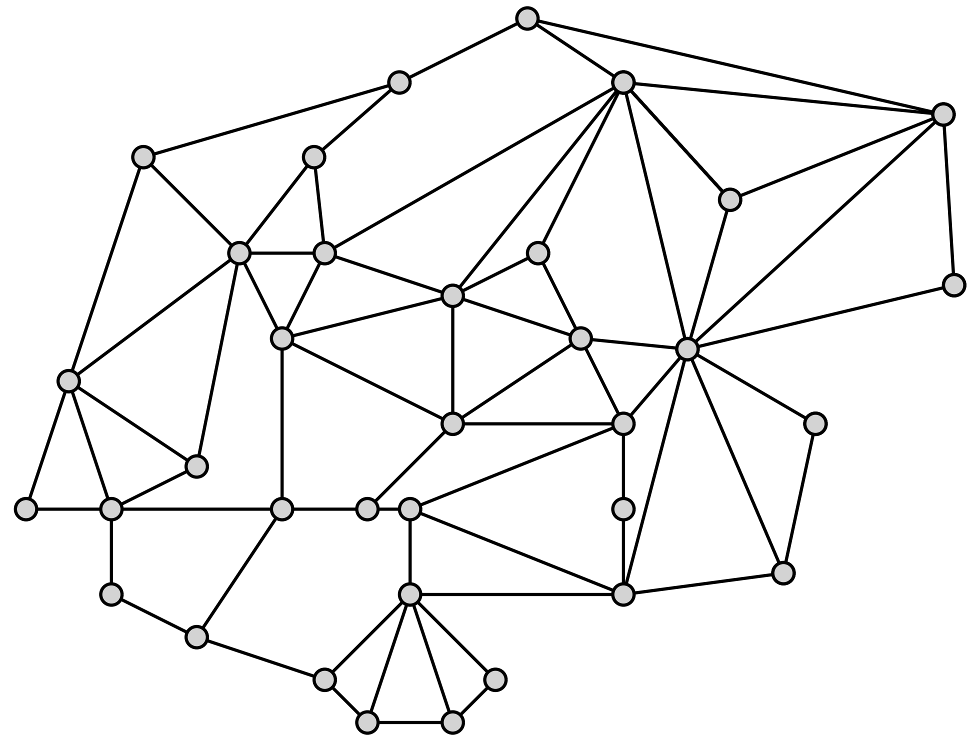
Deterministic Distributed Matching via Rounding

Manuela Fischer

joint work with Mohsen Ghaffari

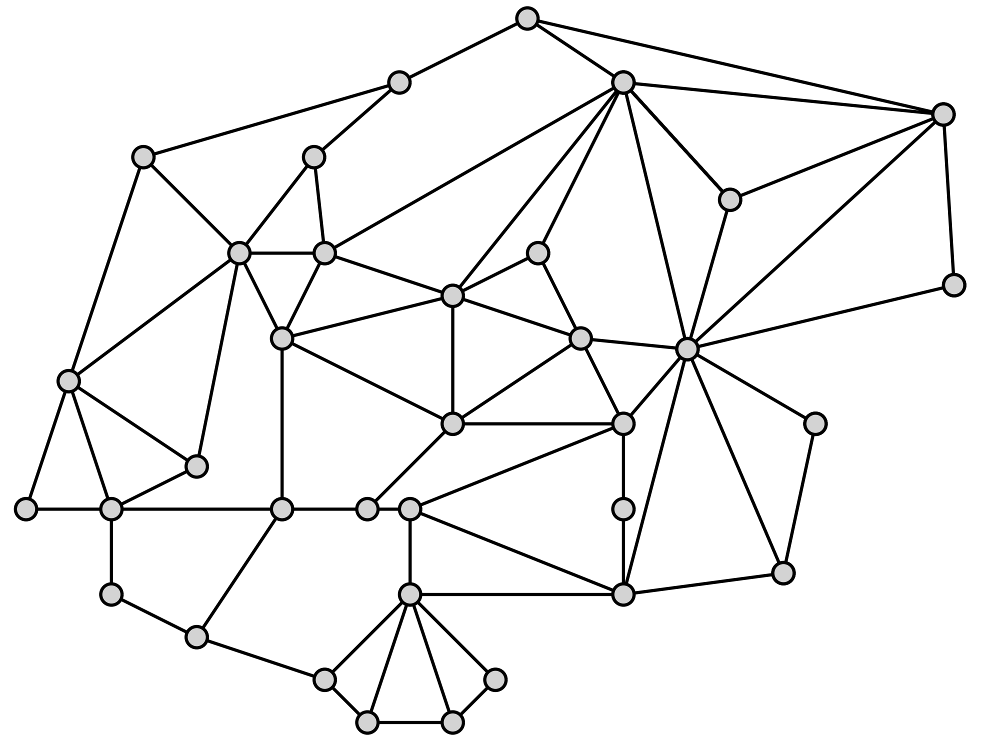
16.05.2017

Distributed LOCAL Model



Distributed LOCAL Model

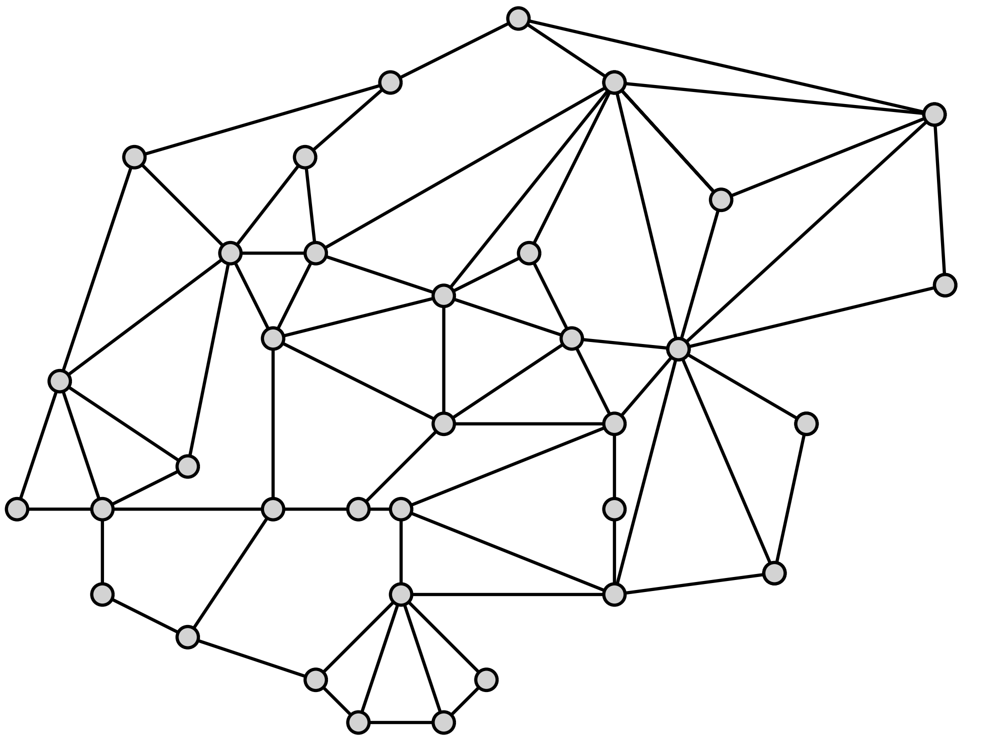
n nodes, maximum degree Δ
unique IDs



Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

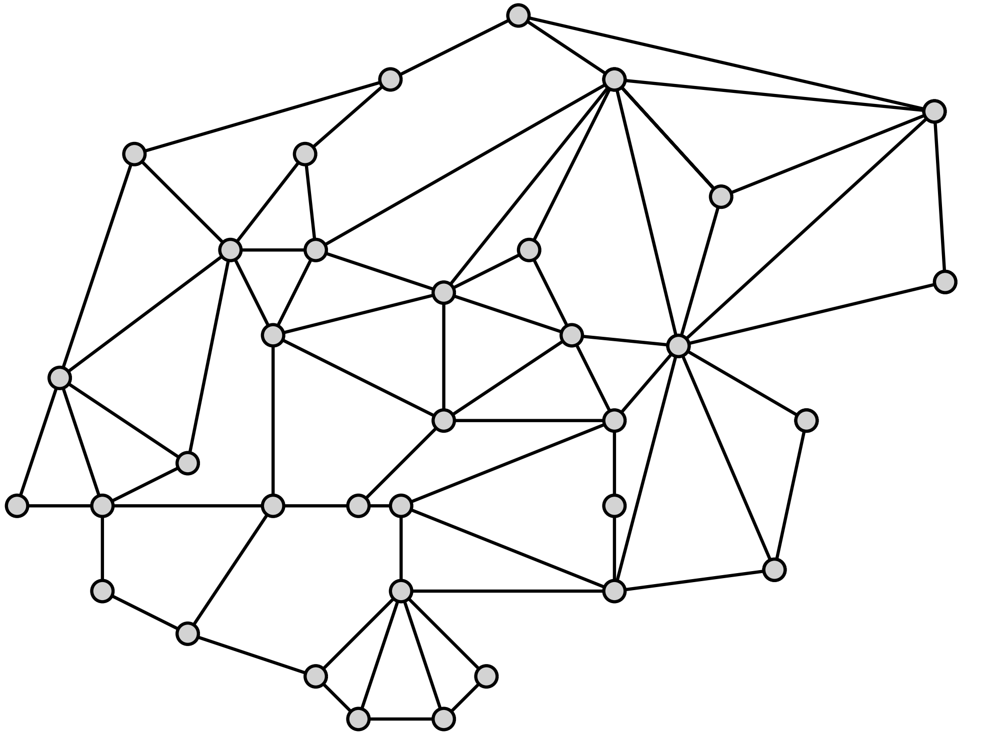
synchronous rounds



Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

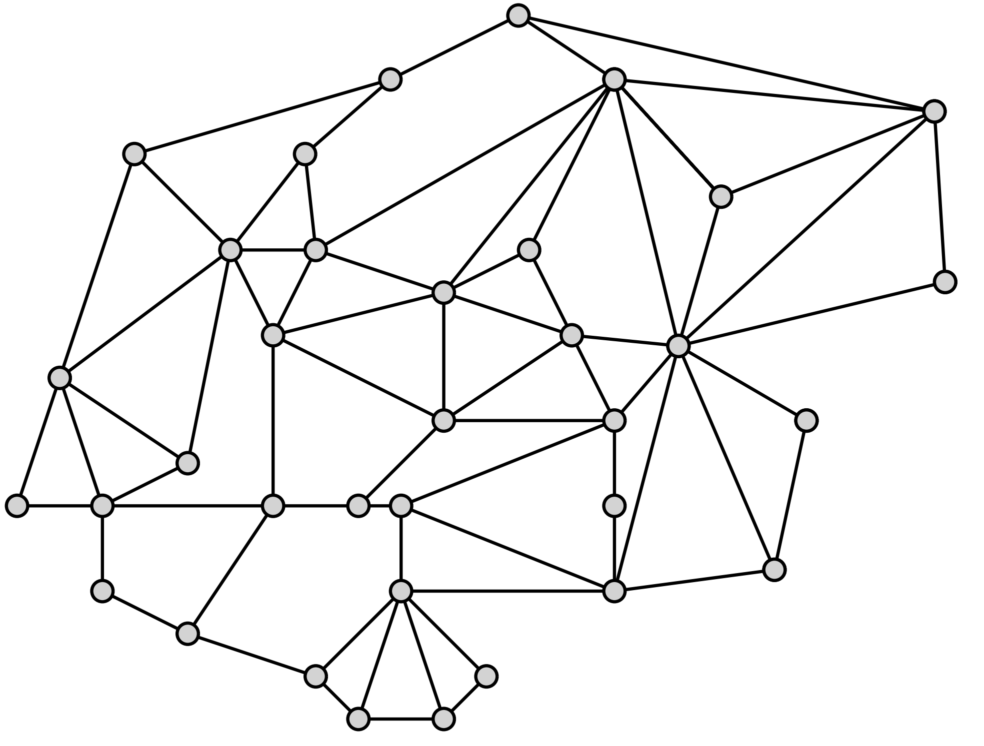
synchronous rounds
unbounded message size



Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

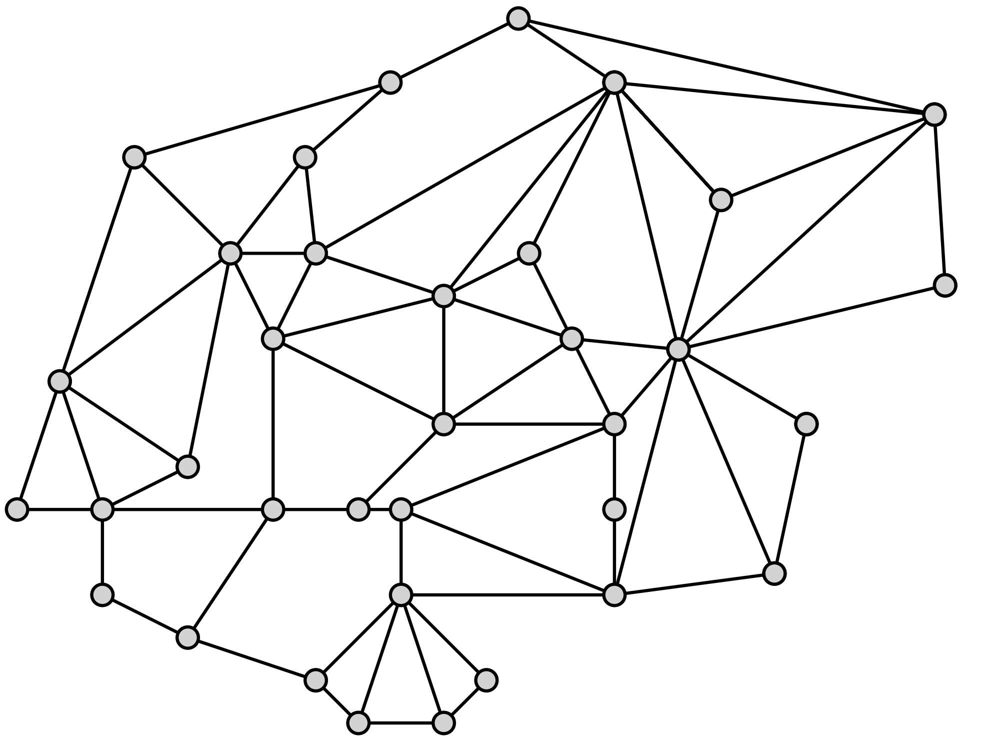
synchronous rounds
unbounded message size
unbounded computational power



Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

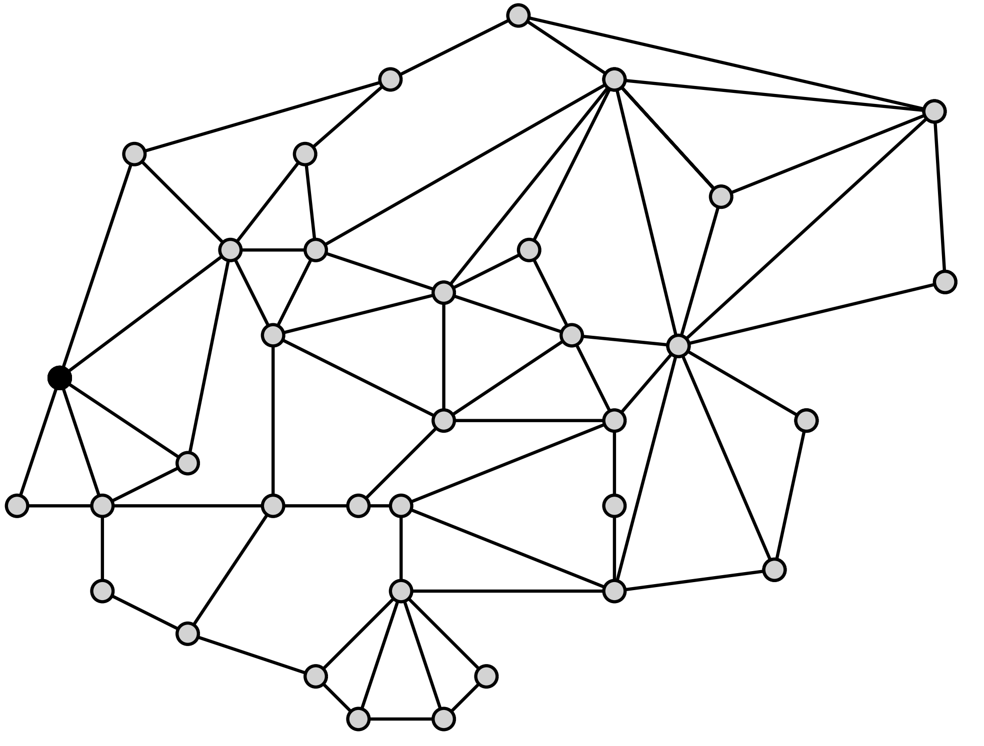


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

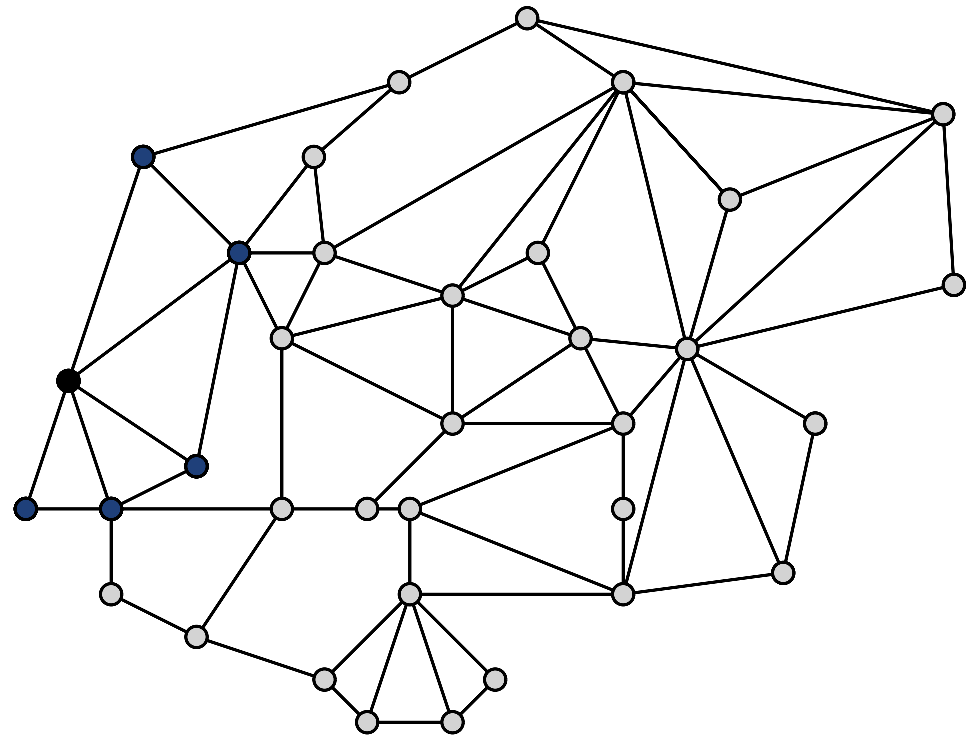


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

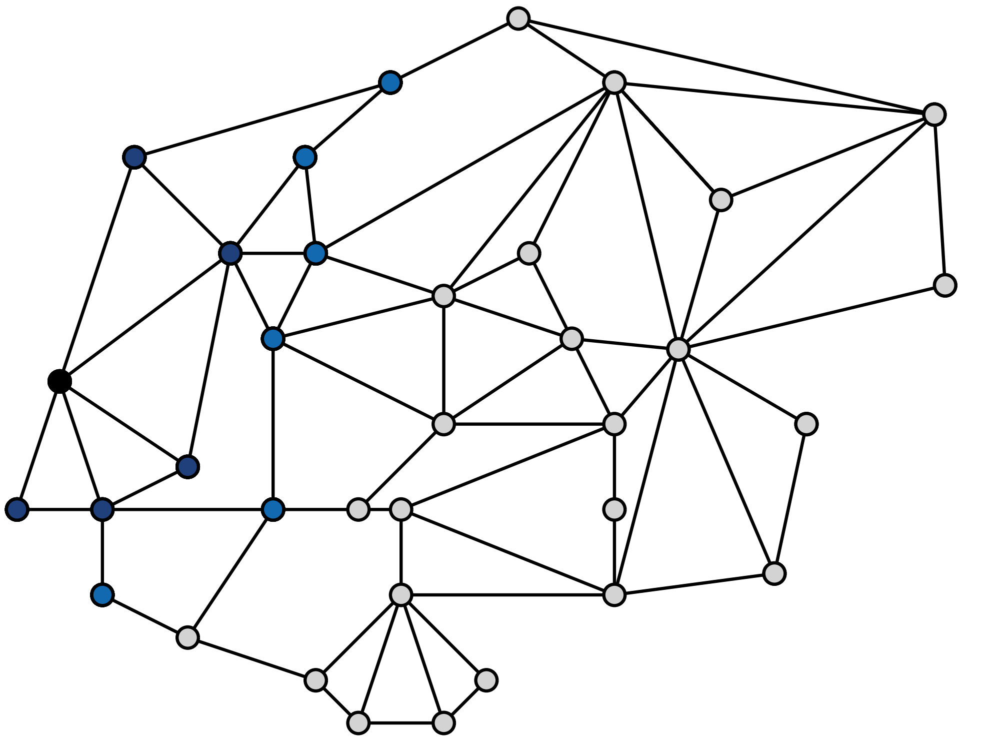


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

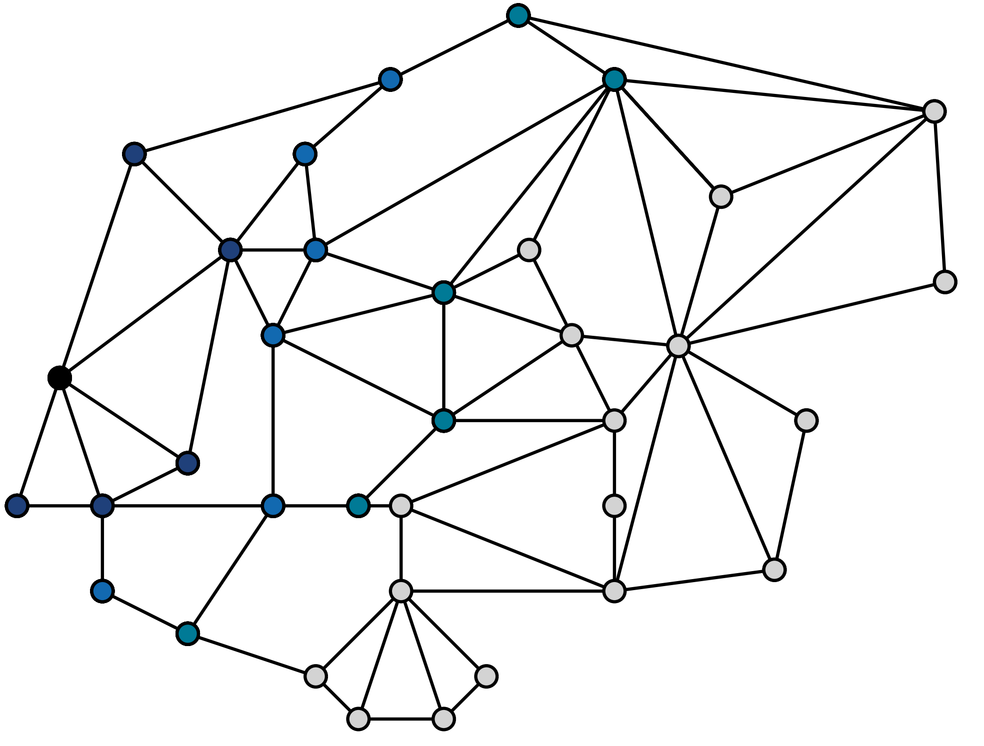


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

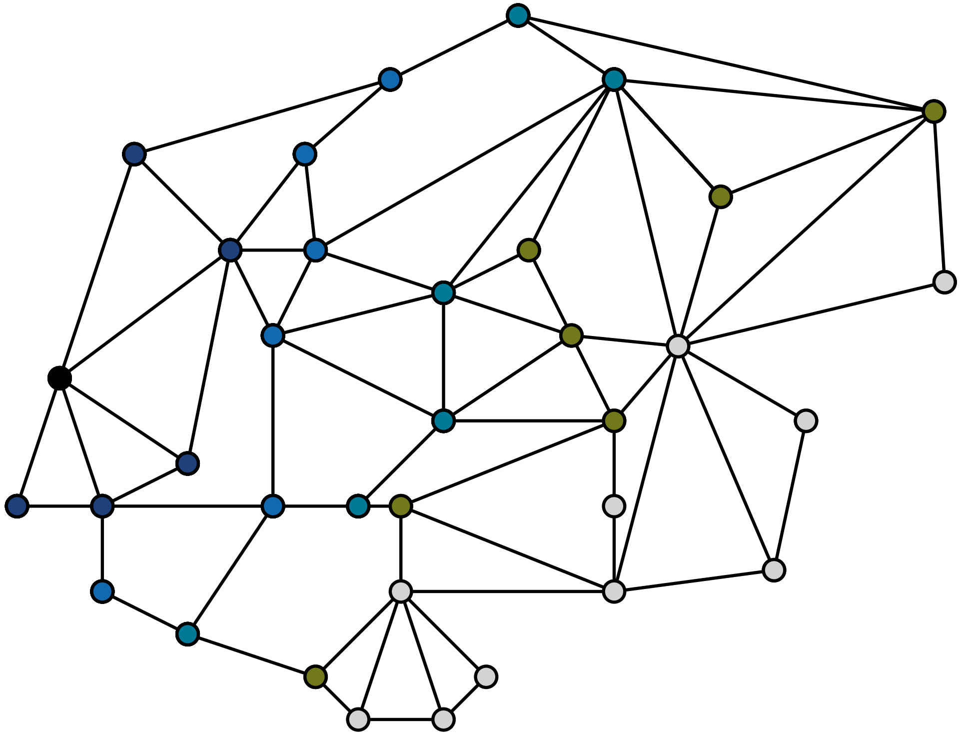


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

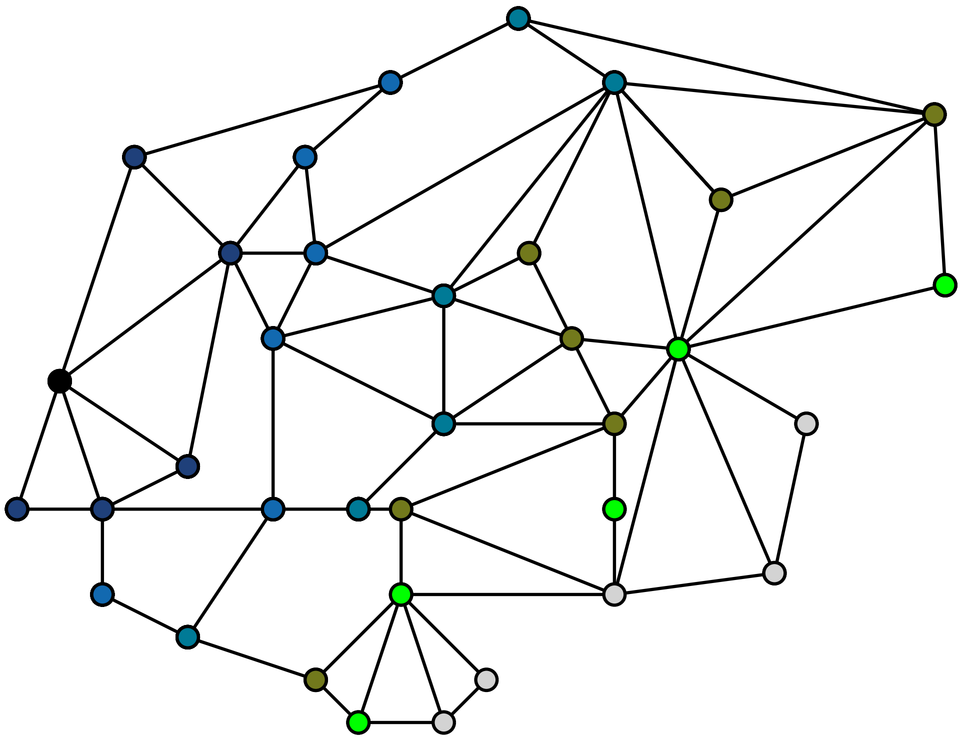


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

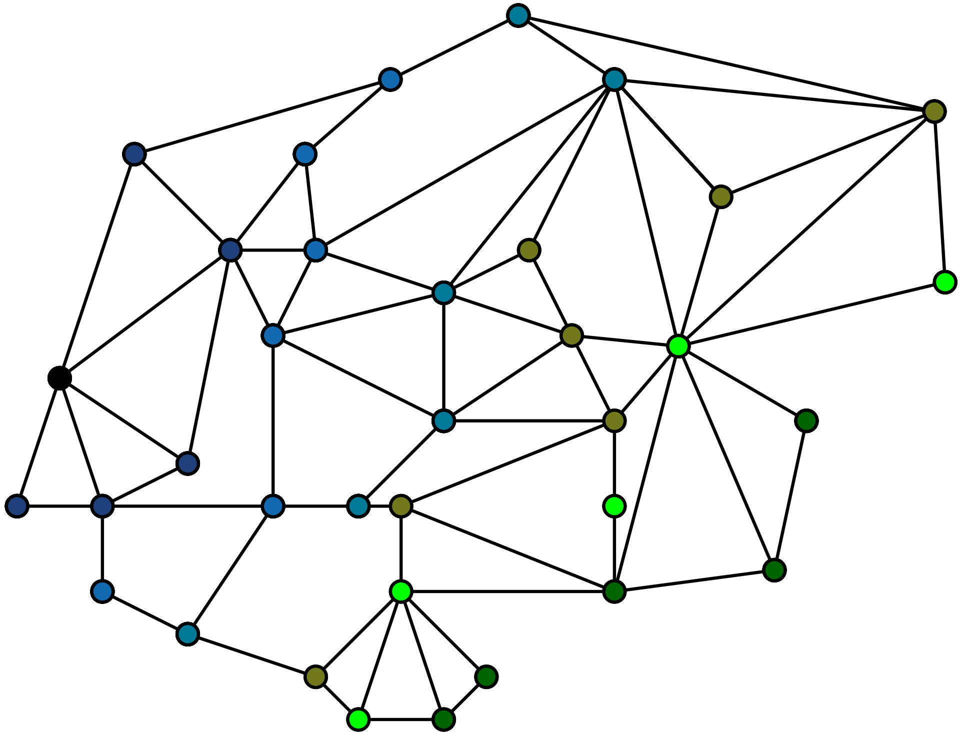


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power

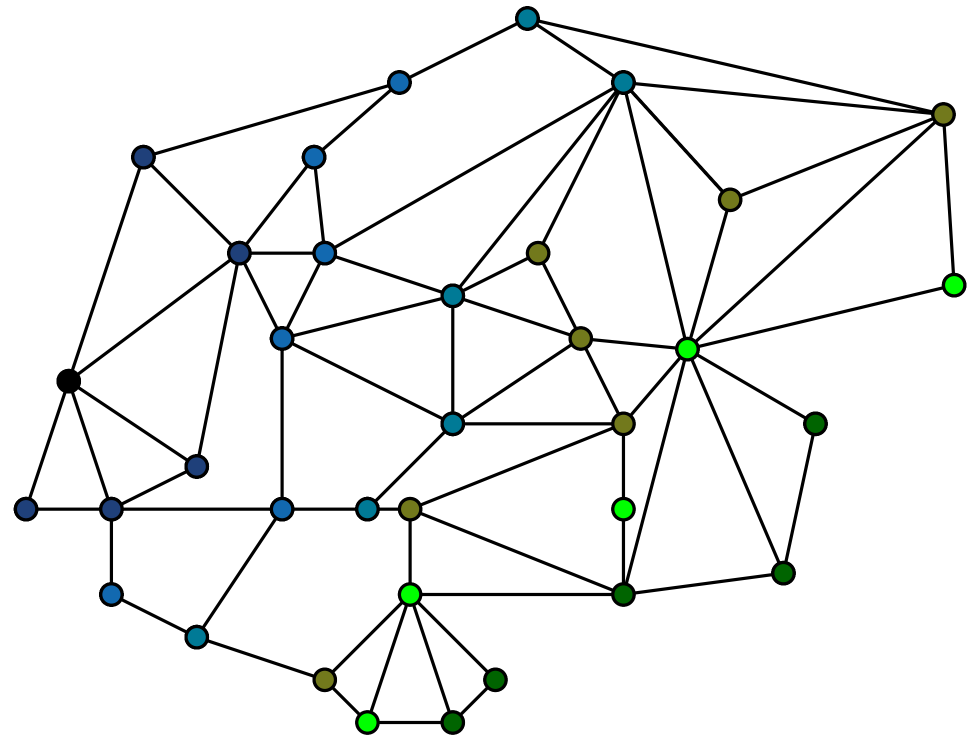


complexity = number of rounds = dependency radius

Distributed LOCAL Model

n nodes, maximum degree Δ
unique IDs

synchronous rounds
unbounded message size
unbounded computational power



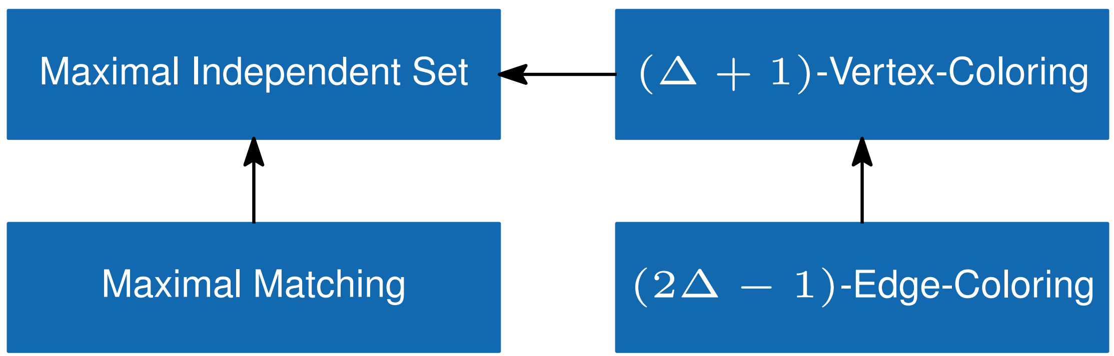
complexity = number of rounds = dependency radius

local problems: $o(D)$ complexity

Distributed LOCAL Model: Bigger Picture

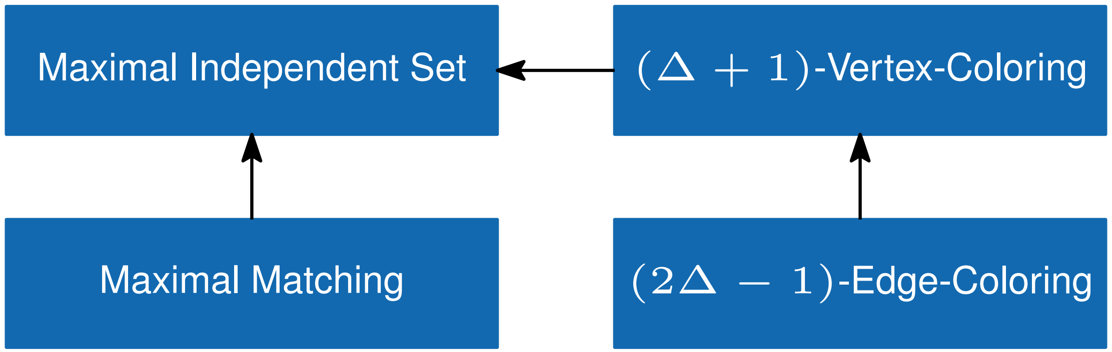
Distributed LOCAL Model: Bigger Picture

Four Classic LOCAL Problems (studied since 1980s)



Distributed LOCAL Model: Bigger Picture

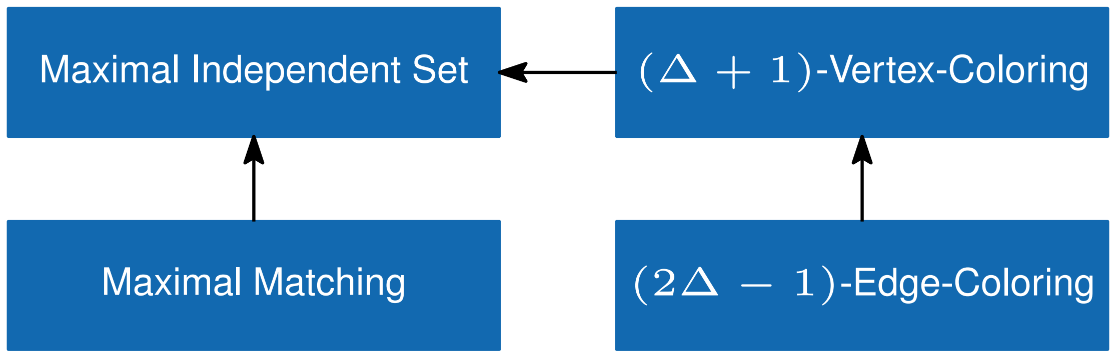
Four Classic LOCAL Problems (studied since 1980s)



goal: efficient deterministic algorithms for these problems

Distributed LOCAL Model: Bigger Picture

Four Classic LOCAL Problems (studied since 1980s)



goal: efficient deterministic algorithms for these problems

completeness of rounding (by Ghaffari, Kuhn, Maus [STOC'17]):

only obstacle for finding efficient deterministic distributed algorithms is

efficient deterministic distributed rounding method

Matching Approximation:
State of the Art & Our Result

Matching Approximation: State of the Art & Our Result

Fast deterministic $O(1)$-approximation

Matching Approximation: State of the Art & Our Result

Fast deterministic $O(1)$-approximation

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

Matching Approximation: State of the Art & Our Result

Fast deterministic $O(1)$-approximation

$O(\log^4 n)$	Hańćkowiak, Karoński, Panconesi [PODC'99]
$O(\Delta + \log^* n)$	Panconesi, Rizzi [DIST'01]

Matching Approximation: State of the Art & Our Result

Fast deterministic $O(1)$-approximation

$O(\log^4 n)$	Hańćkowiak, Karoński, Panconesi [PODC'99]
$O(\Delta + \log^* n)$	Panconesi, Rizzi [DIST'01]

Our Result [F., Ghaffari 2017]

$O(\log^2 \Delta + \log^* n)$	for constant approximation

Matching Approximation: State of the Art & Our Result

Fast deterministic $O(1)$-approximation

$O(\log^4 n)$	Hańćkowiak, Karoński, Panconesi [PODC'99]
$O(\Delta + \log^* n)$	Panconesi, Rizzi [DIST'01]
$\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$	Kuhn, Moscibroda, Wattenhofer [PODC'04], Linial [FOCS'87]

Our Result [F., Ghaffari 2017]

$O(\log^2 \Delta + \log^* n)$	for constant approximation

Matching Approximation: State of the Art & Our Result

Fast deterministic $O(1)$-approximation

$O(\log^4 n)$	Hańćkowiak, Karoński, Panconesi [PODC'99]
$O(\Delta + \log^* n)$	Panconesi, Rizzi [DIST'01]
$\Omega\left(\frac{\log \Delta}{\log\log \Delta} + \log^* n\right)$	Kuhn, Moscibroda, Wattenhofer [PODC'04], Linial [FOCS'87]

Our Result [F., Ghaffari 2017]

$O(\log^2 \Delta + \log^* n)$	for constant approximation
$O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n)$	for $(2+\varepsilon)$-approximation

Matching Approximation: State of the Art & Our Result

Fast deterministic $O(1)$-approximation

$O(\log^4 n)$	Hańćkowiak, Karoński, Panconesi [PODC'99]
$O(\Delta + \log^* n)$	Panconesi, Rizzi [DIST'01]
$\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$	Kuhn, Moscibroda, Wattenhofer [PODC'04], Linial [FOCS'87]

Our Result [F., Ghaffari 2017]

$O(\log^2 \Delta + \log^* n)$	for constant approximation
$O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n)$	for $(2+\varepsilon)$-approximation
$O(\log^2 \Delta \cdot \log n)$	for maximal matching (2-approximation)

Further Extensions & Corollaries of Our Result

$O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon})$	for $(2+\varepsilon)$-approximate weighted matching
$O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon})$	for $(2+\varepsilon)$-approximate (weighted) b-matching
$O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon})$	for an ε-almost maximal matching
$O(\log^2 \Delta \cdot \log \frac{\Delta}{\varepsilon})$	for $(2+\varepsilon)$-approximate minimum edge dominating set

Intuitive Idea

(1) What is simple case?

(2) How to reduce to it?

(1) Simple Case: Bipartite Bounded-Degree Graphs

(1) Simple Case:
Bipartite Bounded-Degree Graphs

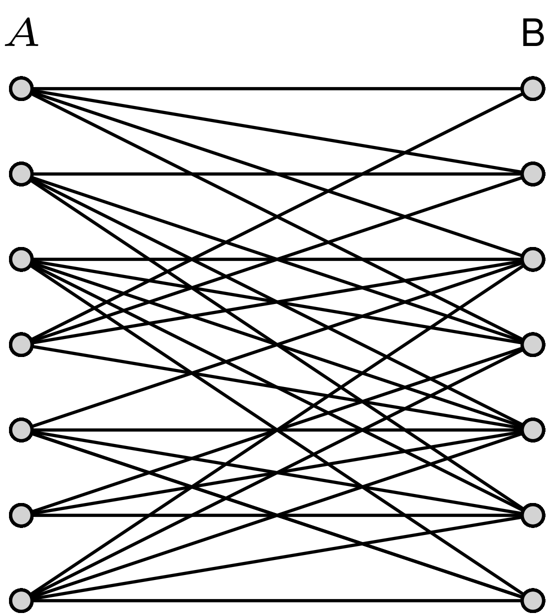
Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

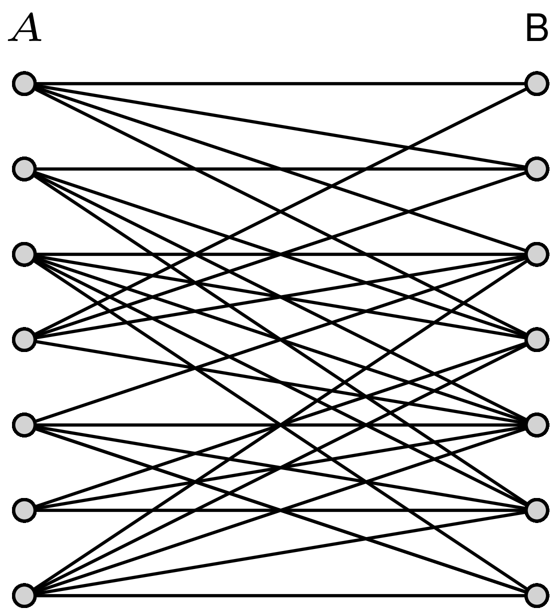
In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.

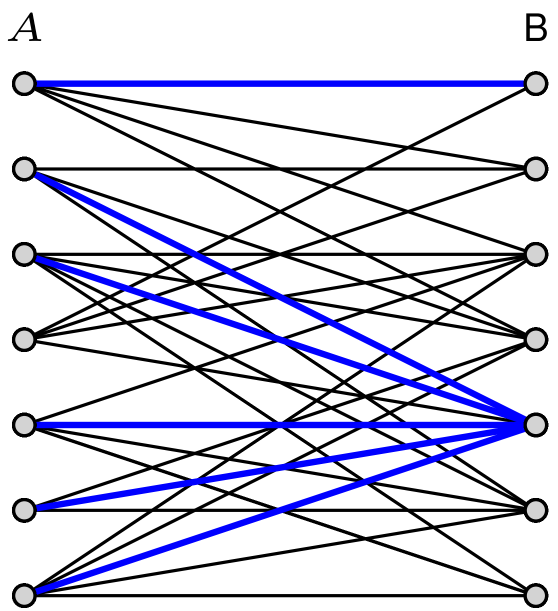


Proposing Algorithm

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



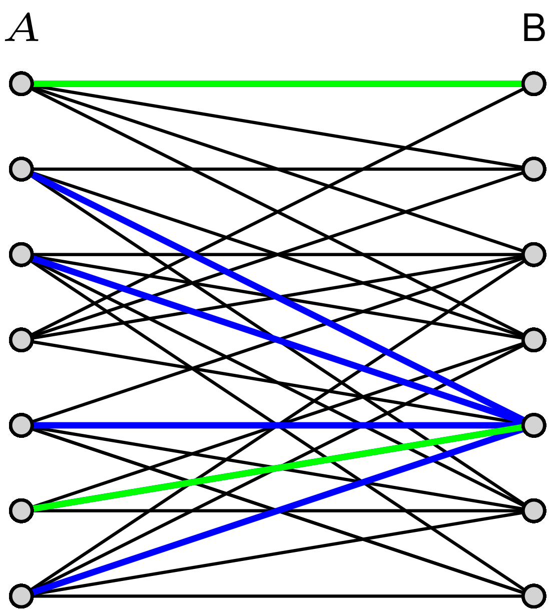
Proposing Algorithm

A-nodes propose to an arbitrary incident edge

(1) Simple Case: Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

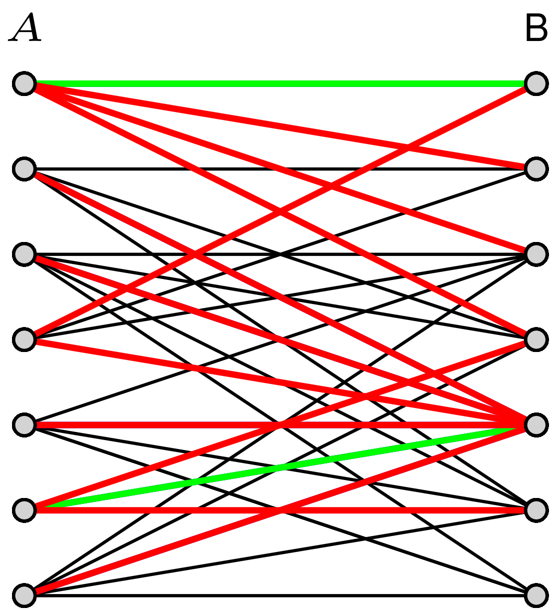
A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

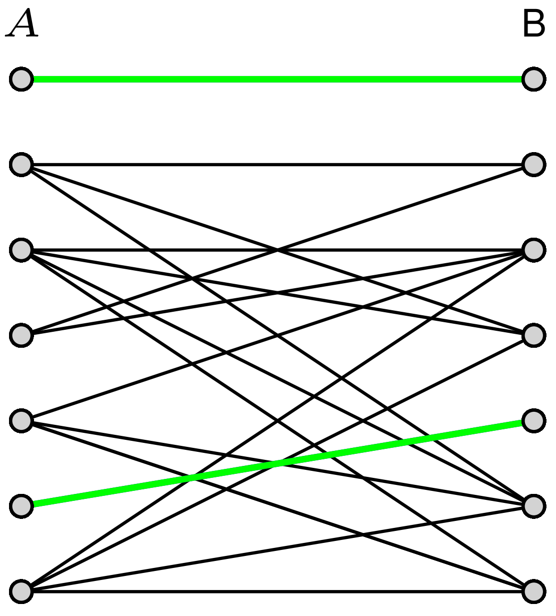
B-nodes match arbitrary proposed edge (if any)

remove all conflicting edges

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

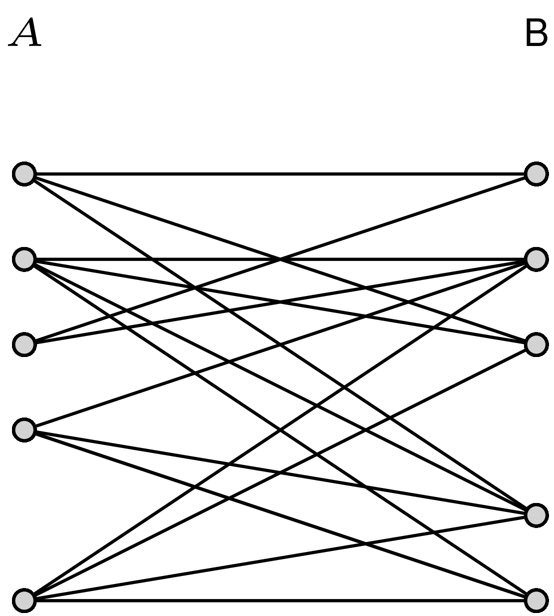
remove all conflicting edges

Manuela Fischer

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

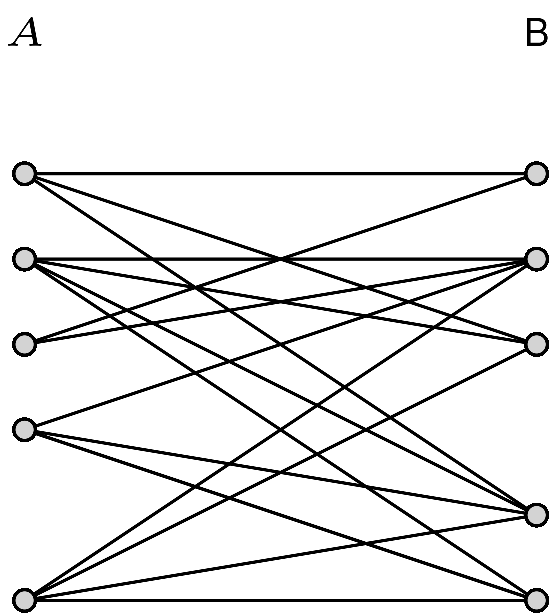
B-nodes match arbitrary proposed edge (if any)

remove all conflicting edges

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

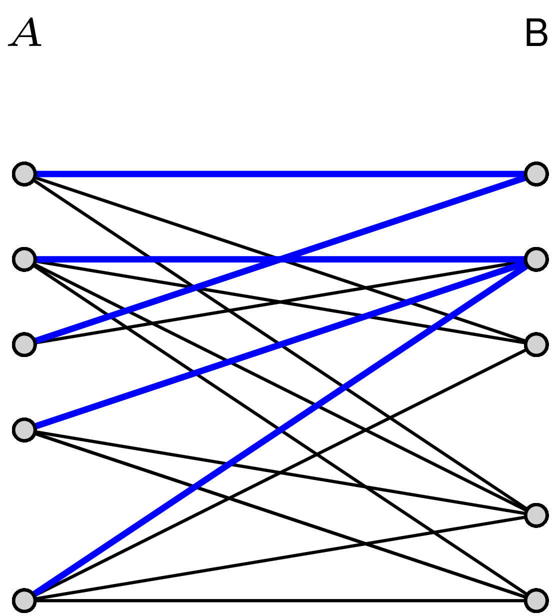
remove all conflicting edges

repeat on remainder graph

(1) Simple Case: Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

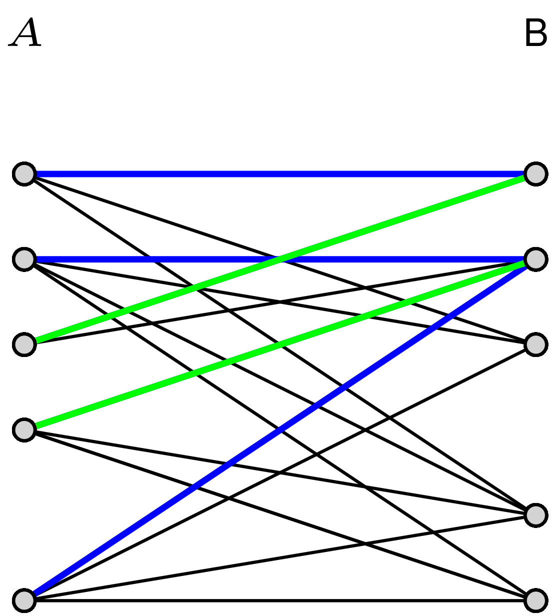
remove all conflicting edges

repeat on remainder graph

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

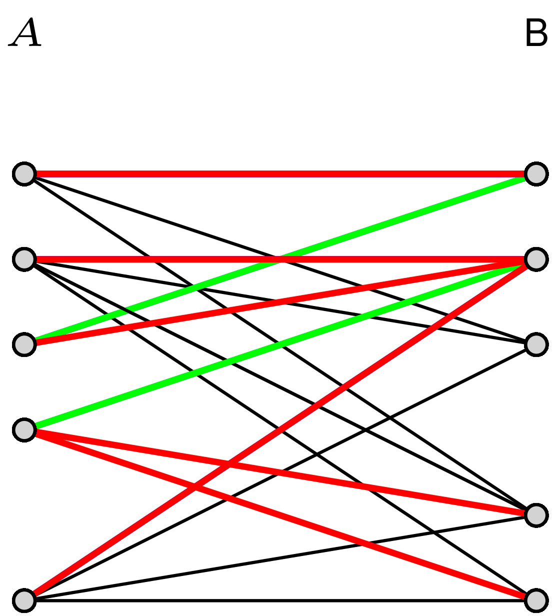
remove all conflicting edges

repeat on remainder graph

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

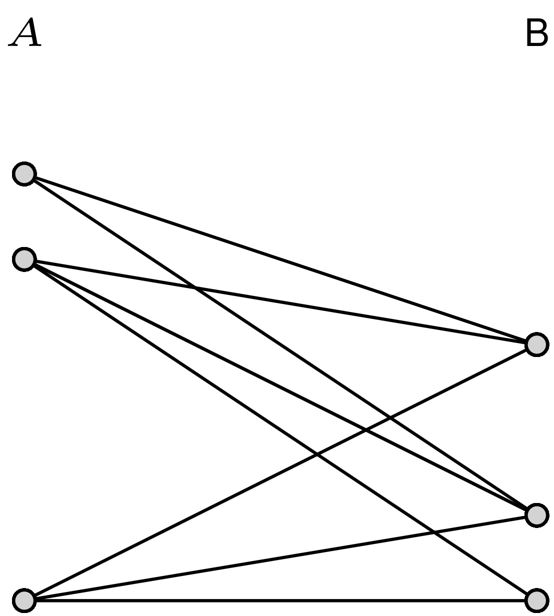
remove all conflicting edges

repeat on remainder graph

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

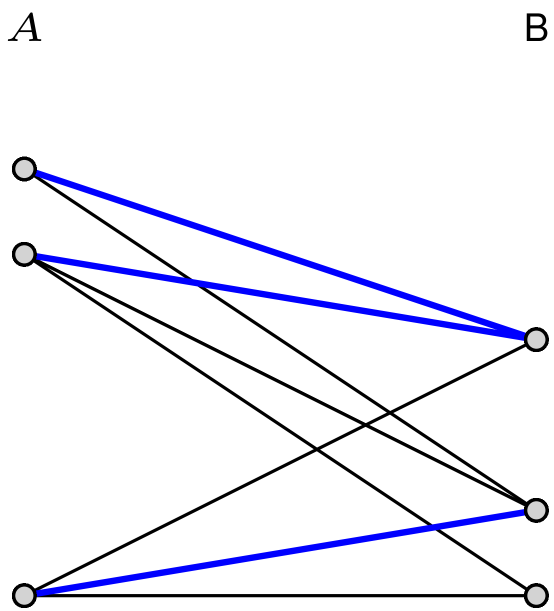
remove all conflicting edges

repeat on remainder graph

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

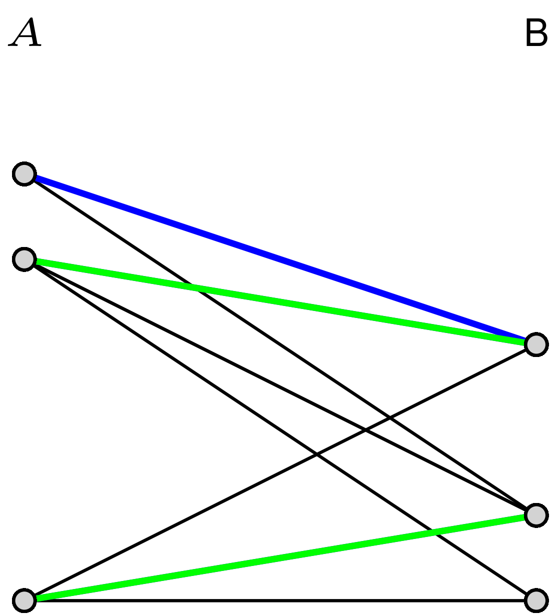
remove all conflicting edges

repeat on remainder graph

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

remove all conflicting edges

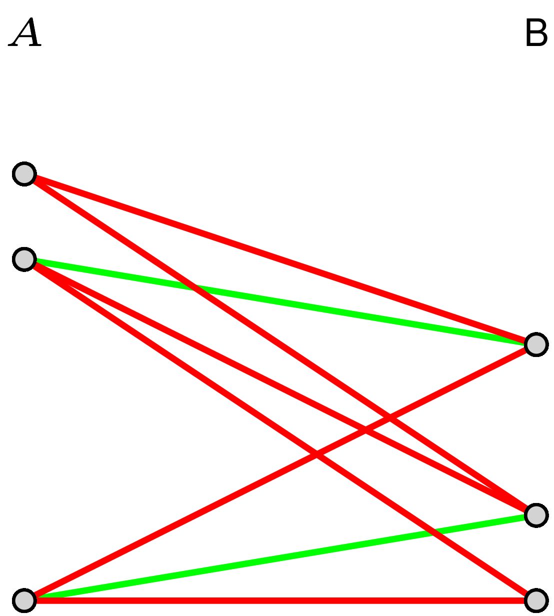
repeat on remainder graph

Manuela Fischer

(1) Simple Case: Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

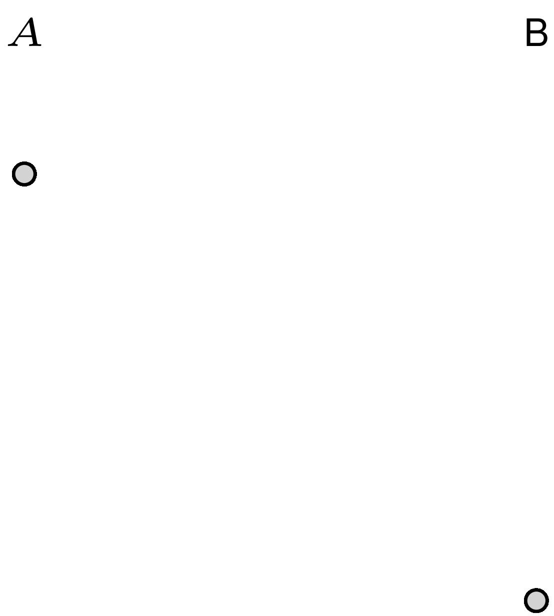
remove all conflicting edges

repeat on remainder graph

(1) Simple Case:
Bipartite Bounded-Degree Graphs

Lemma:

In bipartite graphs,
a maximal matching can be found in $O(\Delta)$ rounds.



Proposing Algorithm

A-nodes propose to an arbitrary incident edge

B-nodes match arbitrary proposed edge (if any)

remove all conflicting edges

repeat on remainder graph

(2) How to reduce to simple case?

Iterative Degree Reduction

(2) How to reduce to simple case? Iterative Degree Reduction

iteratively decrease degree (by deleting edges)

(2) How to reduce to simple case?
Iterative Degree Reduction

iteratively decrease degree (by deleting edges)

while not changing maximum matching size by too much

(2) How to reduce to simple case? Iterative Degree Reduction

iteratively decrease degree (by deleting edges)

while not changing maximum matching size by too much

until maximum degree $= O(1)$

(2) How to reduce to simple case?
Iterative Degree Reduction

iteratively decrease degree (by deleting edges)

by a constant factor

while not changing maximum matching size by too much

until maximum degree $= O(1)$

(2) How to reduce to simple case? Iterative Degree Reduction

iteratively decrease degree (by deleting edges)

by a constant factor

while not changing maximum matching size by too much

by a factor r_i in iteration i

until maximum degree $= O(1)$

(2) How to reduce to simple case? Iterative Degree Reduction

iteratively decrease degree (by deleting edges)

by a constant factor

while not changing maximum matching size by too much

by a factor r_i in iteration i

until maximum degree $= O(1)$

for $O(\log \Delta)$ iterations

(2) How to reduce to simple case? Iterative Degree Reduction

iteratively decrease degree (by deleting edges)

by a constant factor

while not changing maximum matching size by too much

by a factor r_i in iteration i

until maximum degree $= O(1)$

for $O(\log \Delta)$ iterations

$O(\log \Delta)$-round $O(\prod_i r_i)$-approximation

Our Method:

Matching Approximation via Rounding LPs

Fractional Matching

Fractional Matching

(Integral) Matching as Integer Program

Fractional Matching

(Integral) Matching as Integer Program

$x_e \in \{0, 1\}$ for all $e \in E$

Fractional Matching

(Integral) Matching as Integer Program

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

$x_e \in \{0, 1\}$ for all $e \in E$

Fractional Matching

(Integral) Matching as Integer Program

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

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

$x_e \in \{0, 1\}$ for all $e \in E$

Fractional Matching

Fractional Matching as Linear Program

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

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

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

Fractional Matching

Fractional Matching as Linear Program

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

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

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

why useful?

Fractional Matching

Fractional Matching as Linear Program

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

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

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

why useful?

instead of just setting half of edges to 0 and half to 1,

pursue a more gradual approach with fractional matchings

Fractional Matching

Fractional Matching as Linear Program

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

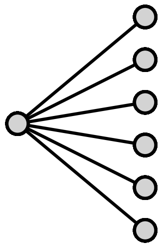
s.t. $\sum_{e \in E(v)} x_e \leq 1$ for all $v \in V$

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

why useful?

instead of just setting half of edges to 0 and half to 1,

pursue a more gradual approach with fractional matchings



Fractional Matching

Fractional Matching as Linear Program

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

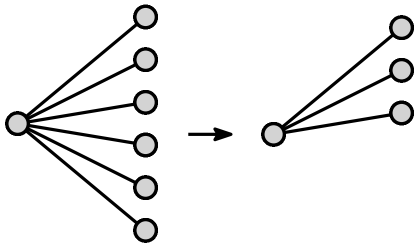
$$\text{s.t.} \sum_{e \in E(v)} x_e \leq 1 \text{ for all } v \in V$$

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

why useful?

instead of just setting half of edges to 0 and half to 1,

pursue a more gradual approach with fractional matchings



Fractional Matching

Fractional Matching as Linear Program

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

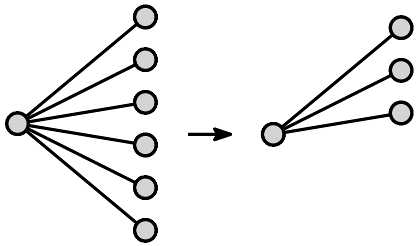
s.t. $\sum_{e \in E(v)} x_e \leq 1$ for all $v \in V$

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

why useful?

instead of just setting half of edges to 0 and half to 1,

pursue a more gradual approach with fractional matchings



lose $\approx$ 2-factor in value

Fractional Matching

Fractional Matching as Linear Program

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

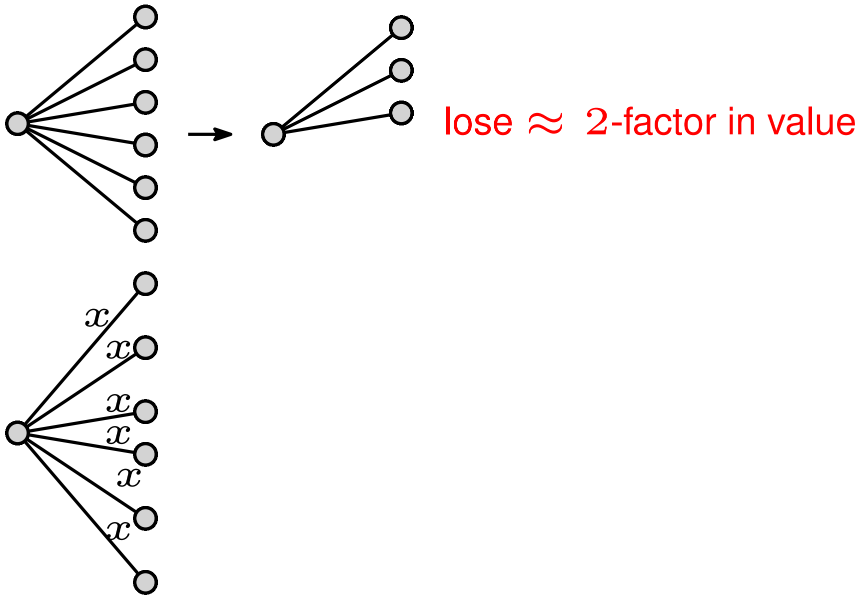
s.t. $\sum_{e \in E(v)} x_e \leq 1$ for all $v \in V$

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

why useful?

instead of just setting half of edges to 0 and half to 1,

pursue a more gradual approach with fractional matchings



Fractional Matching

Fractional Matching as Linear Program

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

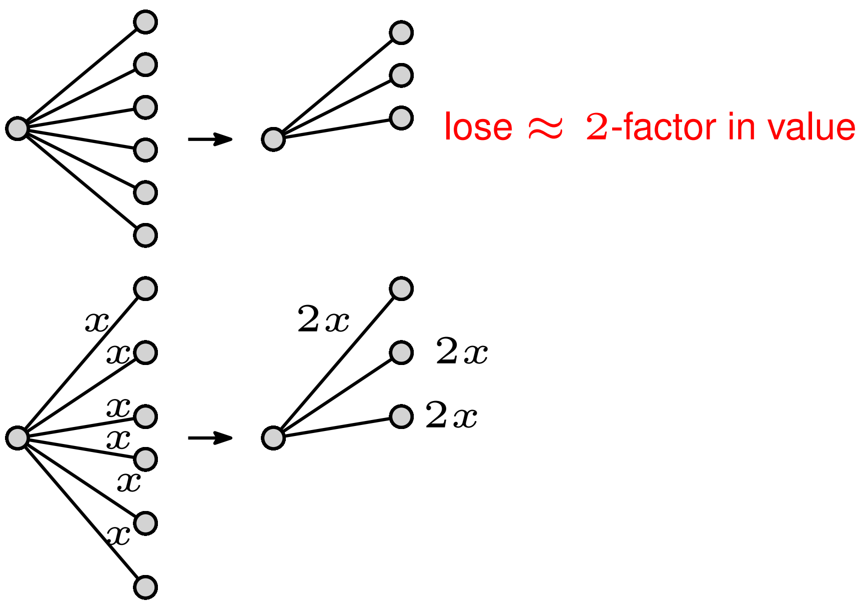
s.t. $\sum_{e \in E(v)} x_e \leq 1$ for all $v \in V$

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

why useful?

instead of just setting half of edges to 0 and half to 1,

pursue a more gradual approach with fractional matchings



Fractional Matching

Fractional Matching as Linear Program

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

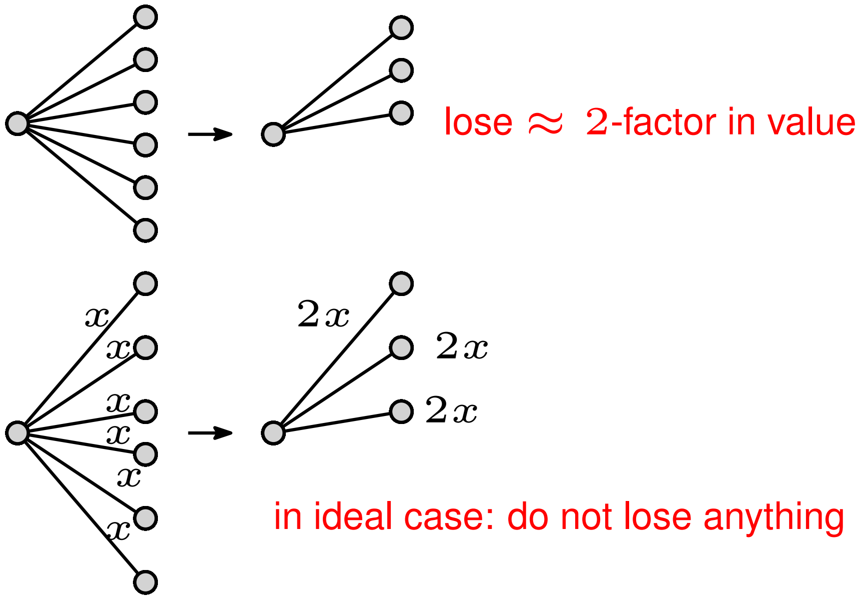
s.t. $\sum_{e \in E(v)} x_e \leq 1$ for all $v \in V$

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

why useful?

instead of just setting half of edges to 0 and half to 1,

pursue a more gradual approach with fractional matchings



Our Matching Approximation Algorithm: Outline

Our Matching Approximation Algorithm: Outline

(1) Fractional matching

Our Matching Approximation Algorithm: Outline

(1) Fractional matching

(2) Iterative rounding

Our Matching Approximation Algorithm: Outline

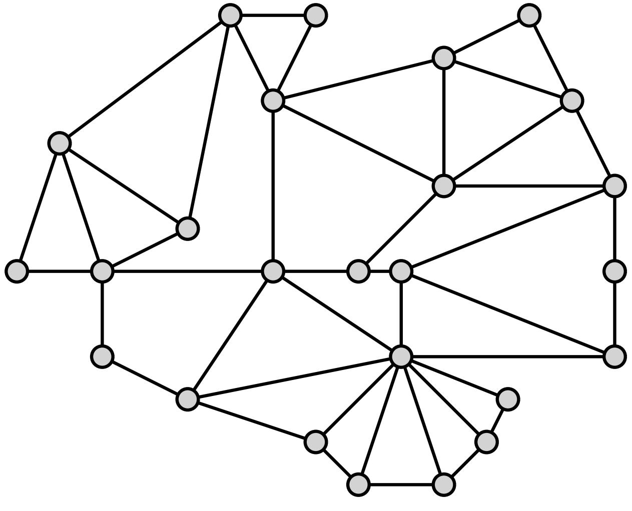
(1) Fractional matching

(2) Iterative rounding

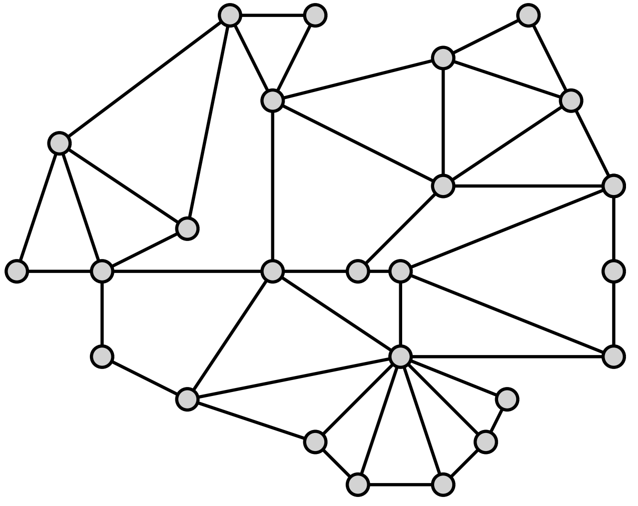
(3) Final Rounding

(1) Find Fractional Matching

(1) Find Fractional Matching

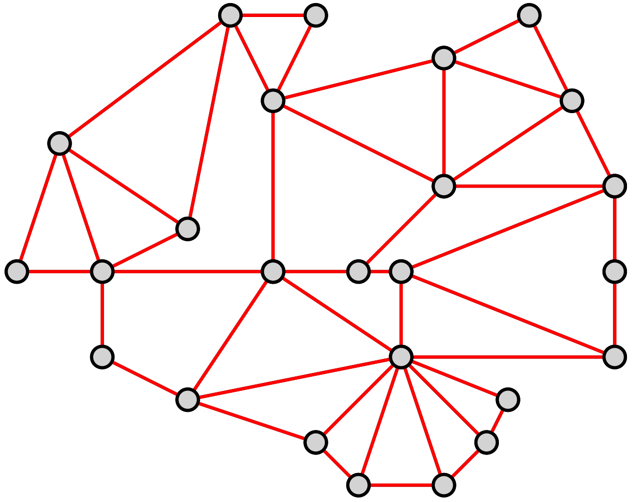


(1) Find Fractional Matching



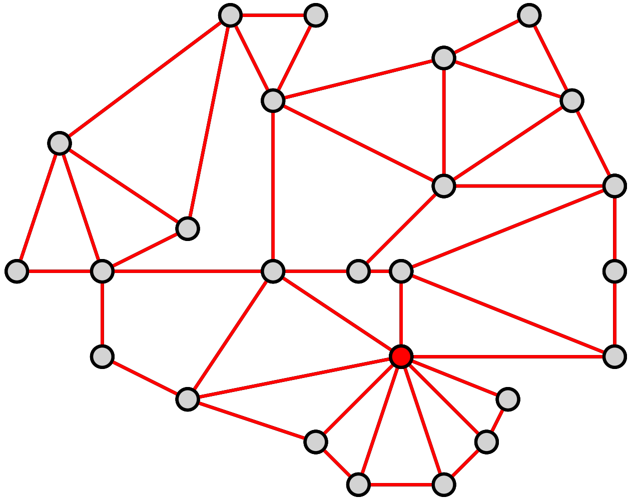
start with $x_e = 2^{-\lceil \log \Delta \rceil}$

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

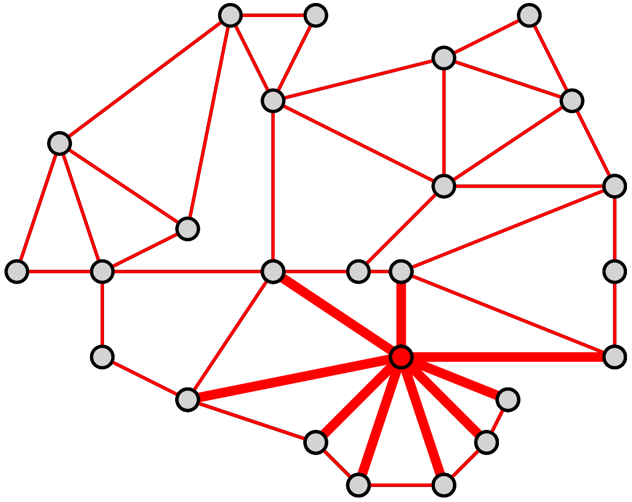
(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

mark $\frac{1}{2}$-tight nodes

(1) Find Fractional Matching

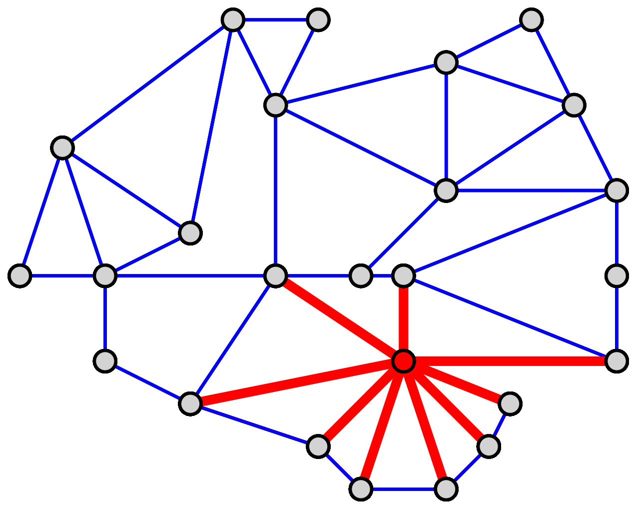


start with $x_e = 2^{-\lceil \log \Delta \rceil}$

mark $\frac{1}{2}$-tight nodes

block its edges

(1) Find Fractional Matching



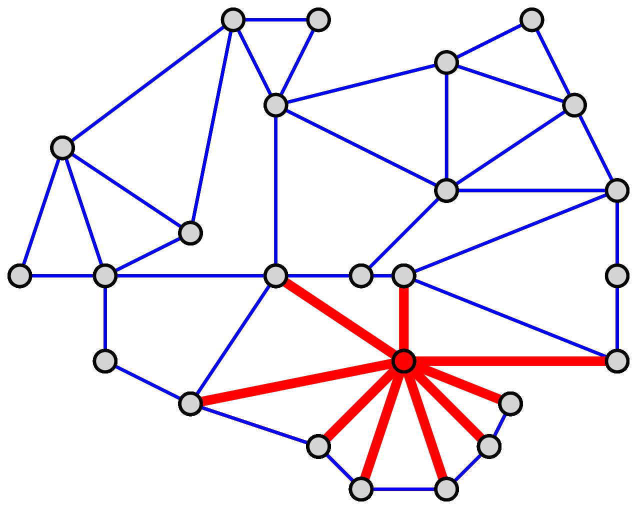
start with $x_e = 2^{-\lceil \log \Delta \rceil}$

mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

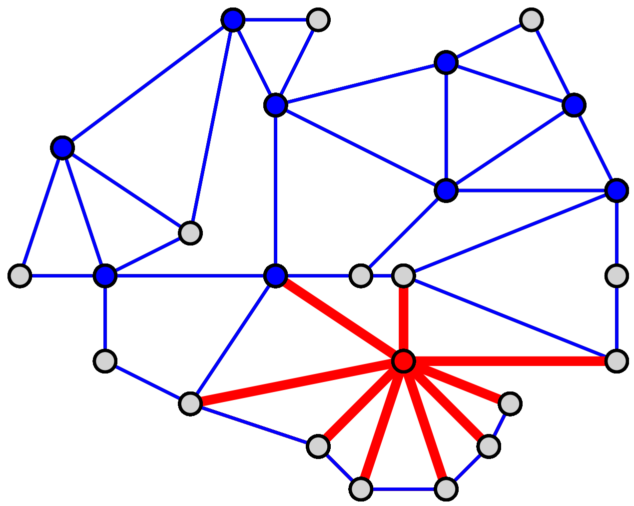
repeat

mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

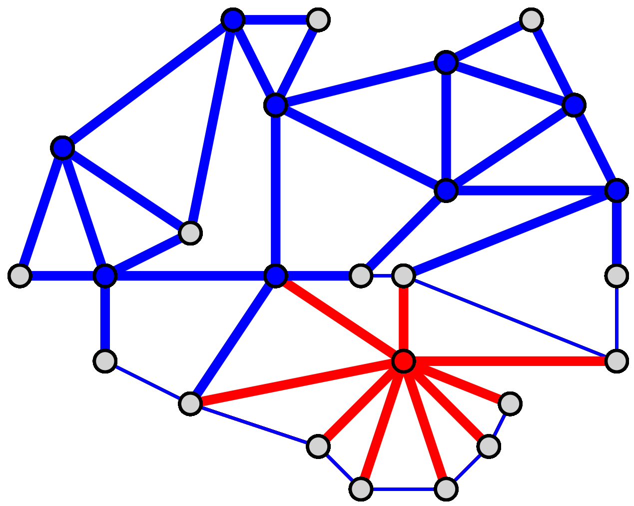
repeat

mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

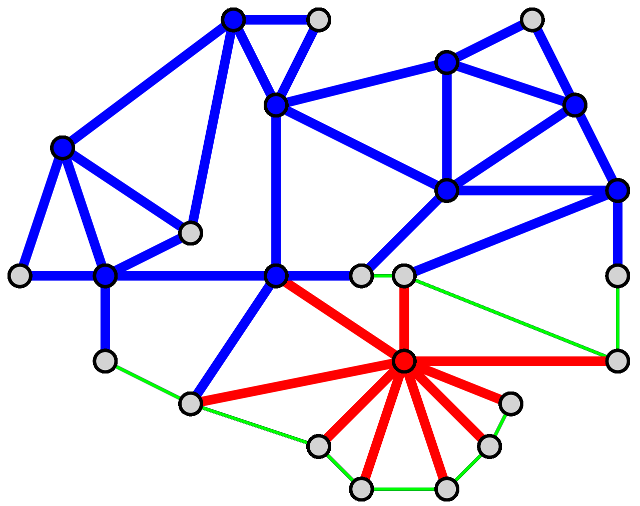
repeat

mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

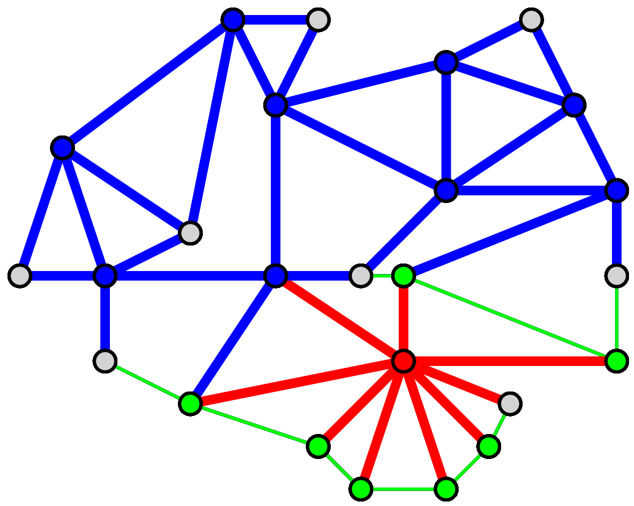
repeat

mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

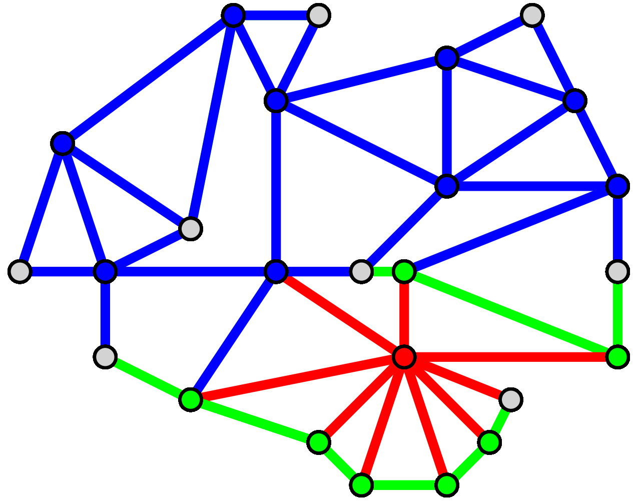
repeat

mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

(1) Find Fractional Matching

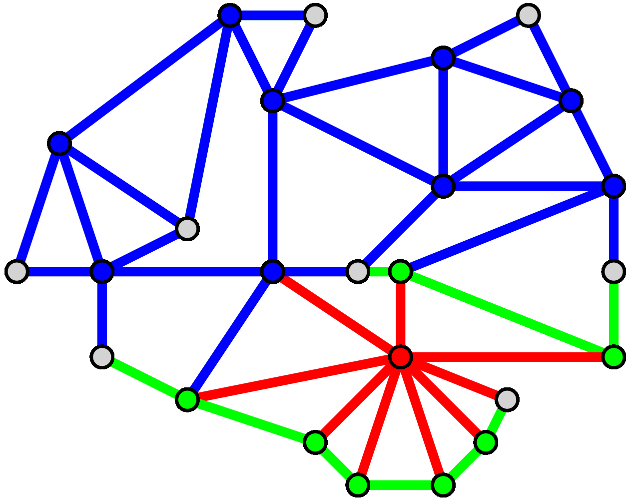


start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

- mark $\frac{1}{2}$-tight nodes
- block its edges
- double value of all other edges

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

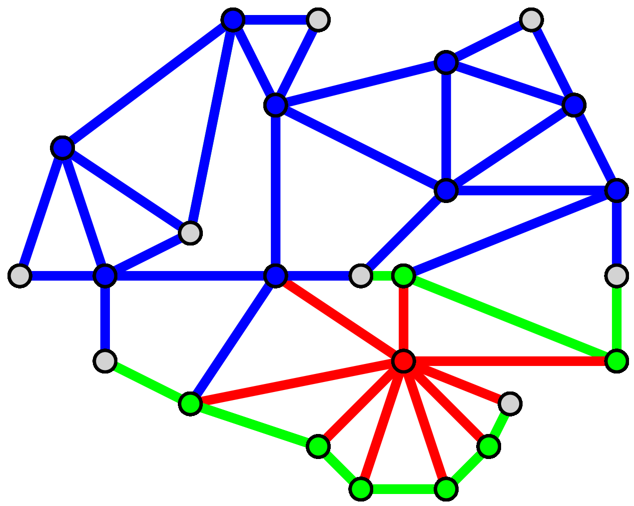
mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

until all edges are blocked

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

mark $\frac{1}{2}$-tight nodes

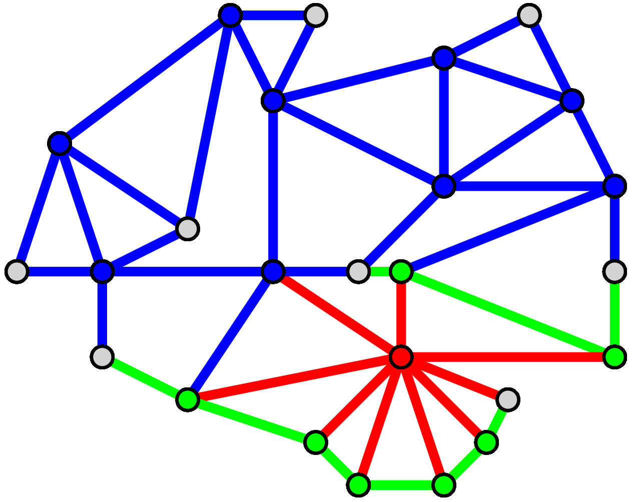
block its edges

double value of all other edges

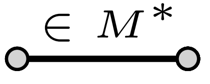
until all edges are blocked

4-approximation

(1) Find Fractional Matching



4-approximation



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

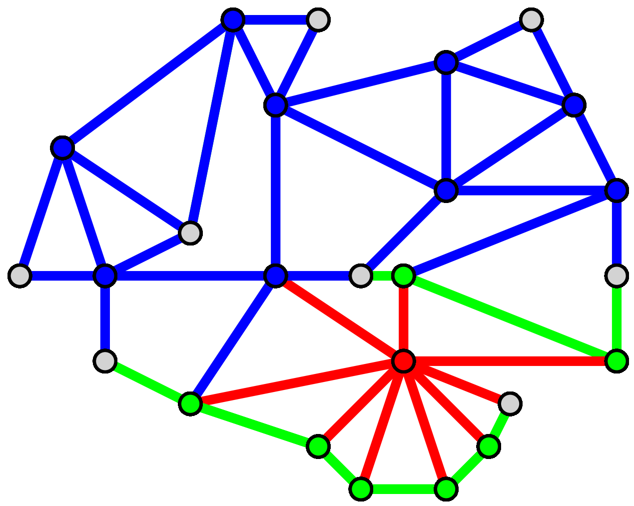
mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

until all edges are blocked

(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

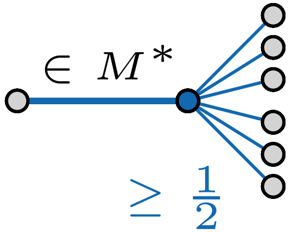
mark $\frac{1}{2}$-tight nodes

block its edges

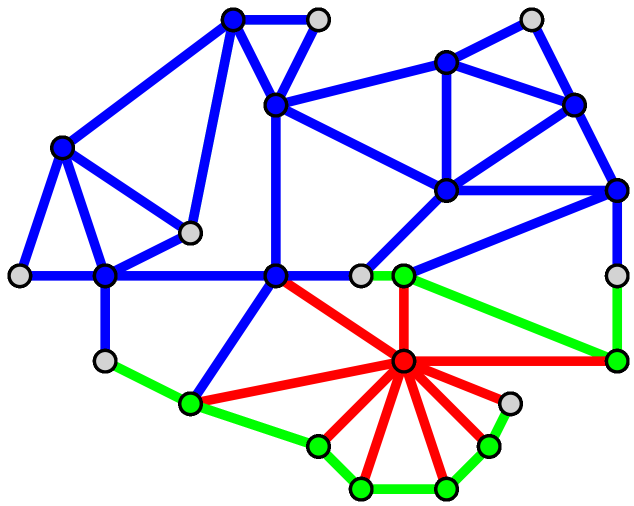
double value of all other edges

until all edges are blocked

4-approximation



(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

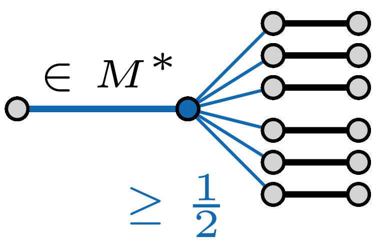
mark $\frac{1}{2}$-tight nodes

block its edges

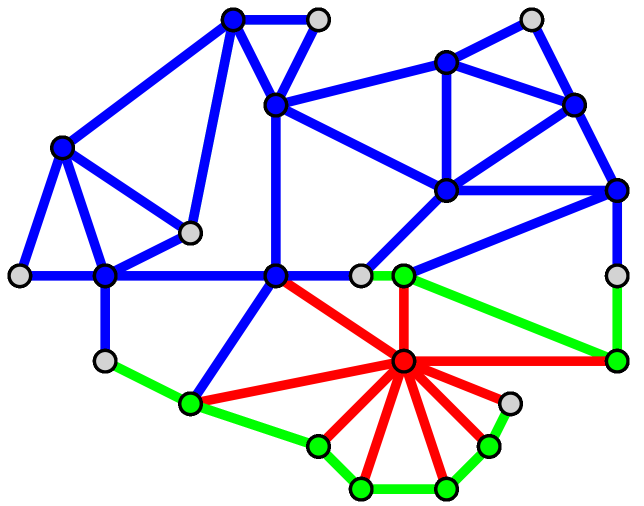
double value of all other edges

until all edges are blocked

4-approximation



(1) Find Fractional Matching



start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

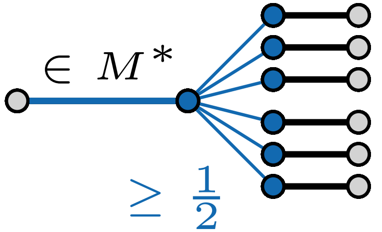
mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

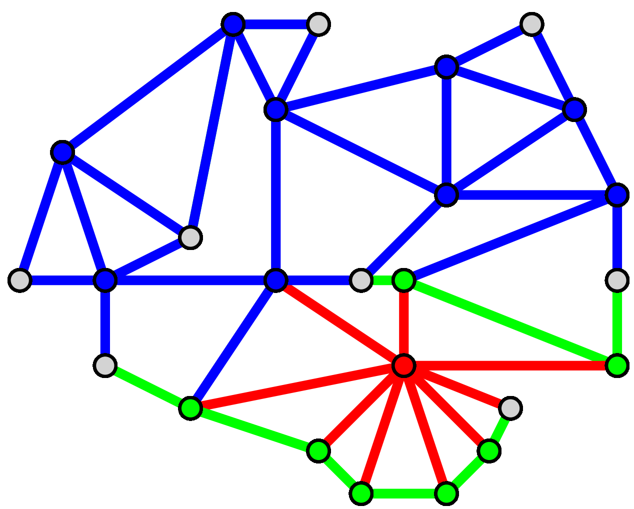
until all edges are blocked

4-approximation



possible overcounting by 2-factor

(1) Find Fractional Matching



4-approximation in $O(\log \Delta)$ rounds

start with $x_e = 2^{-\lceil \log \Delta \rceil}$

repeat

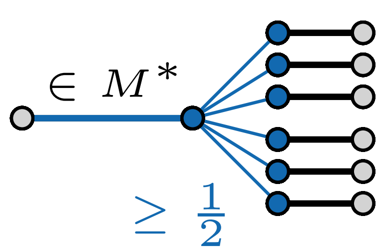
mark $\frac{1}{2}$-tight nodes

block its edges

double value of all other edges

until all edges are blocked

4-approximation



possible overcounting by 2-factor

(2) Iterative Rounding

(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

(2) Iterative Rounding

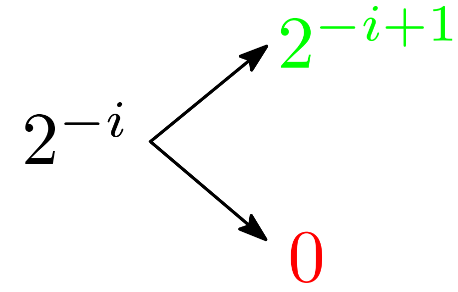
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

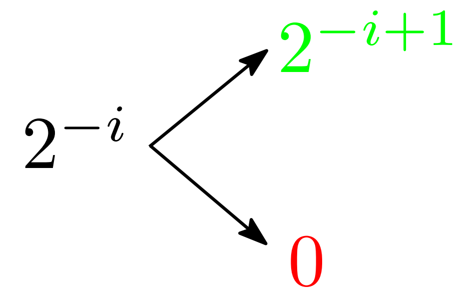
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



(2) Iterative Rounding

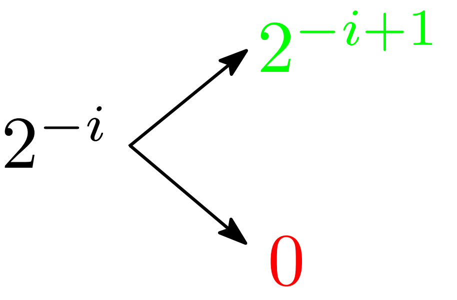
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



graph induced by $x_e = 2^{-i}$

(2) Iterative Rounding

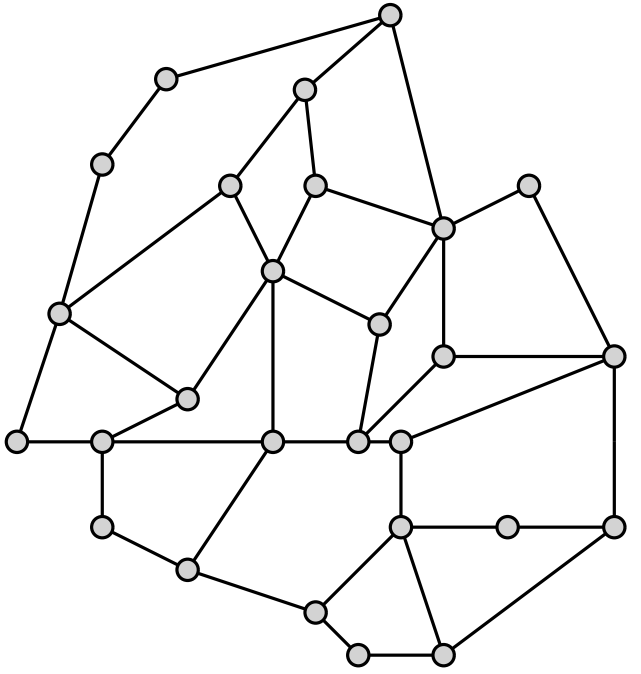


for $\lceil \log \Delta \rceil \geq i \geq 5$

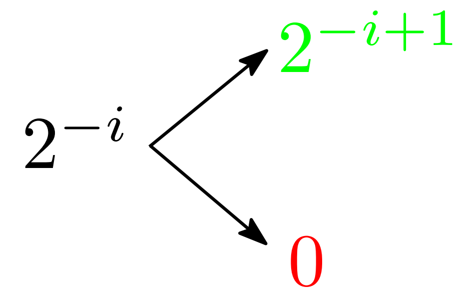
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

graph induced by $x_e = 2^{-i}$

decompose into paths/cycles



(2) Iterative Rounding

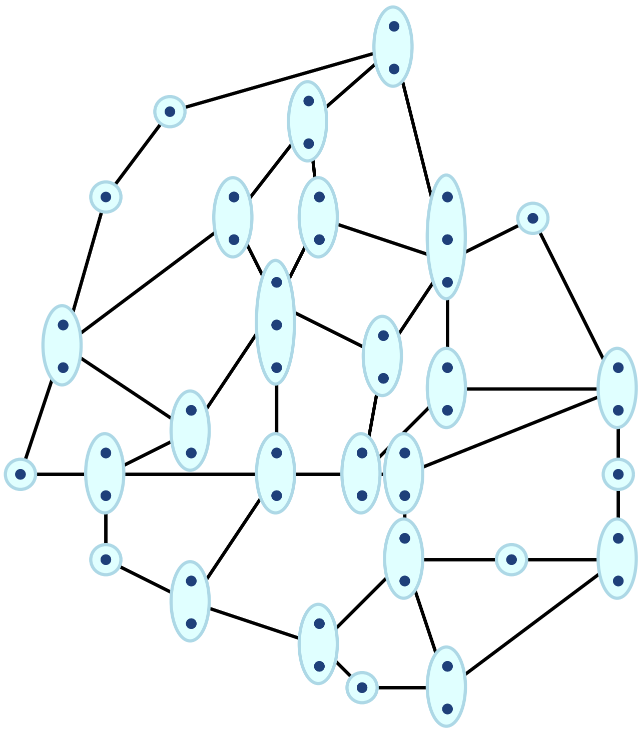


for $\lceil \log \Delta \rceil \geq i \geq 5$

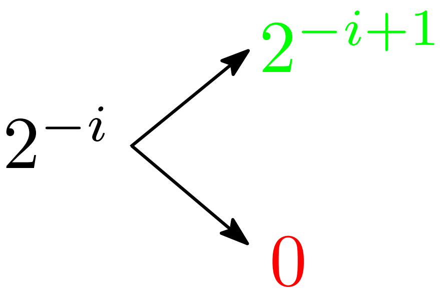
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

graph induced by $x_e = 2^{-i}$

decompose into paths/cycles



(2) Iterative Rounding



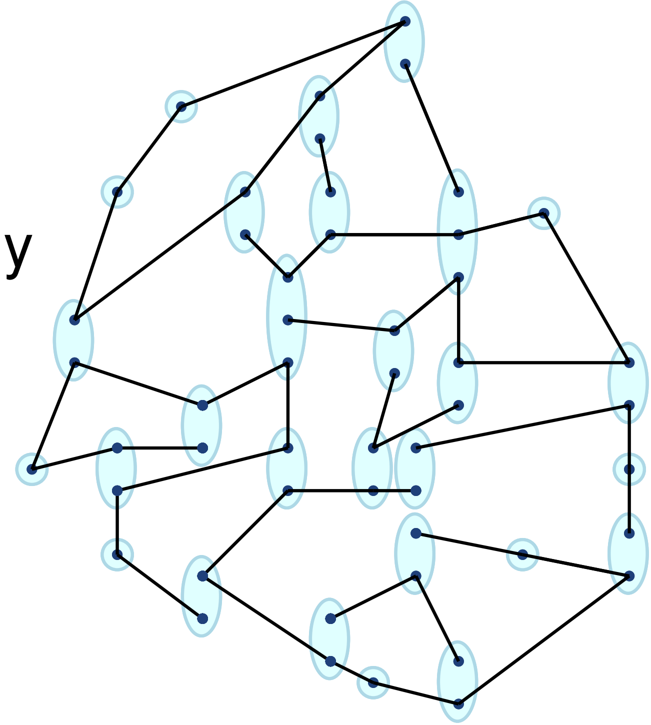
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

graph induced by $x_e = 2^{-i}$

decompose into paths/cycles

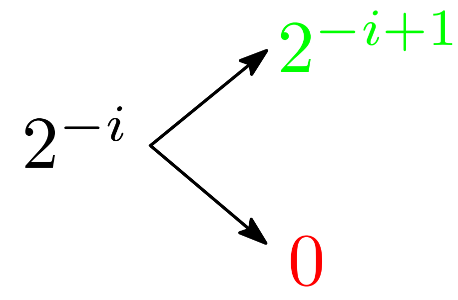
deal with each path/cycle separately



(2) Iterative Rounding

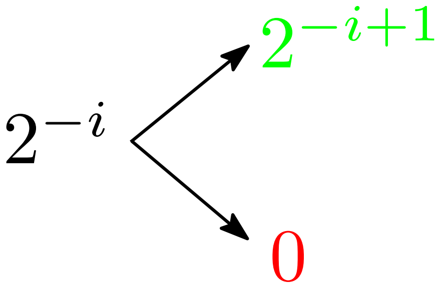
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Cycles

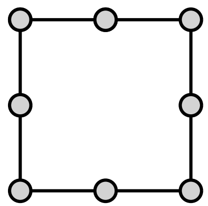
(2) Iterative Rounding



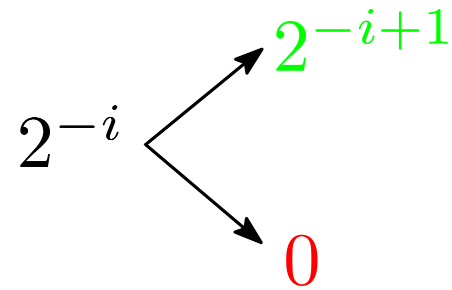
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Cycles



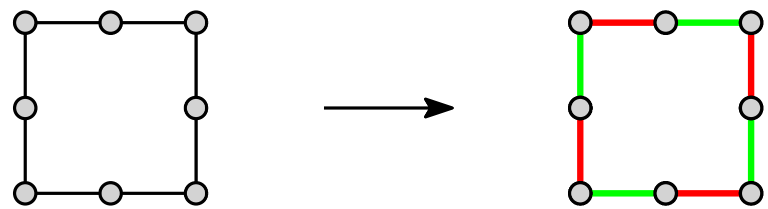
(2) Iterative Rounding



for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

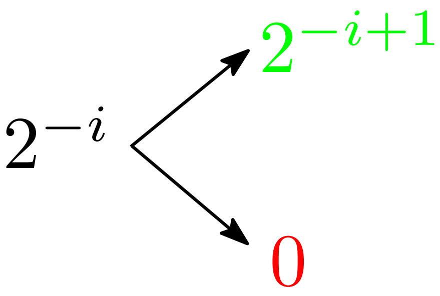
Short Cycles



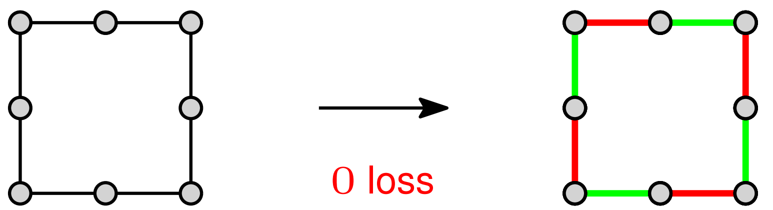
(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

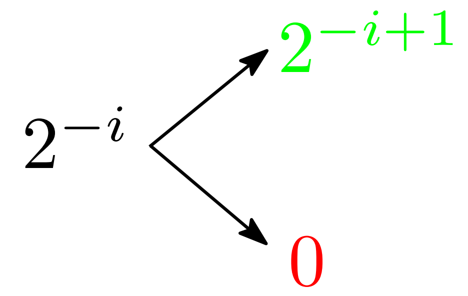
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Cycles



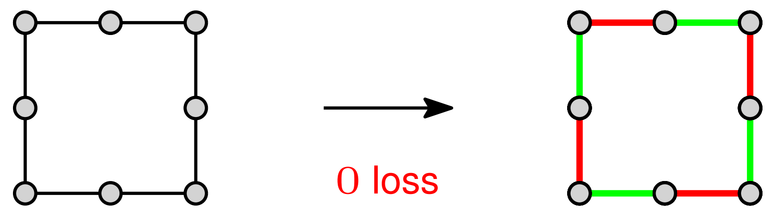
(2) Iterative Rounding



for $\lceil \log \Delta \rceil \geq i \geq 5$

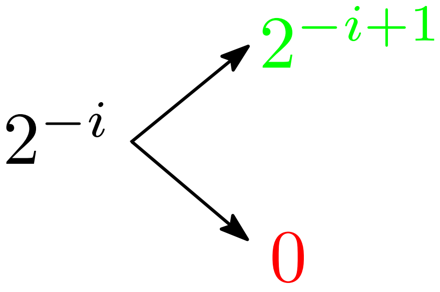
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Cycles



bipartite only!

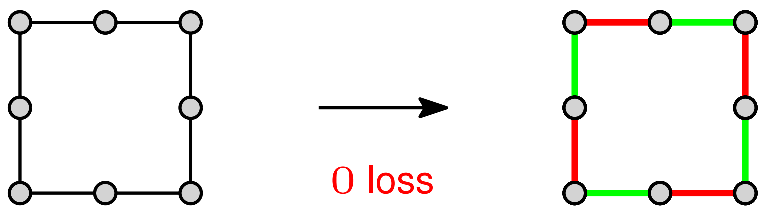
(2) Iterative Rounding



for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

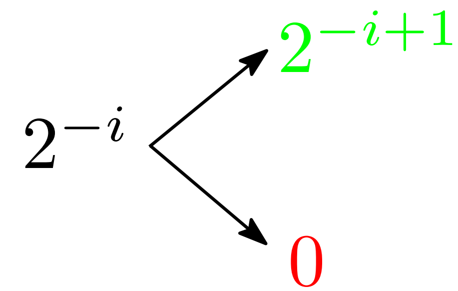
Short Cycles



bipartite only!

otherwise can lose $O(1)$

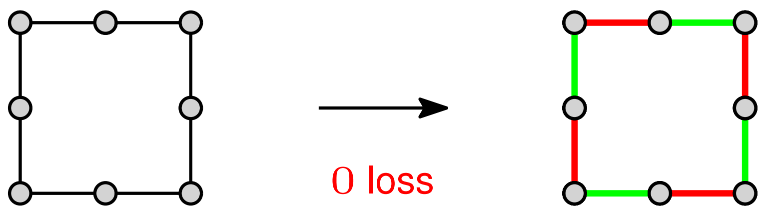
(2) Iterative Rounding



for $\lceil \log \Delta \rceil \geq i \geq 5$

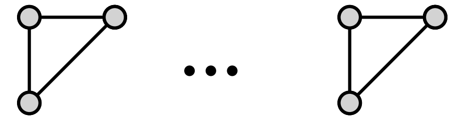
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Cycles

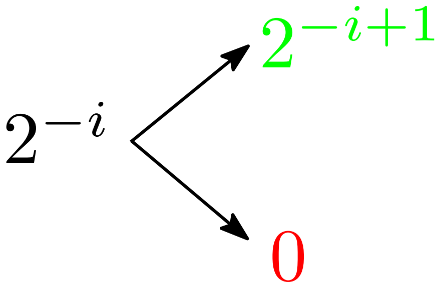


bipartite only!

otherwise can lose $O(1)$



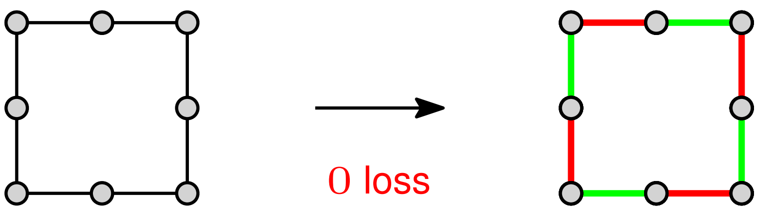
(2) Iterative Rounding



for $\lceil \log \Delta \rceil \geq i \geq 5$

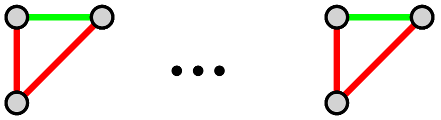
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Cycles



bipartite only!

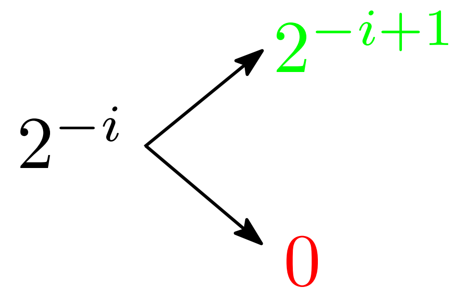
otherwise can lose $O(1)$



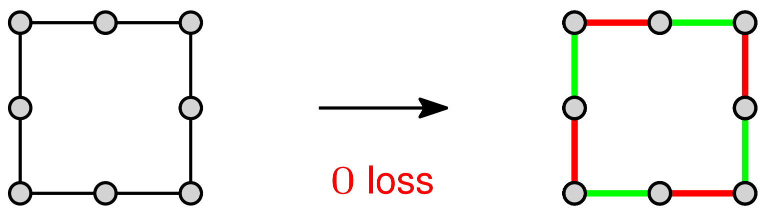
(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

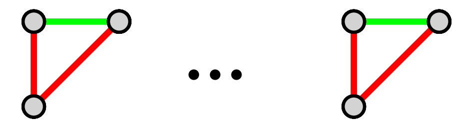


Short Cycles



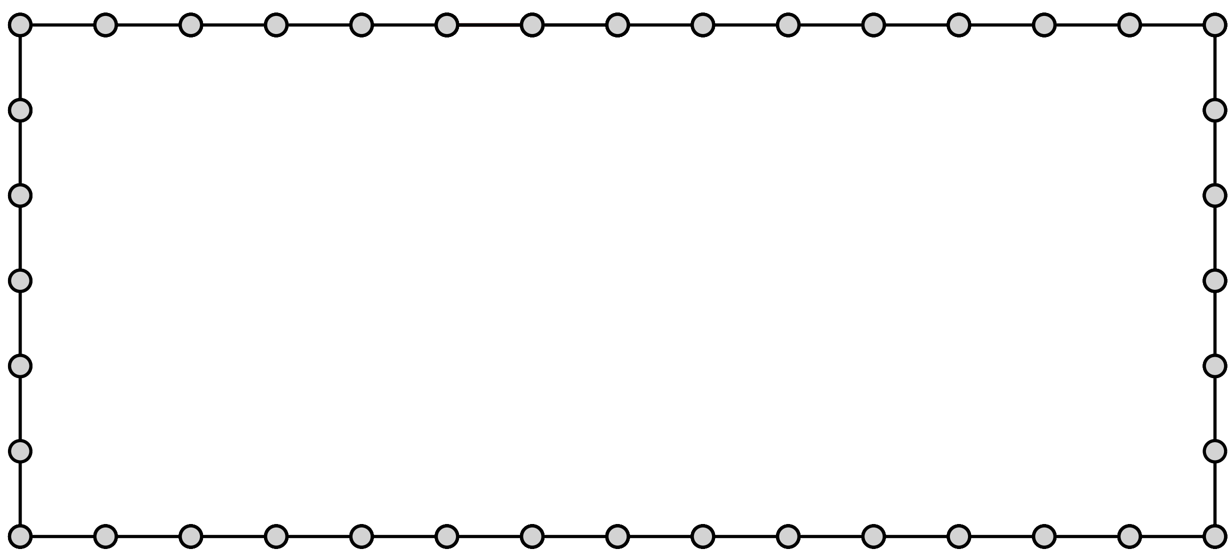
bipartite only!

otherwise can lose $O(1)$



Long Cycles

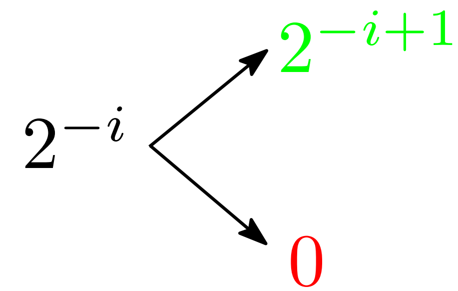
& Long Paths



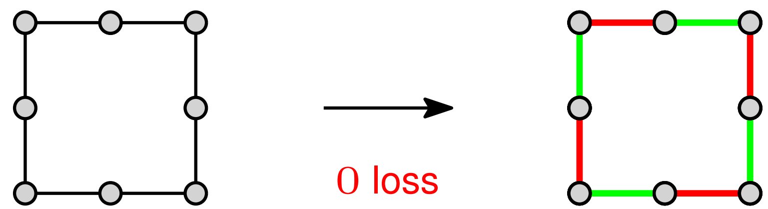
(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

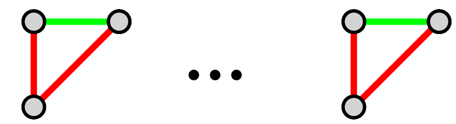


Short Cycles

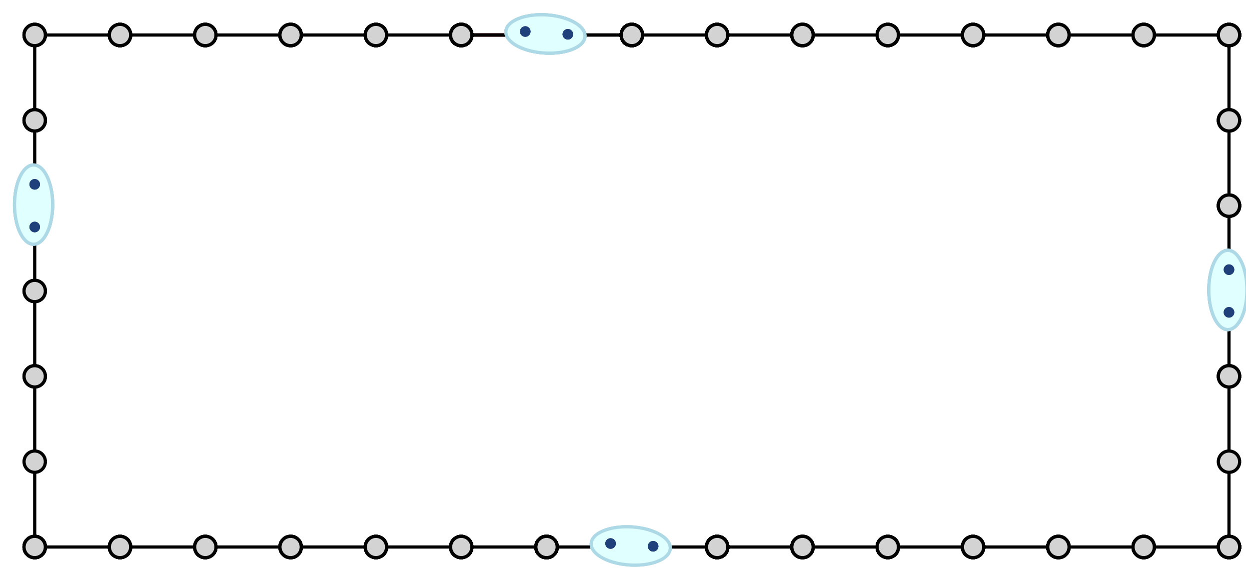


bipartite only!

otherwise can lose $O(1)$



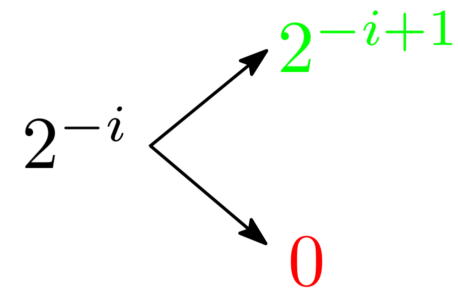
Long Cycles & Long Paths



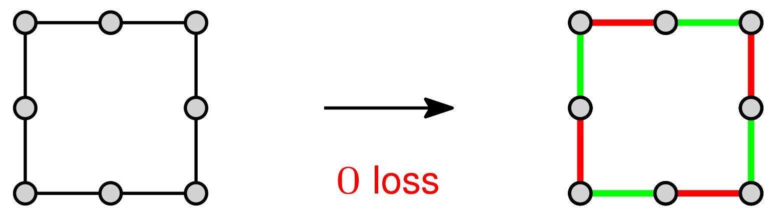
(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

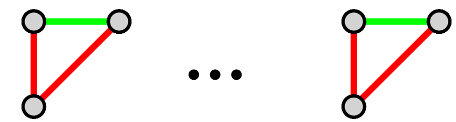


Short Cycles



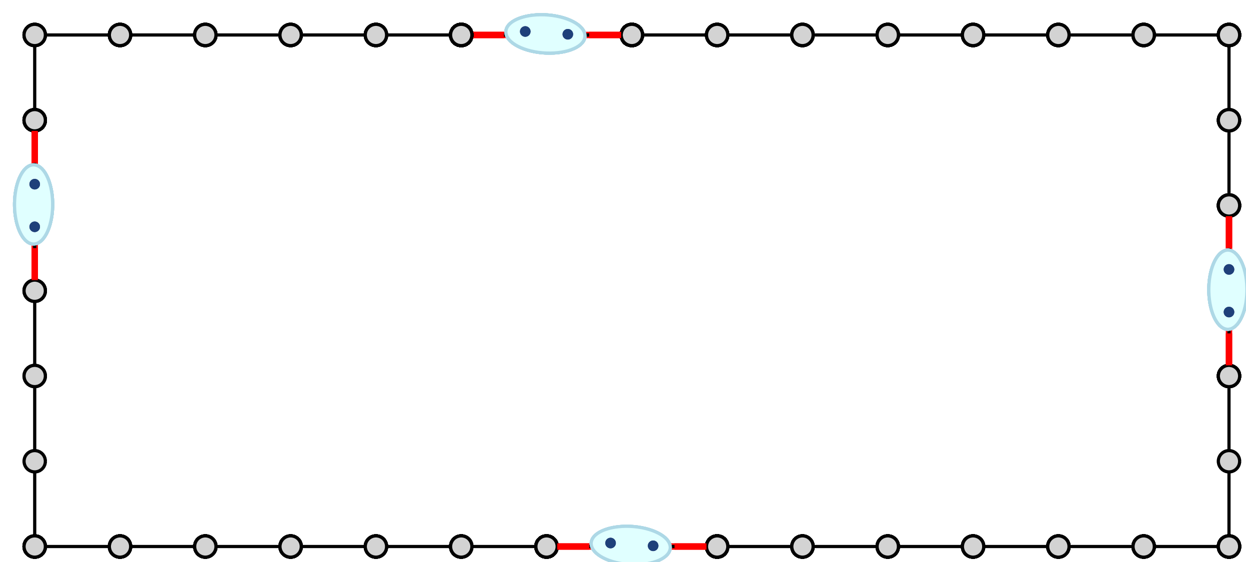
bipartite only!

otherwise can lose $O(1)$



Long Cycles

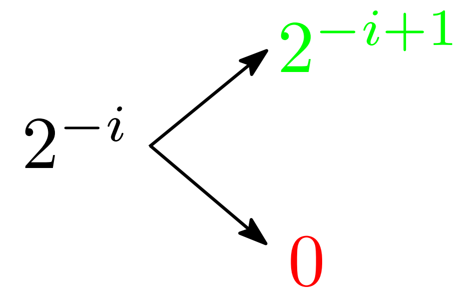
& Long Paths



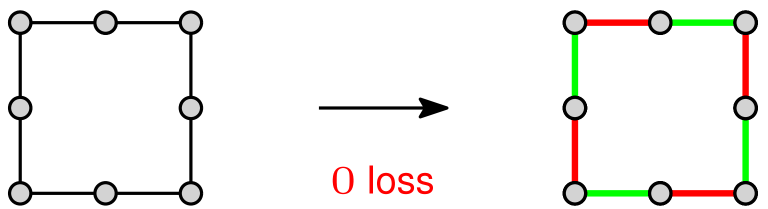
(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

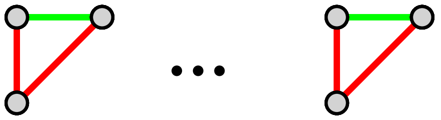


Short Cycles



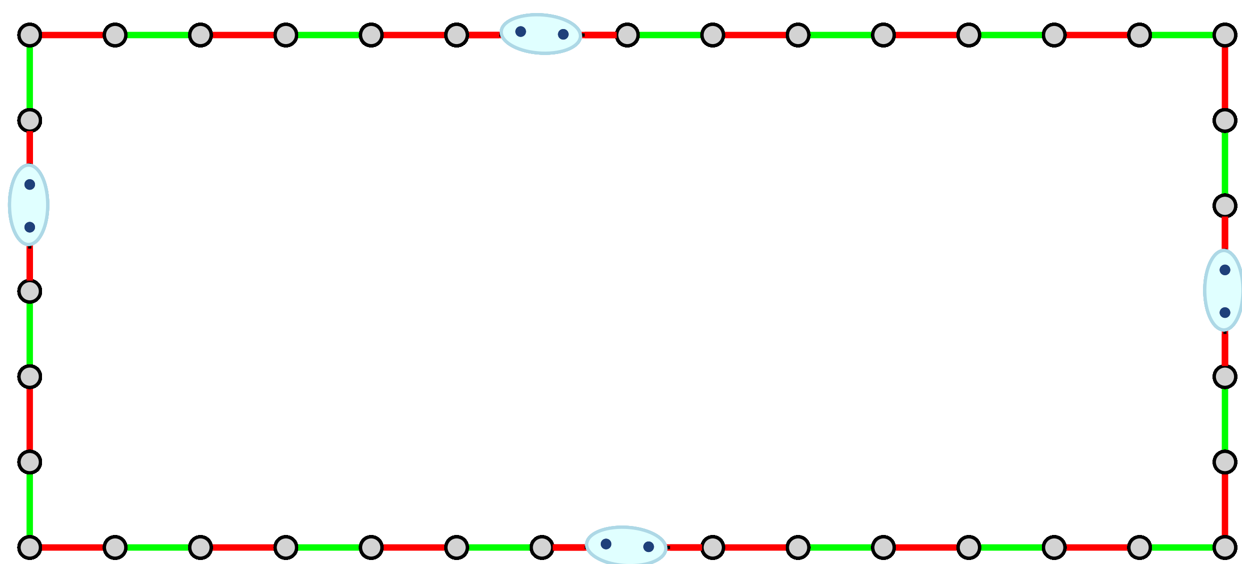
bipartite only!

otherwise can lose $O(1)$



Long Cycles

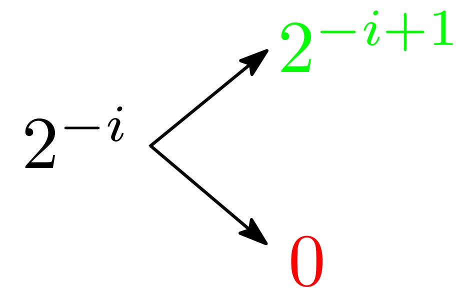
& Long Paths



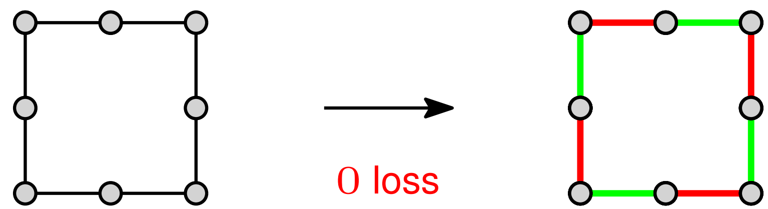
(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

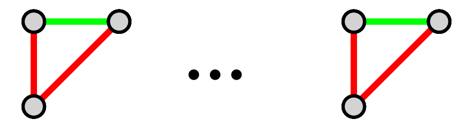


Short Cycles

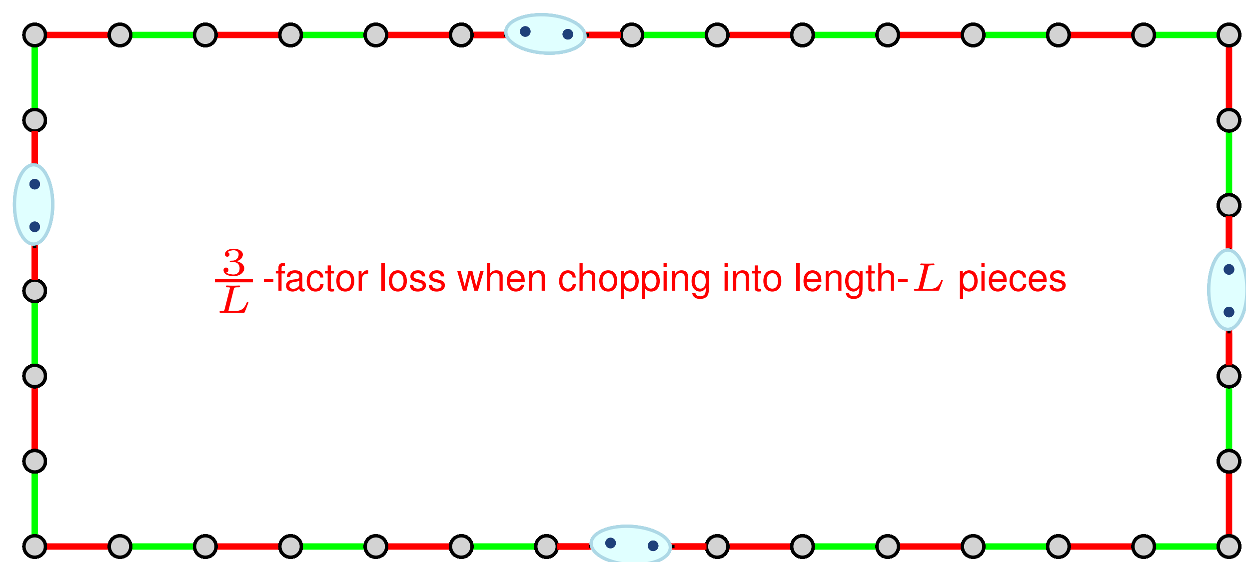


bipartite only!

otherwise can lose $O(1)$



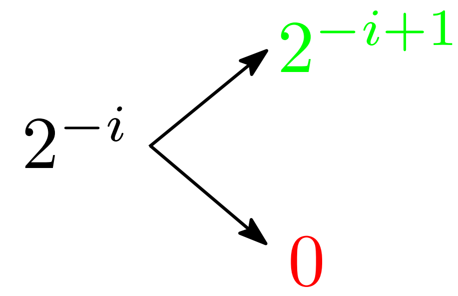
Long Cycles & Long Paths



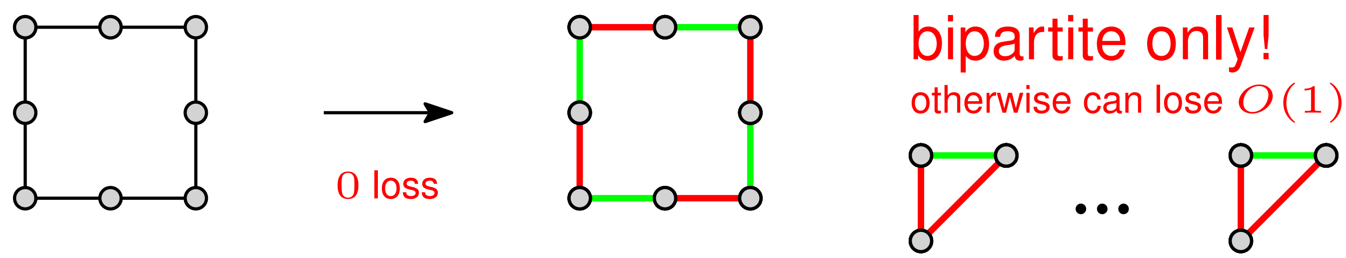
(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

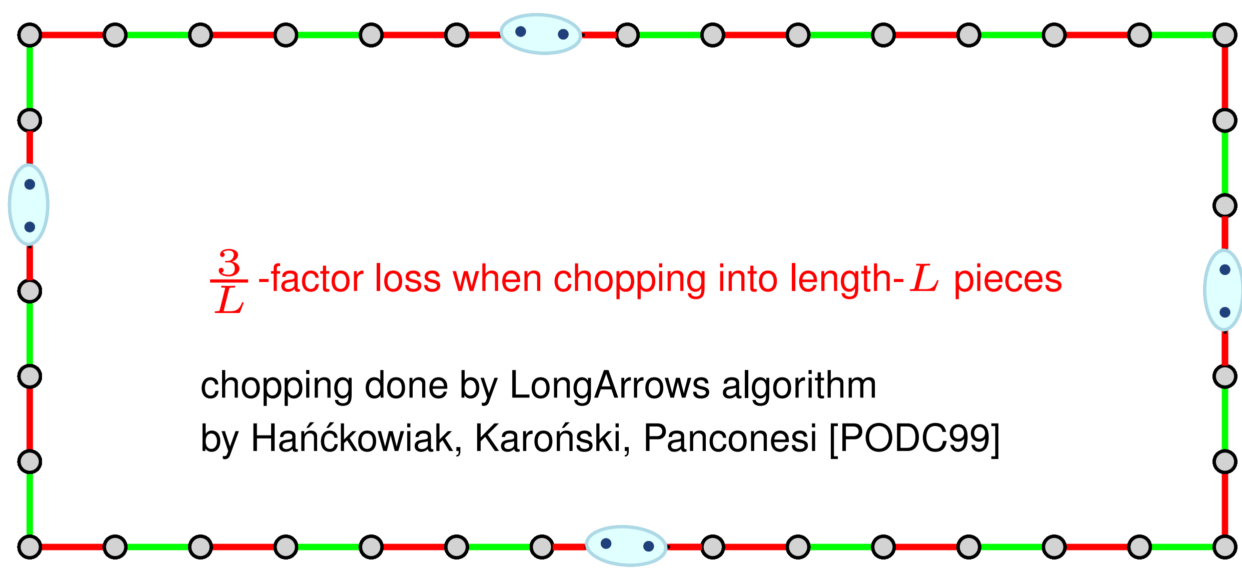


Short Cycles



Long Cycles

& Long Paths

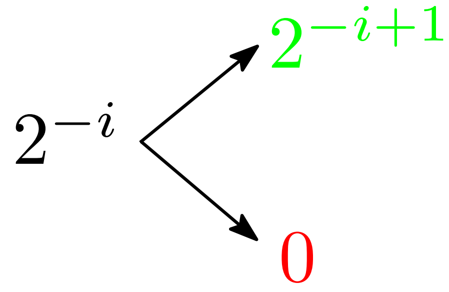


(2) Iterative Rounding

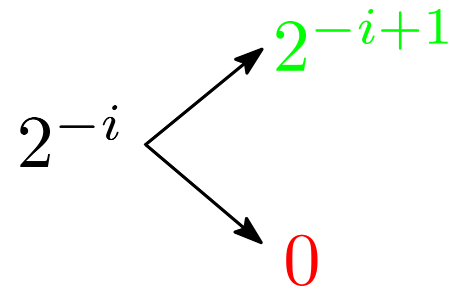
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Paths



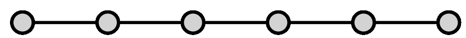
(2) Iterative Rounding



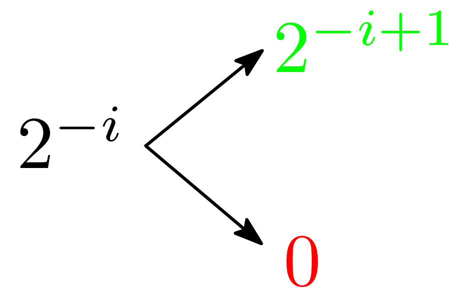
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Paths



(2) Iterative Rounding

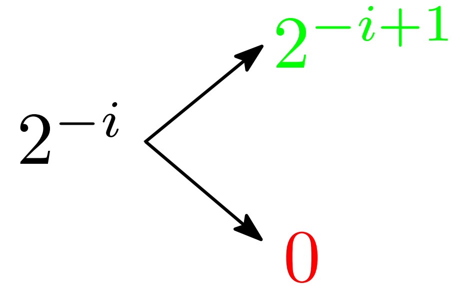


for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Paths 

(2) Iterative Rounding



for $\lceil \log \Delta \rceil \geq i \geq 5$

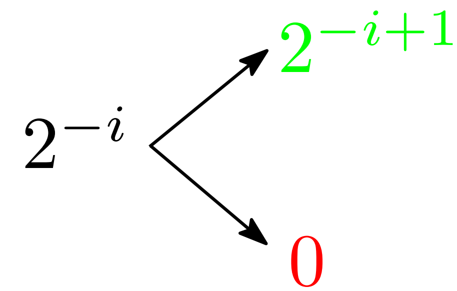
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

Short Paths

(2) Iterative Rounding

for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



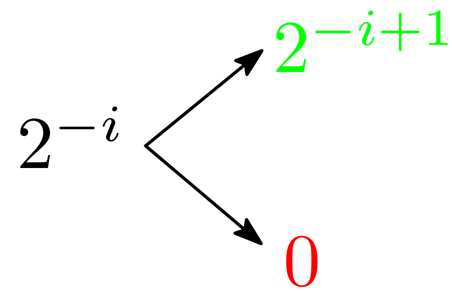
Short Paths

lose constant factor!

(2) Iterative Rounding

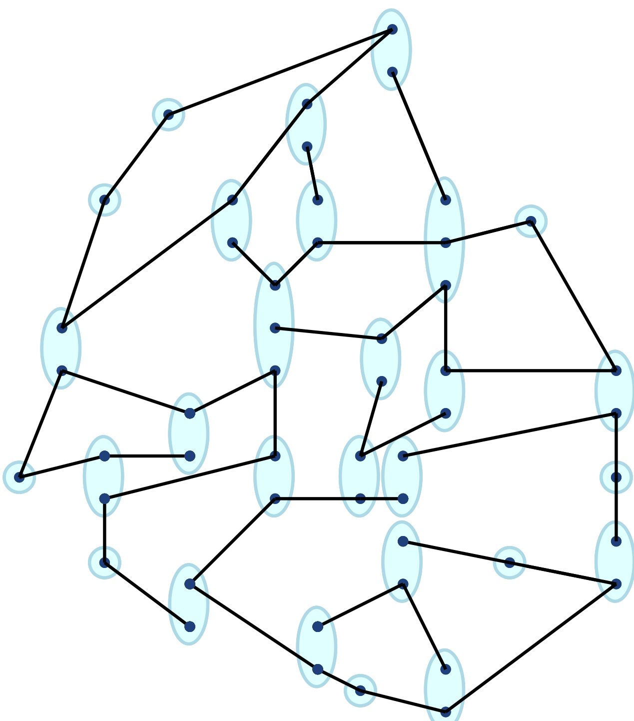
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

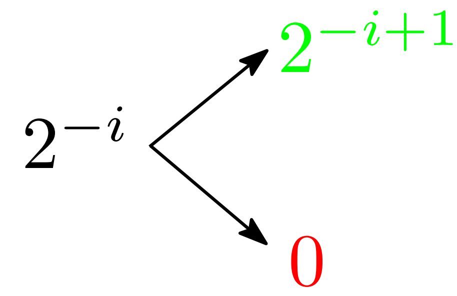
lose constant factor!



(2) Iterative Rounding

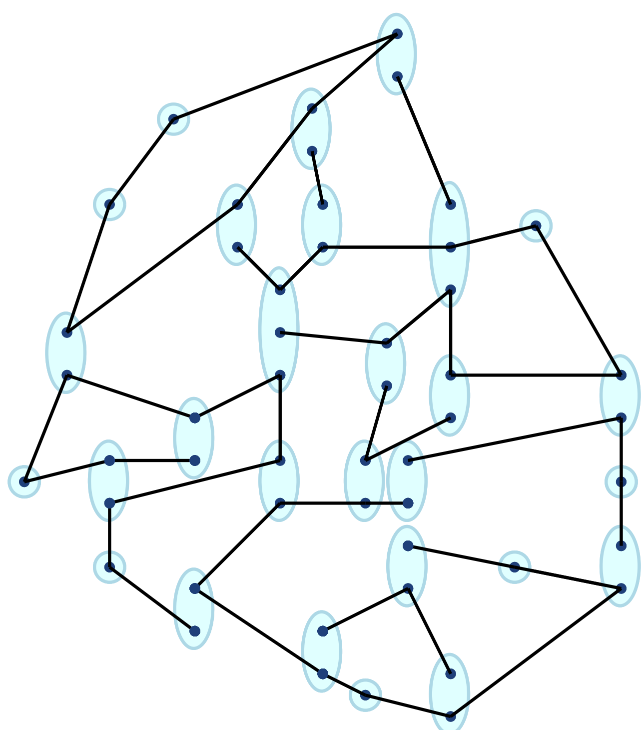
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!



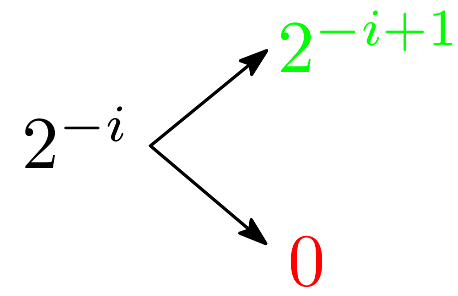
for $\frac{1}{2}$-loose nodes
& even-length paths

Manuela Fischer

(2) Iterative Rounding

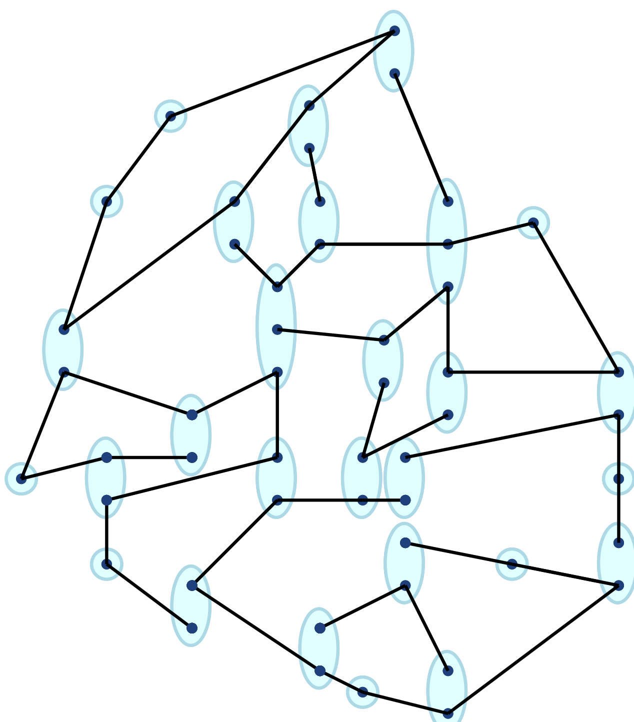
for $\lceil \log \Delta \rceil \geq i \geq 5$

round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}

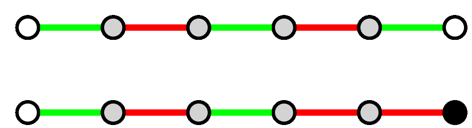


Short Paths

lose constant factor!



for $\frac{1}{2}$-loose nodes
& even-length paths

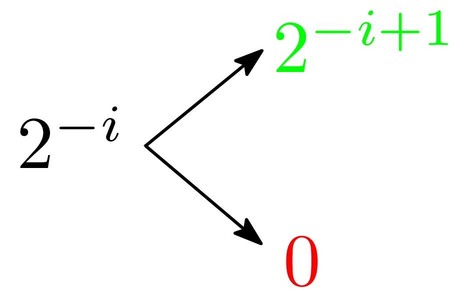


for $\frac{1}{2}$-loose nodes
& odd-length paths

(2) Iterative Rounding

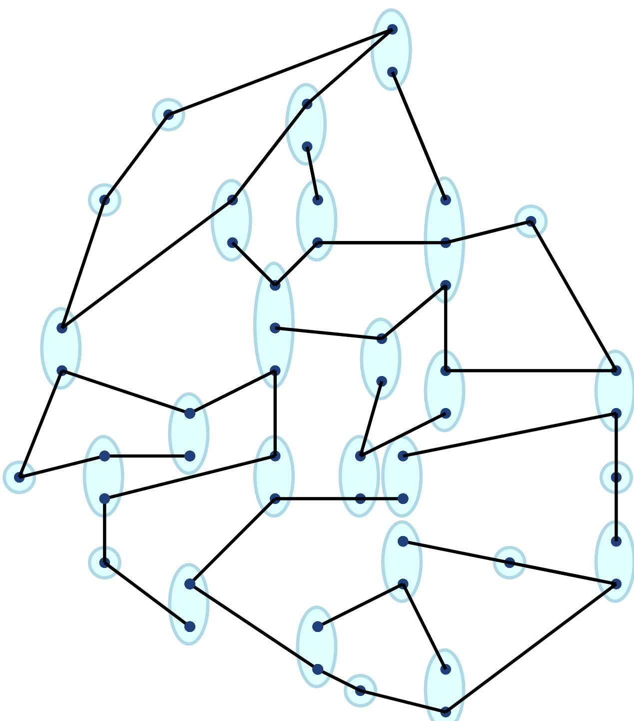
for $\lceil \log \Delta \rceil \geq i \geq 5$

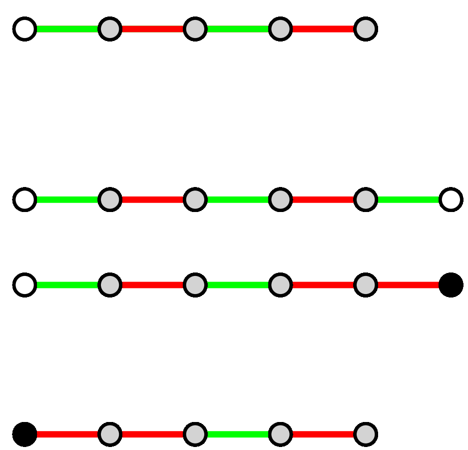
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!





for $\frac{1}{2}$-loose nodes
& even-length paths

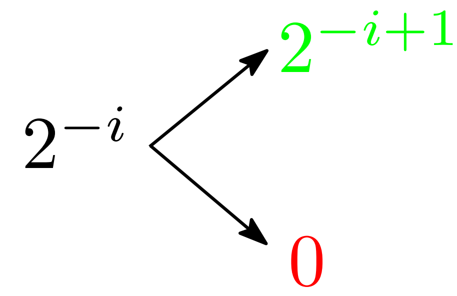
for $\frac{1}{2}$-loose nodes
& odd-length paths

for $\frac{1}{2}$-loose nodes
& even-length paths

(2) Iterative Rounding

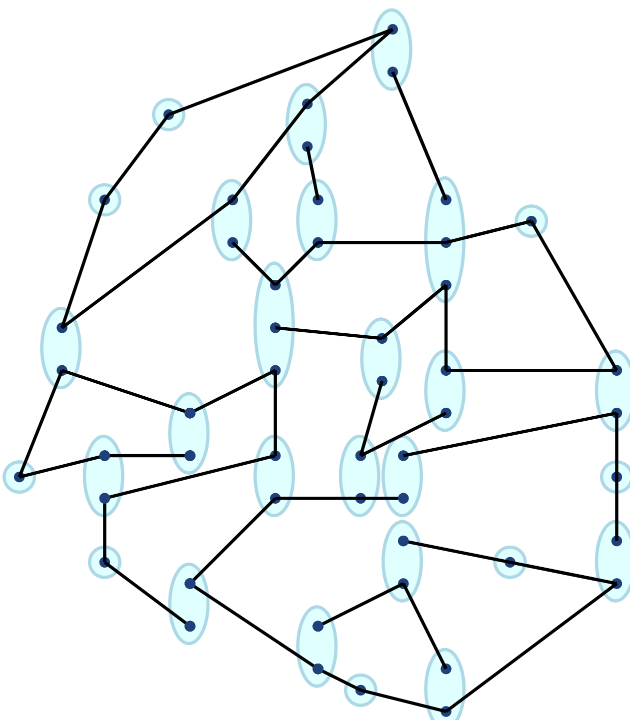
for $\lceil \log \Delta \rceil \geq i \geq 5$

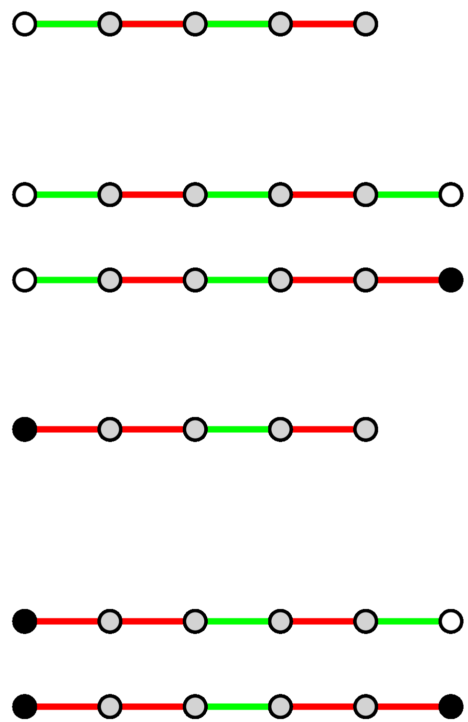
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!





for $\frac{1}{2}$-loose nodes & even-length paths

for $\frac{1}{2}$-loose nodes & odd-length paths

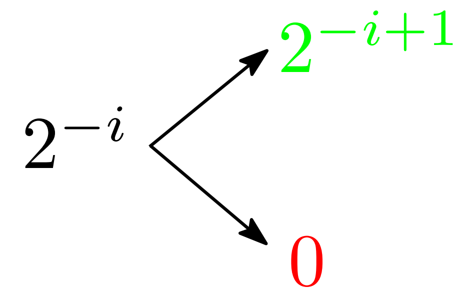
for $\frac{1}{2}$-loose nodes & even-length paths

for $\frac{1}{2}$-tight nodes & odd-length paths

(2) Iterative Rounding

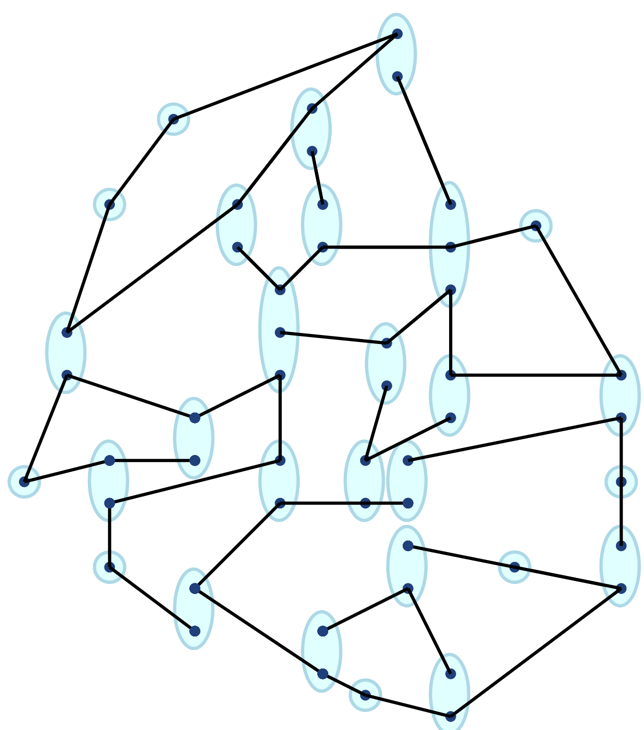
for $\lceil \log \Delta \rceil \geq i \geq 5$

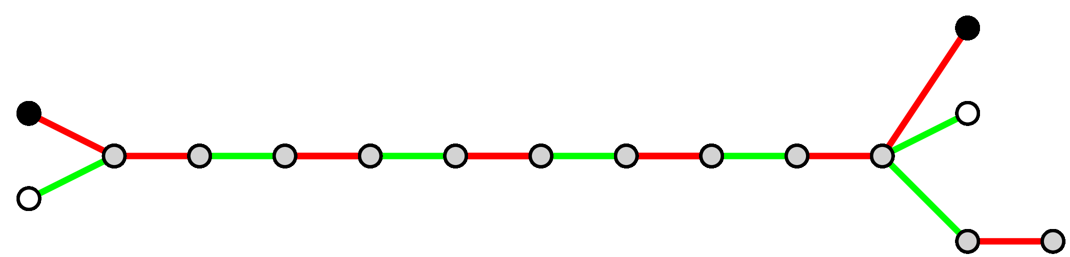
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!

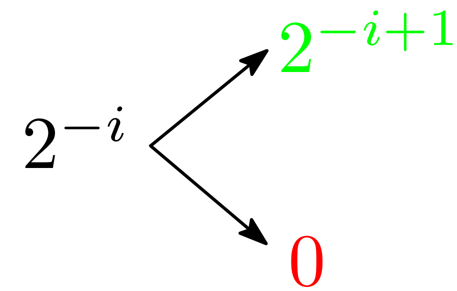




(2) Iterative Rounding

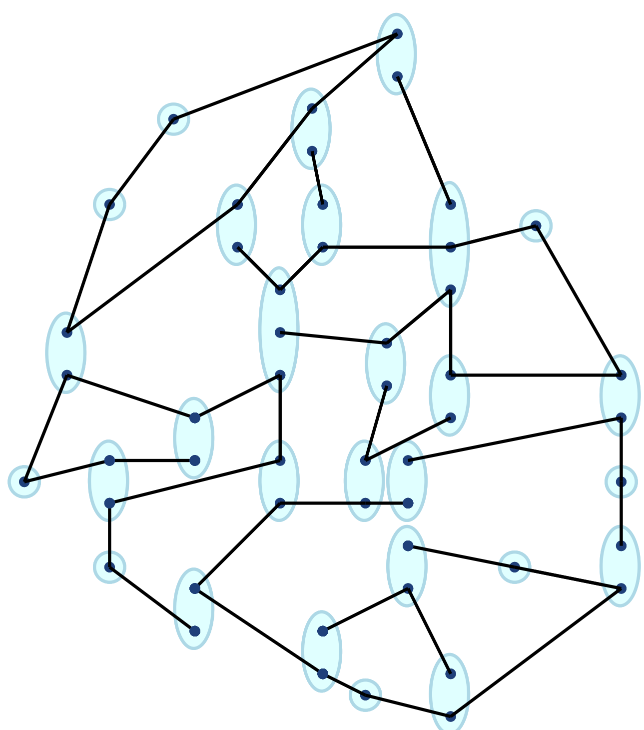
for $\lceil \log \Delta \rceil \geq i \geq 5$

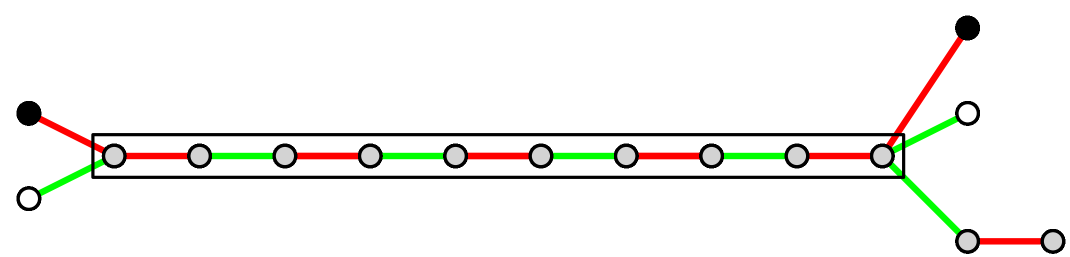
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!

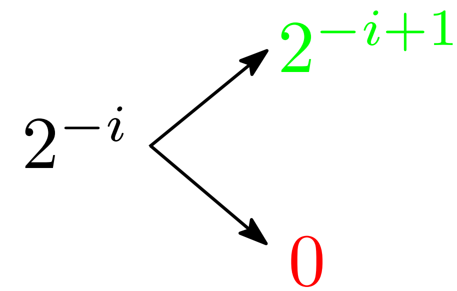




(2) Iterative Rounding

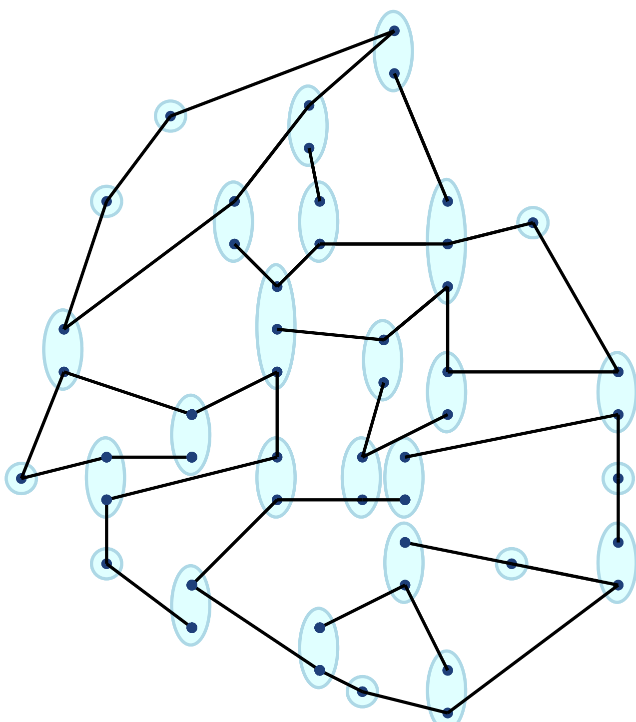
for $\lceil \log \Delta \rceil \geq i \geq 5$

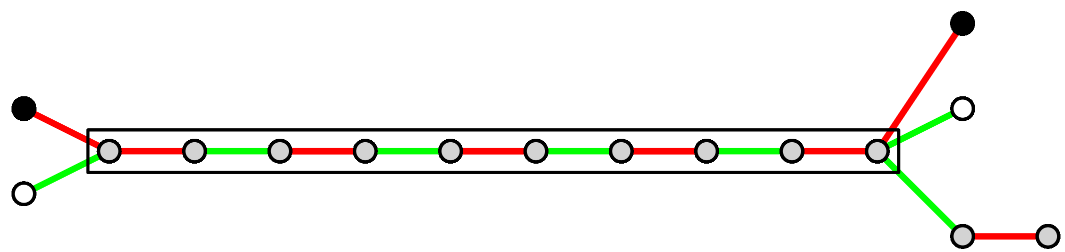
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!



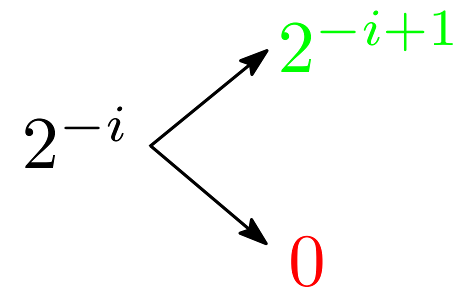


lose at most 2^{-i+1} per $\frac{1}{2}$-tight vertex

(2) Iterative Rounding

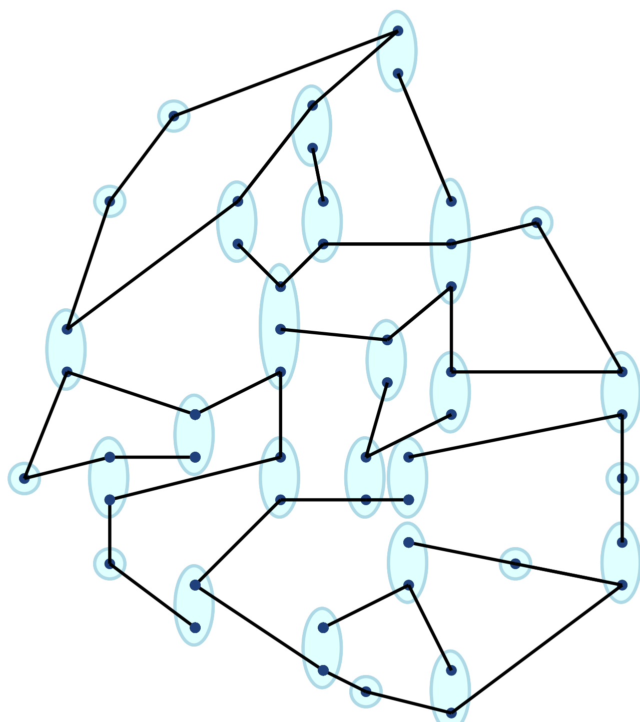
for $\lceil \log \Delta \rceil \geq i \geq 5$

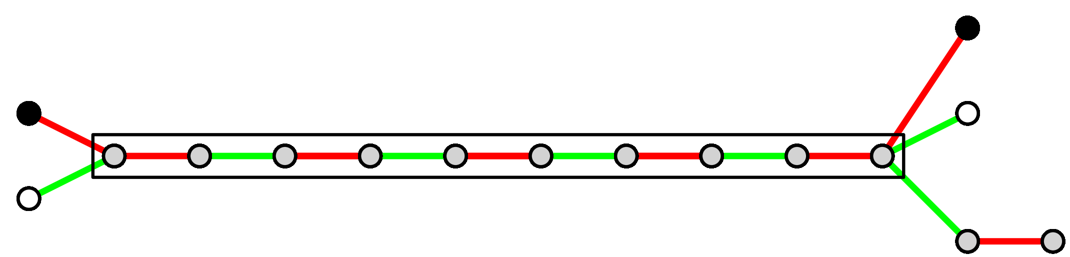
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!





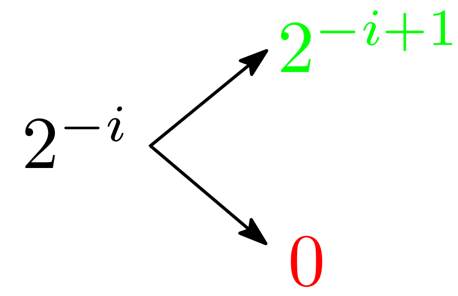
lose at most 2^{-i+1} per $\frac{1}{2}$-tight vertex

thus at most 2^{-i+2}-factor of its value

(2) Iterative Rounding

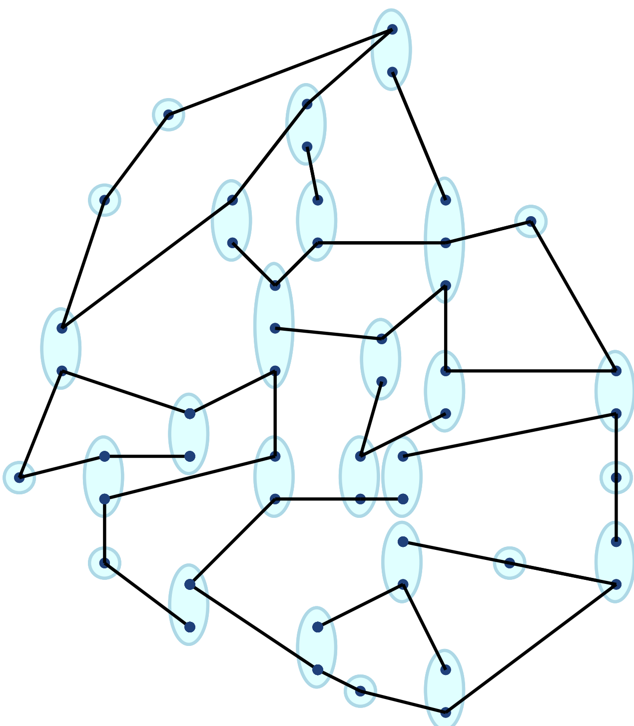
for $\lceil \log \Delta \rceil \geq i \geq 5$

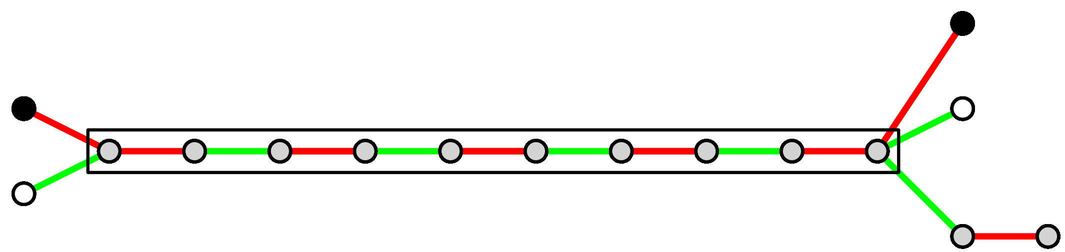
round values $x_e = 2^{-i}$ to 0 or to 2^{-i+1}



Short Paths

lose constant factor!





lose at most 2^{-i+1} per $\frac{1}{2}$-tight vertex

thus at most 2^{-i+2}-factor of its value

overall, lose at most 2^{-i+3}-factor

(2) Iterative Rounding

Short Cycles — no loss at all

Long Cycles & Long Paths — $\frac{3}{L}$-factor loss when chopping into length-L pieces

Short Paths — 2^{-i+3}-factor loss

(2) Iterative Rounding

Short Cycles — no loss at all

Long Cycles & Long Paths — $\frac{3}{L}$-factor loss when chopping into length-L pieces

Short Paths — 2^{-i+3}-factor loss

reduce value to $(1 - \frac{3}{L} - 2^{-i+3})$-factor of previous value

(2) Iterative Rounding

Short Cycles — no loss at all

Long Cycles & Long Paths — $\frac{3}{L}$-factor loss when chopping into length-L pieces

$L = \lceil \log \Delta \rceil$

Short Paths — 2^{-i+3}-factor loss

reduce value to $(1 - \frac{3}{L} - 2^{-i+3})$-factor of previous value

$$\sum_{e \in E} x'_e \geq \prod_{i=5}^{\lceil \log \Delta \rceil} (1 - \frac{3}{L} - 2^{-i+3}) \sum_{e \in E} x_e \geq \frac{1}{5} \sum_{e \in E} x_e$$

(2) Iterative Rounding

5-approximation in $O(\log^2 \Delta)$ rounds

Short Cycles — no loss at all

Long Cycles & Long Paths — $\frac{3}{L}$-factor loss when chopping into length-L pieces

$L = \lceil \log \Delta \rceil$

Short Paths — 2^{-i+3}-factor loss

reduce value to $(1 - \frac{3}{L} - 2^{-i+3})$-factor of previous value

$$\sum_{e \in E} x'_e \geq \prod_{i=5}^{\lceil \log \Delta \rceil} (1 - \tfrac{3}{L} - 2^{-i+3}) \sum_{e \in E} x_e \geq \tfrac{1}{5} \sum_{e \in E} x_e$$

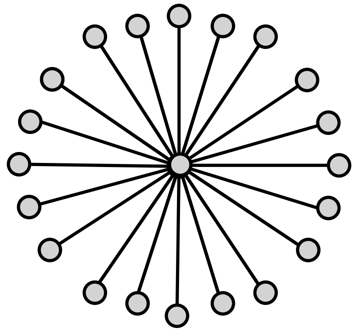
(3) Final Rounding

(3) Final Rounding

values in $\{2^{-i} \mid 4 \geq i \geq 1\} \cup \{0\}$

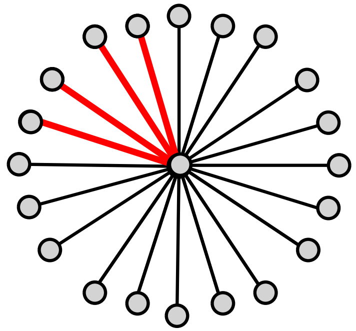
(3) Final Rounding

values in $\{2^{-i} \mid 4 \geq i \geq 1\} \cup \{0\}$



(3) Final Rounding

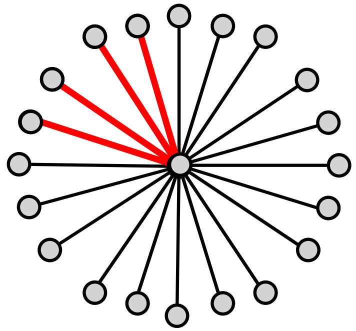
values in $\{2^{-i} \mid 4 \geq i \geq 1\} \cup \{0\}$



maximum degree in induced graph ≤ 16

(3) Final Rounding

values in $\{2^{-i} \mid 4 \geq i \geq 1\} \cup \{0\}$

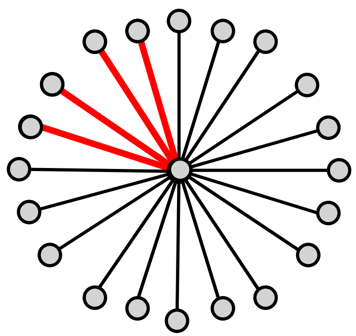


maximum degree in induced graph ≤ 16

find maximal matching in $O(1)$ by Proposing Algorithm

(3) Final Rounding

values in $\{2^{-i} \mid 4 \geq i \geq 1\} \cup \{0\}$



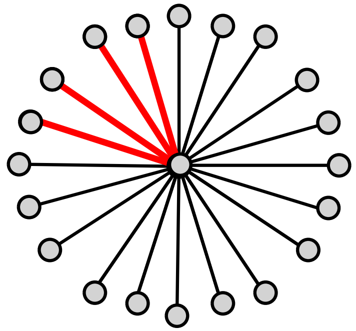
maximum degree in induced graph ≤ 16

find maximal matching in $O(1)$ by Proposing Algorithm

matches constant fraction of the remaining edges

(3) Final Rounding

values in $\{2^{-i} \mid 4 \geq i \geq 1\} \cup \{0\}$



maximum degree in induced graph ≤ 16

find maximal matching in $O(1)$ by Proposing Algorithm

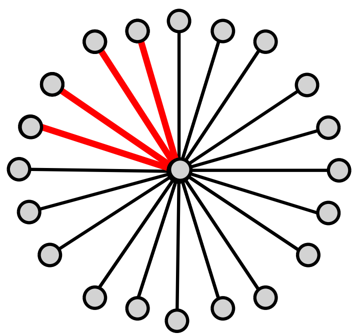
matches constant fraction of the remaining edges

(any maximal matching has size $\geq \frac{1}{2\Delta - 1} |E|$)

(3) Final Rounding

$O(1)$-approximation in $O(1)$ rounds

values in $\{2^{-i} \mid 4 \geq i \geq 1\} \cup \{0\}$



maximum degree in induced graph ≤ 16

find maximal matching in $O(1)$ by Proposing Algorithm

matches constant fraction of the remaining edges

(any maximal matching has size $\geq \frac{1}{2\Delta - 1} |E|$)

Our Matching Approximation Algorithm: Recap

Our Matching Approximation Algorithm: Recap

(1) Fractional matching

Our Matching Approximation Algorithm: Recap

(1) Fractional matching

values in $\{2^{-i} \mid \lceil \log \Delta \rceil \geq i \geq 1\}$

4-approximation

$O(\log \Delta)$ rounds

Our Matching Approximation Algorithm: Recap

(1) Fractional matching

values in $\{2^{-i} \mid \lceil \log \Delta \rceil \geq i \geq 1\}$

4-approximation

$O(\log \Delta)$ rounds

(2) Iterative rounding

Our Matching Approximation Algorithm: Recap

(1) Fractional matching

values in $\{2^{-i} \mid \lceil \log \Delta \rceil \geq i \geq 1\}$

4-approximation

$O(\log \Delta)$ rounds

(2) Iterative rounding

values in $\left\{ \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 0 \right\}$

5-approximation of fractional matching of (1)

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

Our Matching Approximation Algorithm: Recap

(1) Fractional matching

values in $\{2^{-i} \mid \lceil \log \Delta \rceil \geq i \geq 1\}$

4-approximation

$O(\log \Delta)$ rounds

(2) Iterative rounding

values in $\left\{ \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 0 \right\}$

5-approximation of fractional matching of (1)

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

(3) Final Rounding

Our Matching Approximation Algorithm: Recap

(1) Fractional matching

values in $\{2^{-i} \mid \lceil \log \Delta \rceil \geq i \geq 1\}$

4-approximation

$O(\log \Delta)$ rounds

(2) Iterative rounding

values in $\left\{ \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 0 \right\}$

5-approximation of fractional matching of (1)

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

(3) Final Rounding

values in $\{1, 0\}$: integral matching

$O(1)$-approximation of fractional matching of (2)

$O(1)$ rounds

Our Matching Approximation Algorithm: Recap

$O(1)$-approximation in $O(\log^2 \Delta)$ rounds

(1) Fractional matching

values in $\{2^{-i} \mid \lceil \log \Delta \rceil \geq i \geq 1\}$

4-approximation

$O(\log \Delta)$ rounds

(2) Iterative rounding

values in $\left\{ \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 0 \right\}$

5-approximation of fractional matching of (1)

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

(3) Final Rounding

values in $\{1, 0\}$: integral matching

$O(1)$-approximation of fractional matching of (2)

$O(1)$ rounds

Our Matching Approximation Algorithm: Recap

$O(1)$-approximation in $O(\log^2 \Delta)$ rounds in bipartite graphs

(1) Fractional matching

values in $\{2^{-i} \mid \lceil \log \Delta \rceil \geq i \geq 1\}$

4-approximation

$O(\log \Delta)$ rounds

(2) Iterative rounding

values in $\left\{\frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 0\right\}$

5-approximation of fractional matching of (1)

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

(3) Final Rounding

values in $\{1, 0\}$: integral matching

$O(1)$-approximation of fractional matching of (2)

$O(1)$ rounds

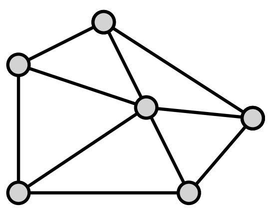
From Bipartite to General Graphs

From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B

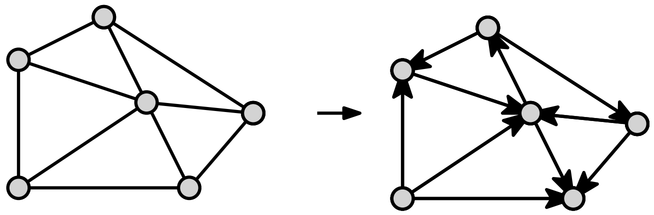
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



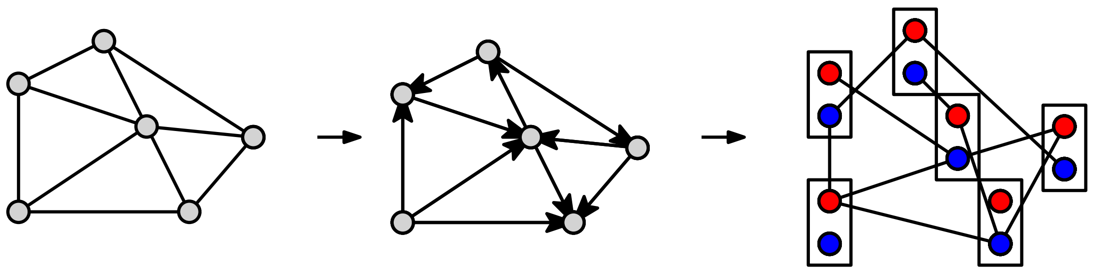
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



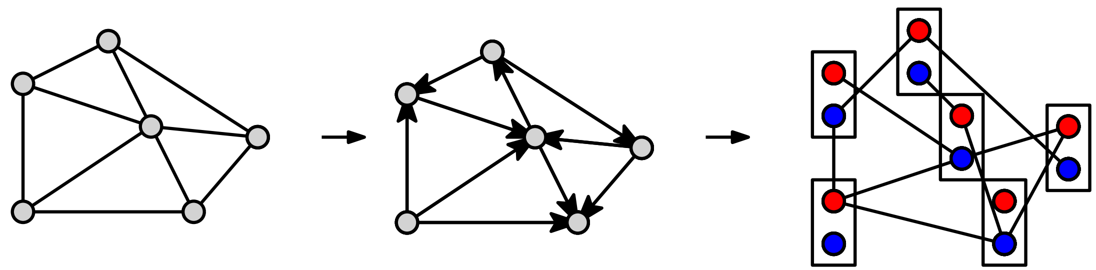
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



From Bipartite to General Graphs

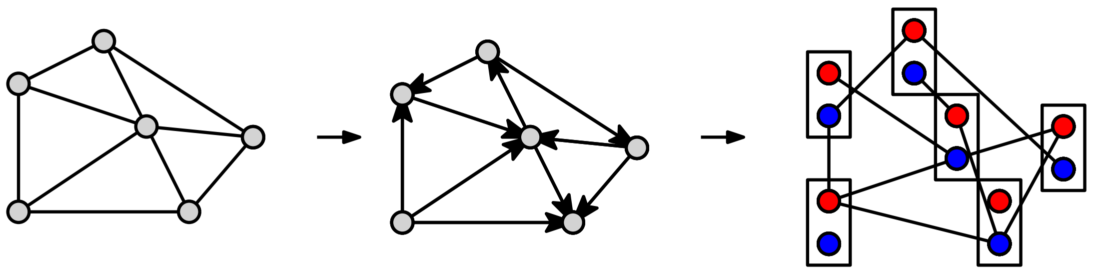
(1) Turn G into auxiliary bipartite graph B



(2) Find $O(1)$-approximate matching M_B in B

From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B

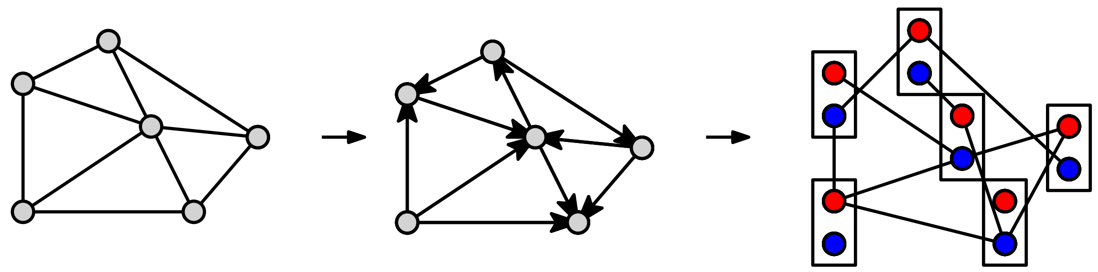


(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G

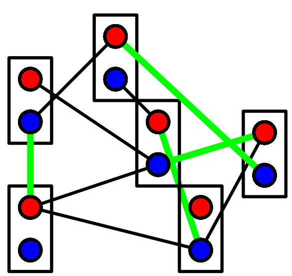
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



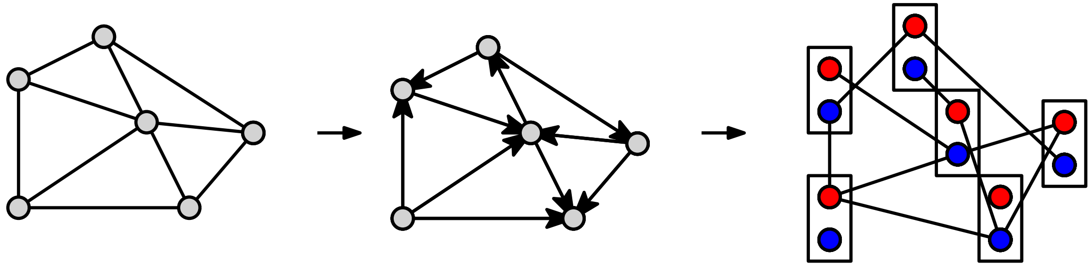
(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G



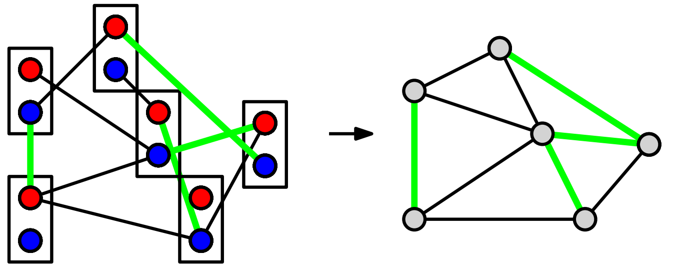
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



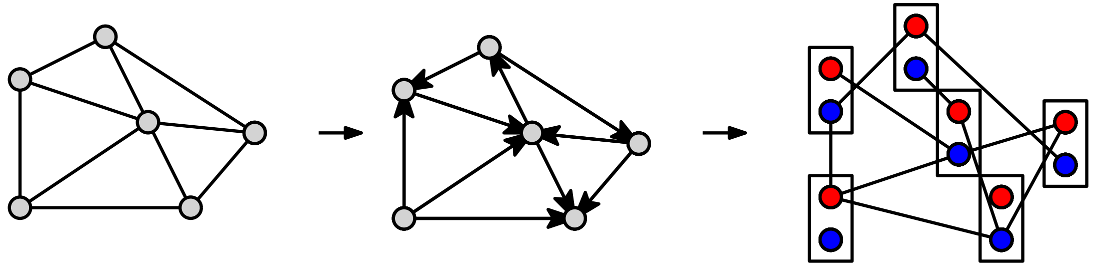
(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G



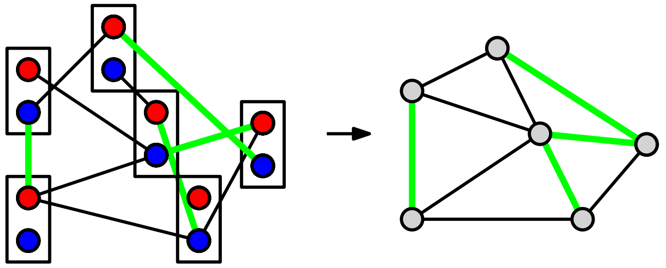
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



(2) Find $O(1)$-approximate matching M_B in B

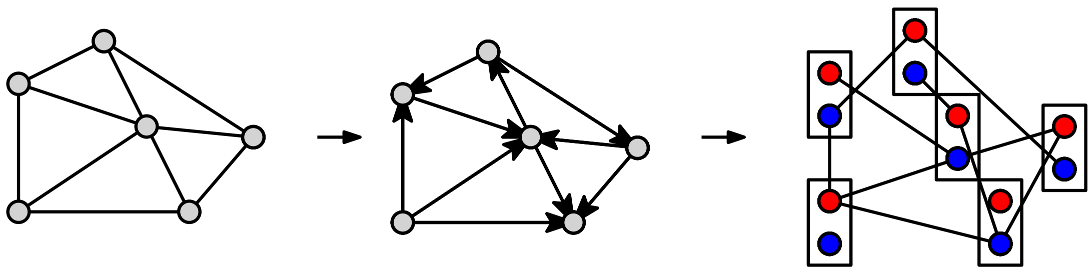
(3) Turn M_B into a matching M_G in G



degree-2 graph

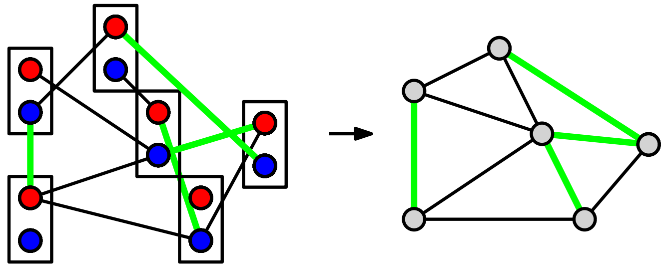
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G

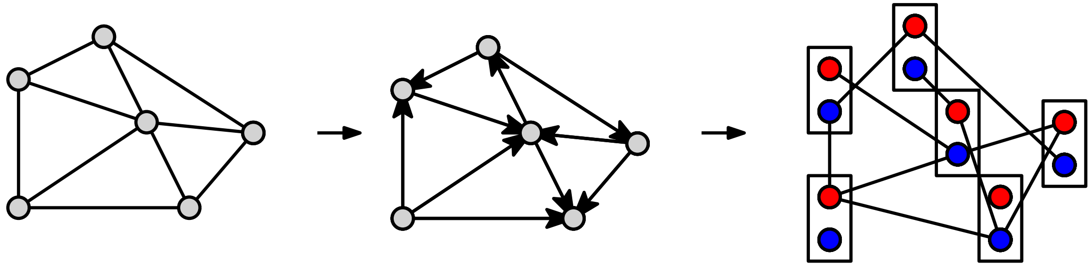


degree-2 graph

but not bipartite

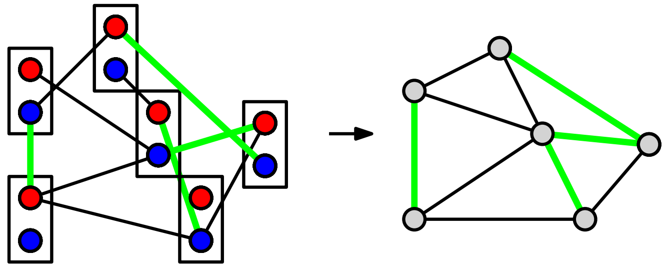
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G



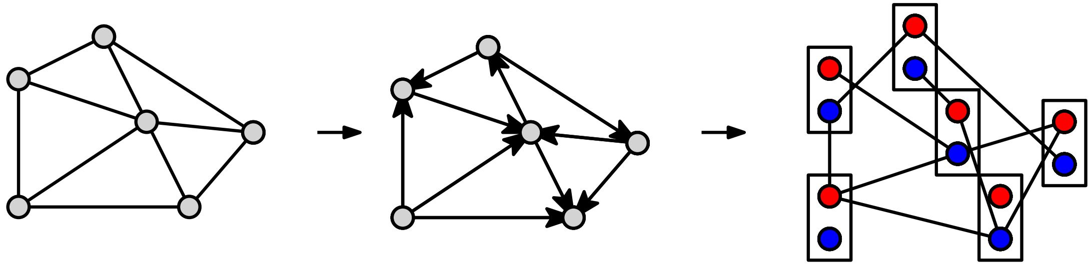
degree-2 graph

but not bipartite

maximal matching in $O(\log^* n)$

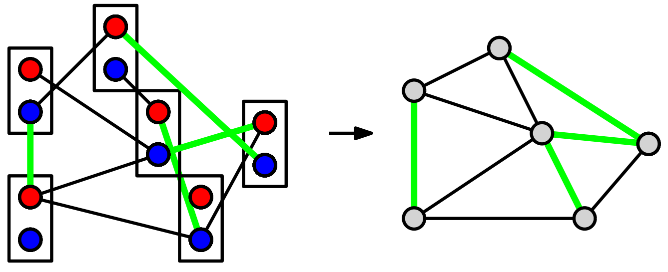
From Bipartite to General Graphs

(1) Turn G into auxiliary bipartite graph B



(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G



degree-2 graph

but not bipartite

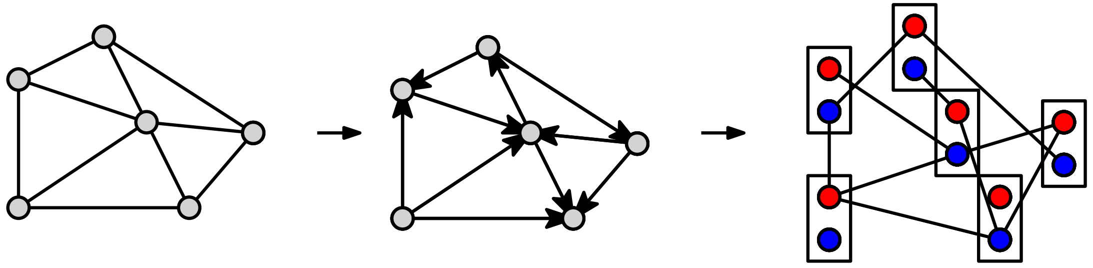
maximal matching in $O(\log^* n)$

matching at least $\frac{1}{3}$-fraction of the edges

From Bipartite to General Graphs

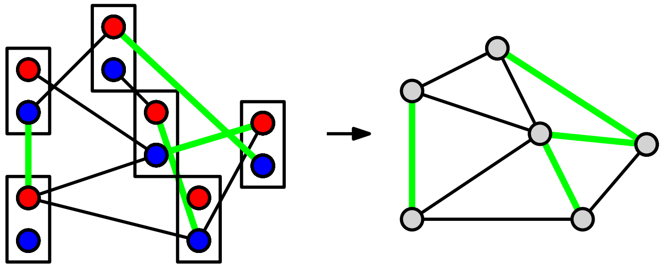
$O(1)$-approximation in $O(\log^2 \Delta + \log^* n)$ rounds

(1) Turn G into auxiliary bipartite graph B



(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G



degree-2 graph

but not bipartite

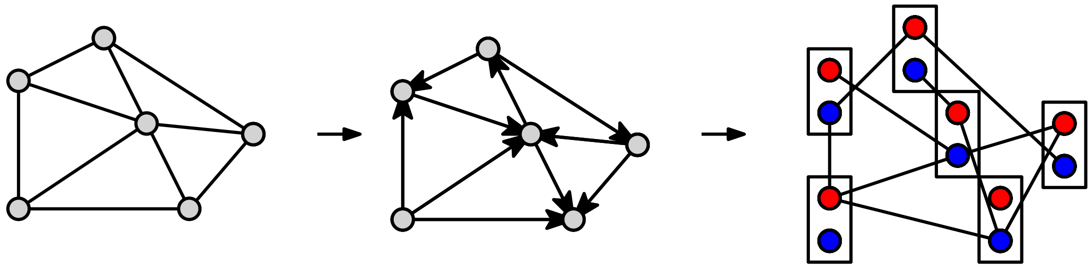
maximal matching in $O(\log^* n)$

matching at least $\frac{1}{3}$-fraction of the edges

From Bipartite to General Graphs

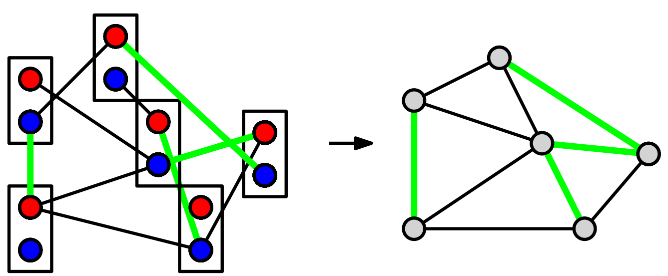
$O(1)$-approximation in $O(\log^2 \Delta + \log^* n)$ rounds in general graphs

(1) Turn G into auxiliary bipartite graph B



(2) Find $O(1)$-approximate matching M_B in B

(3) Turn M_B into a matching M_G in G



degree-2 graph

but not bipartite

maximal matching in $O(\log^* n)$

matching at least $\frac{1}{3}$-fraction of the edges

Improving Approximation Ratio

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor

Improving Approximation Ratio

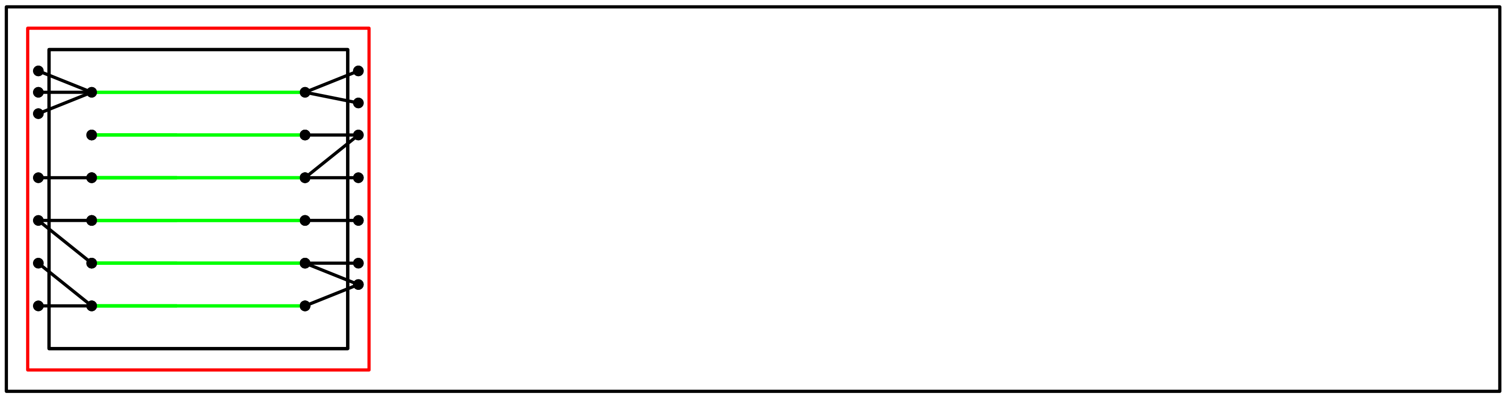
Observation: matching size in remainder graph reduces by $O(1)$-factor

M^*

$|M| \geq \frac{1}{c}|M^*|$

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor



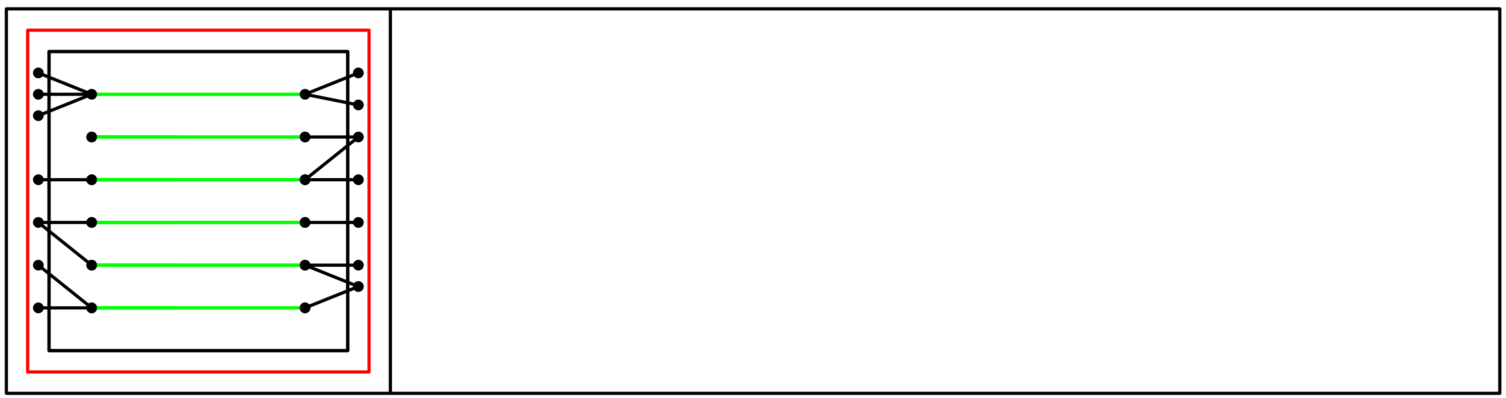
G

M^*

$|M| \geq \frac{1}{c}|M^*|$

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor



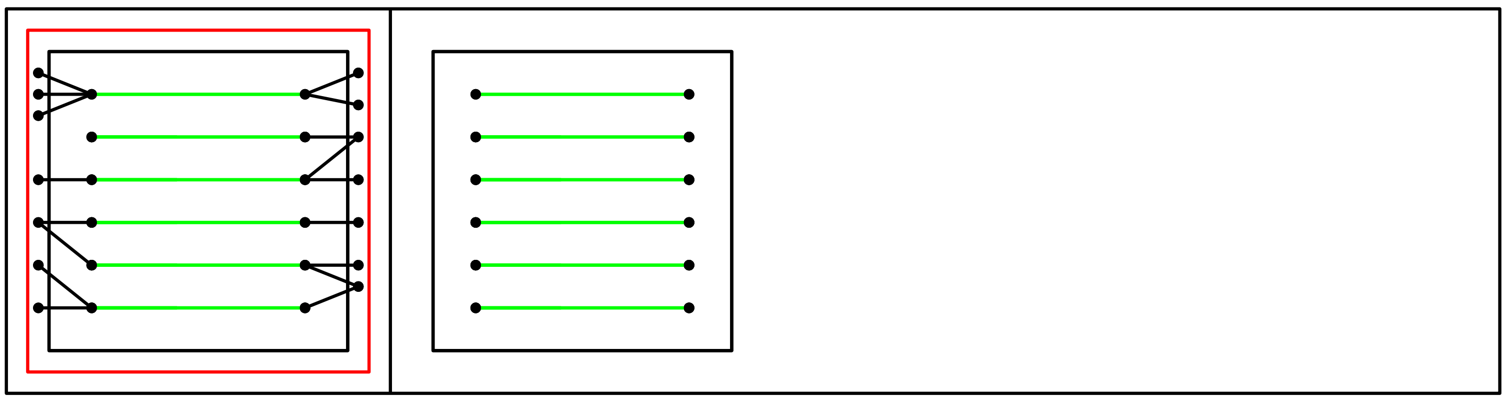
G $\qquad$ G_1

M^* $\qquad$ $|M_1^*| \leq \left(1 - \frac{1}{c}\right) |M^*|$

$|M| \geq \frac{1}{c} |M^*|$

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor



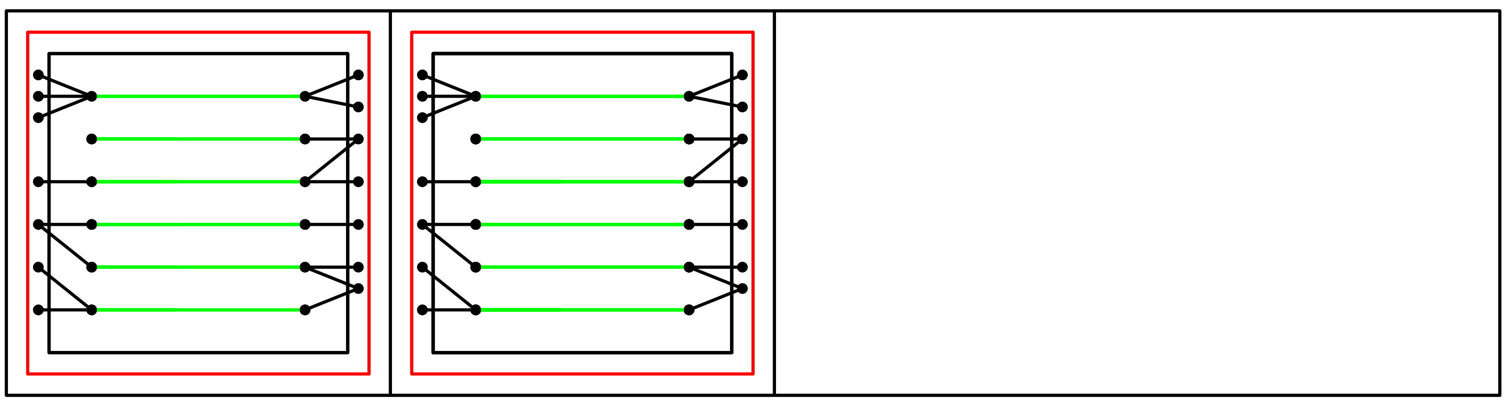
G $\qquad$ G_1

M^* $\qquad$ $|M_1^*| \leq \left(1 - \frac{1}{c}\right) |M^*|$

$|M| \geq \frac{1}{c} |M^*|$ $\qquad$ $|M_1| \geq \frac{1}{c} |M_1^*|$

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor



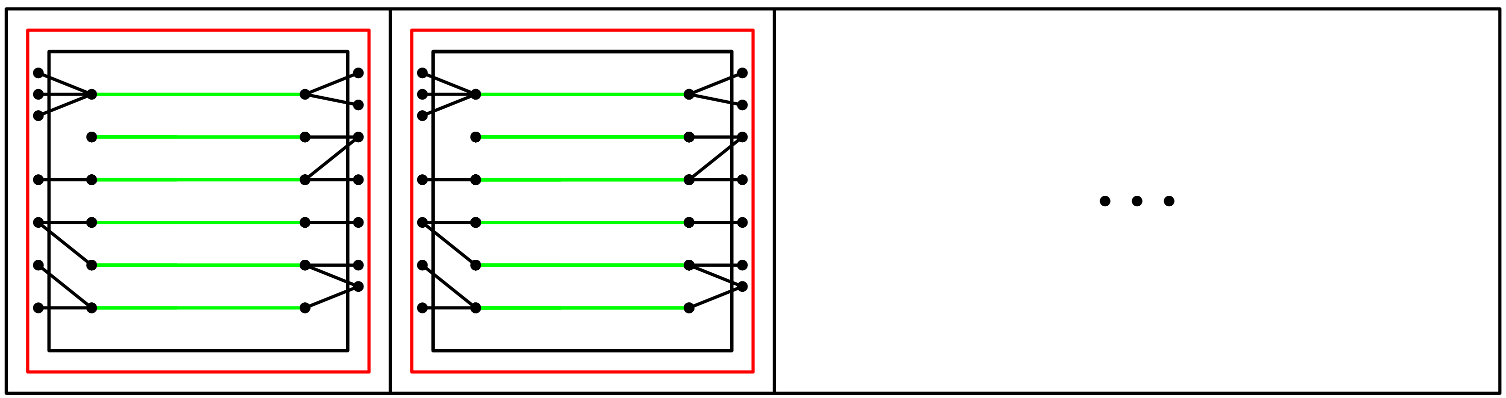
G $\qquad$ G_1

M^* $\qquad$ $|M_1^*| \leq \left(1 - \frac{1}{c}\right) |M^*|$

$|M| \geq \frac{1}{c} |M^*|$ $\qquad$ $|M_1| \geq \frac{1}{c} |M_1^*|$

Improving Approximation Ratio

Observation: matching size in remainder graph reduces by $O(1)$-factor



G | G_1

M^*

$|M_1^*| \leq \left(1 - \frac{1}{c}\right) |M^*|$

$|M| \geq \frac{1}{c} |M^*|$

$|M_1| \geq \frac{1}{c} |M_1^*|$

inductively: $|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$

Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

Improving Approximation Ratio

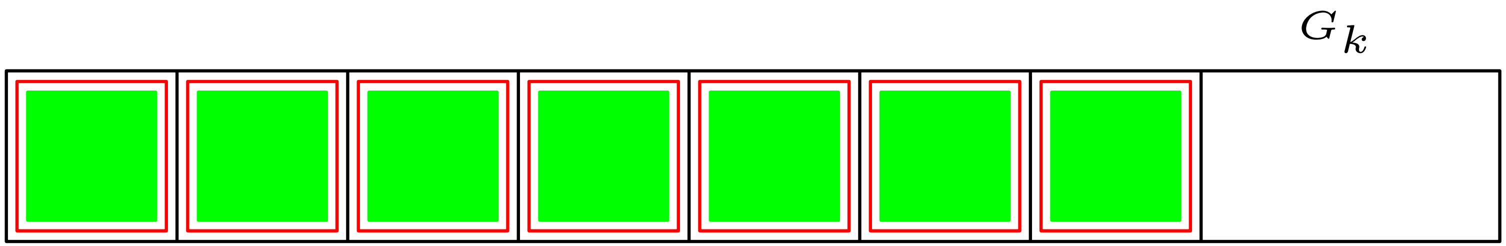
$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

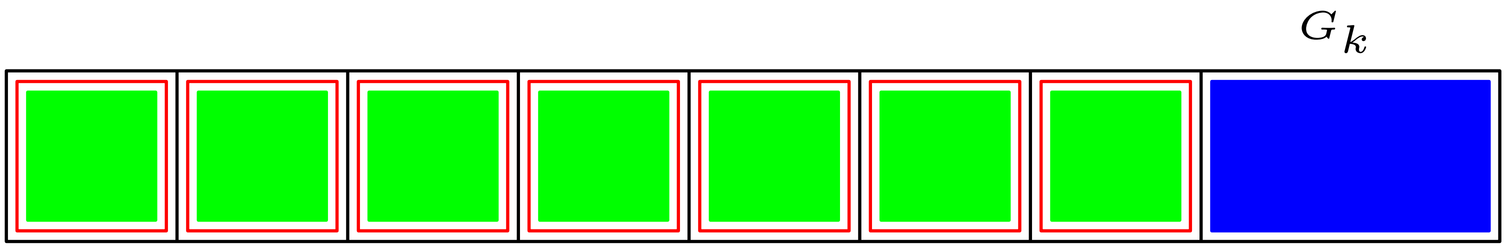
Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$



Improving Approximation Ratio

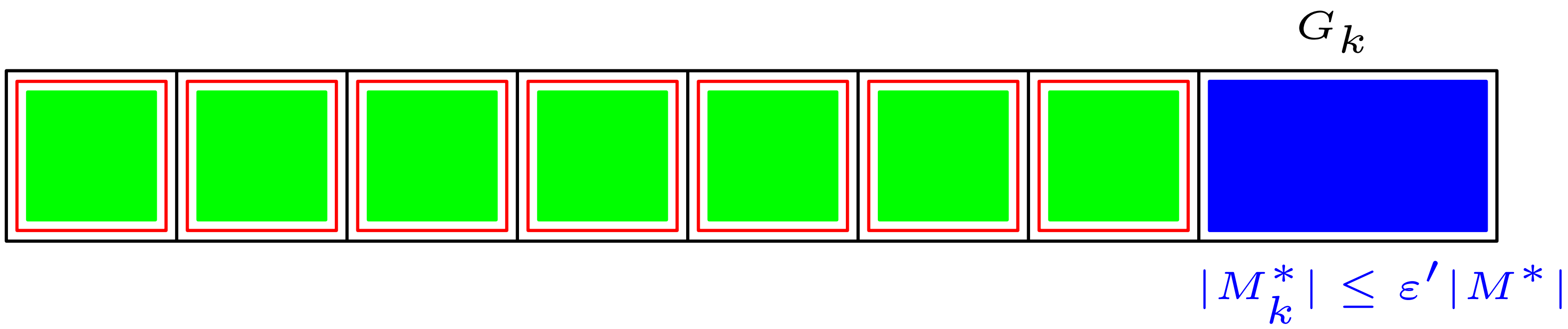
$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$



$|M_k^*| \leq \varepsilon' |M^*|$

Improving Approximation Ratio

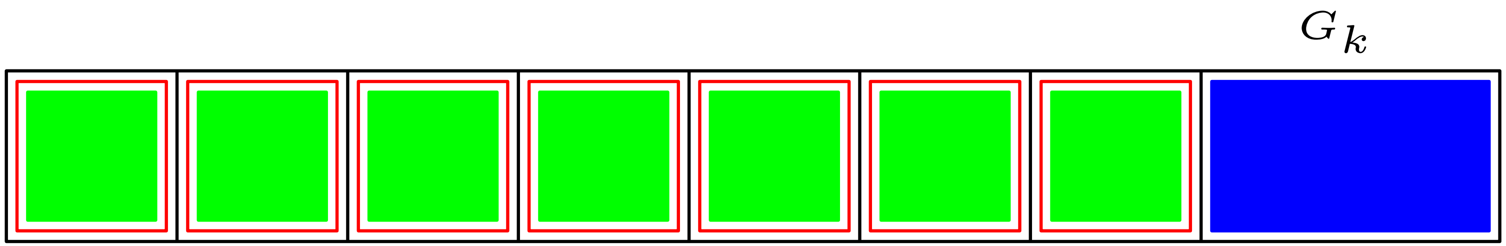
$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$



$M_k^* \cup \bigcup_{i=0}^{k-1} M_i$ is maximal in G

Improving Approximation Ratio

$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$



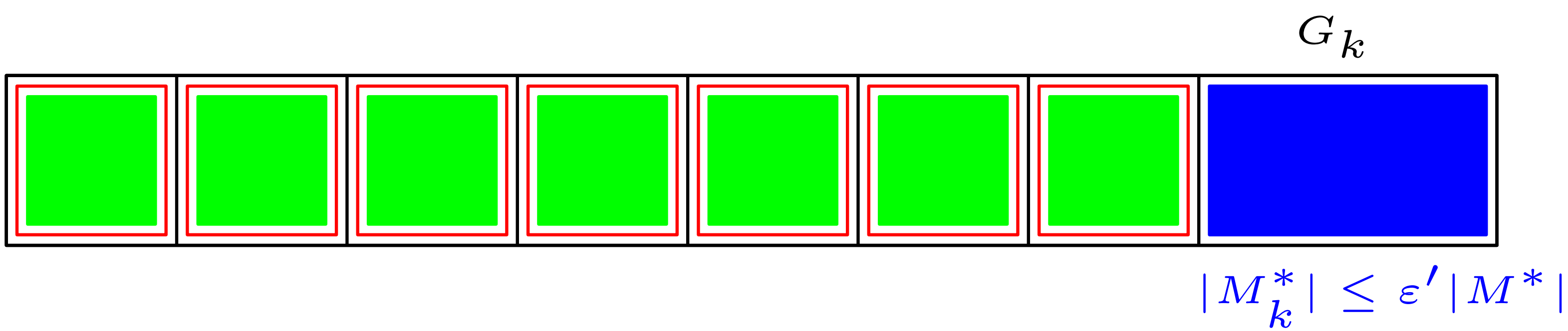
$|M_k^*| \leq \varepsilon' |M^*|$

$M_k^* \cup \bigcup_{i=0}^{k-1} M_i$ is maximal in G

$|M_k^*| + \sum_{i=0}^{k-1} |M_i| \geq \frac{1}{2} |M^*|$

Improving Approximation Ratio

$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$



$M_k^* \cup \bigcup_{i=0}^{k-1} M_i$ is maximal in G

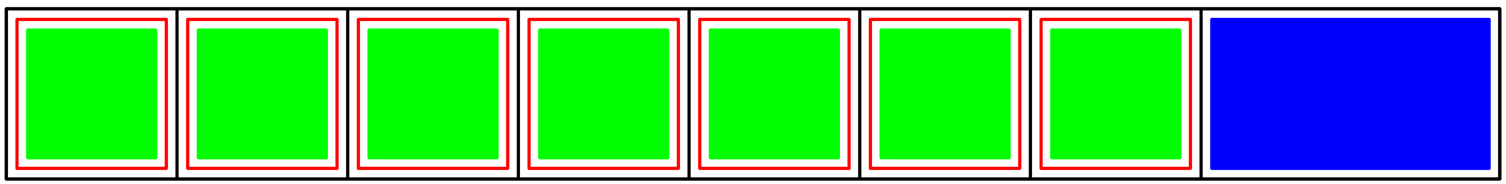
$|M_k^*| + \sum_{i=0}^{k-1} |M_i| \geq \frac{1}{2} |M^*|$

- $O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n)$ rounds for $(2 + \varepsilon)$-approximation

Improving Approximation Ratio

$$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$$

G_k



$|M_k^*| \leq \varepsilon' |M^*|$

$M_k^* \cup \bigcup_{i=0}^{k-1} M_i$ is maximal in G

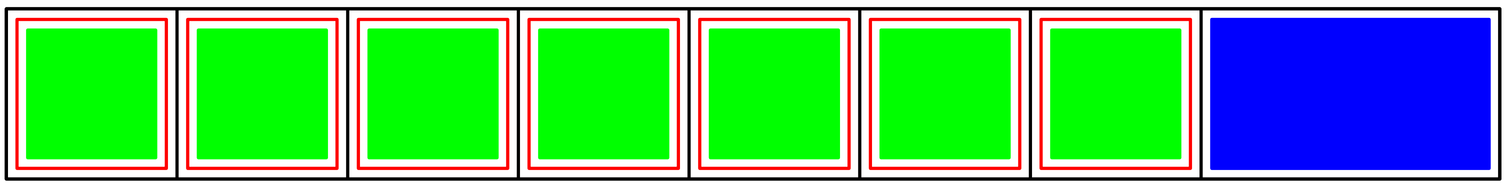
$|M_k^*| + \sum_{i=0}^{k-1} |M_i| \geq \frac{1}{2} |M^*|$

- $O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n)$ rounds for $(2+\varepsilon)$-approximation

  direct: $O((\log^2 \Delta + \log^* n) \log \frac{1}{\varepsilon})$

Improving Approximation Ratio

$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$



$|M_k^*| \leq \varepsilon' |M^*|$

$M_k^* \cup \bigcup_{i=0}^{k-1} M_i$ is maximal in G

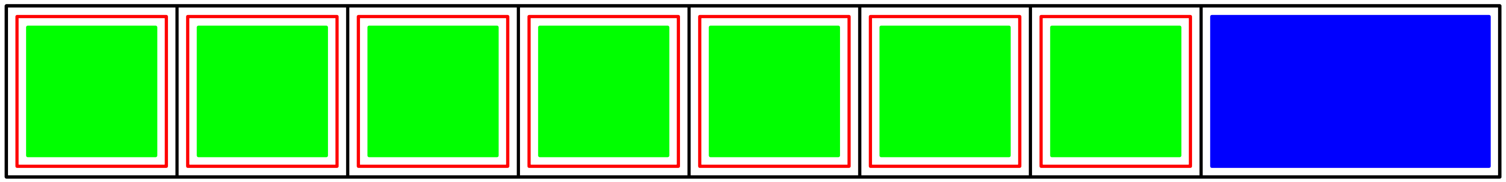
$|M_k^*| + \sum_{i=0}^{k-1} |M_i| \geq \frac{1}{2} |M^*|$

- $O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n)$ rounds for $(2+\varepsilon)$-approximation

  direct: $O((\log^2 \Delta + \log^* n) \log \frac{1}{\varepsilon})$
- $O(\log^2 \Delta \cdot \log n)$ rounds for maximal matching

Improving Approximation Ratio

$|M_i^*| \leq \left(1 - \frac{1}{c}\right)^i |M^*|$



$|M_k^*| \leq \varepsilon' |M^*|$

$M_k^* \cup \bigcup_{i=0}^{k-1} M_i$ is maximal in G $\qquad |M_k^*| + \sum_{i=0}^{k-1} |M_i| \geq \frac{1}{2} |M^*|$

- $O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n)$ rounds for $(2+\varepsilon)$-approximation

  direct: $O((\log^2 \Delta + \log^* n) \log \frac{1}{\varepsilon})$

- $O(\log^2 \Delta \cdot \log n)$ rounds for maximal matching

  after $k = O(\log n)$ iterations: $|M_k^*| \leq \frac{1}{2n} |M^*| < 1$

Recap: Our Results

Recap: Our Results

Our Result [F., Ghaffari 2017]

$O(\log^2 \Delta + \log^* n)$	for constant approximation
$O(\log^2 \Delta \cdot \log \frac{1}{\varepsilon} + \log^* n)$	for $(2+\varepsilon)$-approximation
$O(\log^2 \Delta \cdot \log n)$	for maximal matching (2-approximation)

using an $O\left(\log^2 \Delta\right)$-round $O(1)$-approximation algorithm for bipartite graphs

$O(\log^4 n)$	Hańćkowiak, Karoński, Panconesi [PODC'99]
$O(\Delta + \log^* n)$	Panconesi, Rizzi [DIST'01]
$\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$	Kuhn, Moscibroda, Wattenhofer [PODC'04], Linial [FOCS'87]

Open Questions & Conclusion

Open Questions & Conclusion

- Deterministic Complexity of Matching

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC'14])

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC’14])

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

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC'14])

    $O(\log^2 \Delta \cdot \log n)$
  - prove (or disprove) a lower bound $\Omega(\log n)$ for deterministic maximal matching (for $\Delta = \Omega(\log n)$)

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC'14])

    $O(\log^2 \Delta \cdot \log n)$
  - prove (or disprove) a lower bound $\Omega(\log n)$ for deterministic maximal matching (for $\Delta = \Omega(\log n)$)
- Rounding Method

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC'14])

    $O(\log^2 \Delta \cdot \log n)$
  - prove (or disprove) a lower bound $\Omega(\log n)$ for deterministic maximal matching (for $\Delta = \Omega(\log n)$)
- Rounding Method
  - completeness of rounding (Ghaffari, Kuhn, Maus [STOC'17])

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC'14])

    $O(\log^2 \Delta \cdot \log n)$
  - prove (or disprove) a lower bound $\Omega(\log n)$ for deterministic maximal matching (for $\Delta = \Omega(\log n)$)
- Rounding Method
  - completeness of rounding (Ghaffari, Kuhn, Maus [STOC'17])

    matching admits an efficient deterministic algorithm because
    matching admits efficient deterministic rounding

Open Questions & Conclusion

- Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log \log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC'14])

    $O(\log^2 \Delta \cdot \log n)$
  - prove (or disprove) a lower bound $\Omega(\log n)$ for deterministic maximal matching (for $\Delta = \Omega(\log n)$)
- Rounding Method
  - completeness of rounding (Ghaffari, Kuhn, Maus [STOC'17])

    matching admits an efficient deterministic algorithm because
    matching admits efficient deterministic rounding
  - deterministic poly $\log n$-round algorithm for $(2\Delta - 1)$-edge-coloring (F., Ghaffari, Kuhn (2017)) using deterministic rounding (for hypergraph matching)

Open Questions & Conclusion

- ## Deterministic Complexity of Matching
  - lower bound $\Omega\left(\frac{\log \Delta}{\log\log \Delta} + \log^* n\right)$ for any $O(1)$-approximation
  - conjecture: no $o(\Delta) + O(\log^* n)$ for maximal matching (Göös, Hirvonen, Suomela [PODC'14])

    $O(\log^2 \Delta \cdot \log n)$
  - prove (or disprove) a lower bound $\Omega(\log n)$ for deterministic maximal matching (for $\Delta = \Omega(\log n)$)

- ## Rounding Method
  - completeness of rounding (Ghaffari, Kuhn, Maus [STOC'17])

    matching admits an efficient deterministic algorithm because
    matching admits efficient deterministic rounding
  - deterministic poly $\log n$-round algorithm for $(2\Delta - 1)$-edge-coloring (F., Ghaffari, Kuhn (2017)) using deterministic rounding (for hypergraph matching)
  - more general deterministic rounding method?