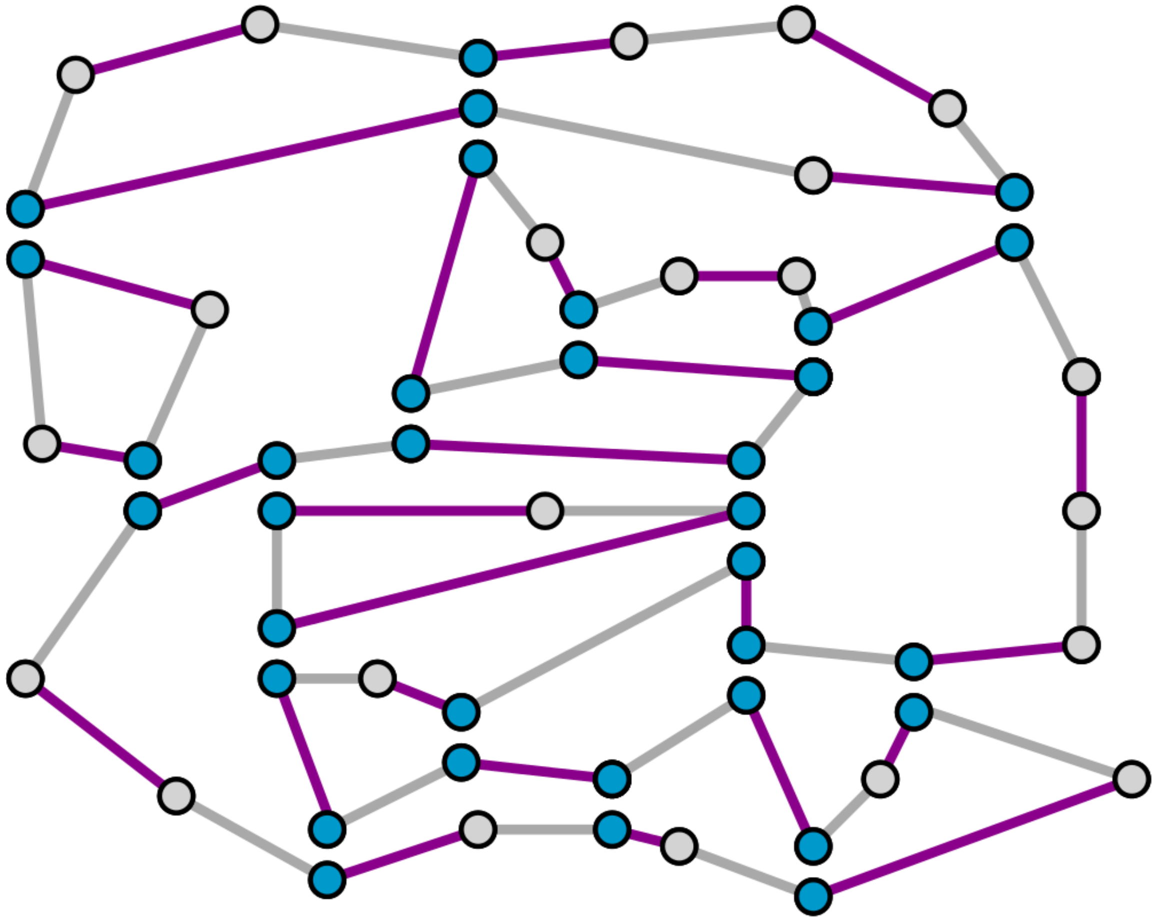
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate ■□

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
  A) **Short cycles** of length $O(\log \Delta)$
    alternate ■□
  B) **Long cycles**
    chop at length $\Theta(\log \Delta)$
    set boundary to 0
    alternate ■□ in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

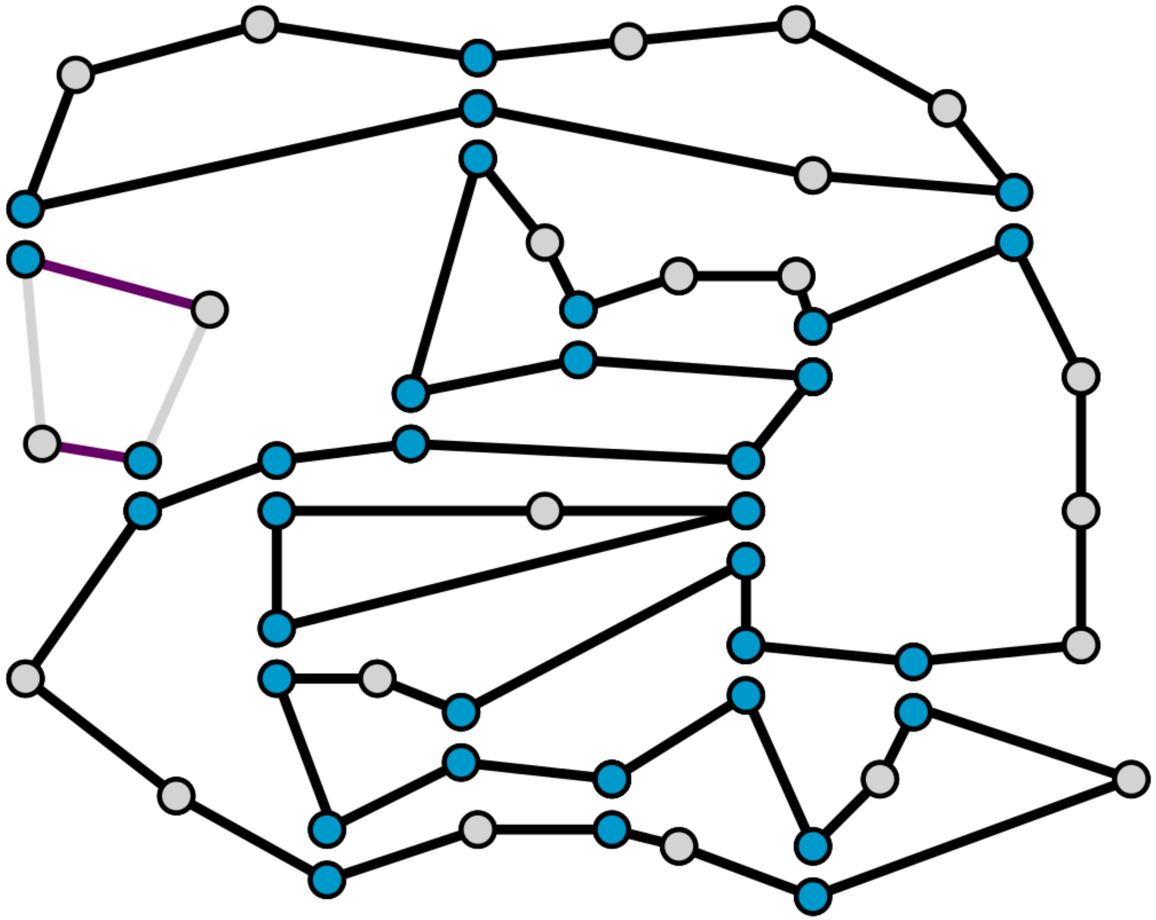
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
 pick arbitrary cycle
 alternate ■□

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
 A) **Short cycles** of length $O(\log \Delta)$
  alternate ■□
 B) **Long cycles**
  chop at length $\Theta(\log \Delta)$
  set boundary to 0
  alternate ■□ in between



* **bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

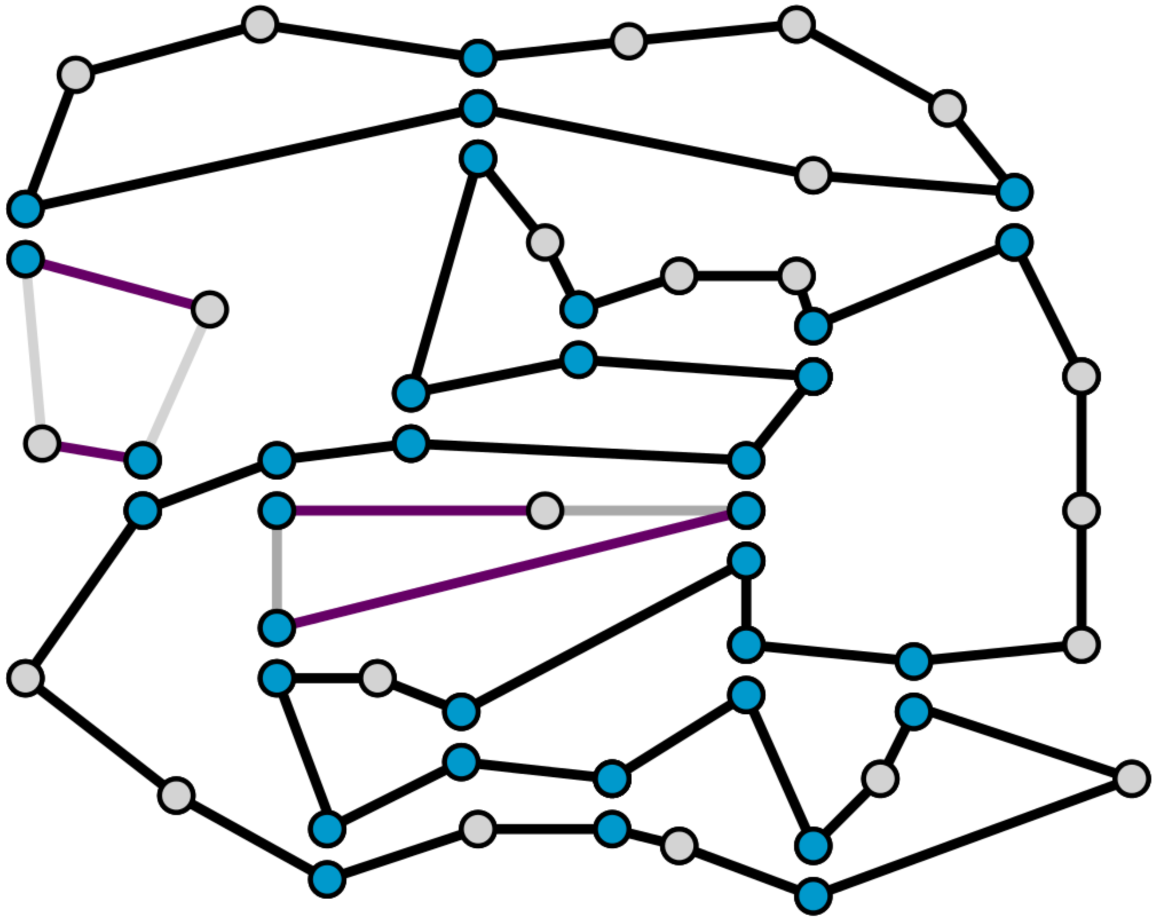
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
  A) **Short cycles** of length $O(\log \Delta)$
    alternate
  B) **Long cycles**
    chop at length $\Theta(\log \Delta)$
    set boundary to 0
    alternate in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

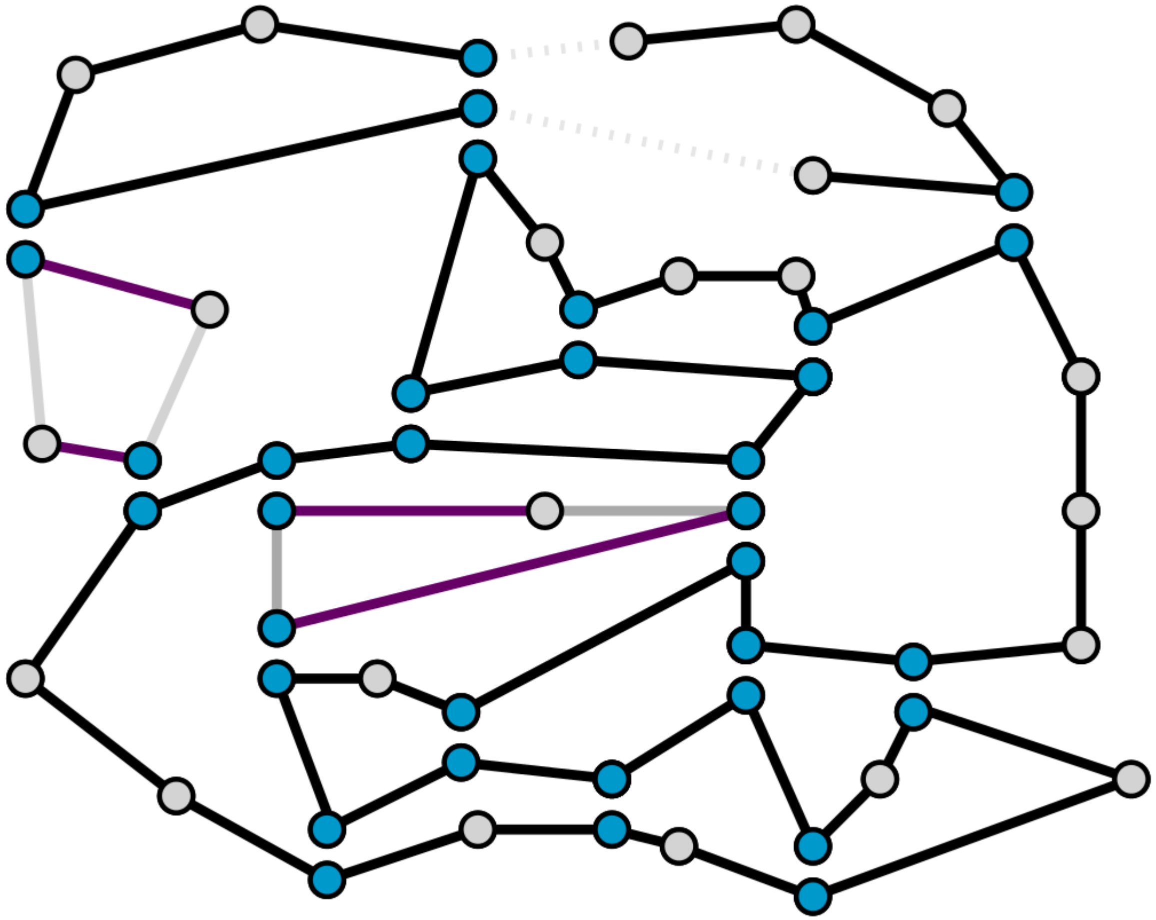
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
  A) **Short cycles** of length $O(\log \Delta)$
    alternate
  B) **Long cycles**
    chop at length $\Theta(\log \Delta)$
    set boundary to 0
    alternate in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

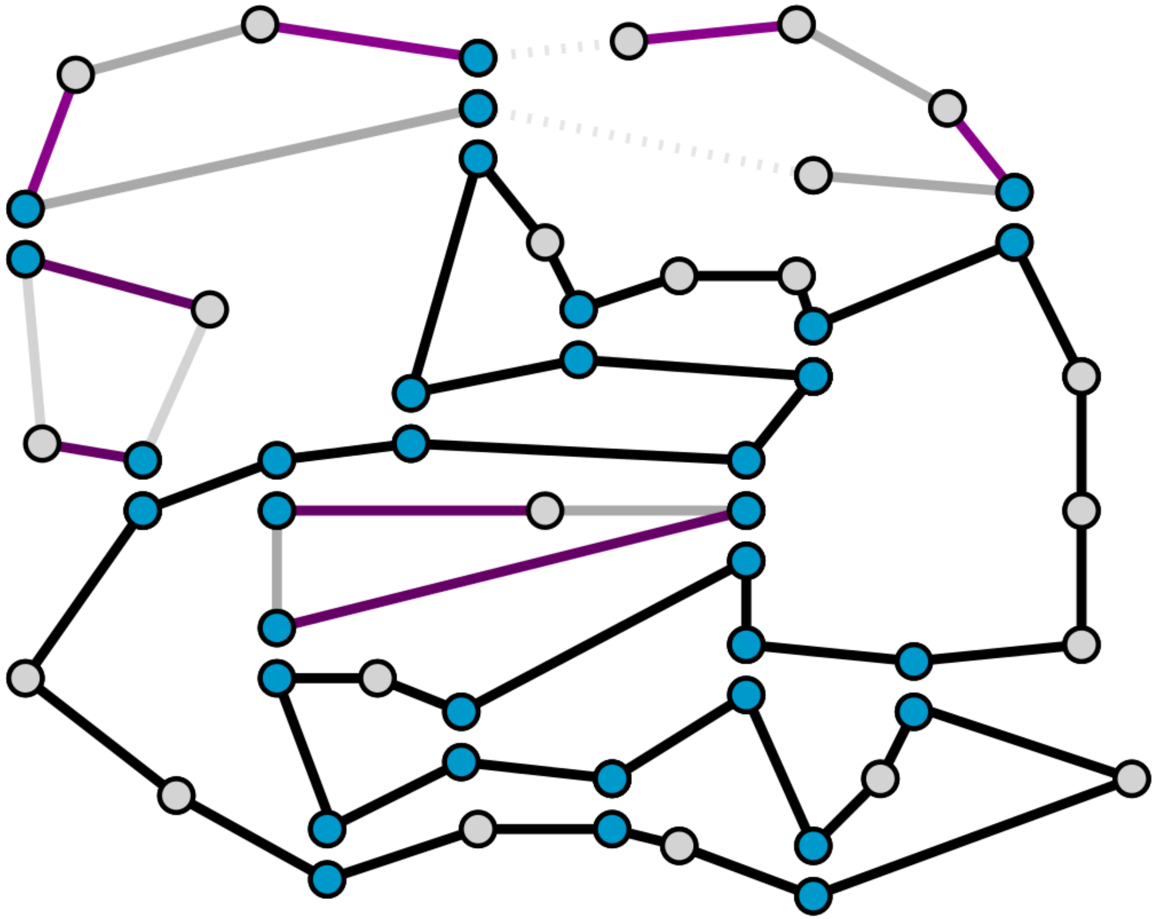
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
- pick arbitrary cycle
- alternate ■□

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
- A) **<u>Short cycles</u>** of length $O(\log \Delta)$
  - alternate ■□
- B) **<u>Long cycles</u>**
  - chop at length $\Theta(\log \Delta)$
  - set boundary to 0
  - alternate ■□ in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

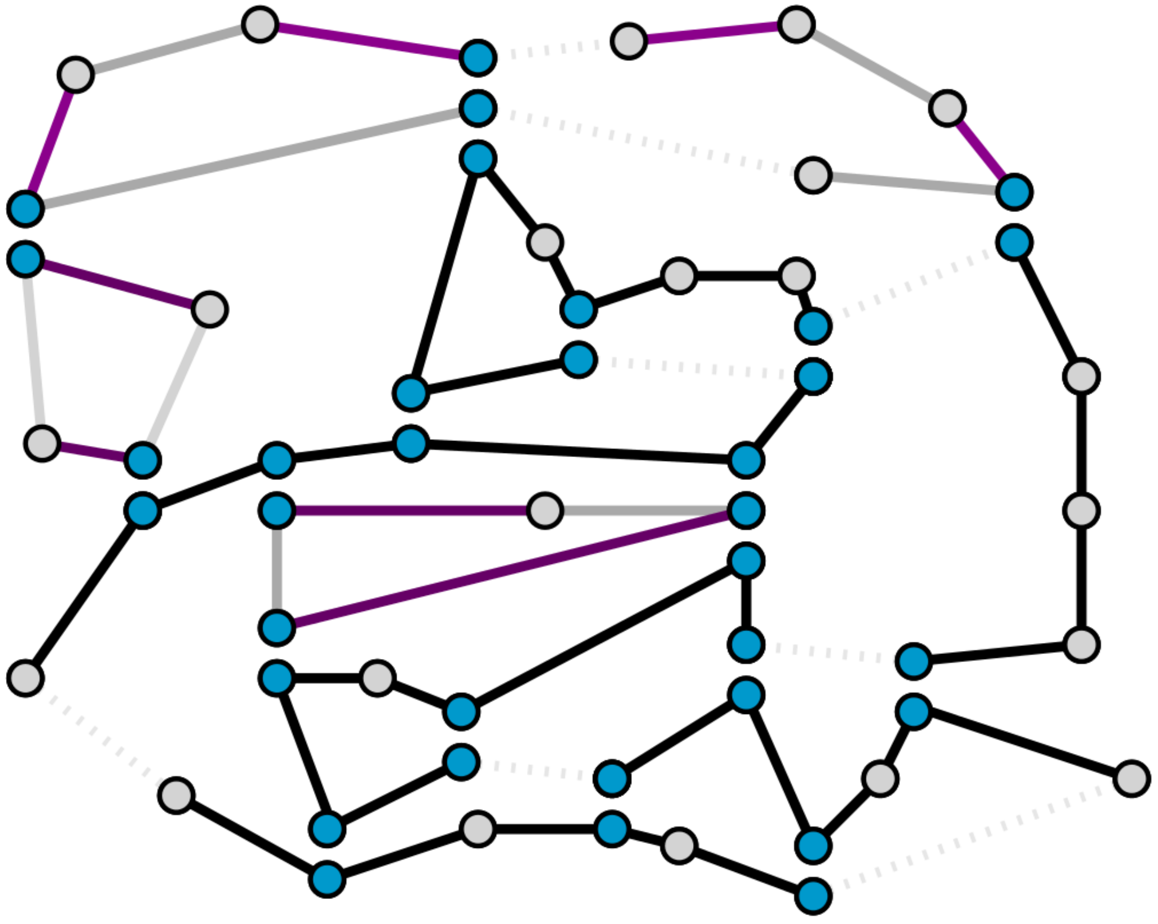
Repeat until all edges colored
- pick arbitrary cycle
- alternate ■■

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles

In parallel, for all cycles

- A) **Short cycles** of length $O(\log \Delta)$
  - alternate ■■
- B) **Long cycles**
  - chop at length $\Theta(\log \Delta)$
  - set boundary to 0
  - alternate ■■ in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

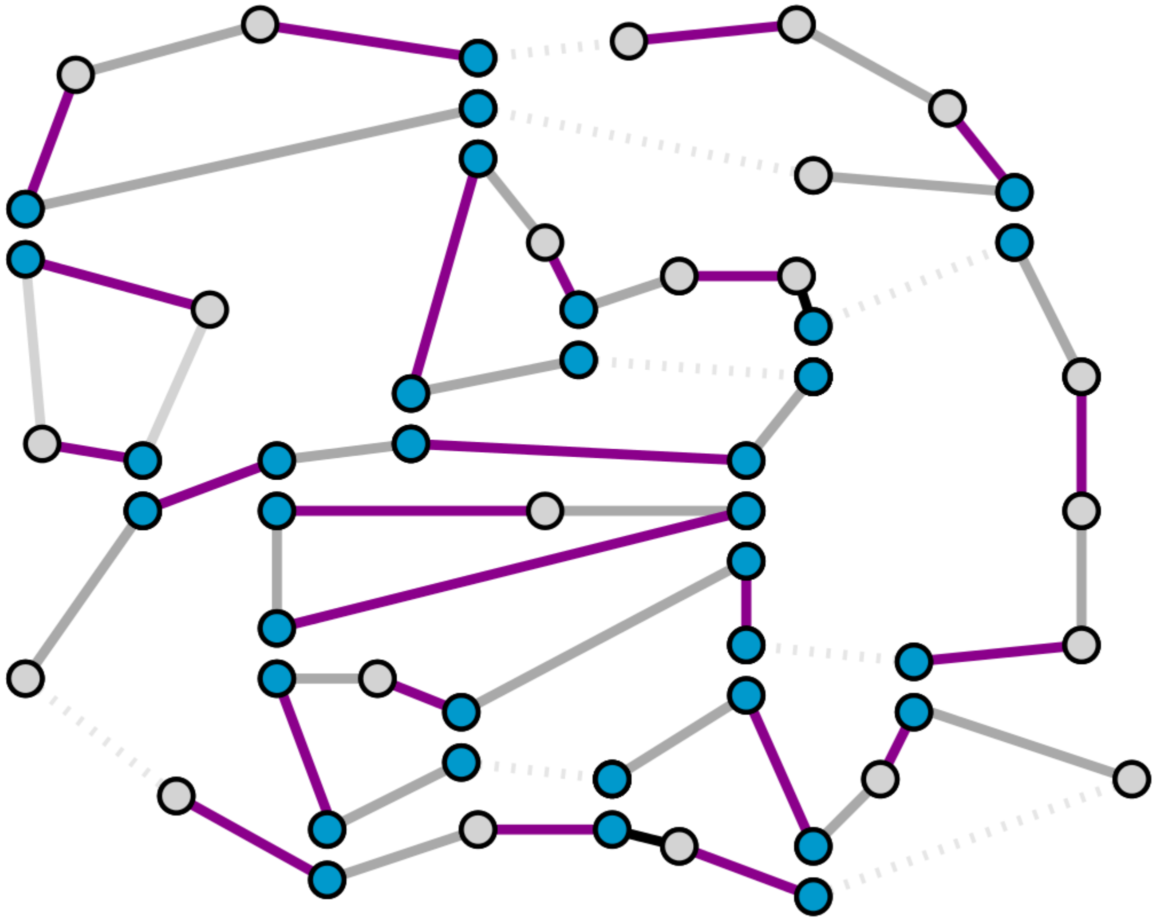
$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
- pick arbitrary cycle
- alternate ■□

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
- A) **<u>Short cycles</u>** of length $O(\log \Delta)$
  - alternate ■□
- B) **<u>Long cycles</u>**
  - chop at length $\Theta(\log \Delta)$
  - set boundary to 0
  - alternate ■□ in between



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

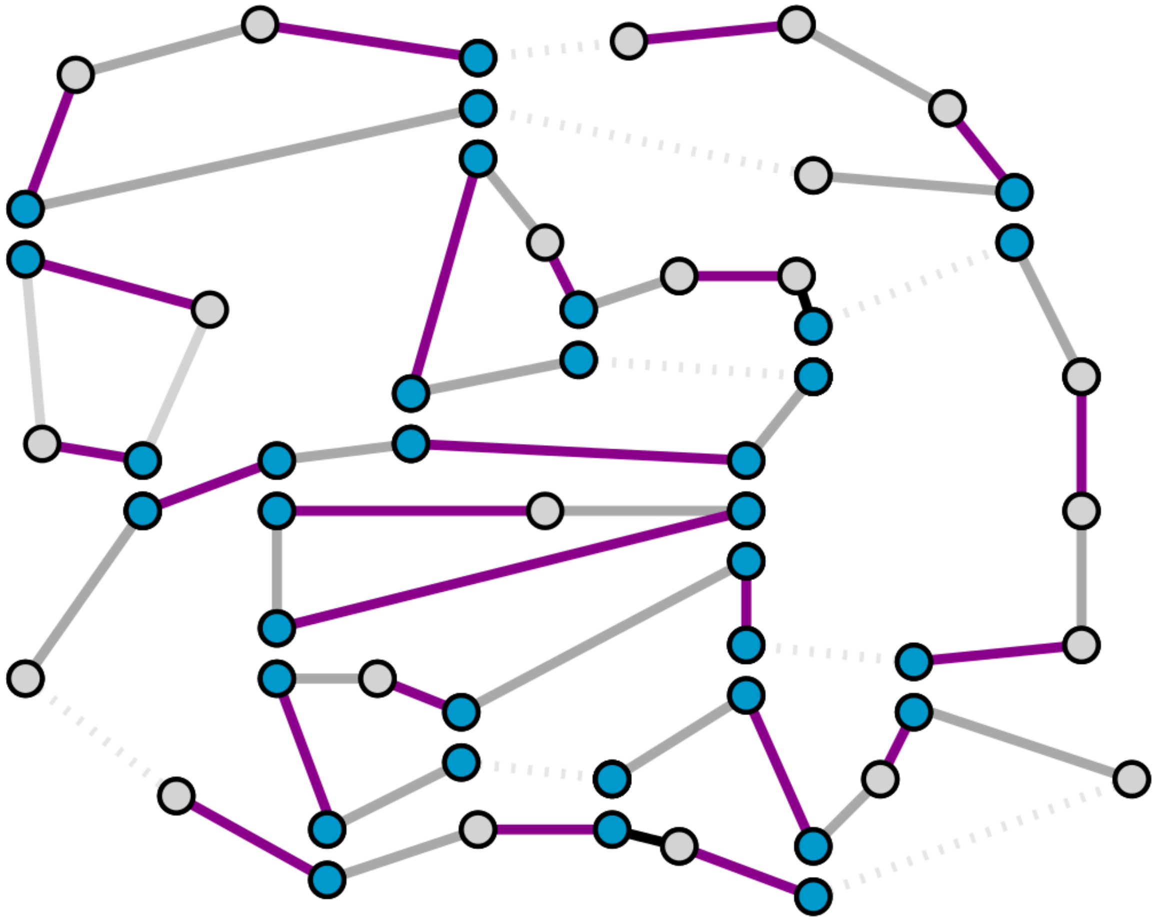
Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate ■□

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
  A) **Short cycles** of length $O(\log \Delta)$
    alternate ■□
  B) **Long cycles**
    chop at length $\Theta(\log \Delta)$
    set boundary to 0
    alternate ■□ in between

$\Theta\left(\frac{1}{\log \Delta}\right)$ loss



*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

Sequential Perfect Splitting*

Repeat until all edges colored
pick arbitrary cycle
alternate ■□

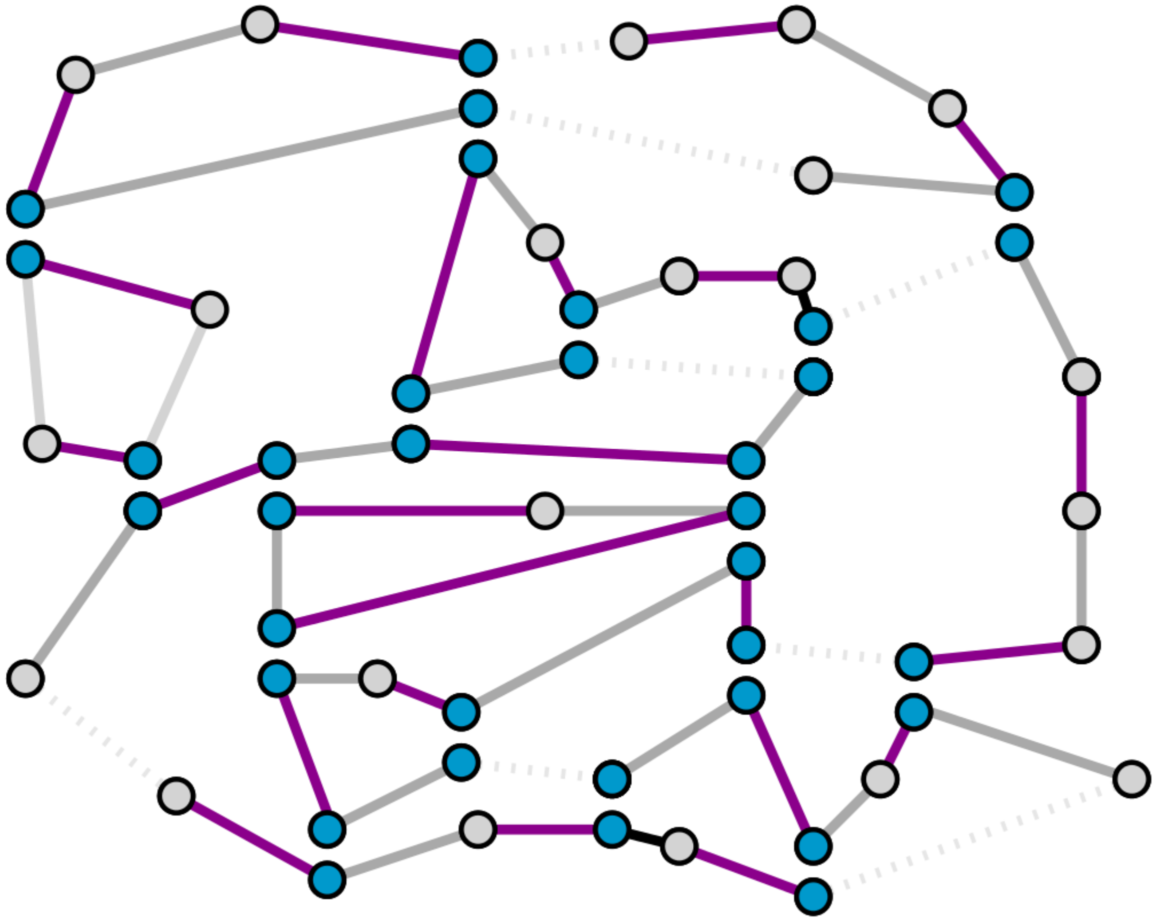
LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles

A) **<u>Short cycles</u>** of length $O(\log \Delta)$
alternate ■□

B) **<u>Long cycles</u>**
chop* at length $\Theta(\log \Delta)$
set boundary to 0
alternate ■□ in between

$\Theta\left(\frac{1}{\log \Delta}\right)$ loss



* by *Hańćkowiak, Karoński, Panconesi* [SODA'98,PODC'99] in $O(\log \Delta)$

*** bipartite and even degree!**

II) Rounding Fractional Bipartite Matching

$O(\log^2 \Delta)$ rounds, $O(1)$ loss

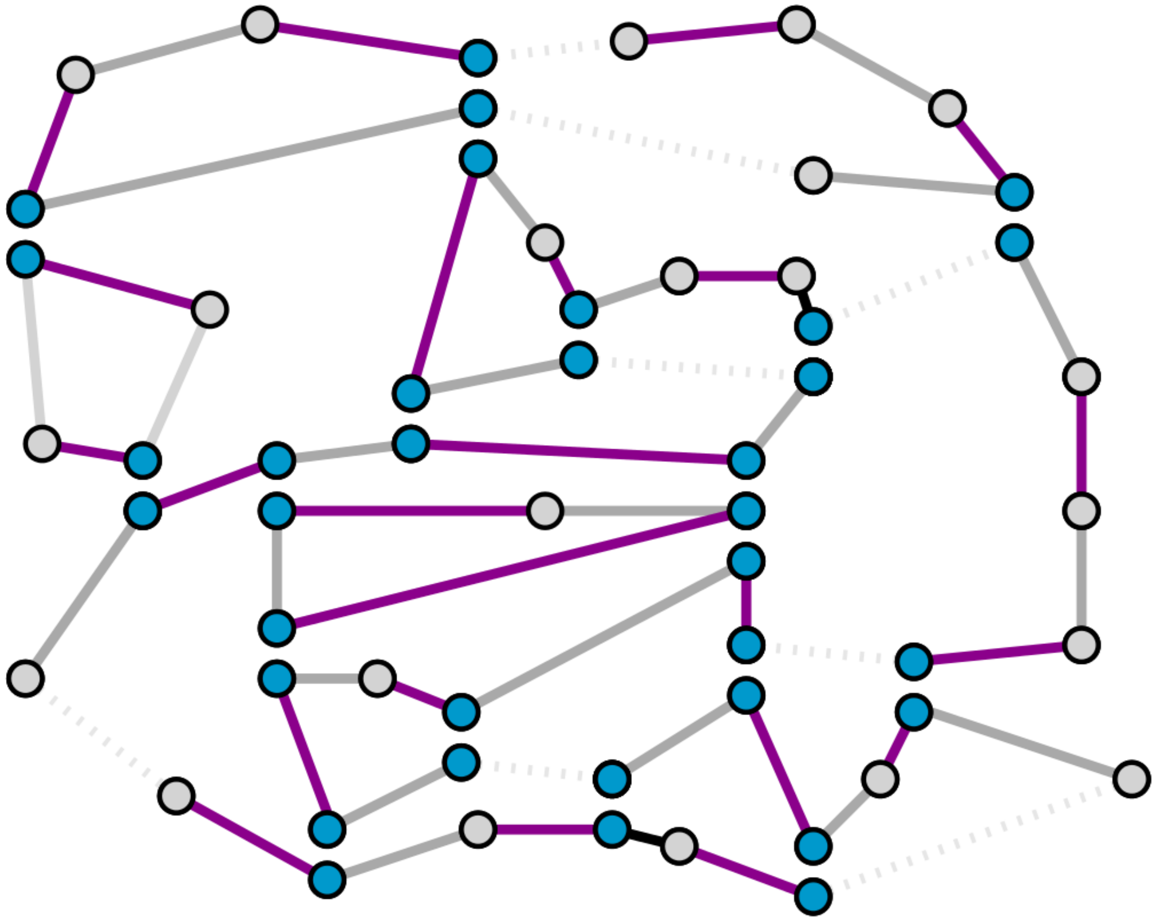
Sequential Perfect Splitting*

Repeat until all edges colored
  pick arbitrary cycle
  alternate ■□

LOCAL Almost-Perfect Splitting*

Decompose into edge-disjoint cycles
In parallel, for all cycles
  A) **Short cycles** of length $O(\log \Delta)$
    alternate ■□
  B) **Long cycles**
    chop* at length $\Theta(\log \Delta)$
    set boundary to 0
    alternate ■□ in between

$\Theta\left(\frac{1}{\log \Delta}\right)$ loss



* by *Hańćkowiak, Karoński, Panconesi* [SODA'98,PODC'99] in $O(\log \Delta)$

Over all O($\log \Delta$) rounding iterations, total loss still constant!

*** bipartite and even degree!**

Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds

I) 4-Approximate Fractional Matching

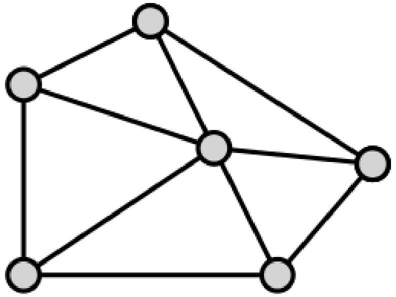
$O(\log \Delta)$ rounds

II) Rounding Fractional Bipartite Matching

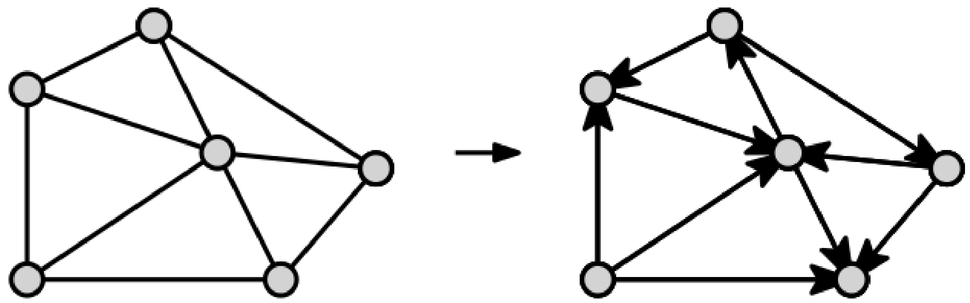
$\mathrm{O}(\log^2 \Delta)$ rounds, $\mathrm{O}(1)$ loss

Constant - Approximate ~~Bipartite~~ Matching $O(\log^2 \Delta)$ rounds

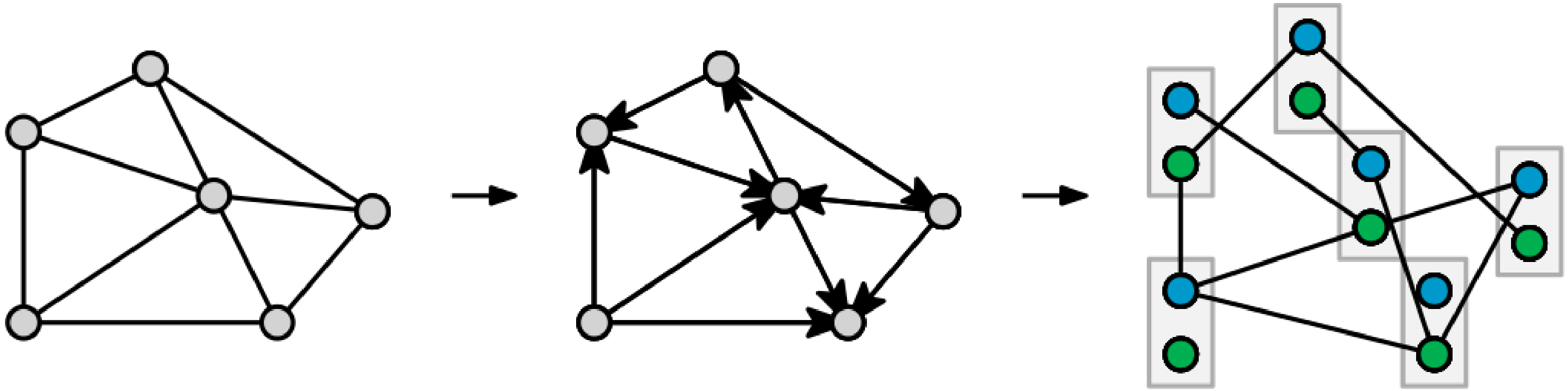
Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds



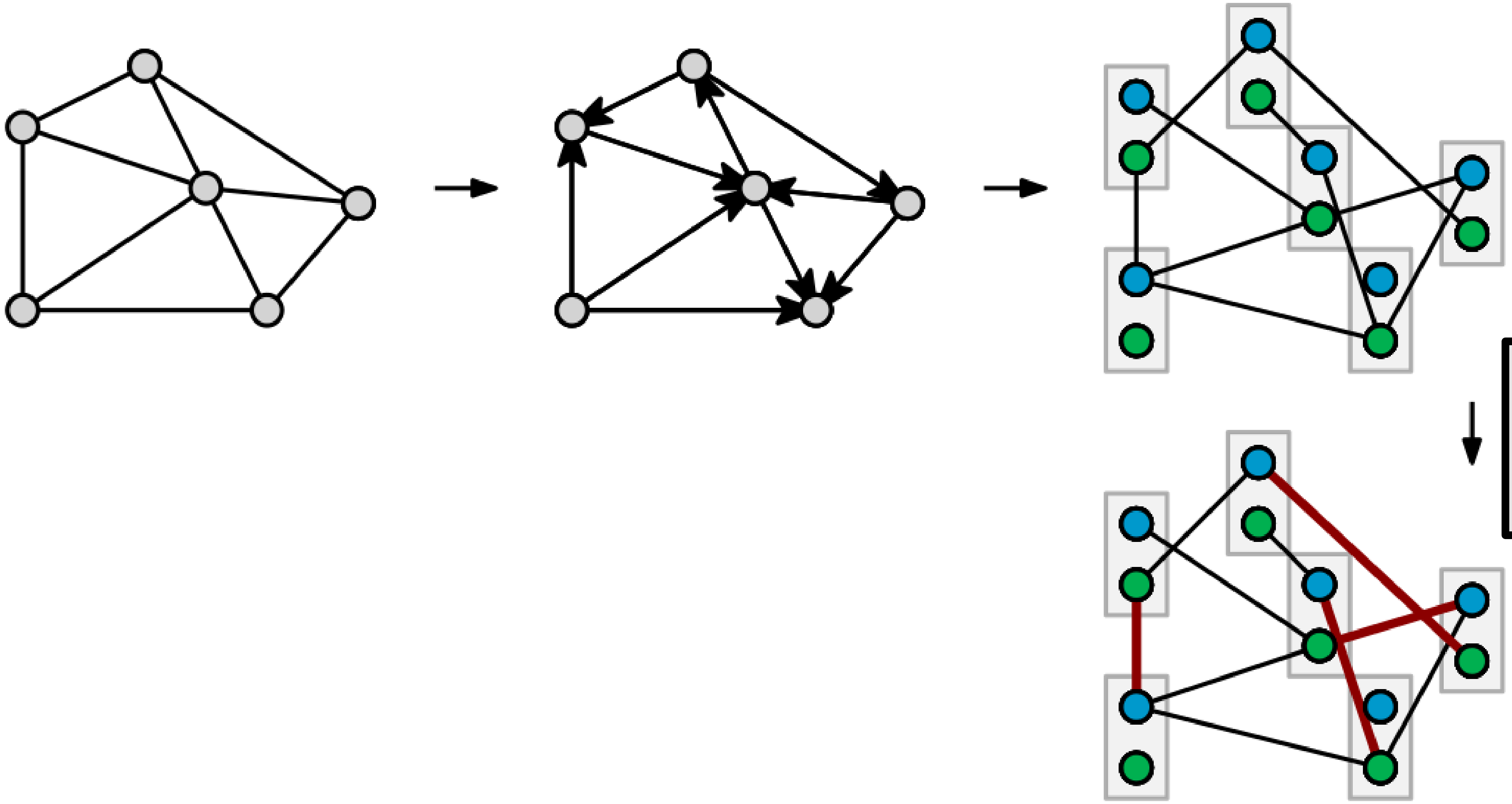
Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds



Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds

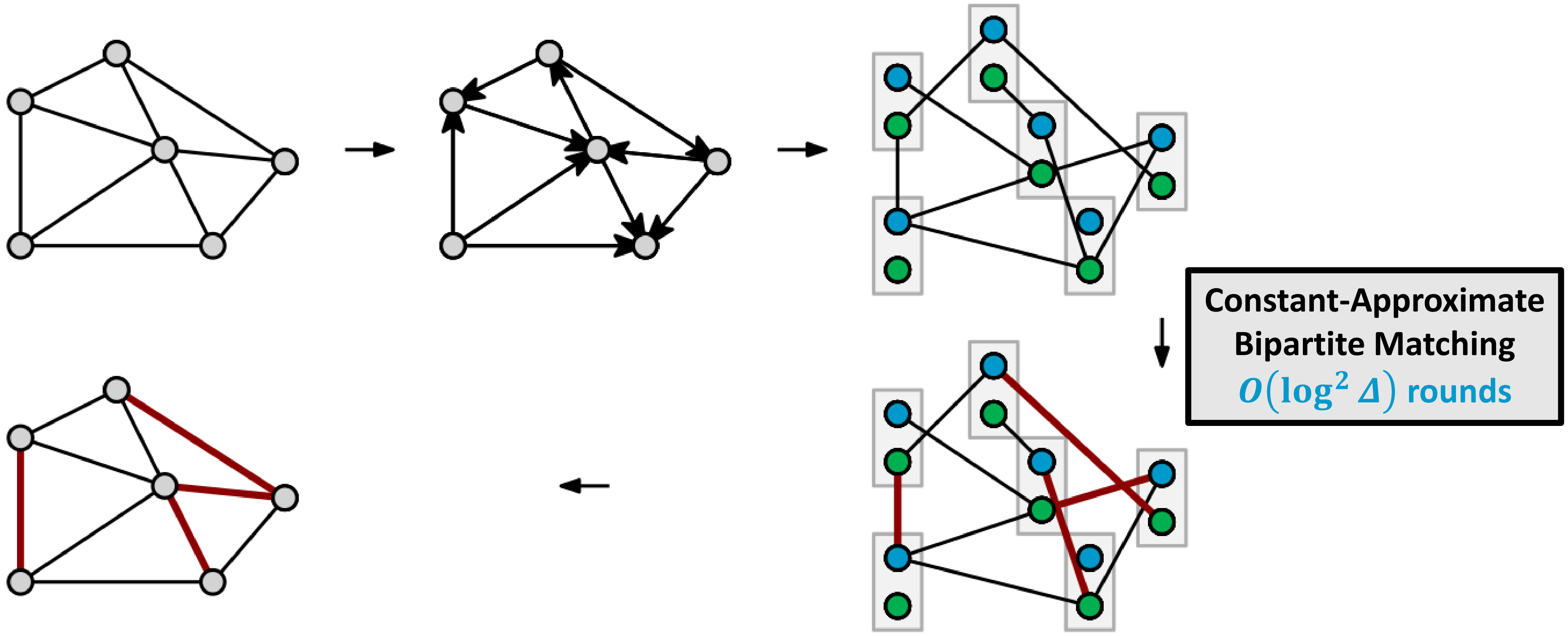


Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds

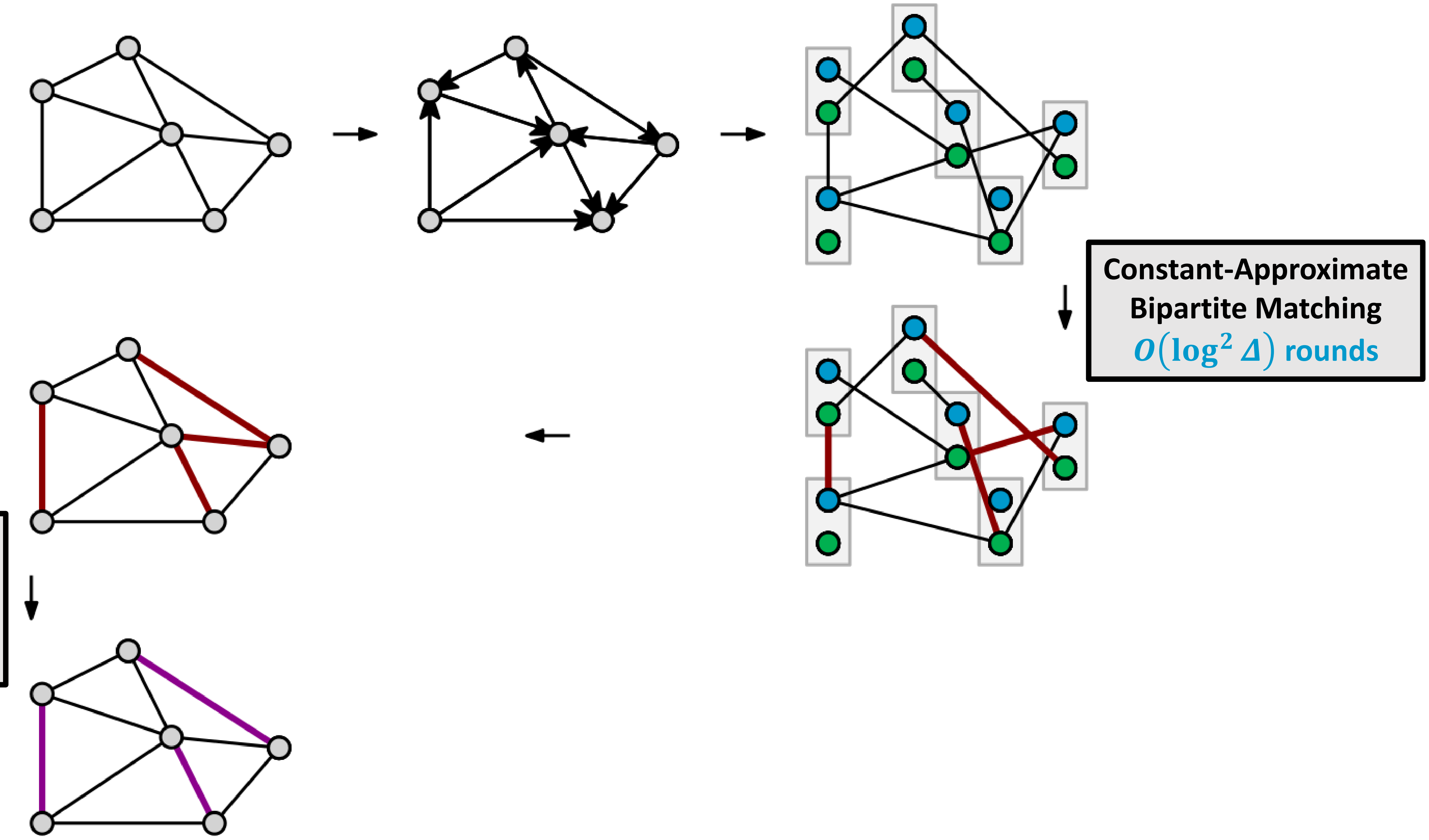


Constant-Approximate Bipartite Matching
$O(\log^2 \Delta)$ rounds

Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds



Constant - Approximate Bipartite Matching $O(\log^2 \Delta)$ rounds



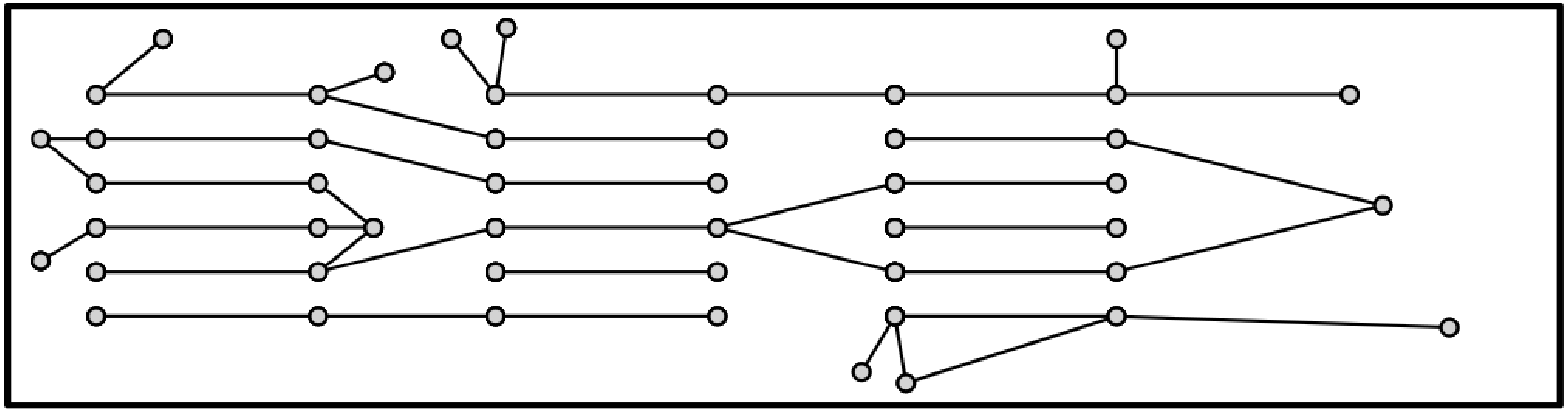
Panconesi, Rizzi [DIST'01]

Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $\boldsymbol{O\left(\log^2 \Delta \cdot \log n\right)}$

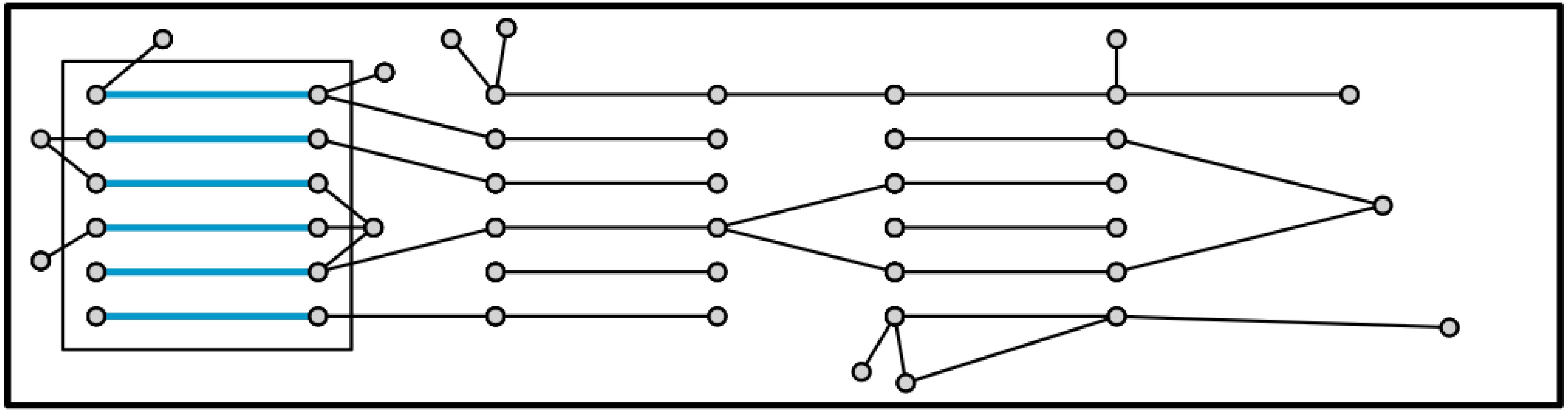
Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $\boldsymbol{O(\log^2 \Delta \cdot \log n)}$



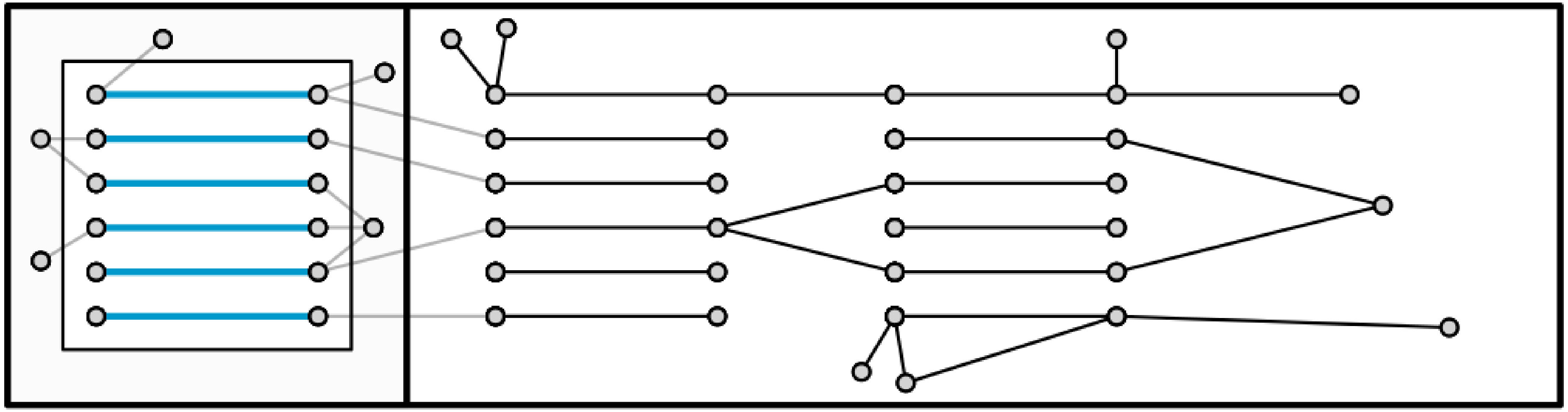
Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $O(\log^2 \Delta \cdot \log n)$



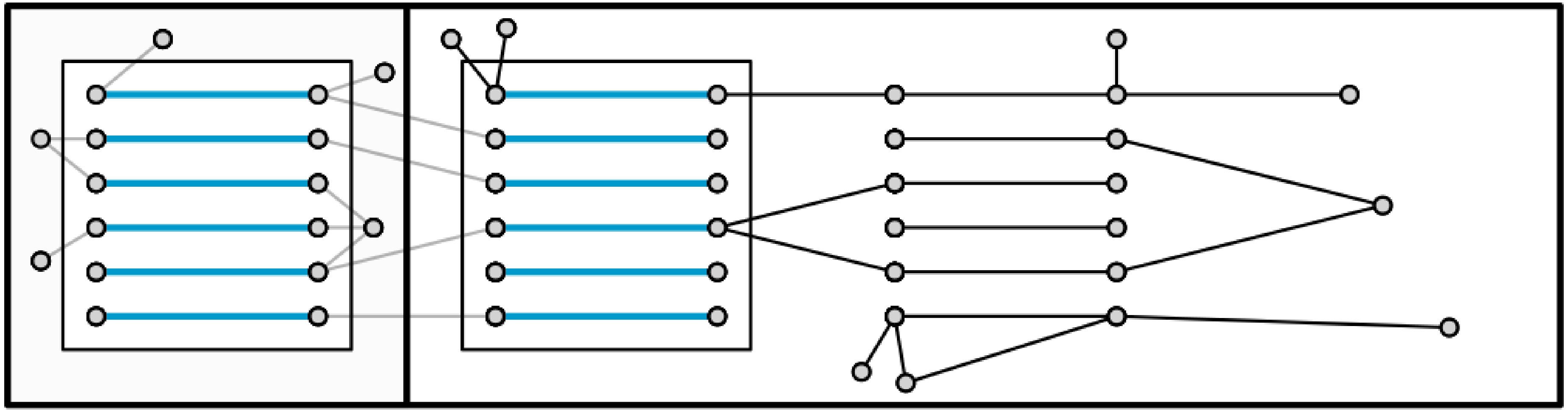
Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $\boldsymbol{O(\log^2 \Delta \cdot \log n)}$



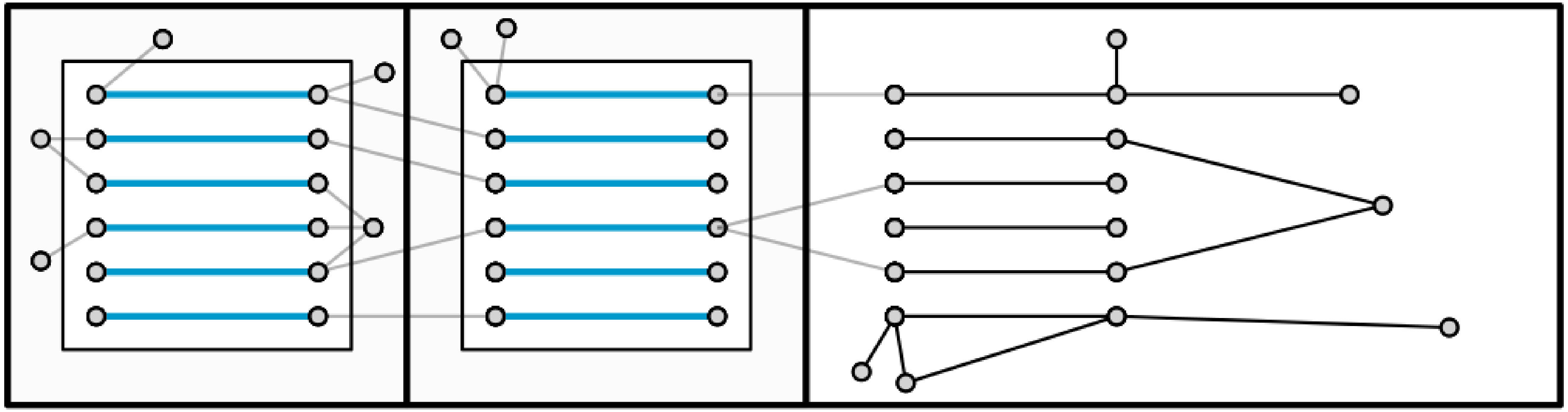
Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $\boldsymbol{O(\log^2 \Delta \cdot \log n)}$

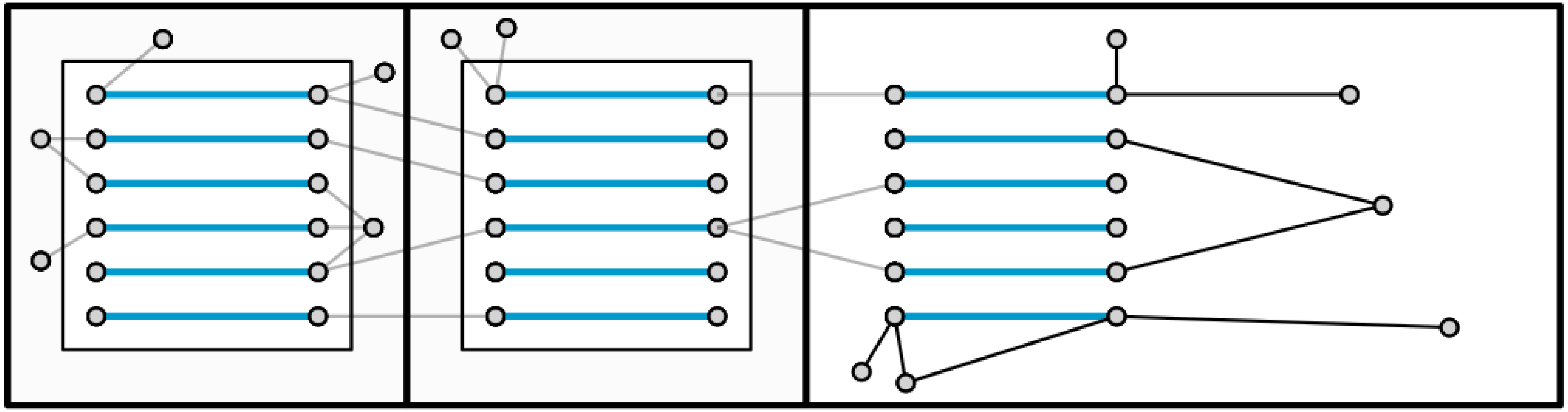


Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $\boldsymbol{O\left(\log^2 \Delta \cdot \log n\right)}$



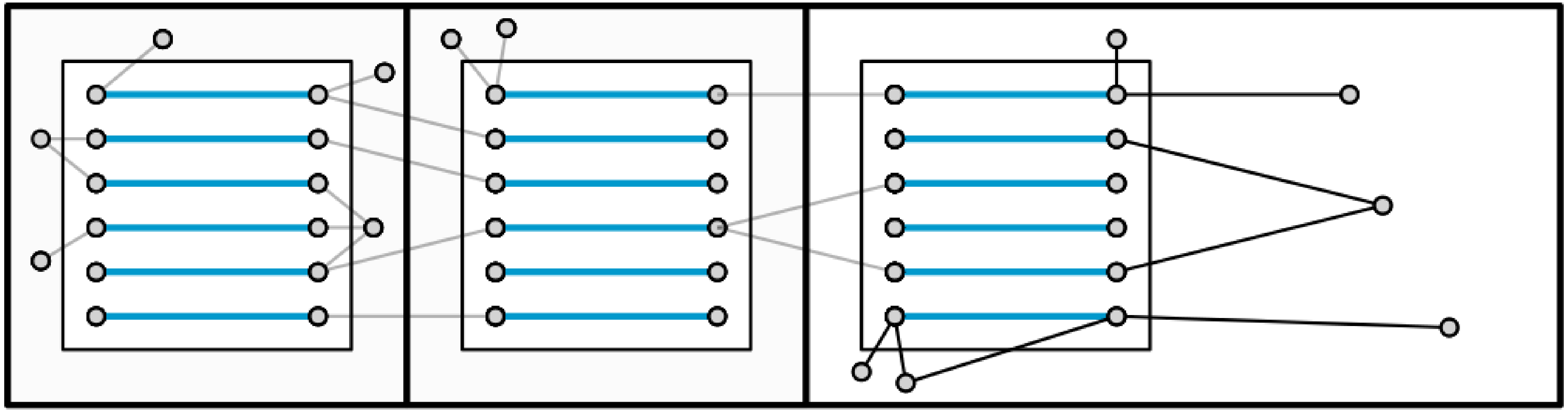
Maximal $O(\log^2 \Delta \cdot \log n)$



Constant - Approximate Matching

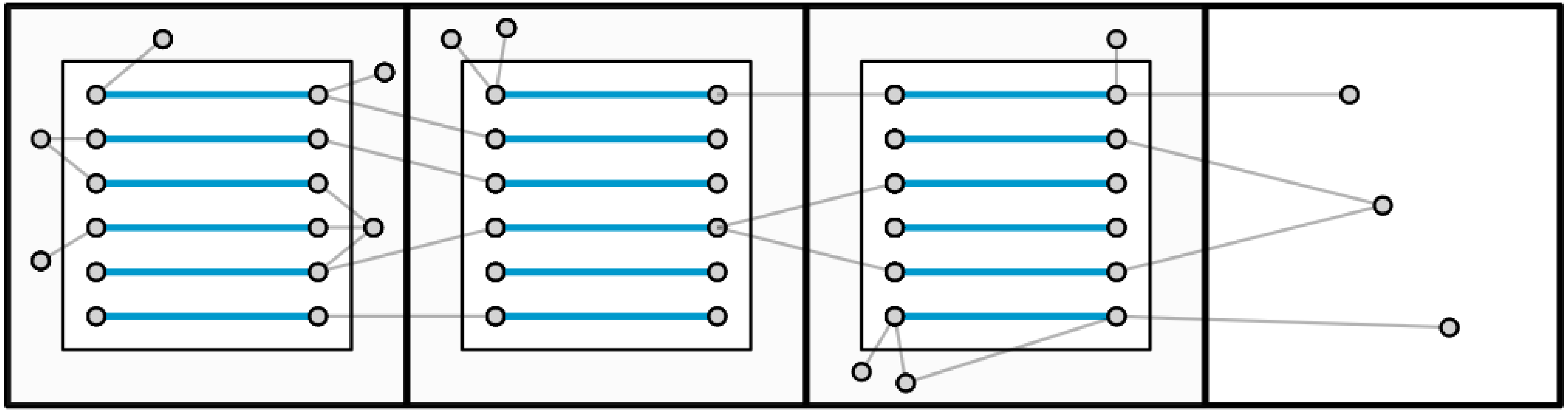
$O(\log^2 \Delta)$ rounds

Maximal $\boldsymbol{O\left(\log^2 \Delta \cdot \log n\right)}$

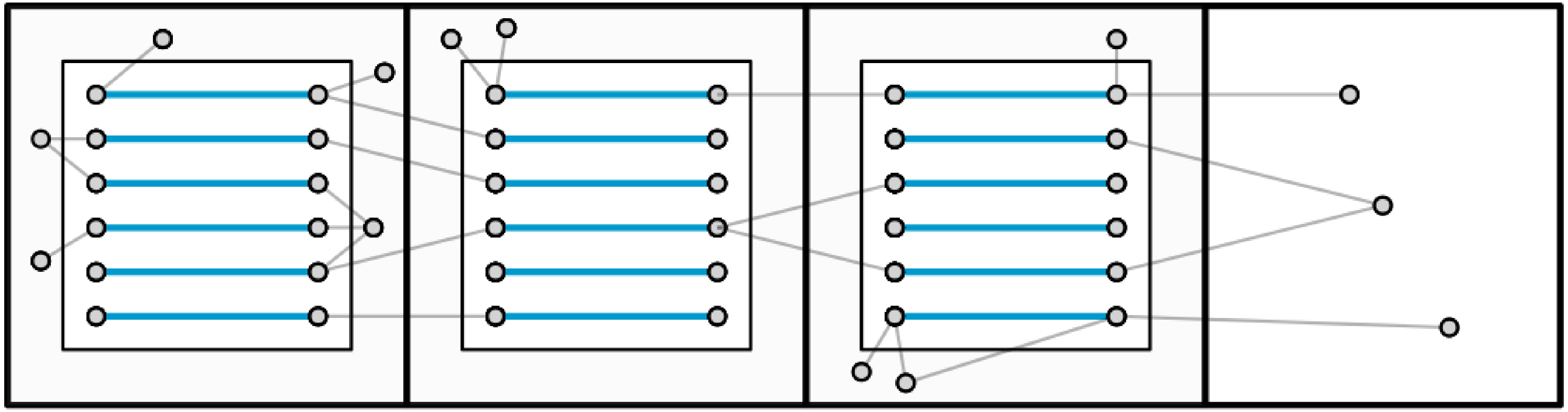


Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $\boldsymbol{O(\log^2 \Delta \cdot \log n)}$



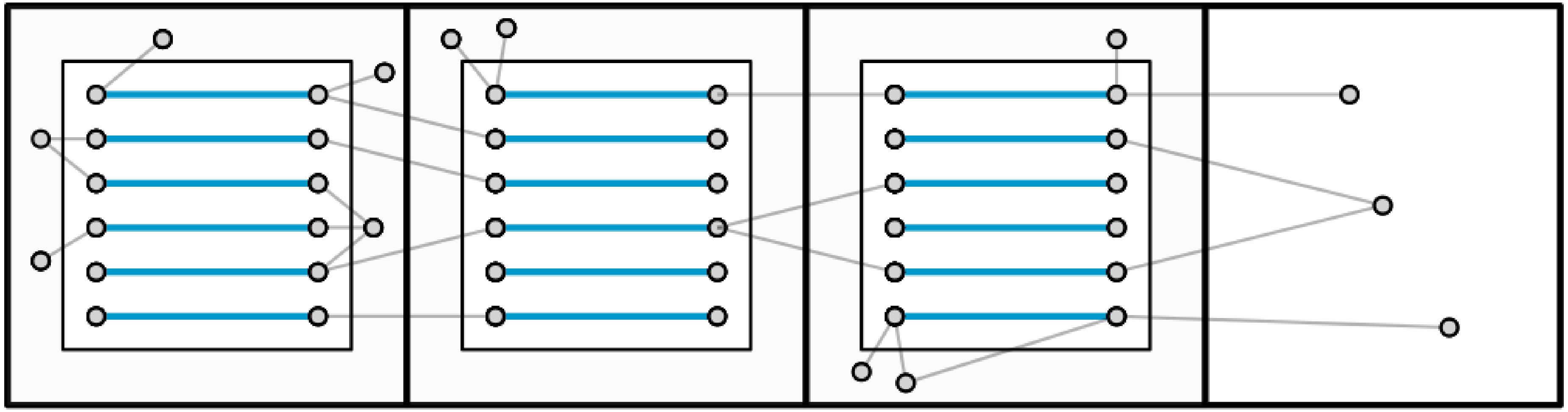
Maximal $O(\log^2 \Delta \cdot \log n)$



maximum matching size in remainder graph decreases by constant factor

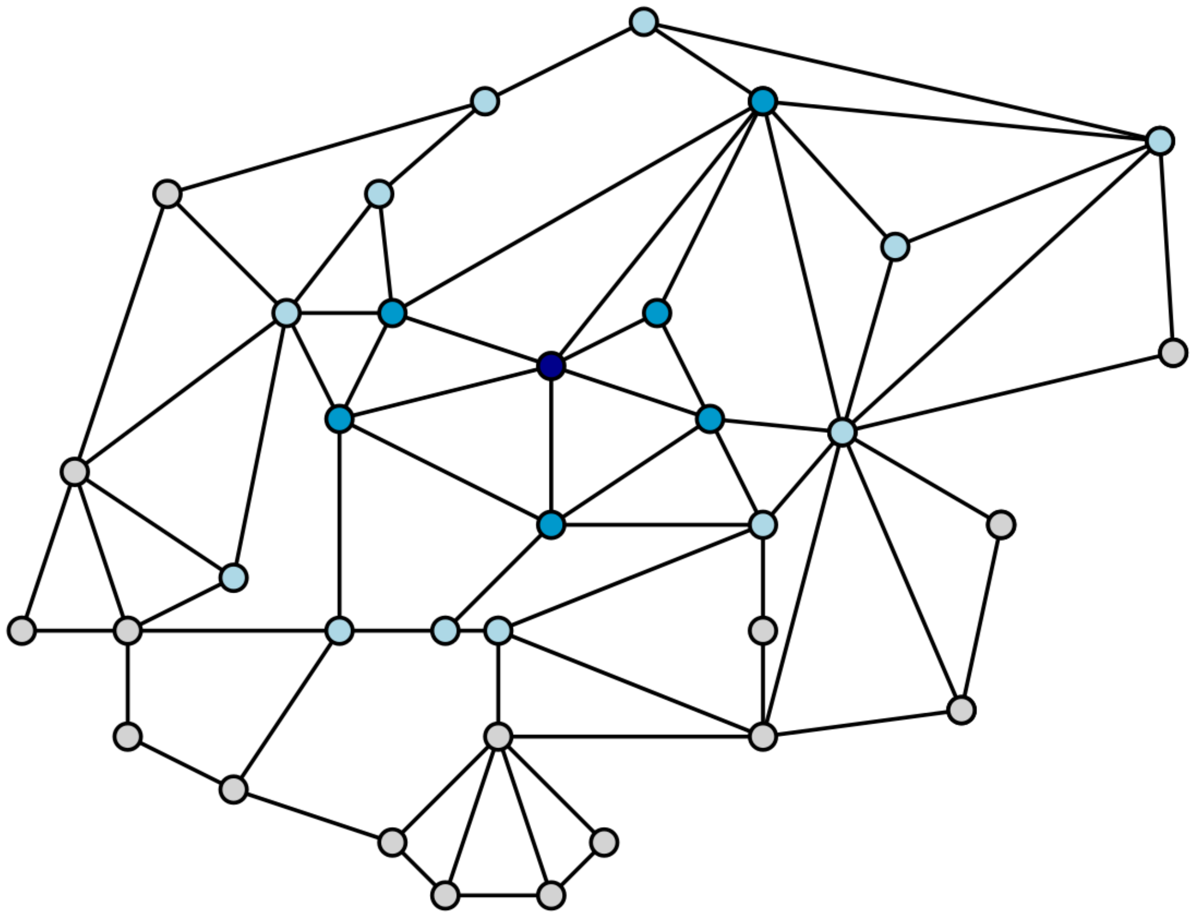
Constant - Approximate Matching $O(\log^2 \Delta)$ rounds

Maximal $O(\log^2 \Delta \cdot \log n)$



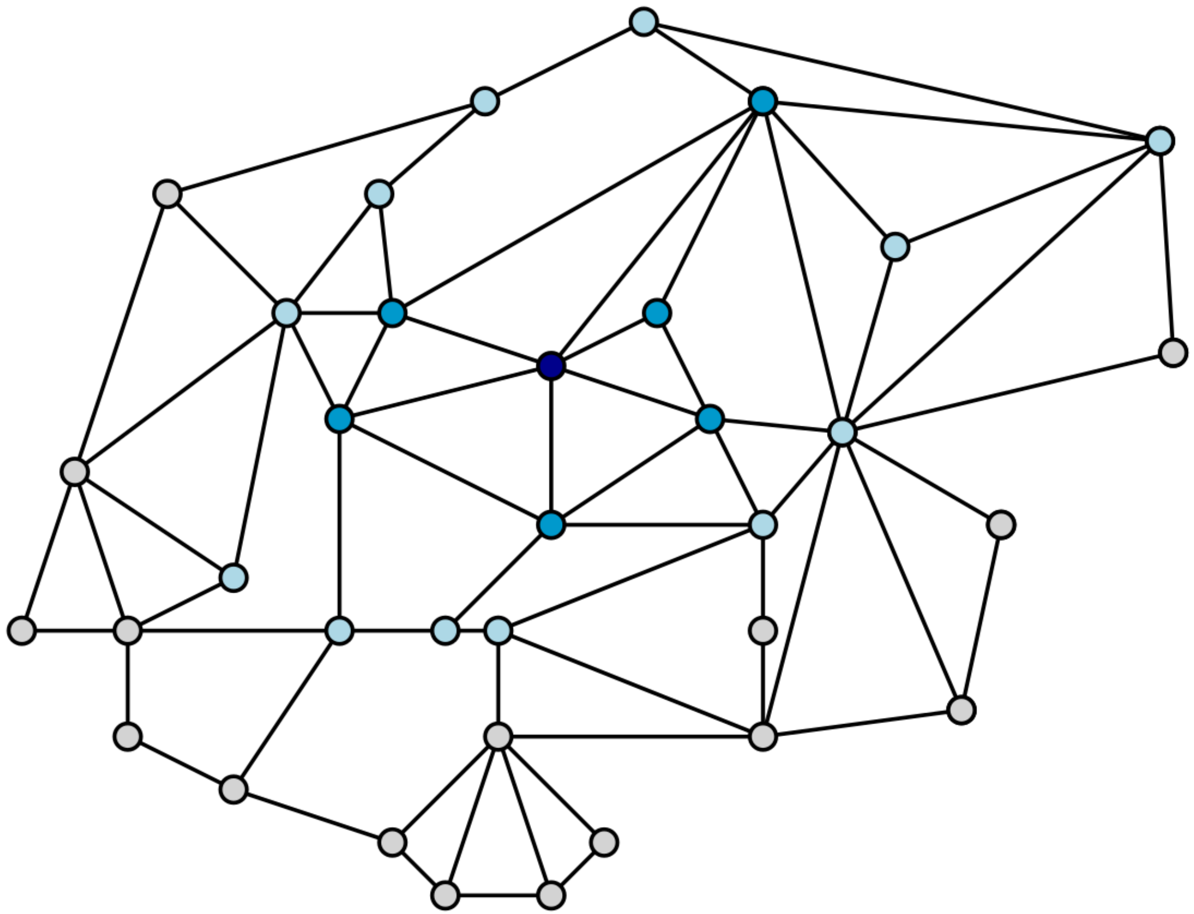
maximum matching size in remainder graph decreases by constant factor

after $O(\log n)$ iterations, maximum matching size is 0, hence graph empty



Open Question: $O(\log\Delta \cdot \log n)$?

What is Locality of Maximal Matching?



Thank you!

Open Question: $O(\log \Delta \cdot \log n)$?

What is Locality of Maximal Matching?