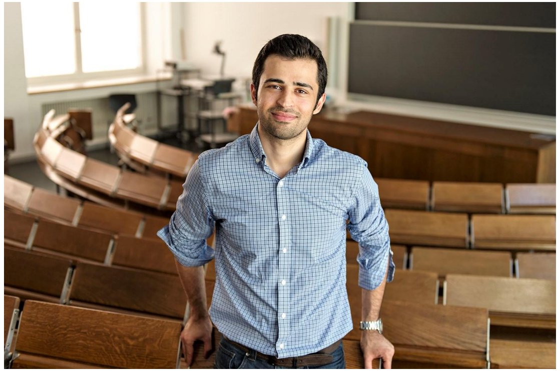
Distributed Derandomization via Network Decomposition

Longer talk



Mohsen Ghaffari (ETH), Vasek Rozhon (ETH)

Plan

1. More on **LOCAL** and **CONGEST** model
2. A deterministic algorithm for network decomposition.
    a. Sequential algorithm
    b. Distributed algorithm
3. Applications
    a. Derandomization and a bigger picture of the **LOCAL** model
    b. Δ+1 coloring, MIS, Lovász local lemma
    c. **CONGEST** model and open problems

Plan

1. More on **LOCAL** and **CONGEST** model

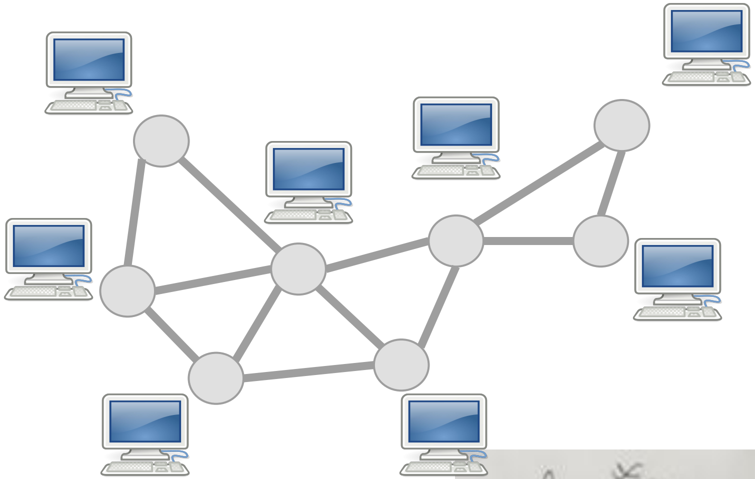
The **LOCAL** model of distributed graph algorithms

- Undirected graph on n nodes
- One computer in each node

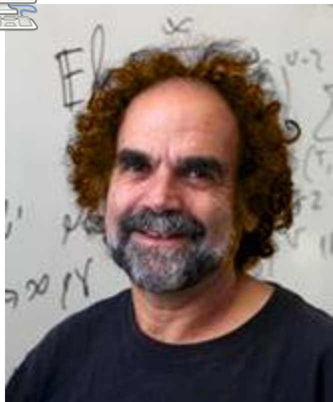
- Synchronous message passing rounds
- Unbounded message size and computation

- Initially, nodes know only (upper bound on) n and their unique $O(\log n)$ bit identifier
- In the end, each node should know its part of output

- Time complexity: number of rounds



LOCAL model
[Linial FOCS'87]



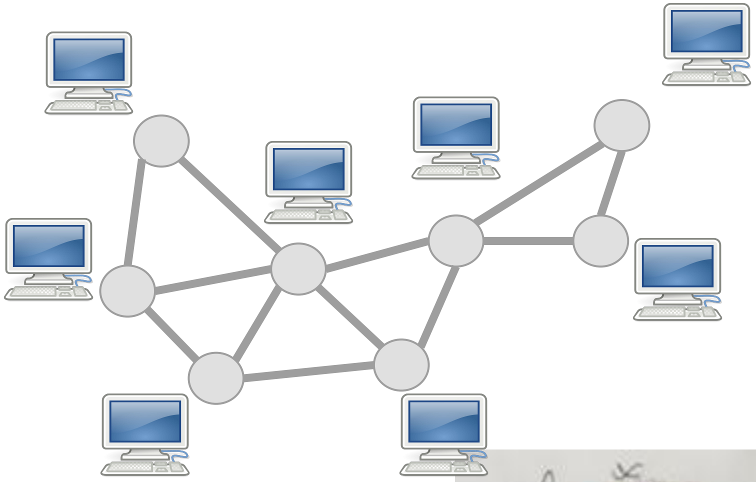
The **LOCAL** model of distributed graph algorithms

- Undirected graph on n nodes
- One computer in each node

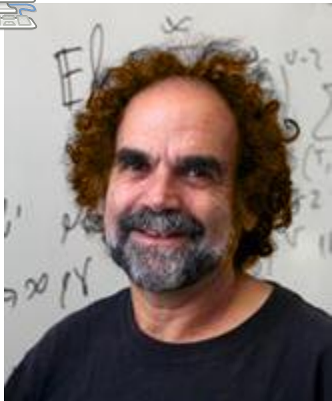
- Synchronous message passing rounds
- Unbounded message size and computation

- Initially, nodes know only (upper bound on) n and their unique $O(\log n)$ bit identifier
- In the end, each node should know its part of output

- Time complexity: number of rounds



LOCAL model
[Linial FOCS’87]



The **LOCAL** model of distributed graph algorithms

“*unbounded message size and computation*”:

CONGEST model: message size bounded to $O(\log n)$.

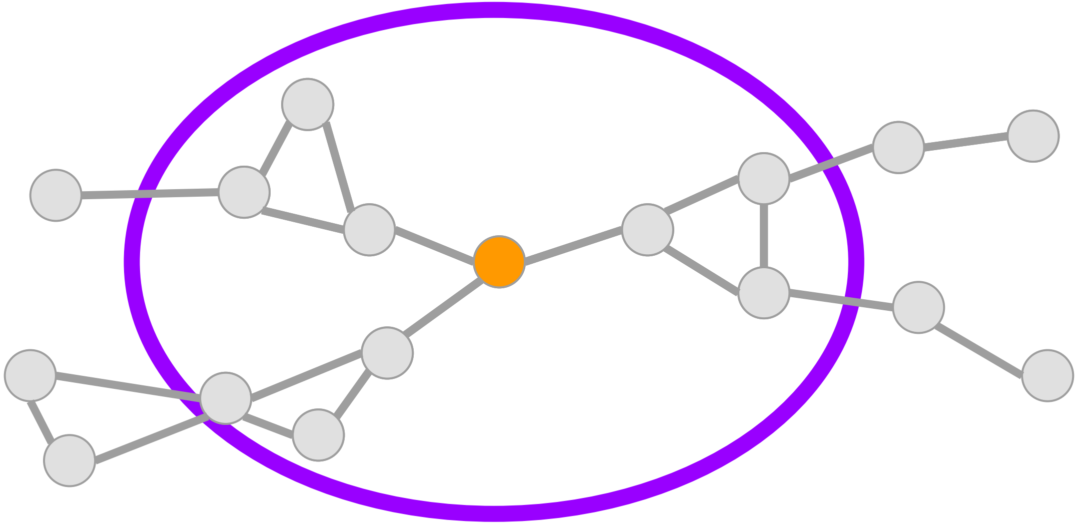
deterministic **LOCAL** algorithm is a function mapping neighbourhoods to labels.

The **LOCAL** model of distributed graph algorithms

"*unbounded message size and computation*":

CONGEST model: message size bounded to $O(\log n)$.

deterministic **LOCAL** algorithm is a function mapping neighbourhoods to labels.

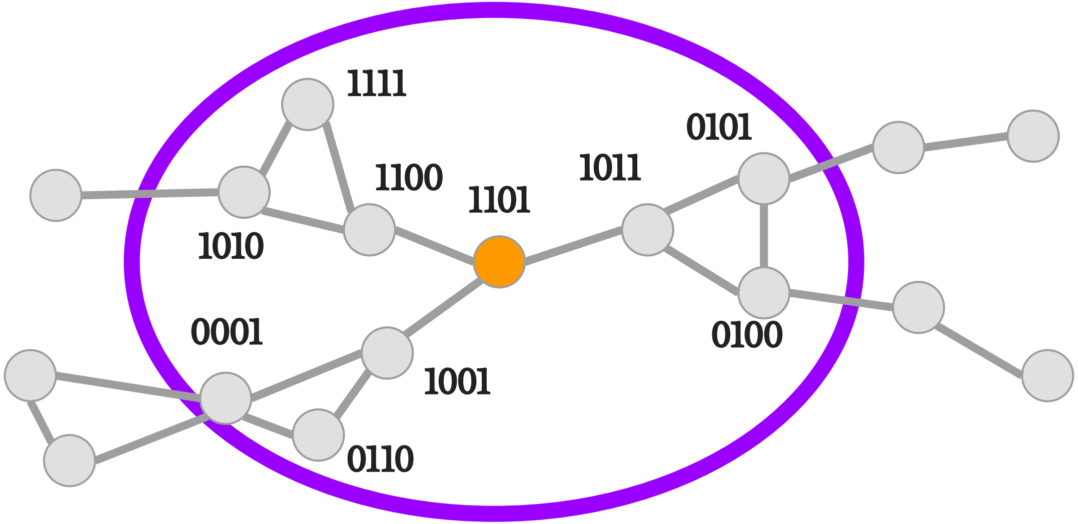


The **LOCAL** model of distributed graph algorithms

"*unbounded message size and computation*":

CONGEST model: message size bounded to $O(\log n)$.

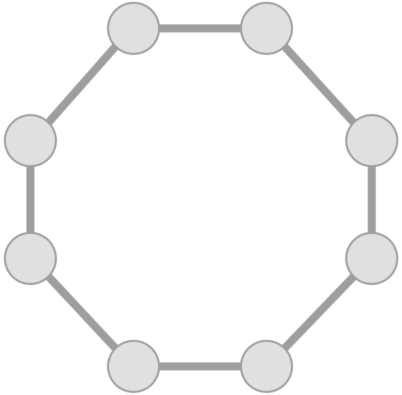
deterministic **LOCAL** algorithm is a function mapping neighbourhoods to labels.



The **LOCAL** model of distributed graph algorithms

Why unique $O(\log n)$ bit identifier?

Otherwise, not much to be done with deterministic algorithms, especially on vertex-transitive graphs!



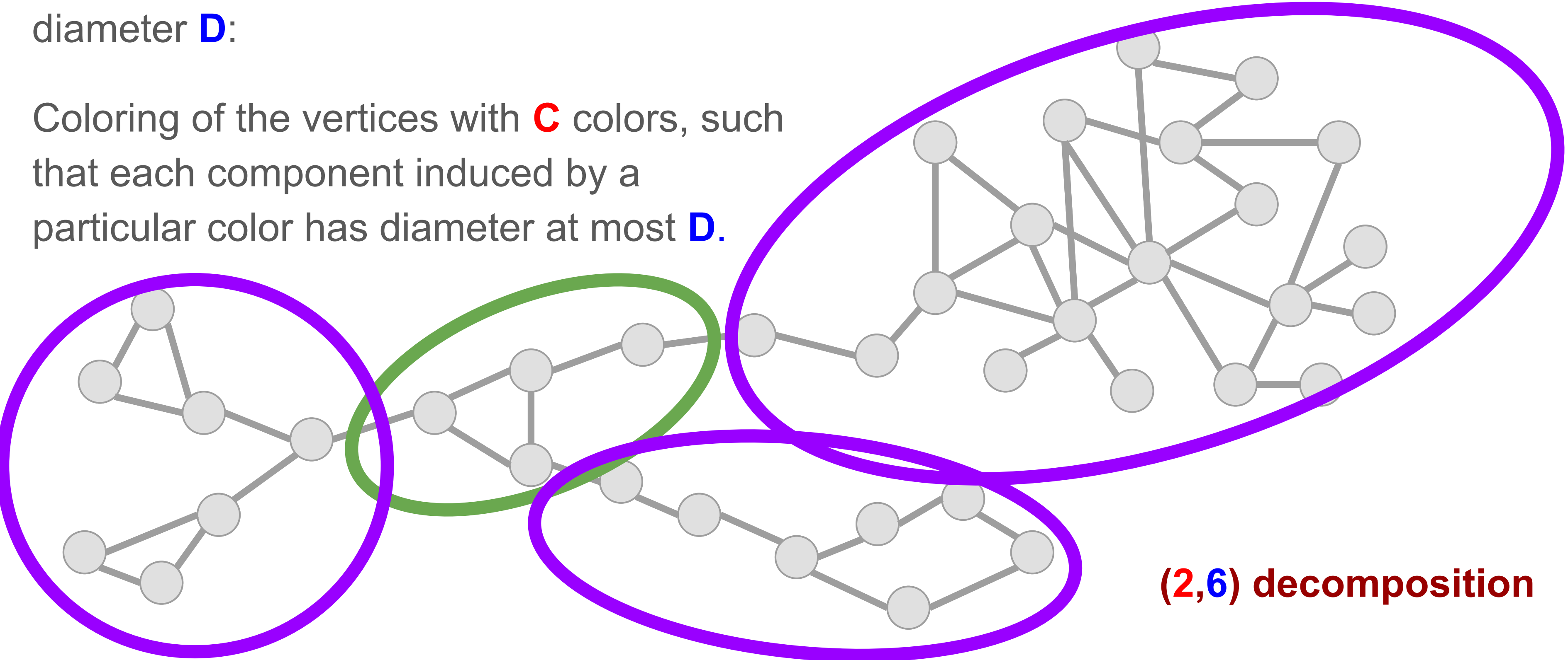
Plan

1. More on **LOCAL** and **CONGEST** model
2. A deterministic algorithm for network decomposition.
    a. Sequential algorithm

Network decomposition

Network decomposition with **C** colors and diameter **D**:

Coloring of the vertices with **C** colors, such that each component induced by a particular color has diameter at most **D**.

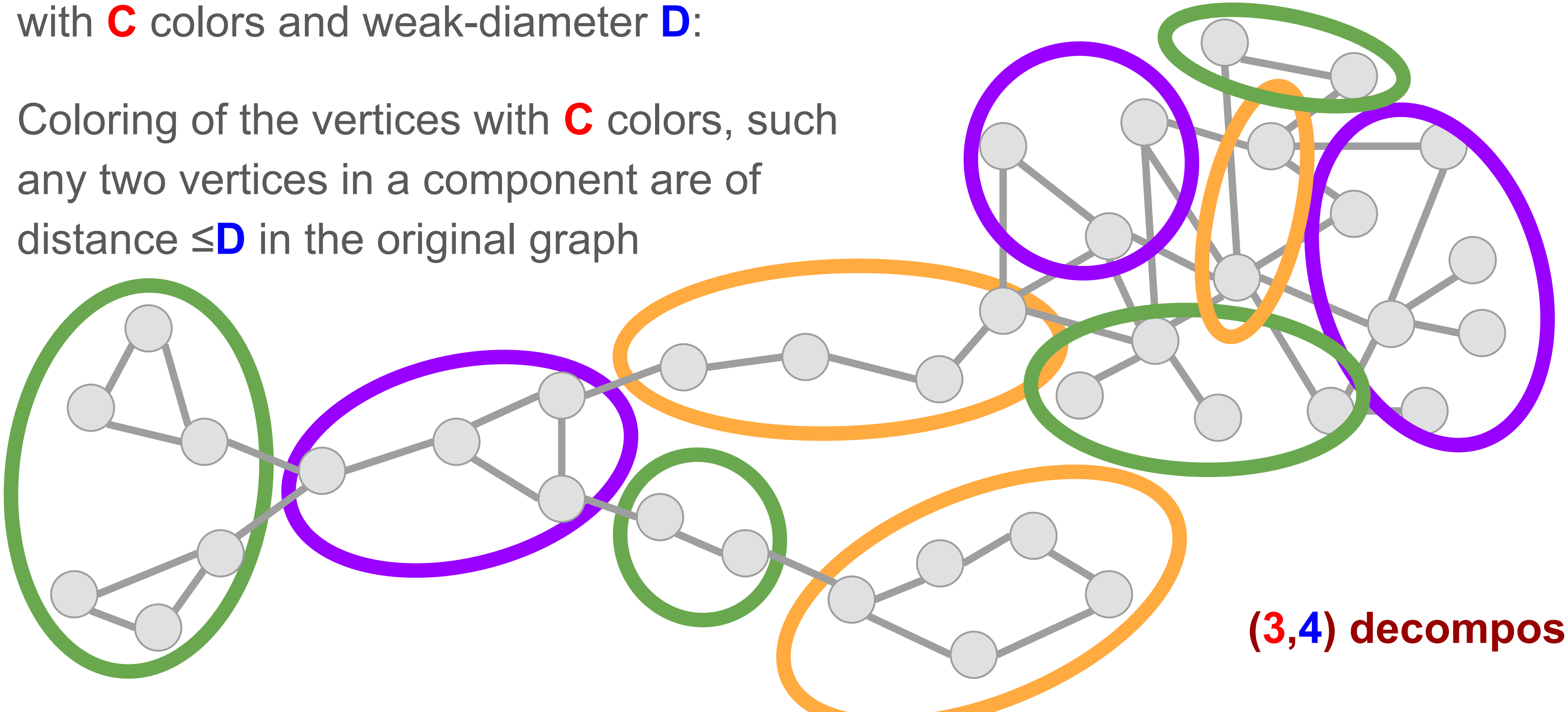


(2,6) decomposition

Weak-diameter network decomposition

Weak-diameter network decomposition
with **C** colors and weak-diameter **D**:

Coloring of the vertices with **C** colors, such any two vertices in a component are of distance ≤**D** in the original graph



(3,4) decomposition

Network decomposition

But is there such a thing (with reasonable parameters)?

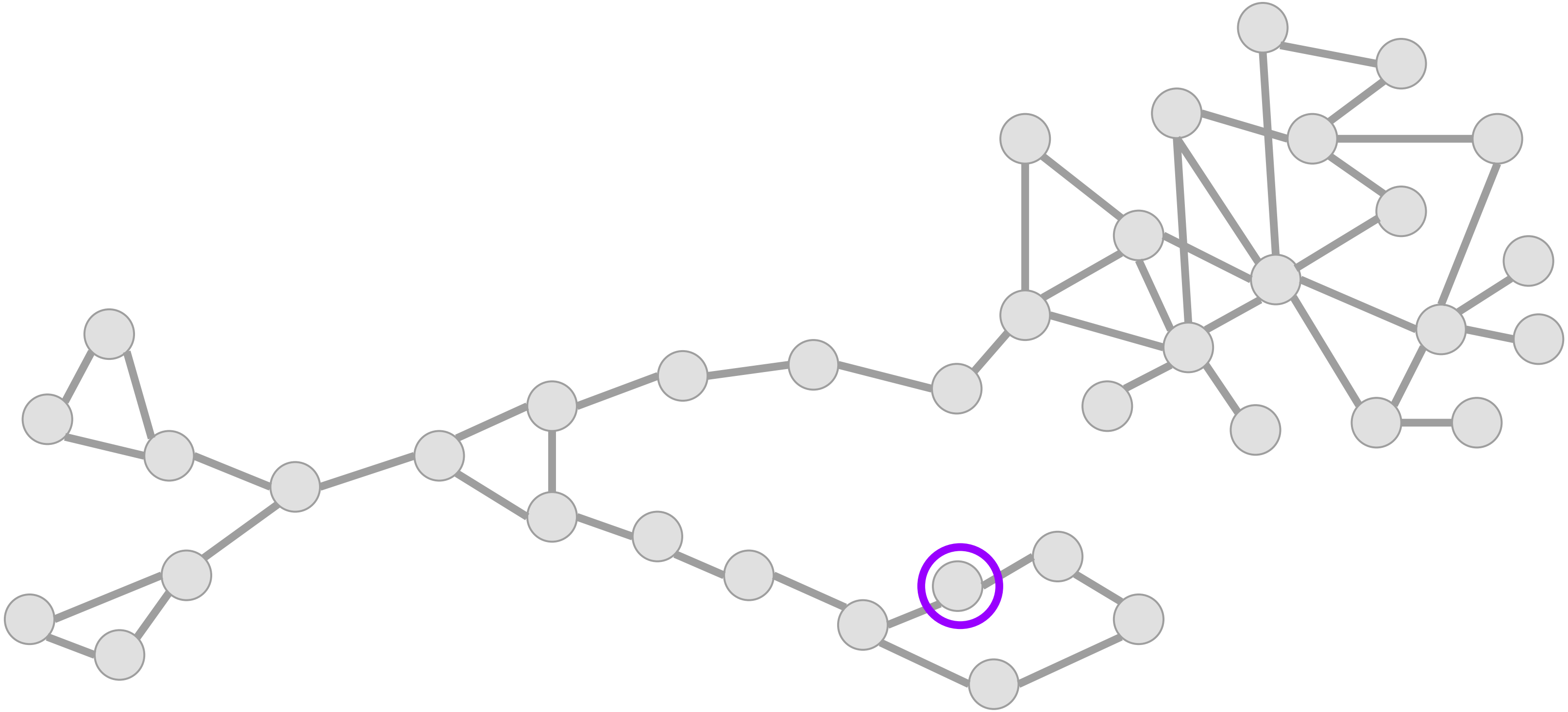
Yes, we let's see a sequential algorithm for ($O(\log n)$, $O(\log n)$) network decomposition.

(Sequential) ball carving

1. clusters at least ½ fraction of vertices
2. such that each cluster has diameter $O(\log n)$ and
3. clusters are non-adjacent

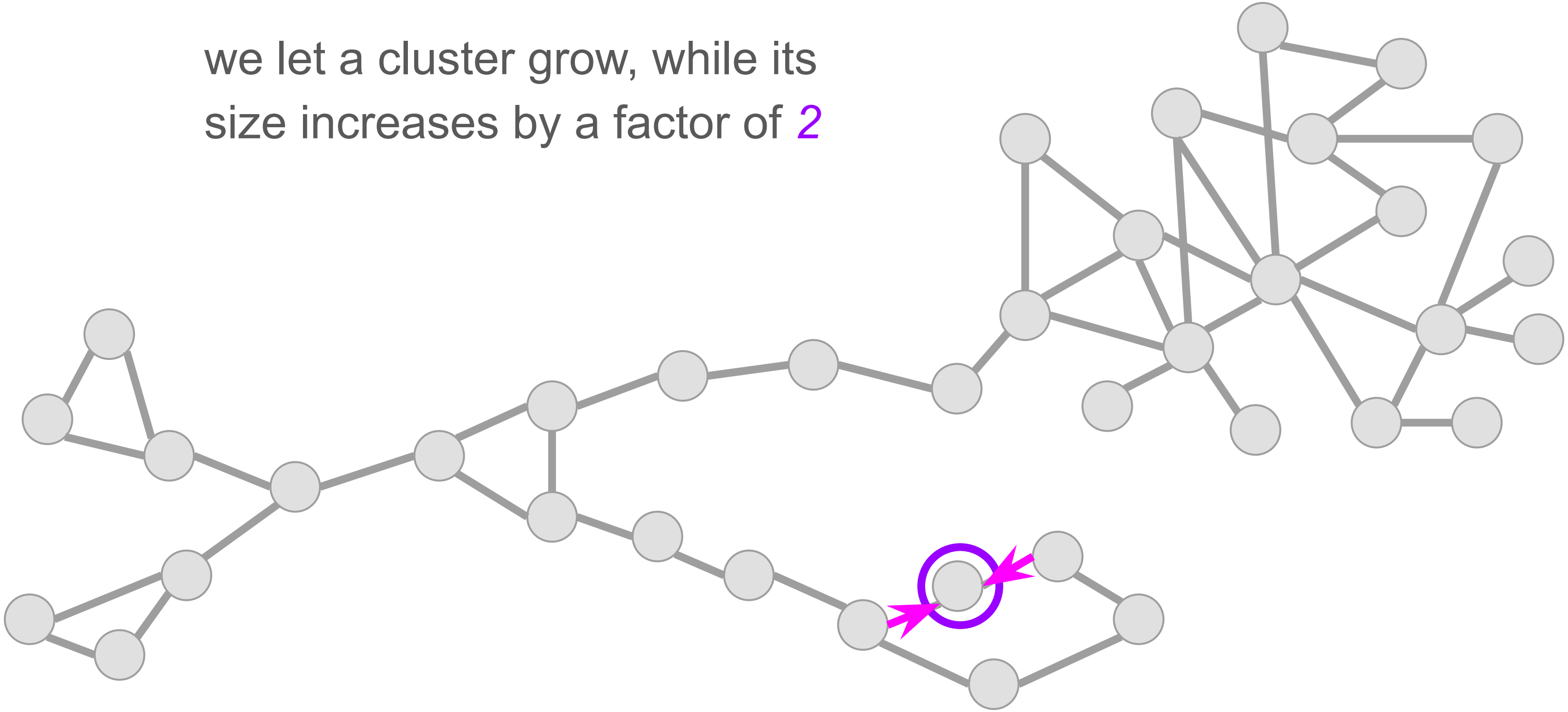
⇒($O(\log n)$, $O(\log n)$) network decomposition by repeated application

Sequential ball carving

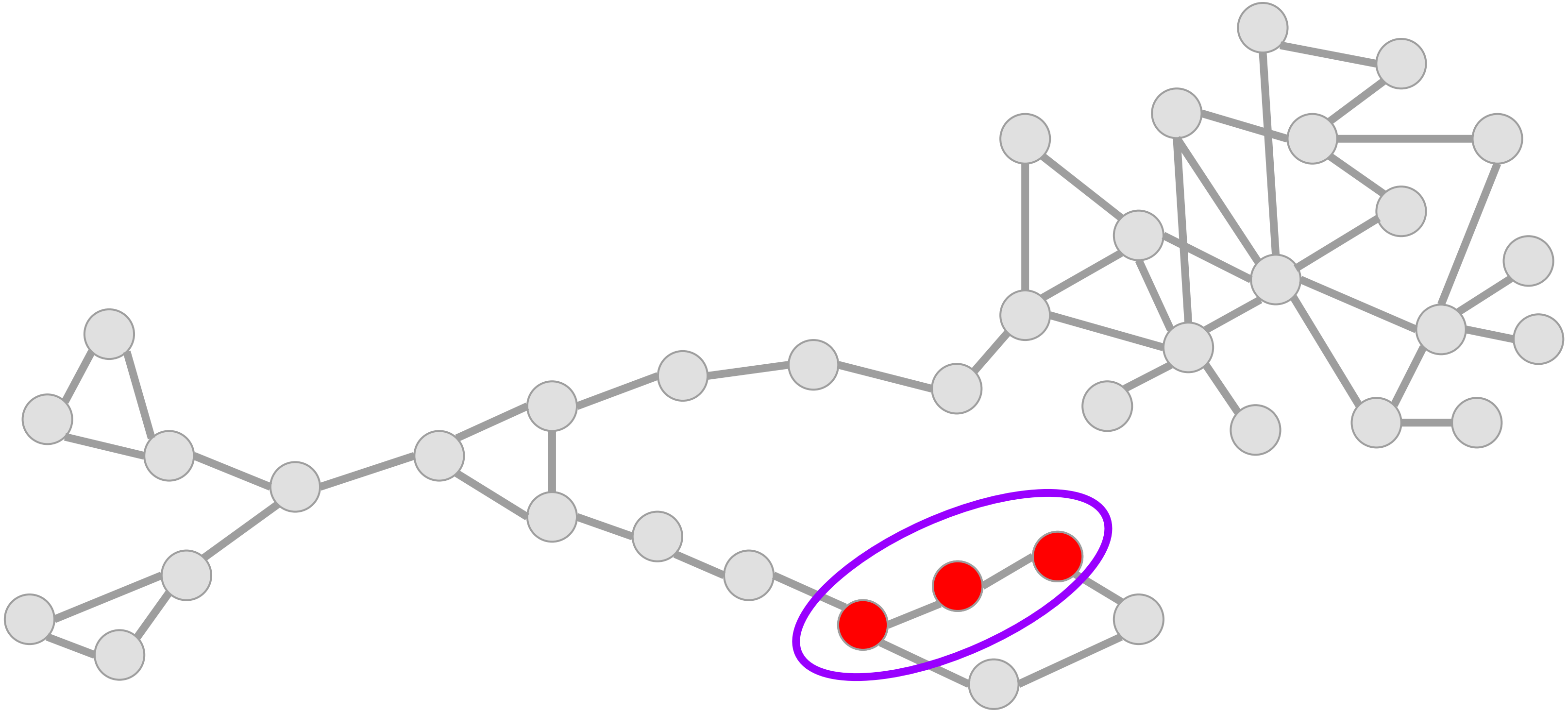


Sequential ball carving

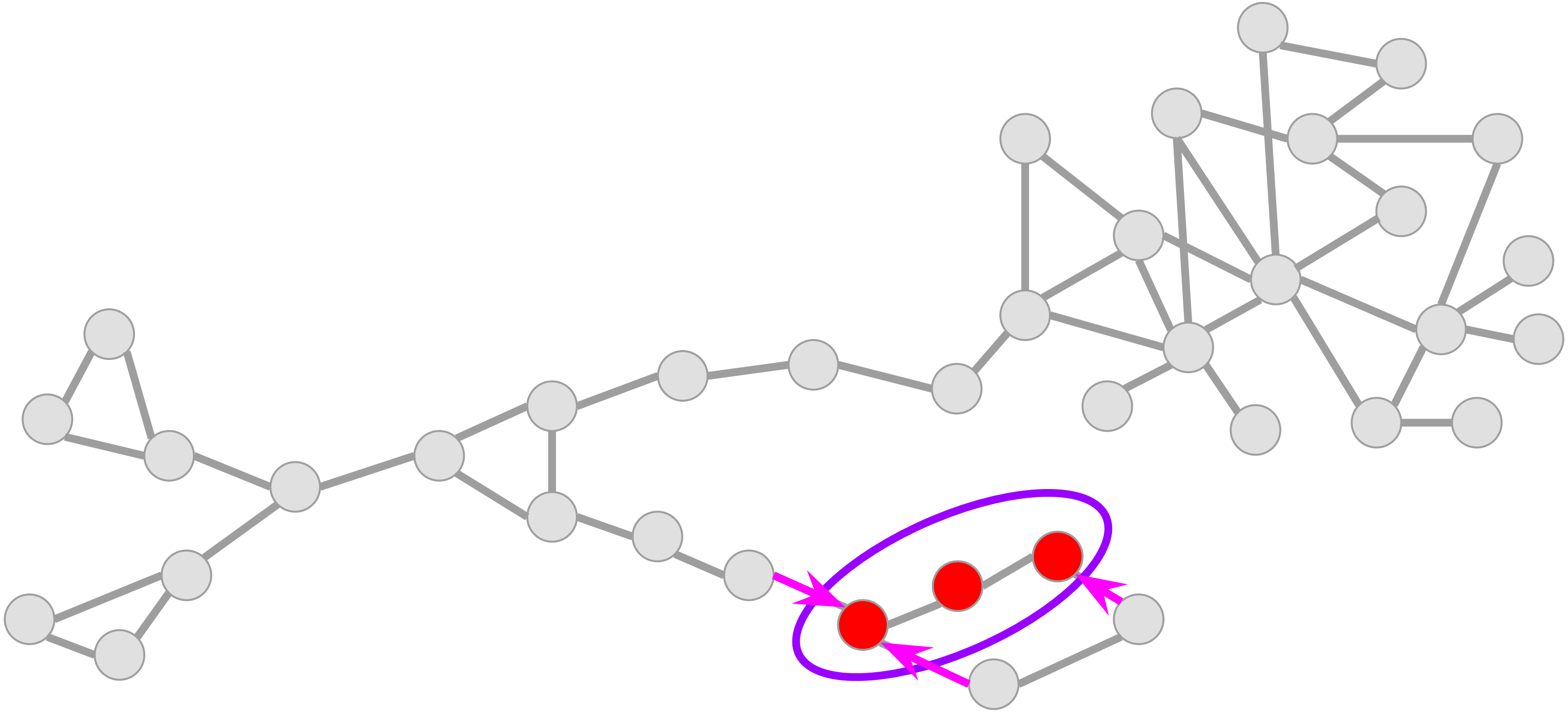
we let a cluster grow, while its
size increases by a factor of 2



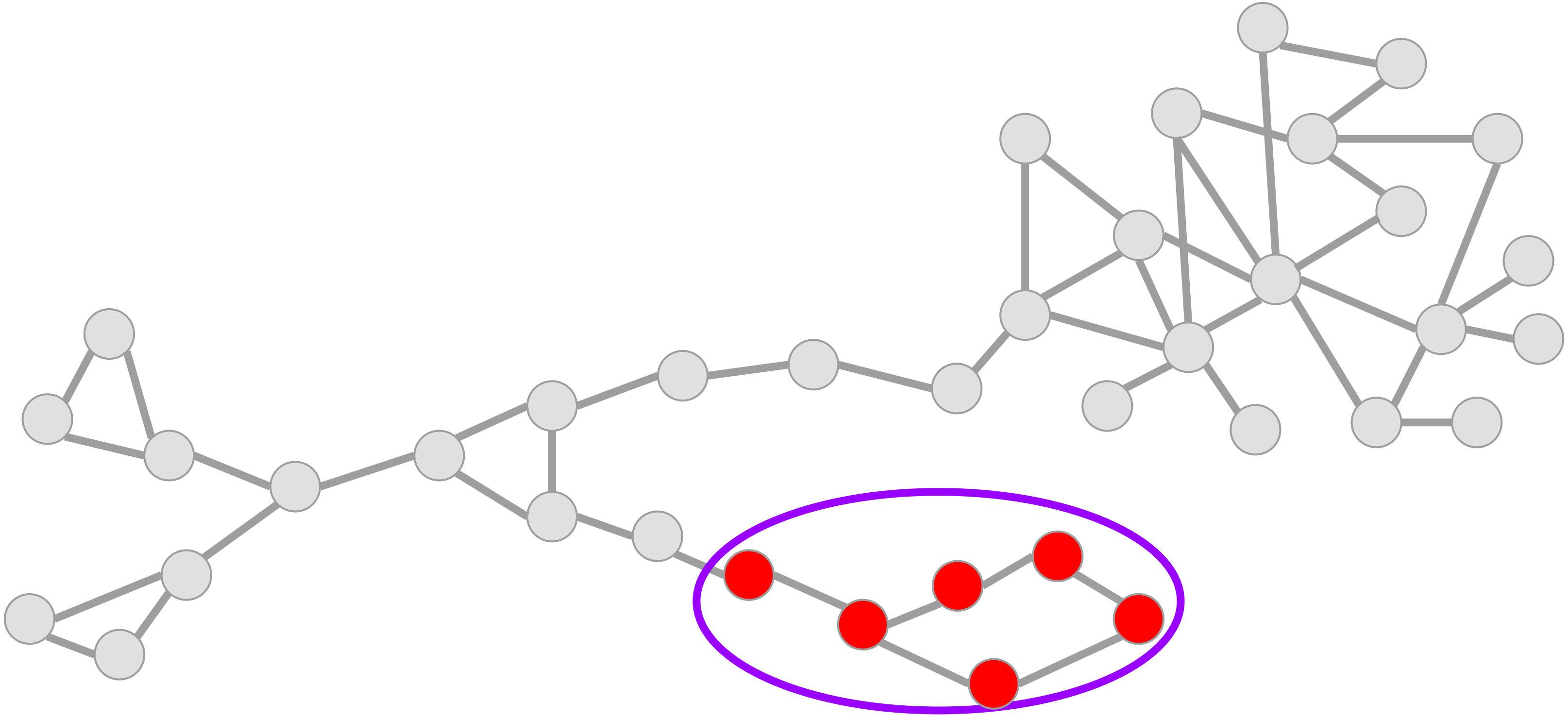
Sequential ball carving



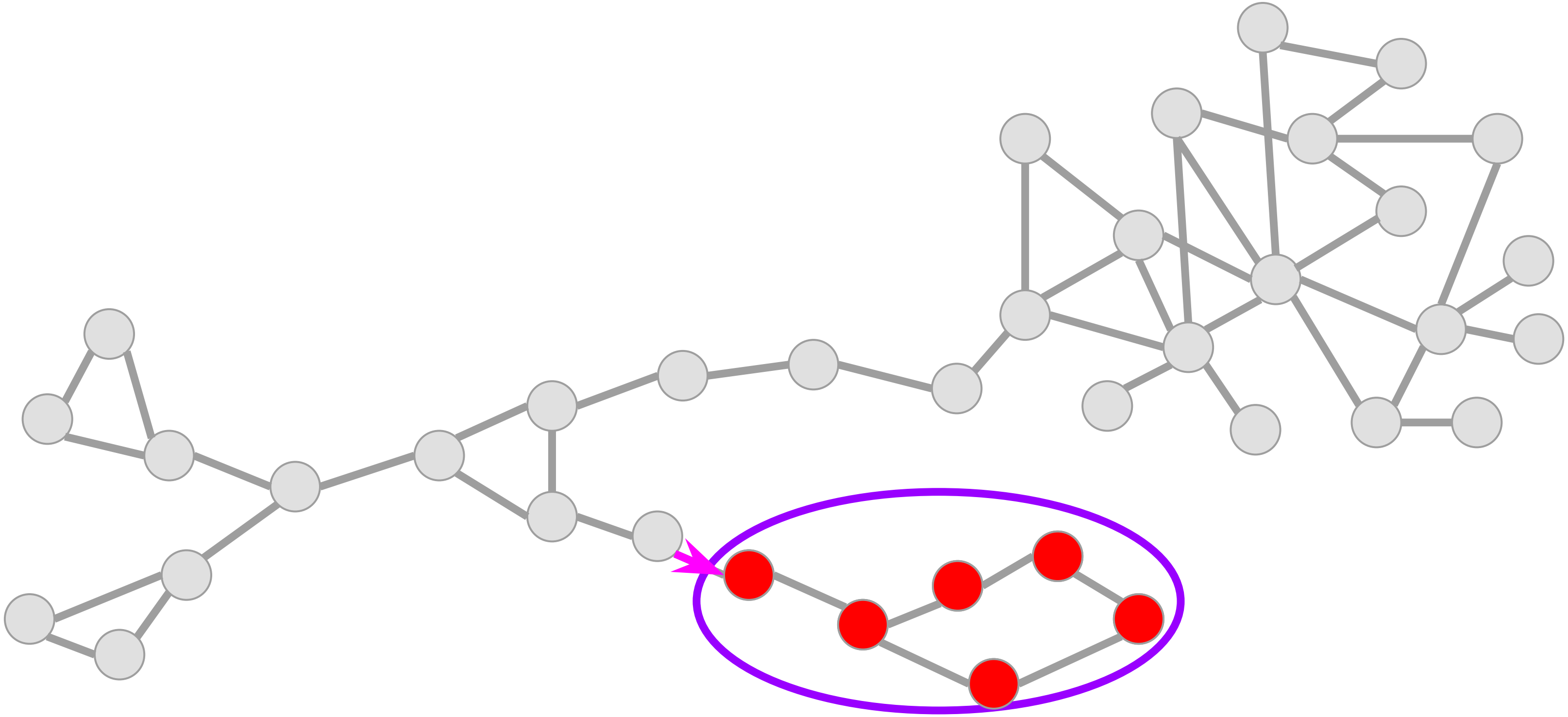
Sequential ball carving



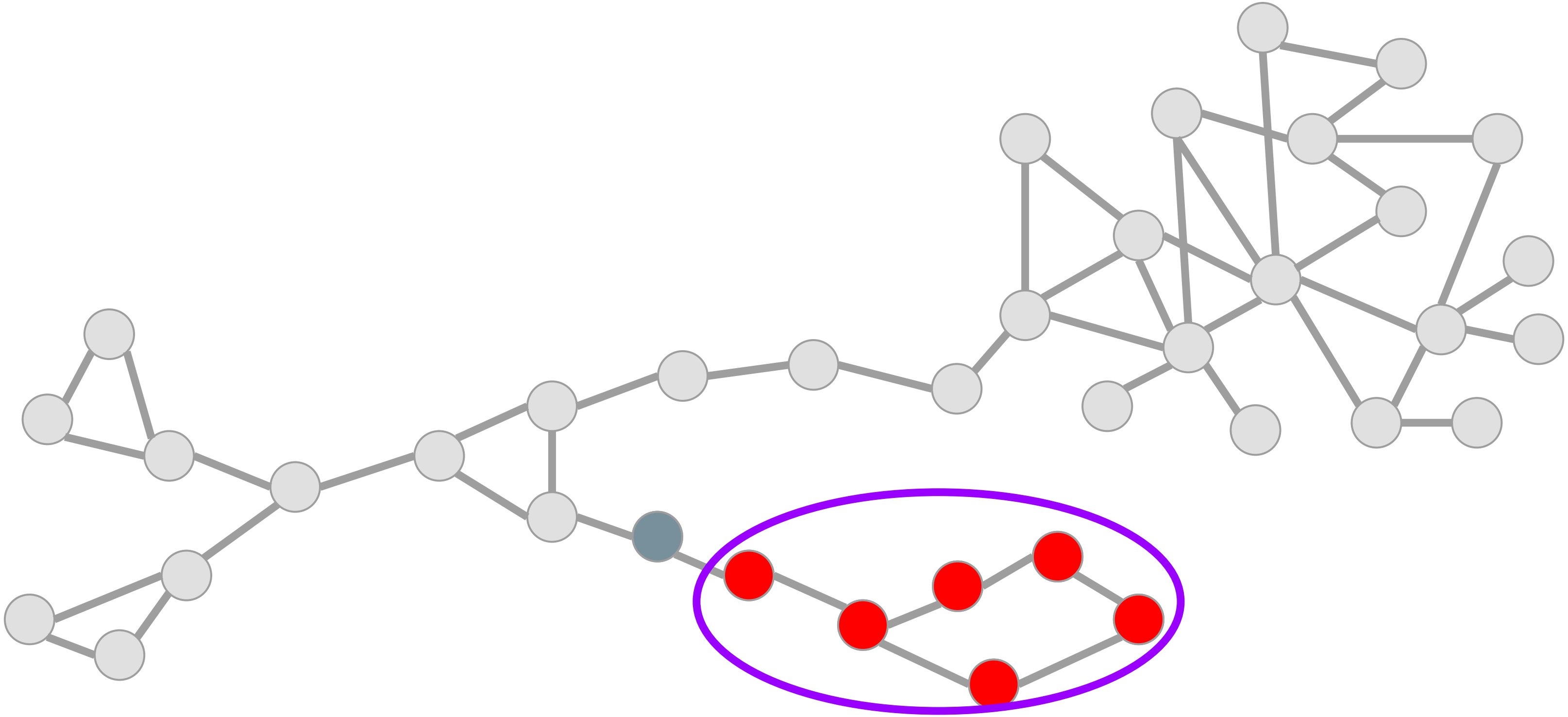
Sequential ball carving



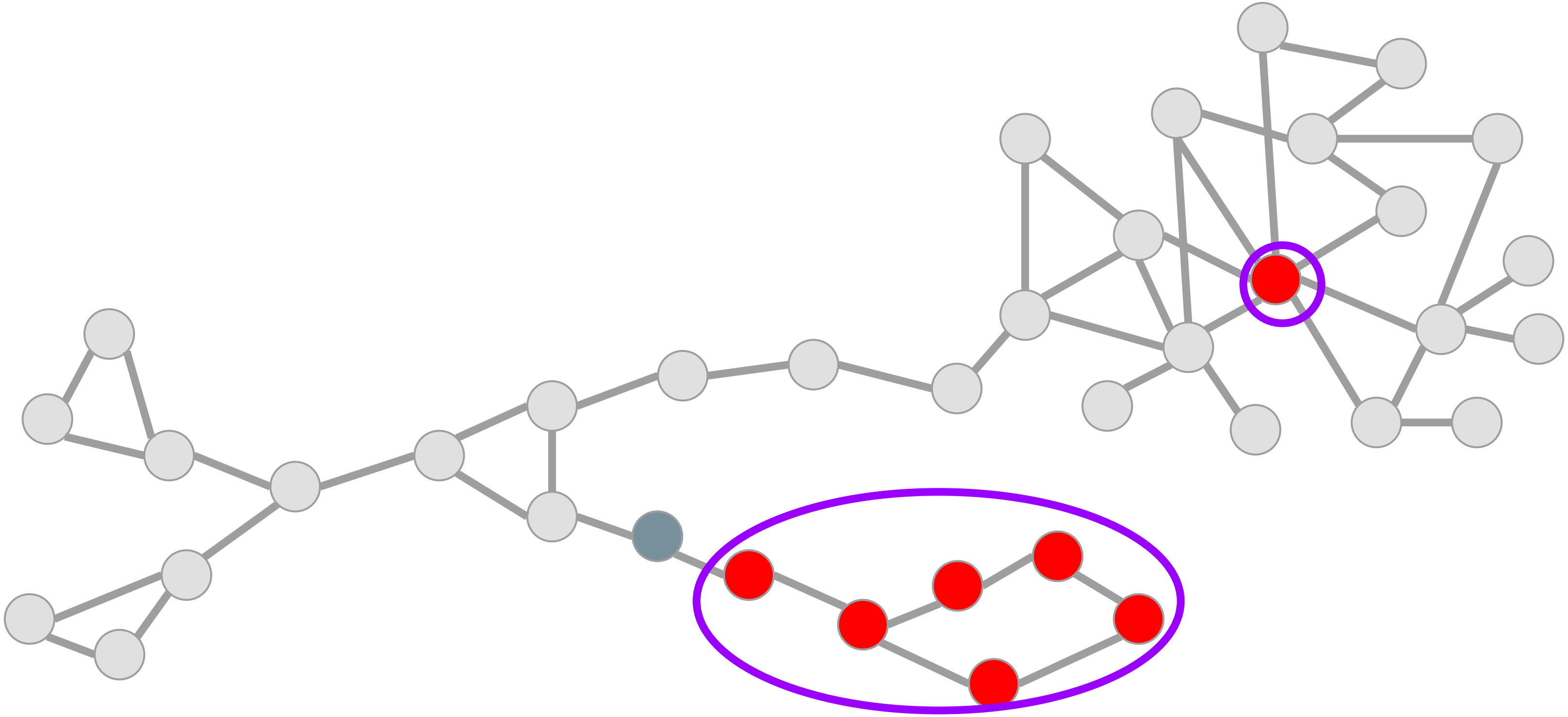
Sequential ball carving



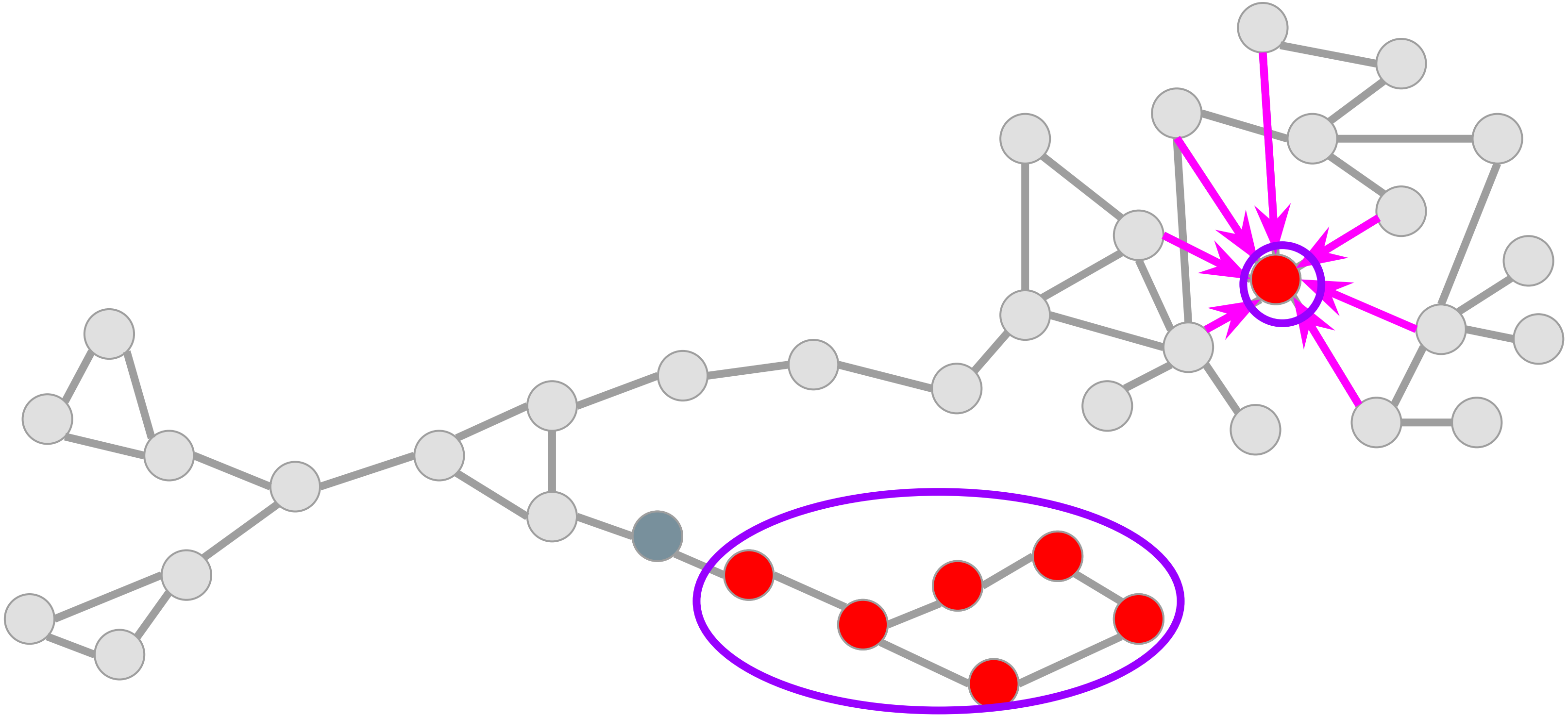
Sequential ball carving



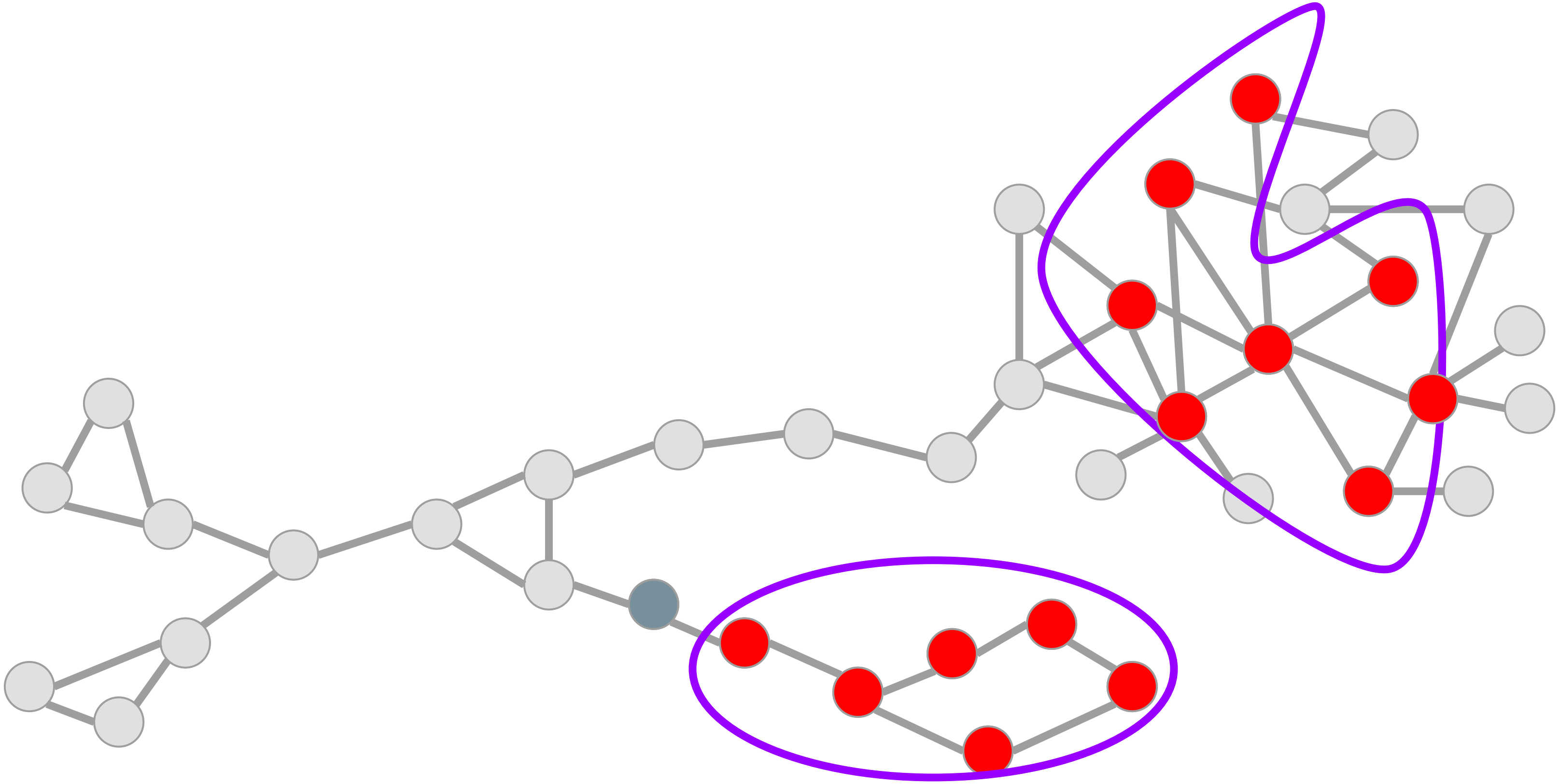
Sequential ball carving



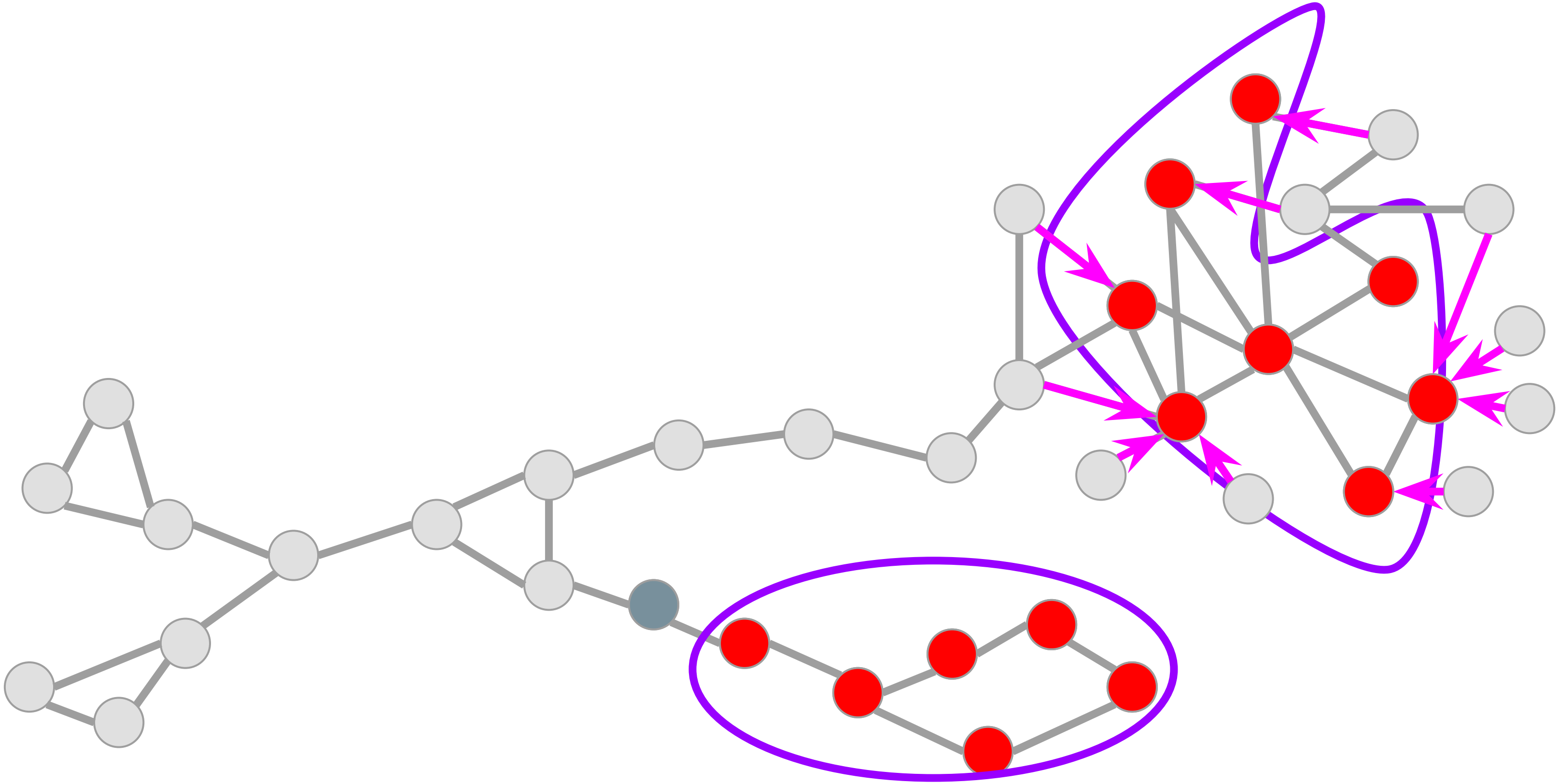
Sequential ball carving



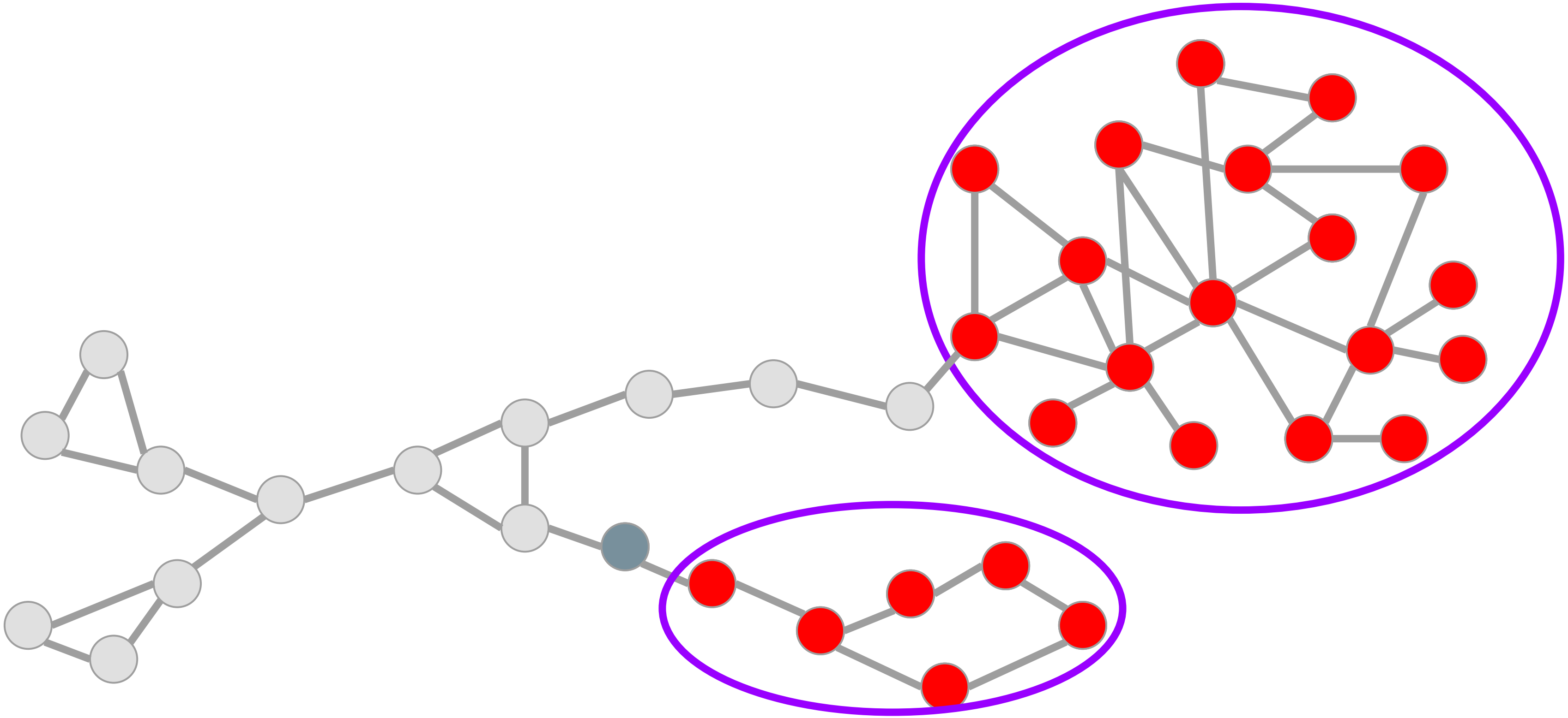
Sequential ball carving



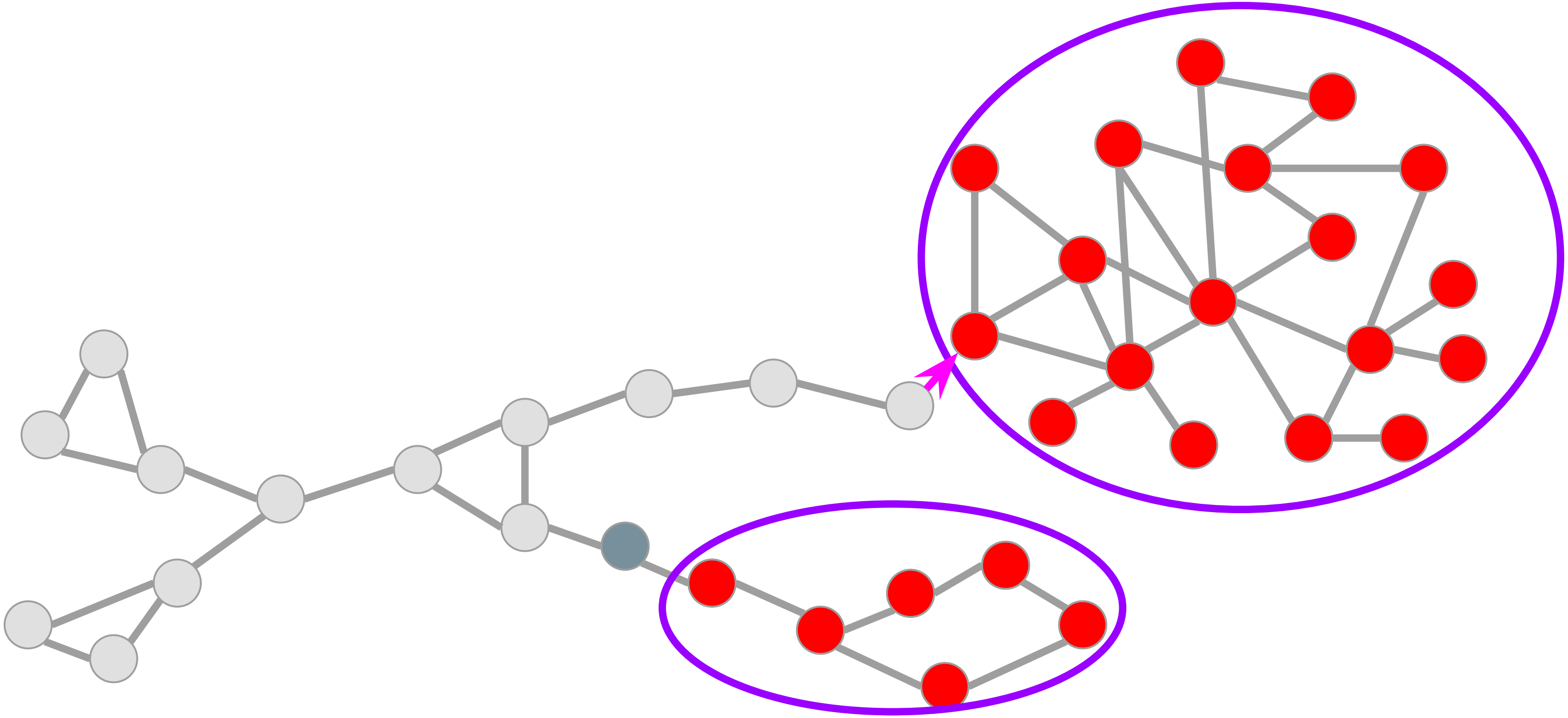
Sequential ball carving



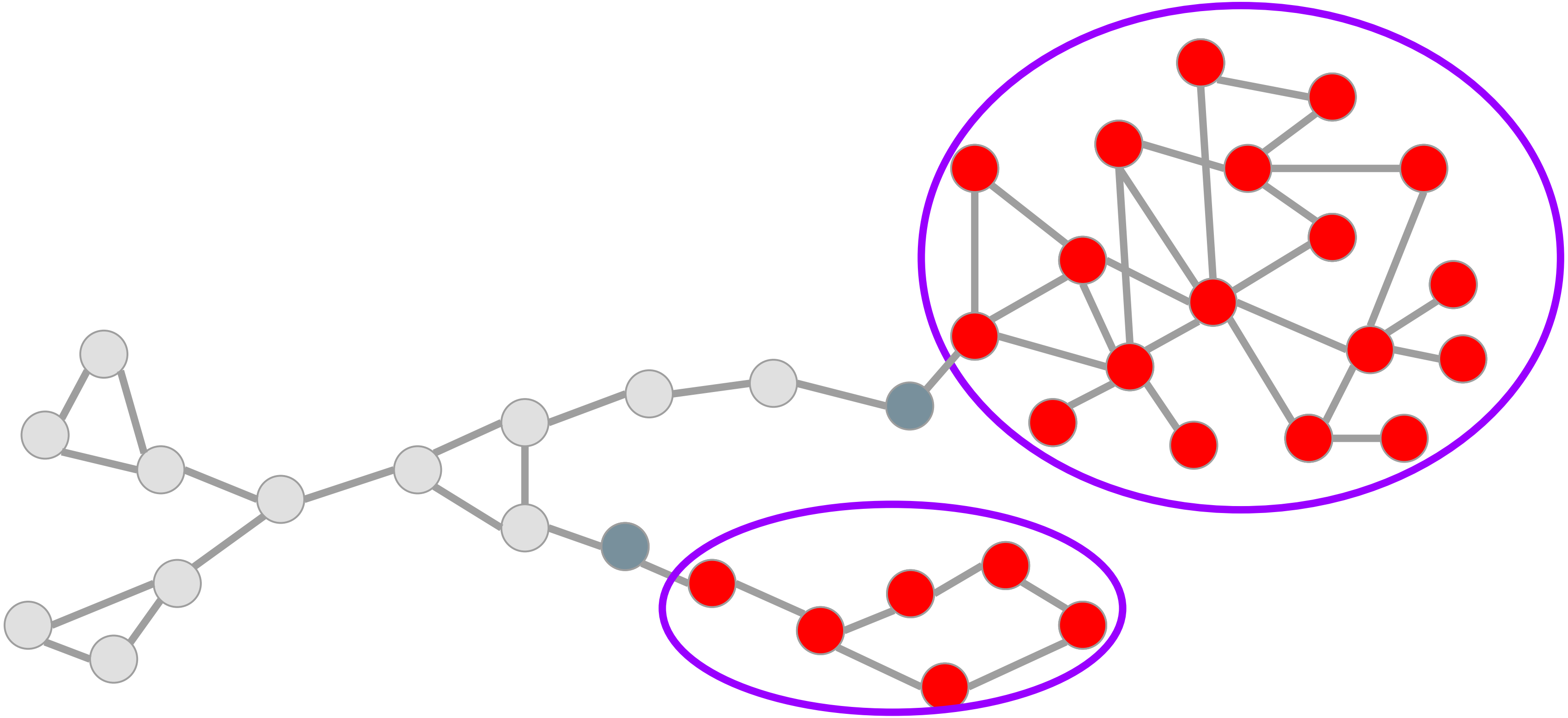
Sequential ball carving



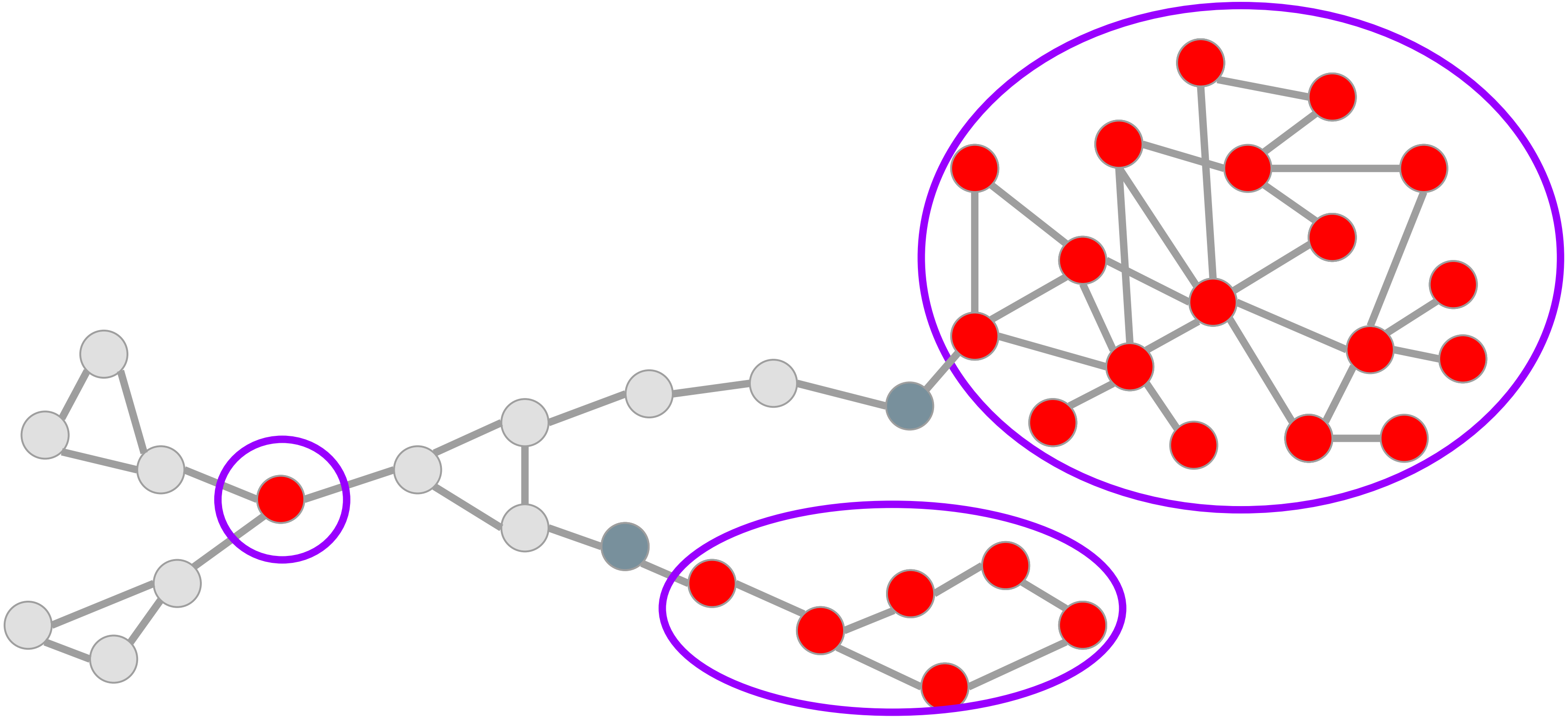
Sequential ball carving



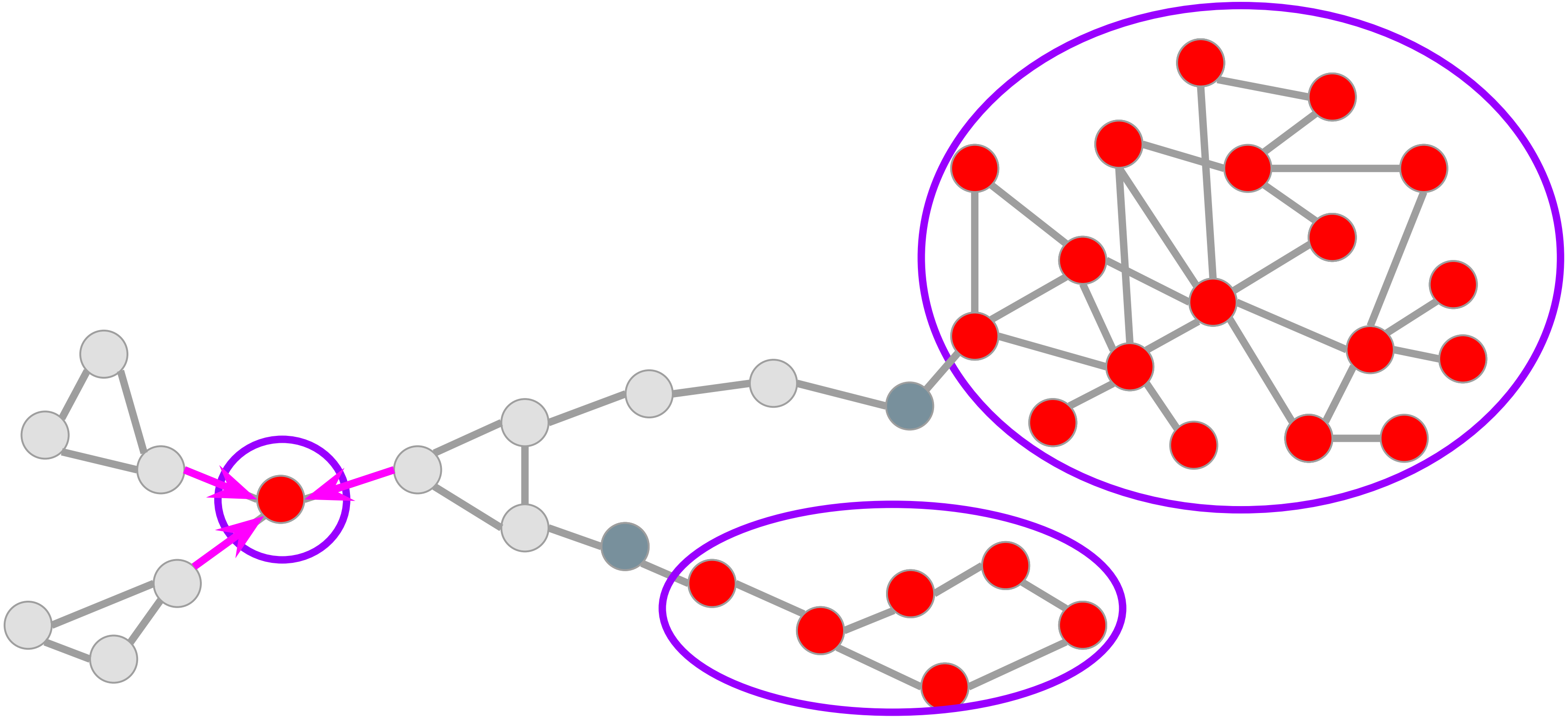
Sequential ball carving



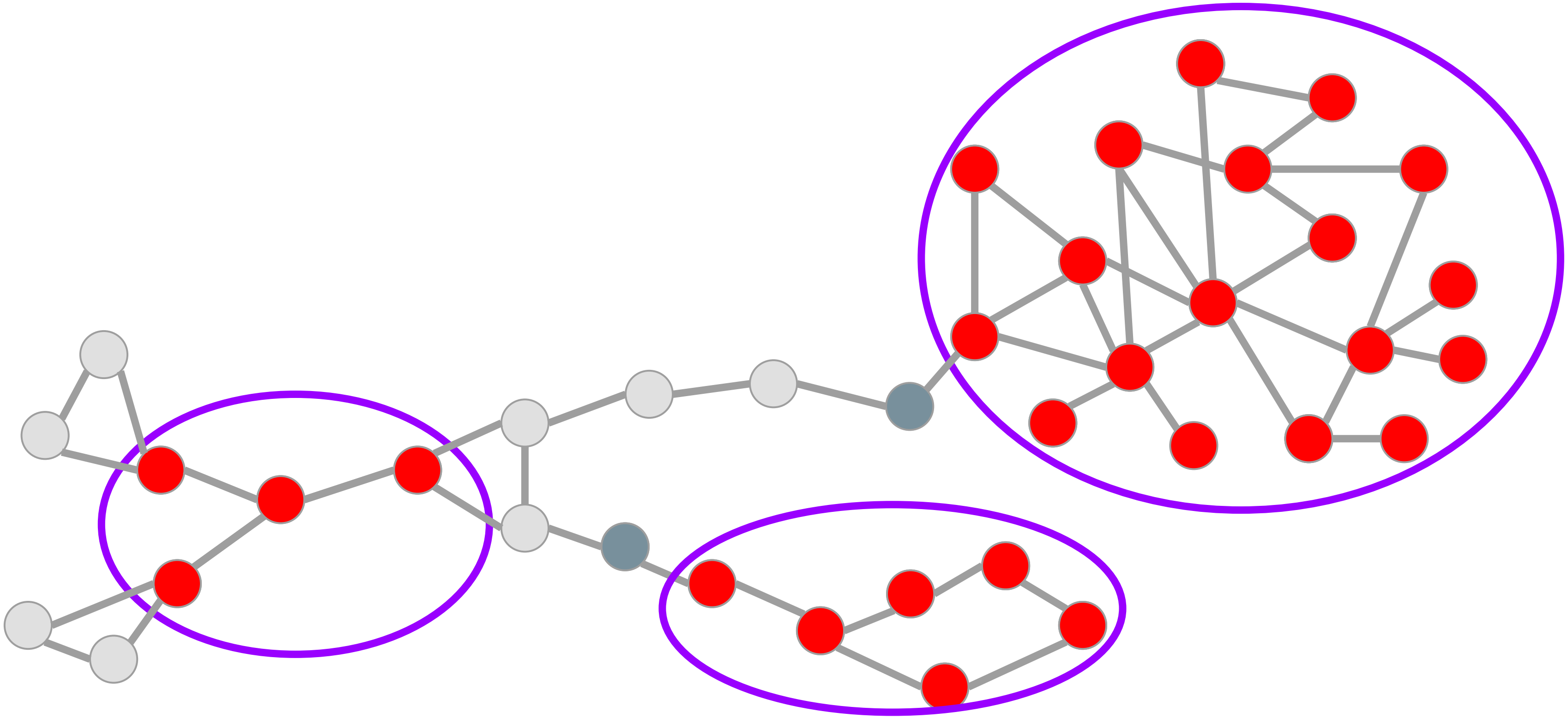
Sequential ball carving



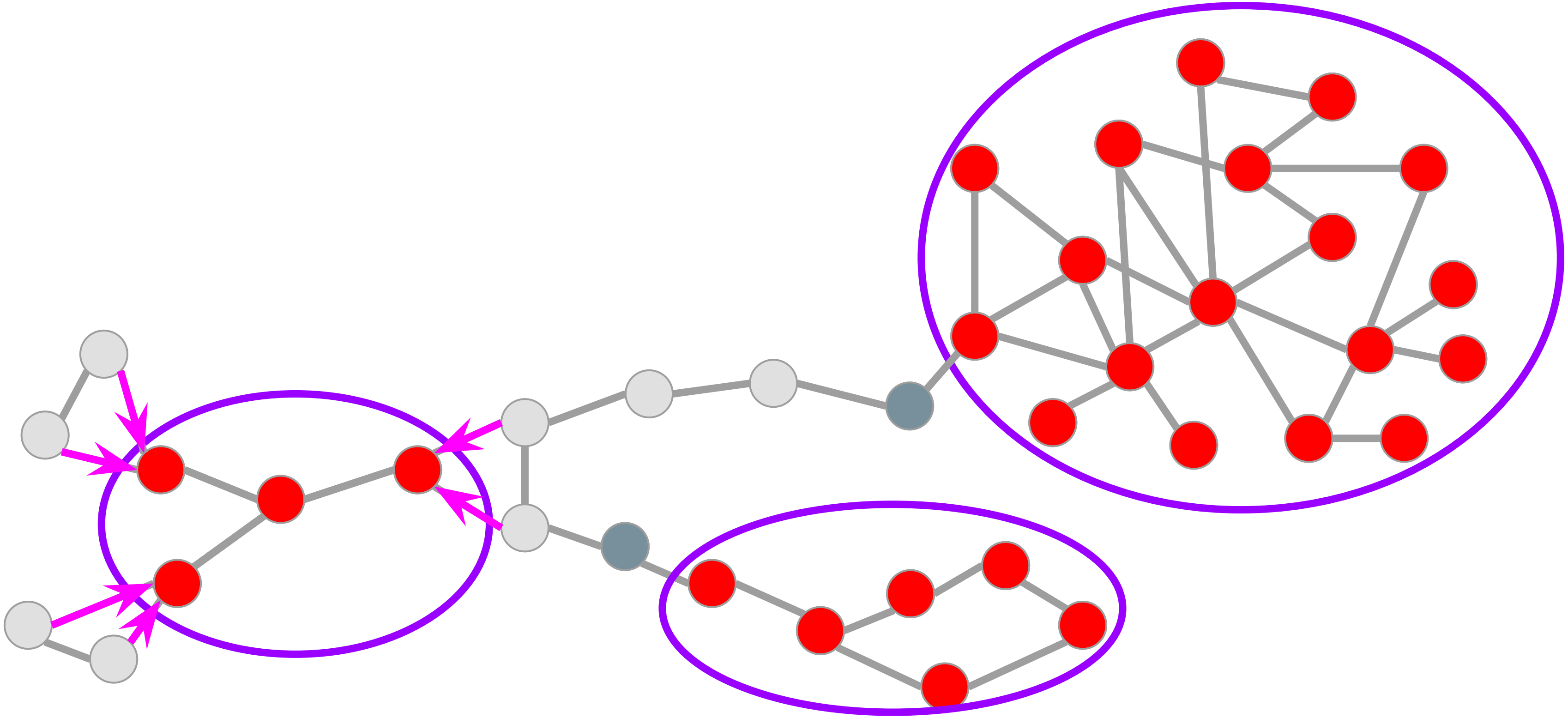
Sequential ball carving



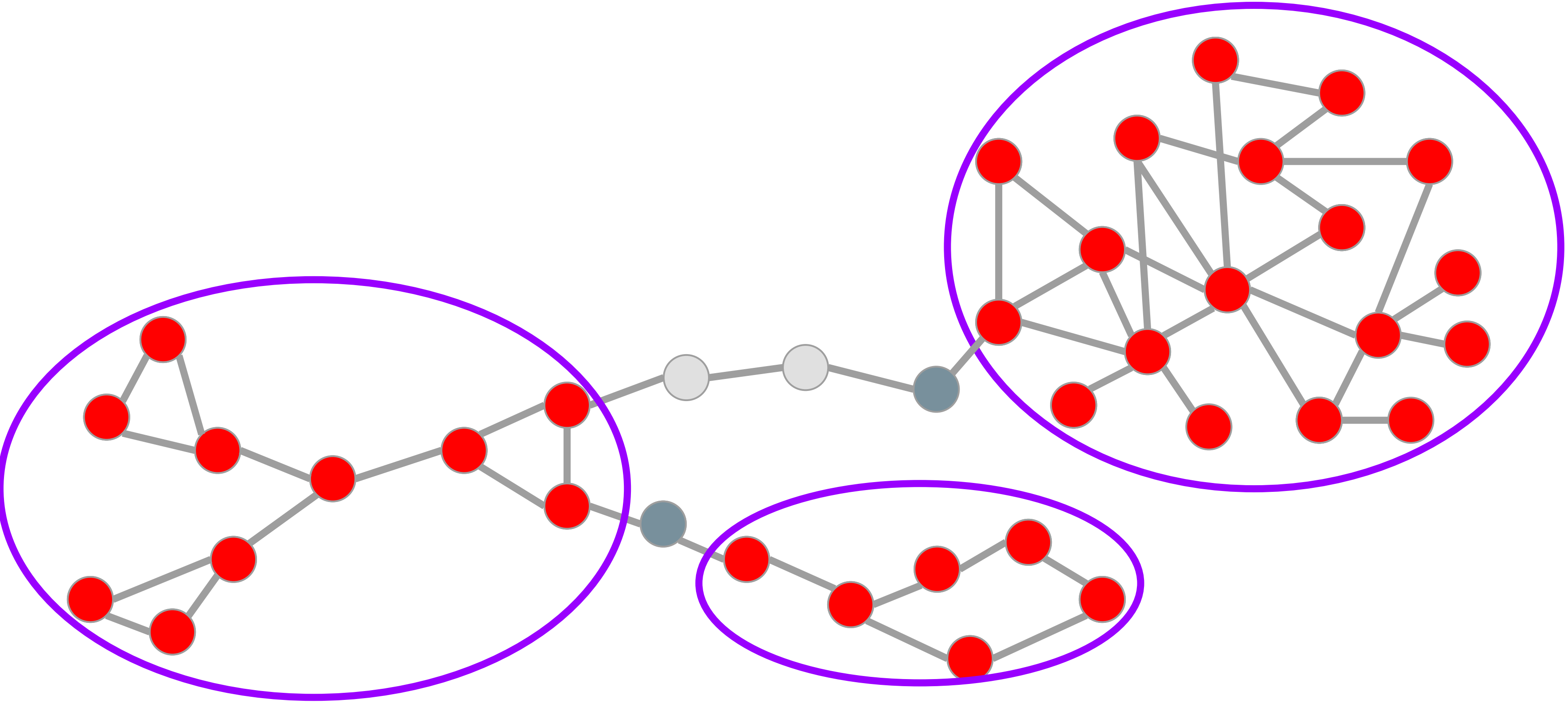
Sequential ball carving



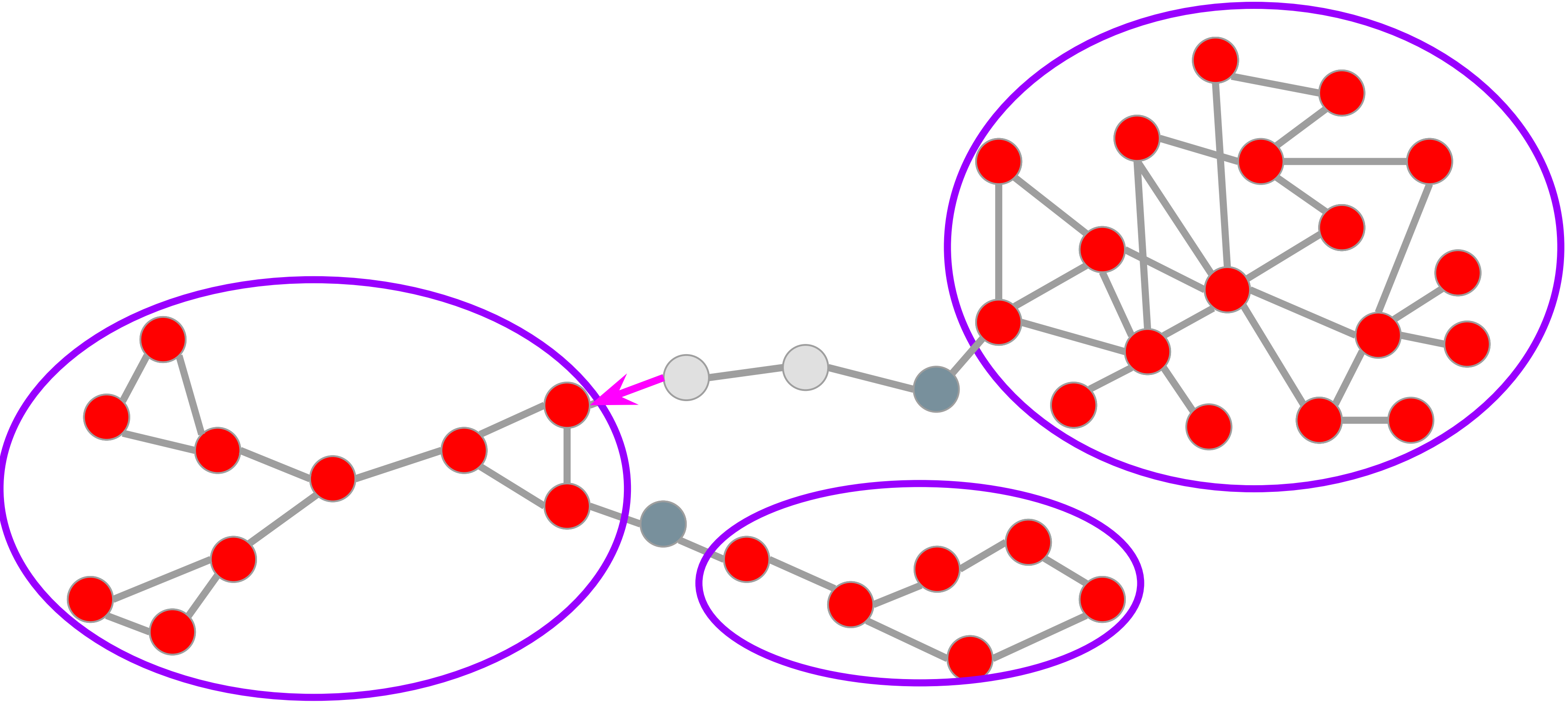
Sequential ball carving



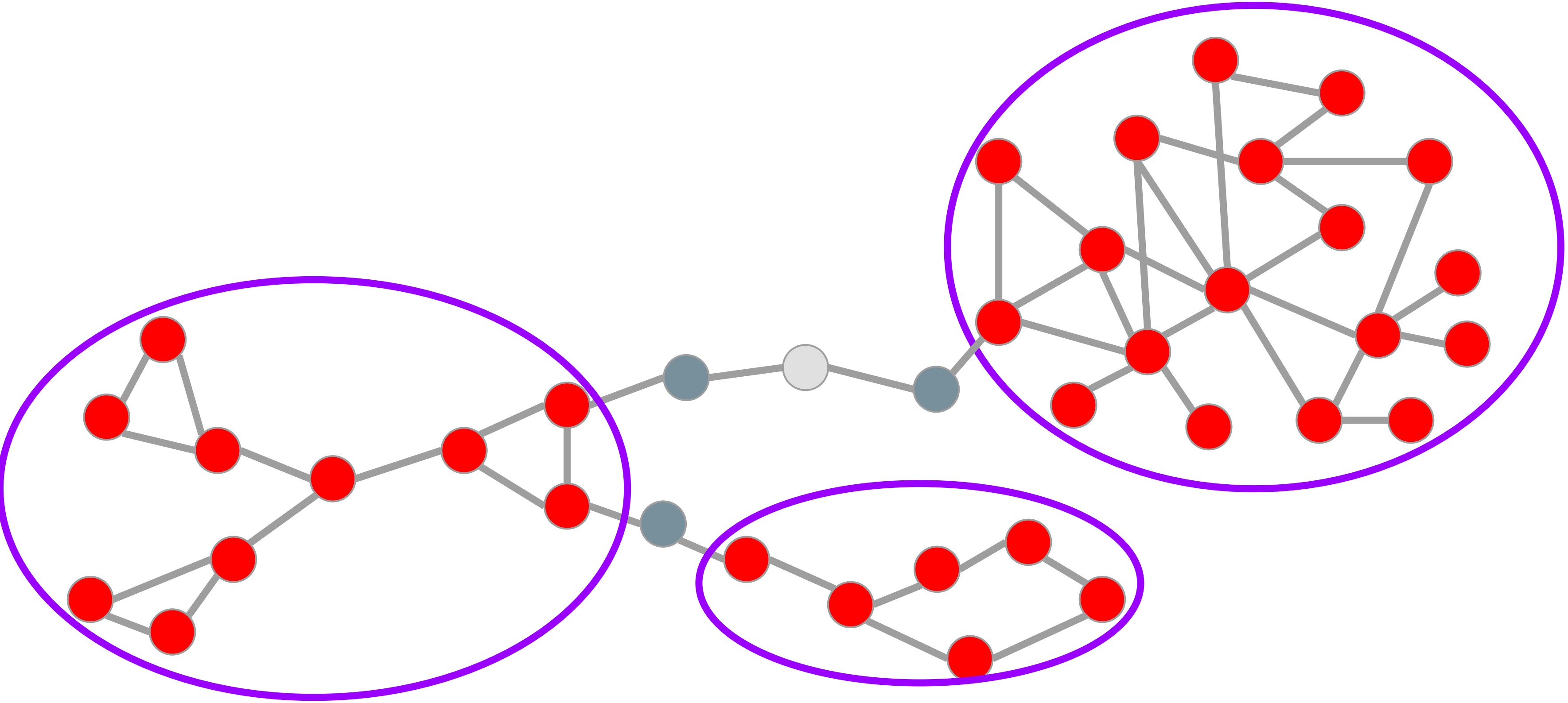
Sequential ball carving



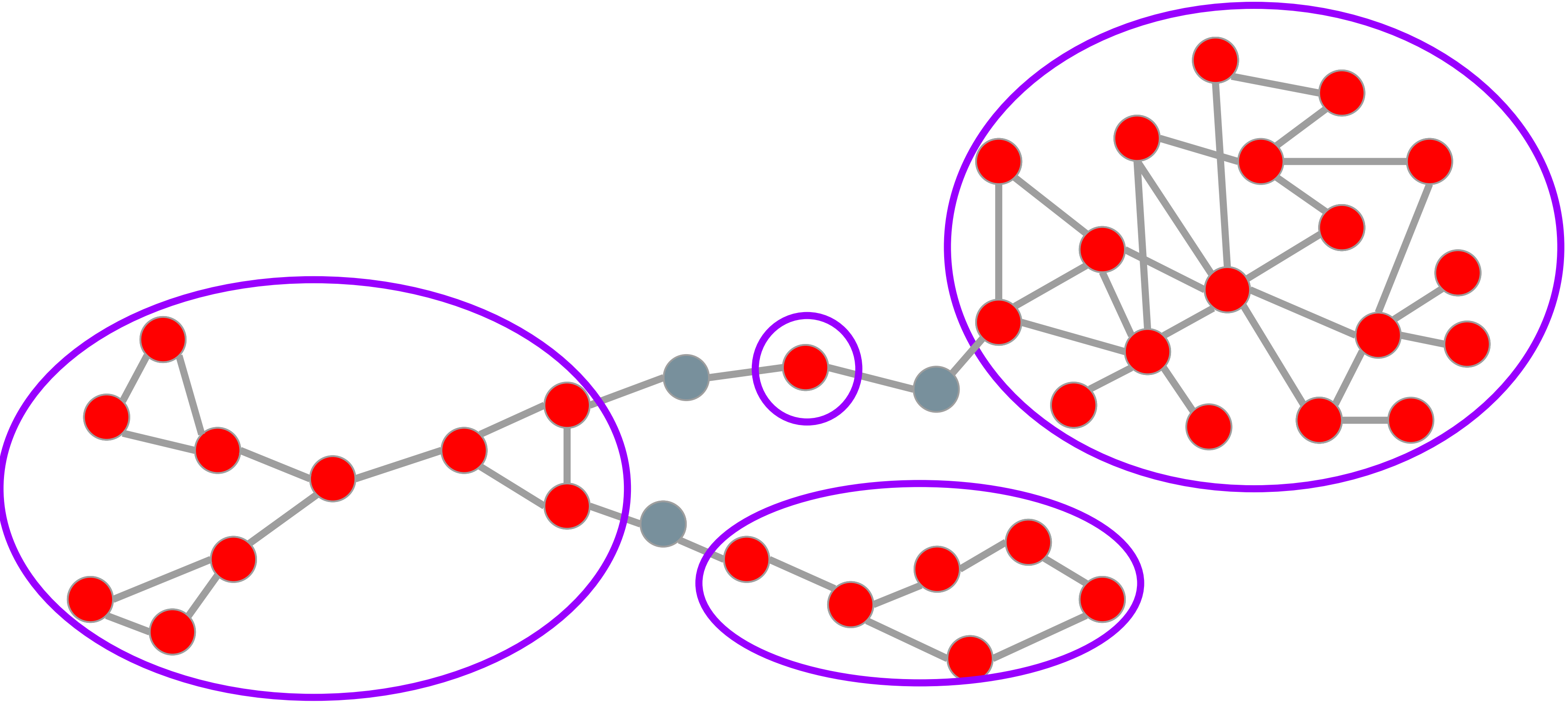
Sequential ball carving



Sequential ball carving



Sequential ball carving



Sequential ball carving

1. clusters at least ½ fraction of vertices

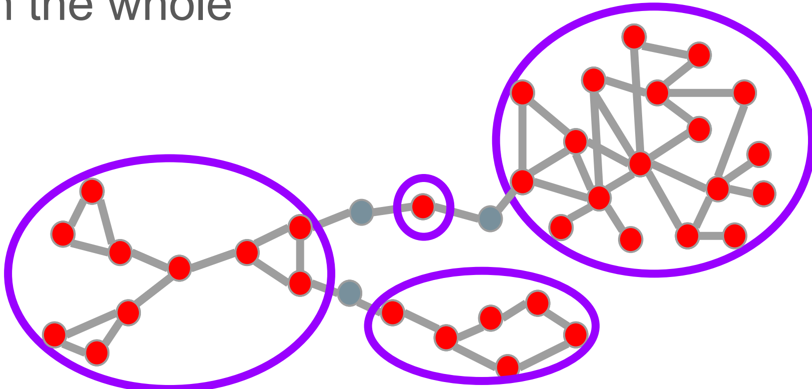
   Each cluster C is responsible for deleting $< |C|$ vertices
   ⇒ $<$ ½ fraction of vertices deleted.

2. each cluster has diameter $O(\log n)$

   After $1+\log n$ steps, a cluster would contain the whole graph, as $2^{1+\log n} > n$.

3. clusters are non-adjacent

   By construction.



Plan

1. More on **LOCAL** and **CONGEST** model
2. A deterministic algorithm for network decomposition.
   a. Sequential algorithm
   b. Distributed algorithm

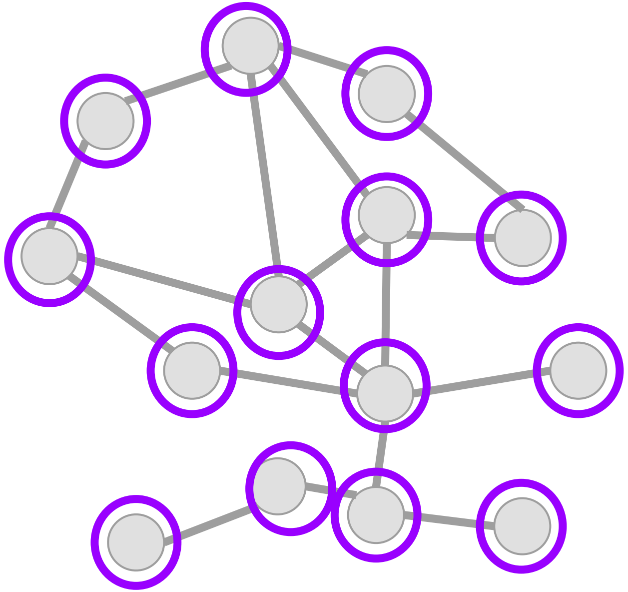
Distributed ball carving

We follow the sequential strategy

and show a deterministic poly(log n)-round algorithm that

1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$ and
3. clusters are non-adjacent

Distributed ball carving



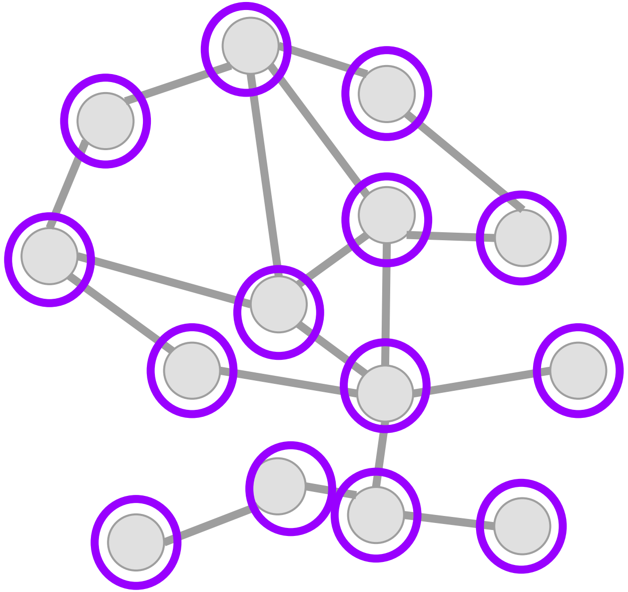
We follow the sequential strategy

and show a deterministic poly(log *n*)-round algorithm that

1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$ and
3. clusters are non-adjacent

At the beginning, each vertex thinks of itself as the root of a cluster

Distributed ball carving



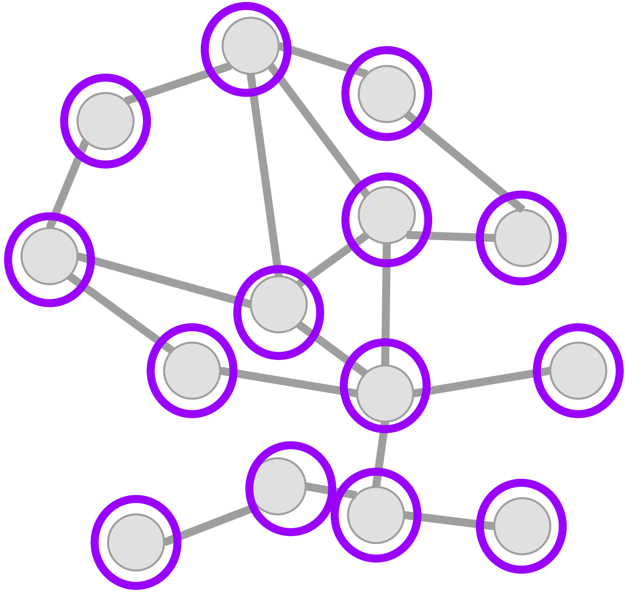
We follow the sequential strategy

and show a deterministic poly(log *n*)-round algorithm that

✓ 1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$ and
3. clusters are non-adjacent

At the beginning, each vertex thinks of itself as the root of a cluster

Distributed ball carving



We follow the sequential strategy

and show a deterministic poly(log n)-round algorithm that

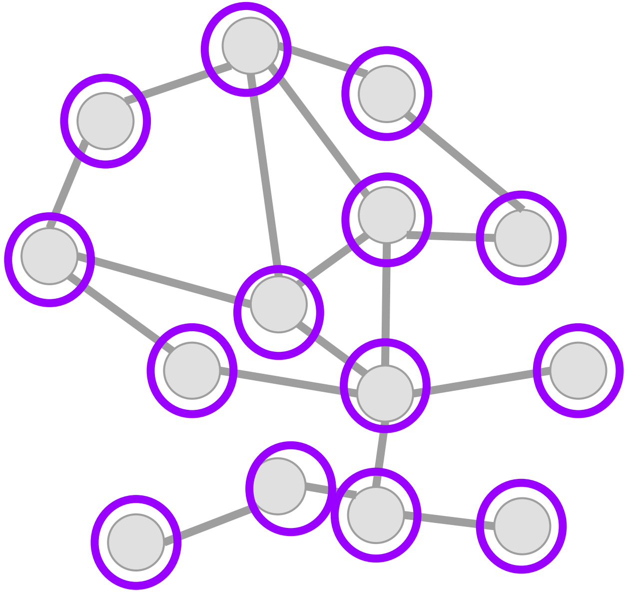
✓ 1. clusters at least ½ fraction of vertices

✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$ and

3. clusters are non-adjacent

At the beginning, each vertex thinks of itself as the root of a cluster

Distributed ball carving



We follow the sequential strategy

and show a deterministic poly(log n)-round algorithm that

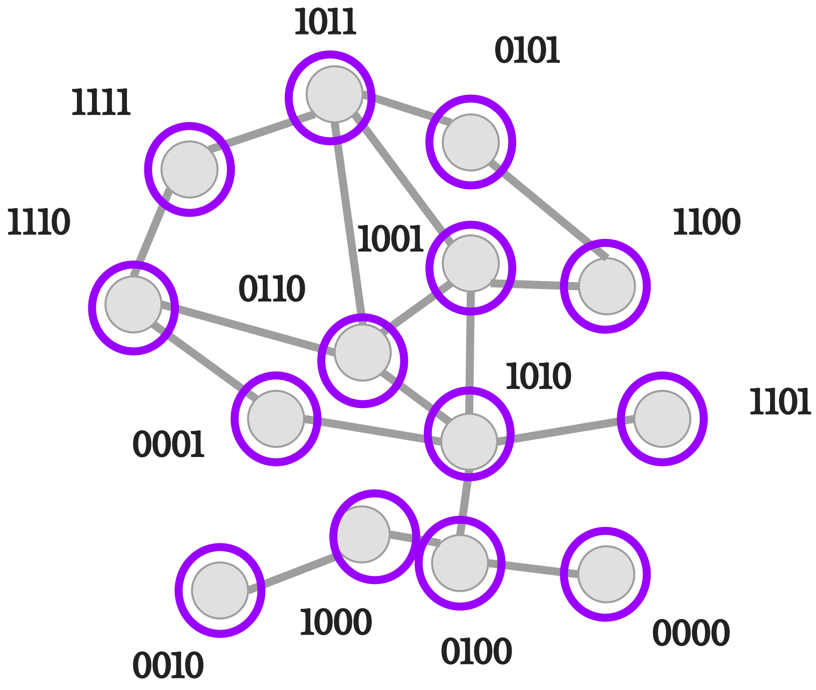
✓ 1. clusters at least ½ fraction of vertices

✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$ and

✗ 3. clusters are non-adjacent

At the beginning, each vertex thinks of itself as the root of a cluster

Distributed ball carving

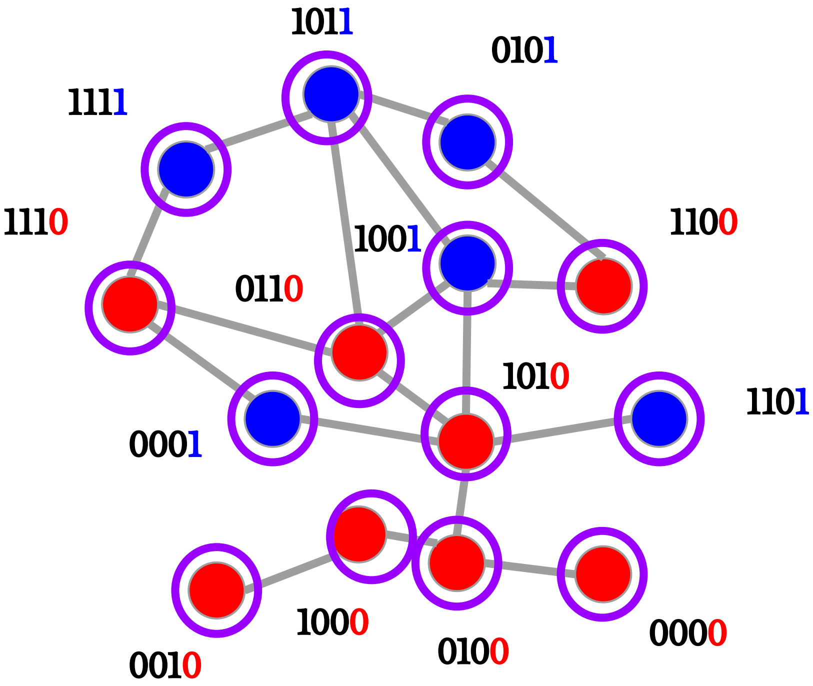


The identifiers have $B = O(\log n)$ bits.

The algorithm has B phases.

In the i-th phase we deal with "bad edges" between clusters whose identifiers differ in the i-th bit.

Distributed ball carving

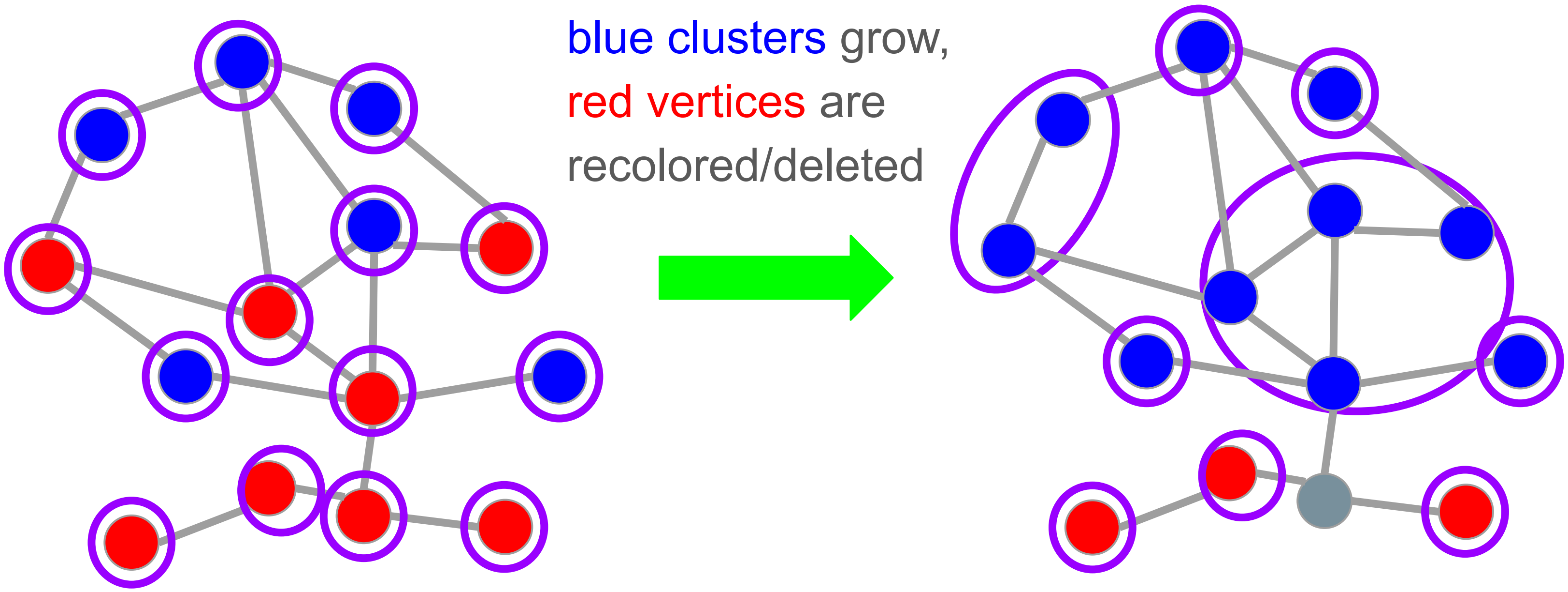


The identifiers have $B = O(\log n)$ bits.

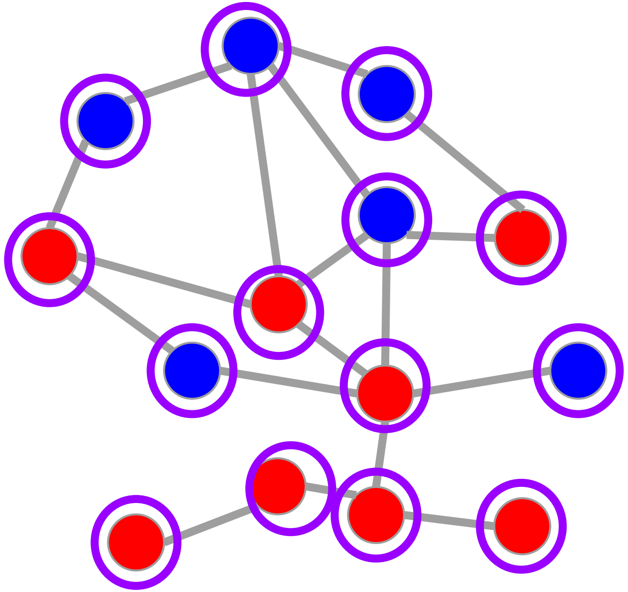
The algorithm has B phases.

In the i-th phase we deal with “bad edges” between clusters whose identifiers differ in the i-th bit.

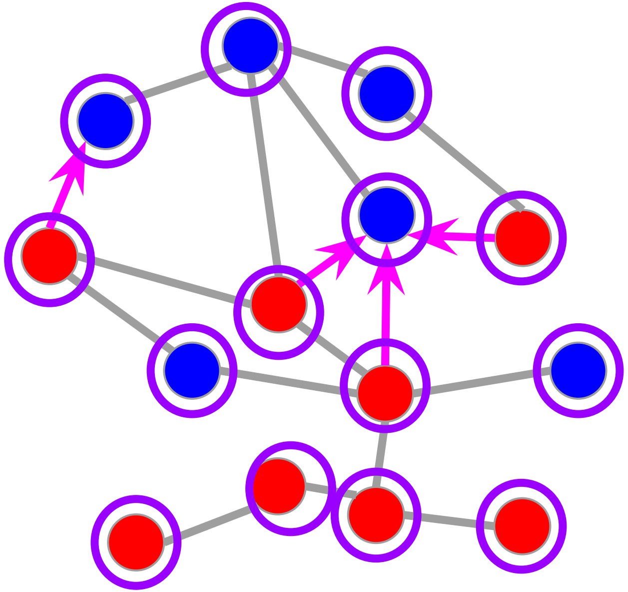
Distributed ball carving



Distributed ball carving



Distributed ball carving

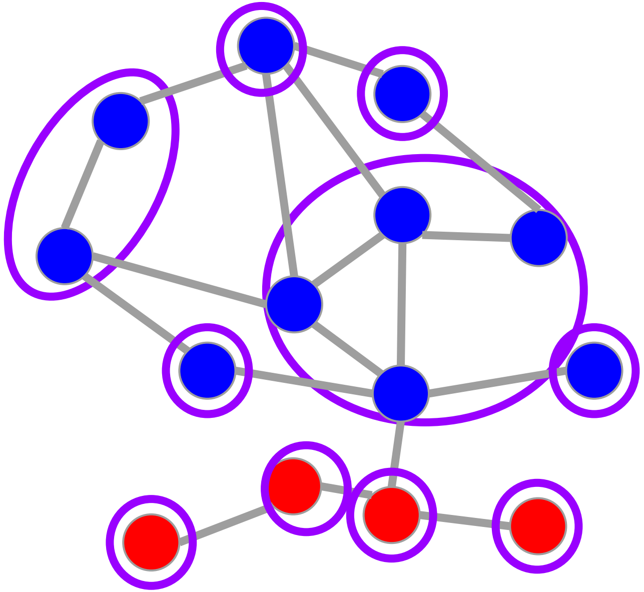


Red vertices propose to join an arbitrary neighbouring blue cluster.

A blue cluster C accepts all proposals if at least $|C|/(2B)$ vertices are proposing.

Otherwise, it denies all of them, therefore deleting proposing red nodes permanently.

Distributed ball carving

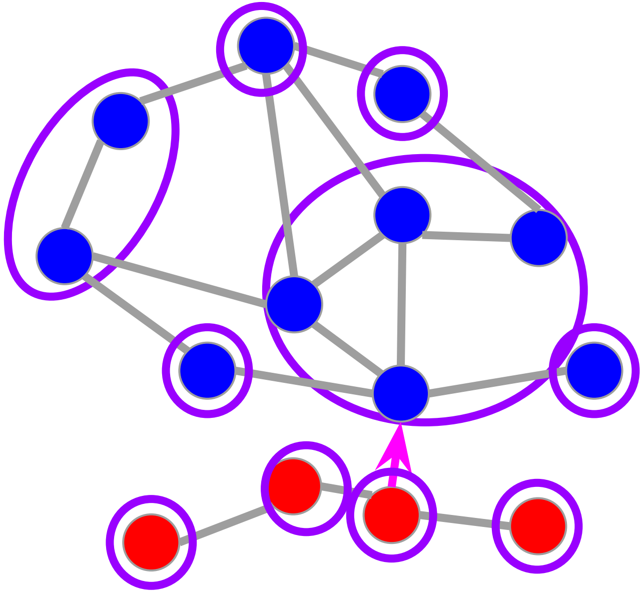


Red vertices propose to join an arbitrary neighbouring blue cluster.

A blue cluster C accepts all proposals if at least $|C|/(2B)$ vertices are proposing.

Otherwise, it denies all of them, therefore deleting proposing red nodes permanently.

Distributed ball carving

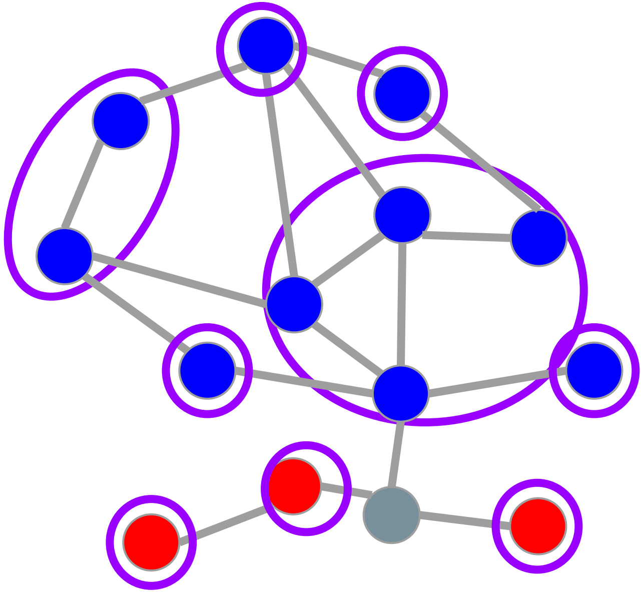


Red vertices propose to join an arbitrary neighbouring blue cluster.

A blue cluster *C* accepts all proposals if at least $|C|/(2B)$ vertices are proposing.

Otherwise, it denies all of them, therefore deleting proposing red nodes permanently.

Distributed ball carving

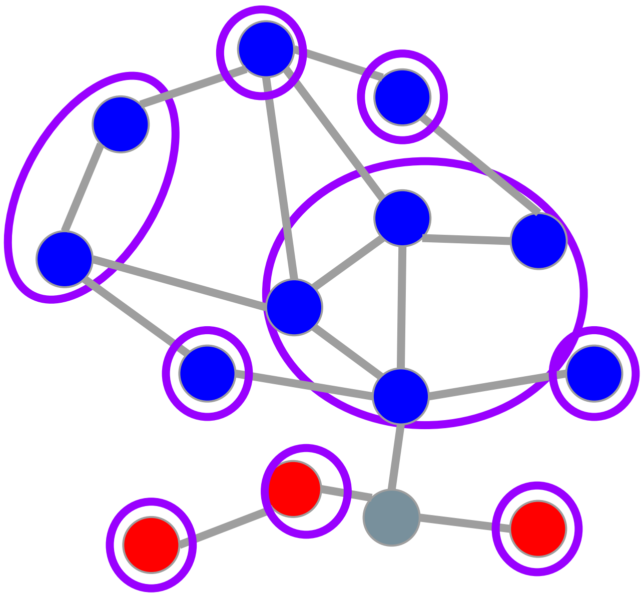


Red vertices propose to join an arbitrary neighbouring blue cluster.

A blue cluster C accepts all proposals if at least $|C|/(2B)$ vertices are proposing.

Otherwise, it denies all of them, therefore deleting proposing red nodes permanently.

Distributed ball carving



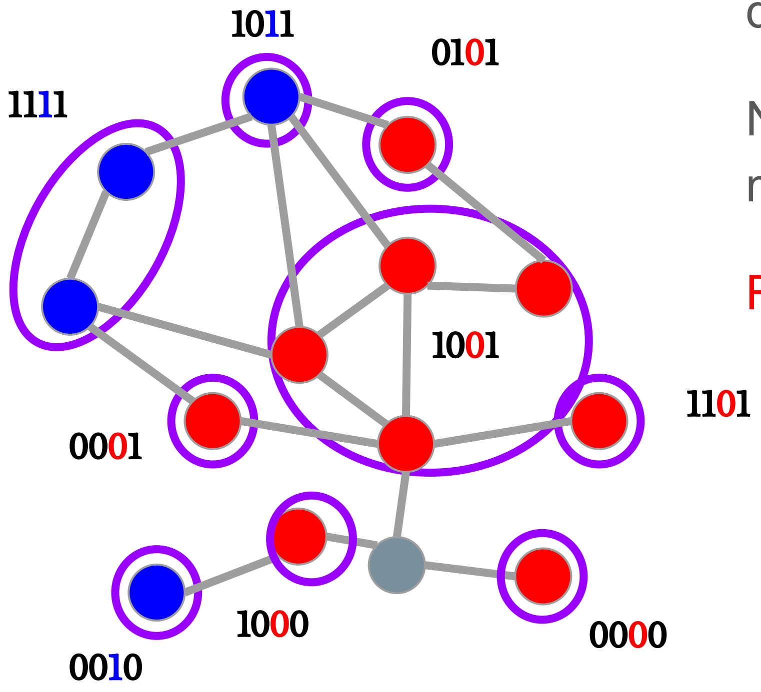
Red vertices propose to join an arbitrary neighbouring blue cluster.

A blue cluster C accepts all proposals if at least $|C|/(2B)$ vertices are proposing.

Otherwise, it denies all of them, therefore deleting proposing red nodes permanently.

We let this process run for $4B \ln n$ steps.

Distributed ball carving

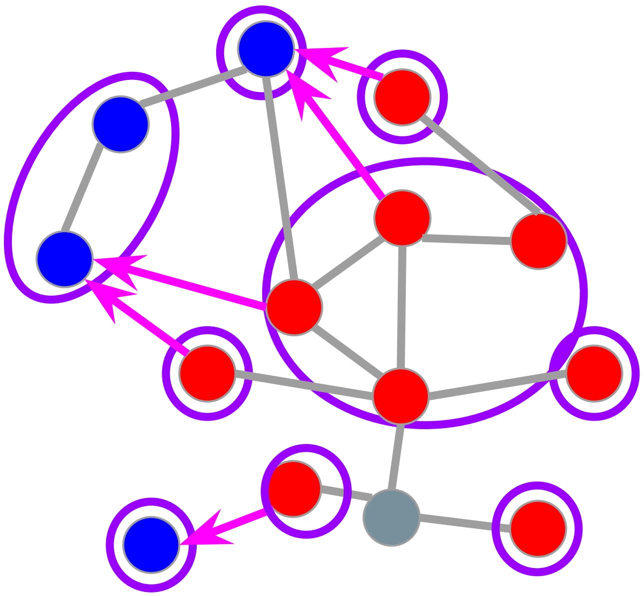


In the second phase (and other phases) we do the same, based on the *i*-th rightmost bit.

Note that the coloring here has different meaning than in the first phase.

Red vertices propose, not whole clusters.

Distributed ball carving

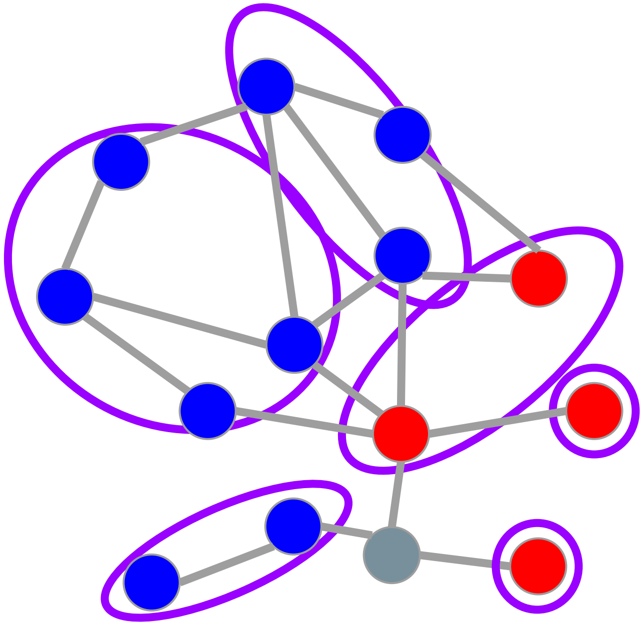


In the second phase (and other phases) we do the same, based on the *i*-th rightmost bit.

Note that the coloring here has different meaning than in the first phase.

Red vertices propose, not whole clusters.

Distributed ball carving

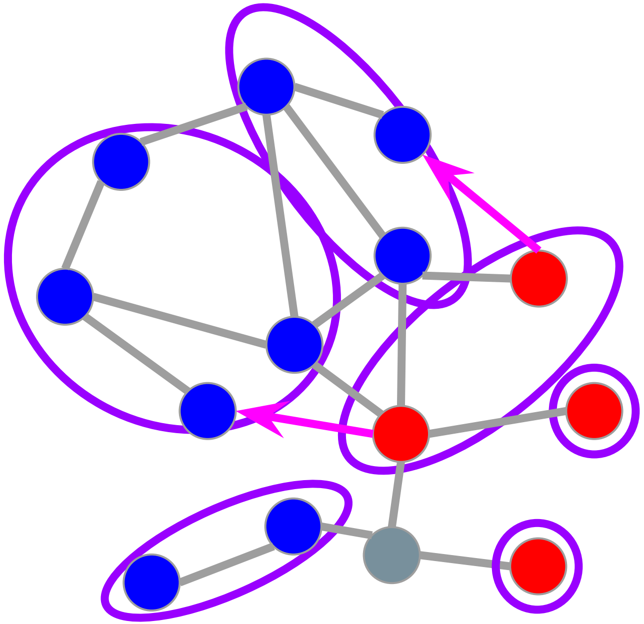


In the second phase (and other phases) we do the same, based on the *i*-th rightmost bit.

Note that the coloring here has different meaning than in the first phase.

Red vertices propose, not whole clusters.

Distributed ball carving

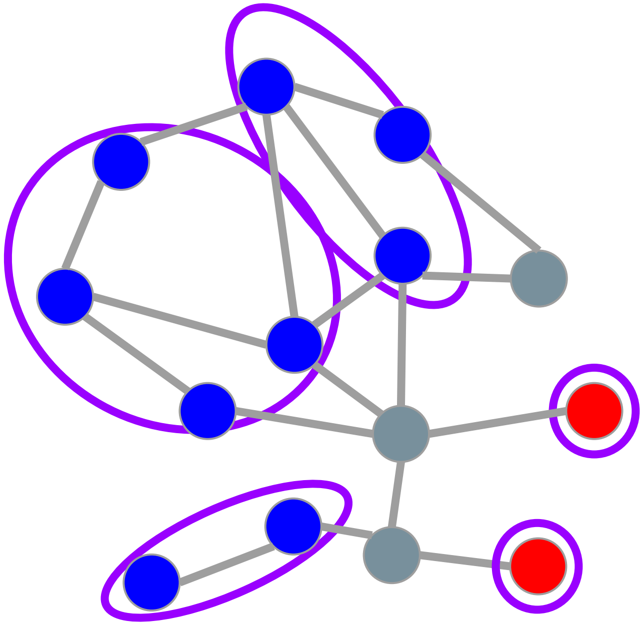


In the second phase (and other phases) we do the same, based on the *i*-th rightmost bit.

Note that the coloring here has different meaning than in the first phase.

Red vertices propose, not whole clusters.

Distributed ball carving

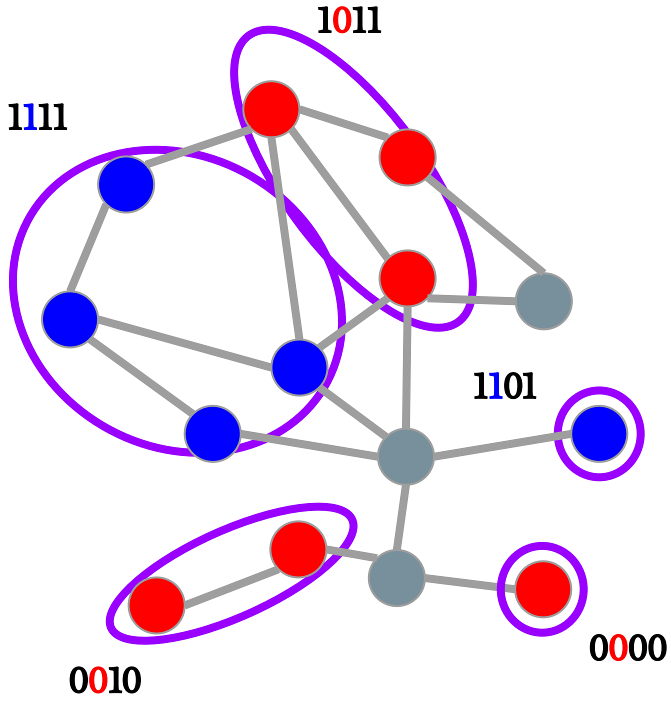


In the second phase (and other phases) we do the same, based on the *i*-th rightmost bit.

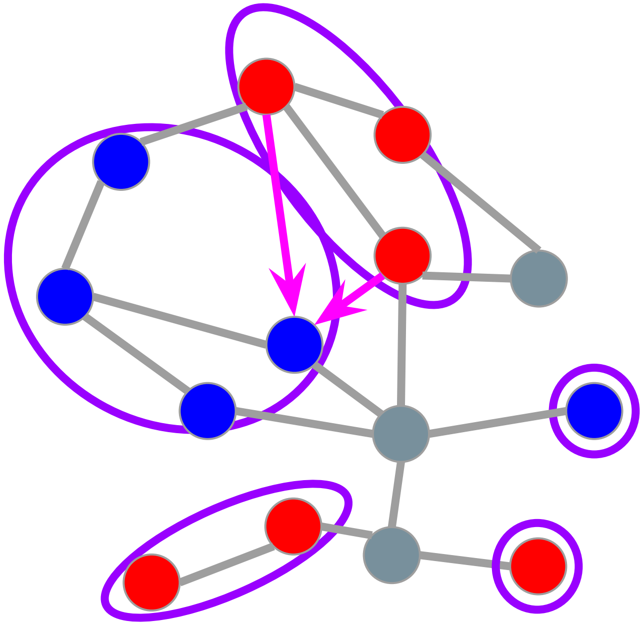
Note that the coloring here has different meaning than in the first phase.

Red vertices propose, not whole clusters.

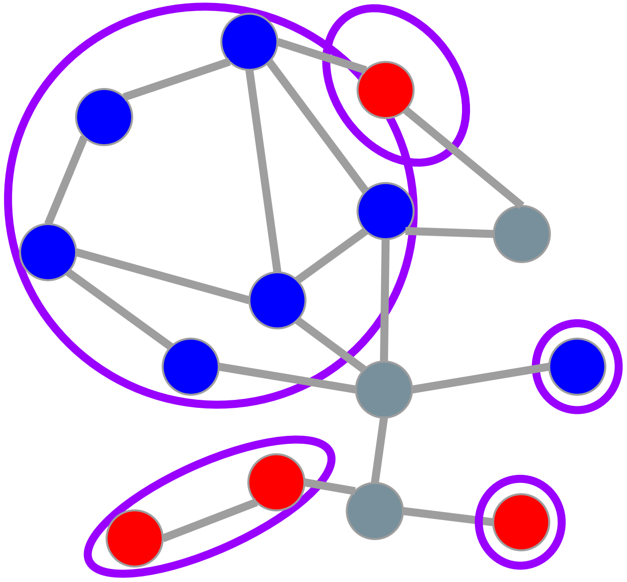
Distributed ball carving



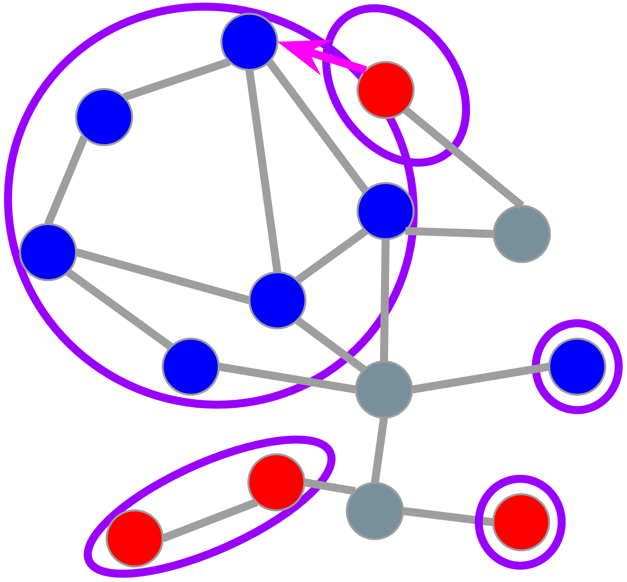
Distributed ball carving



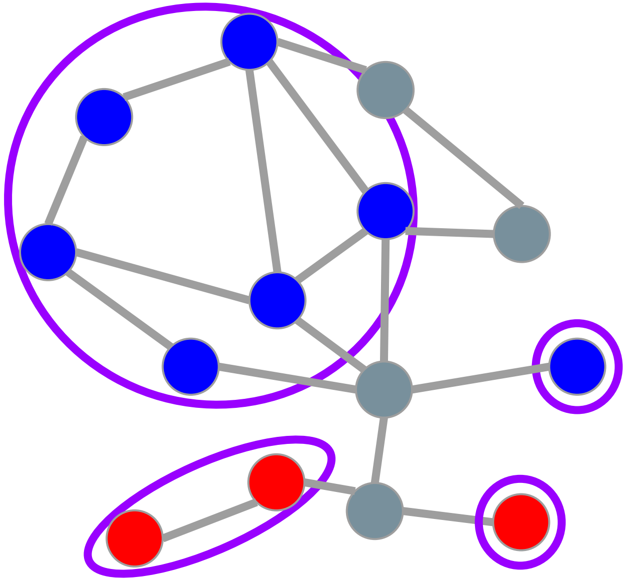
Distributed ball carving



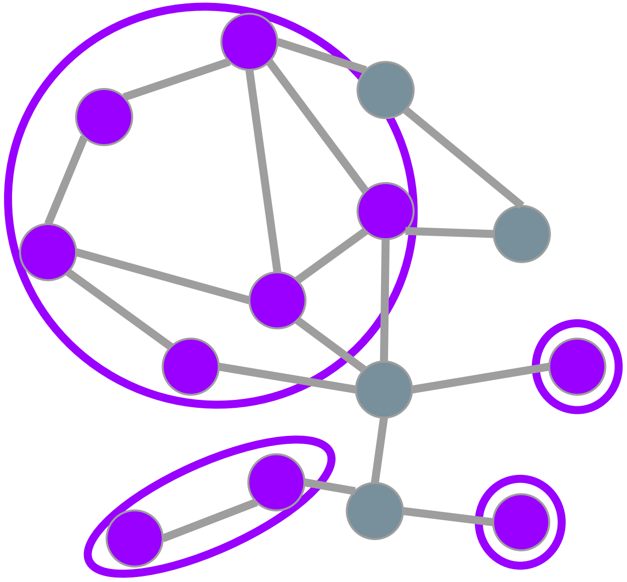
Distributed ball carving



Distributed ball carving



Distributed ball carving

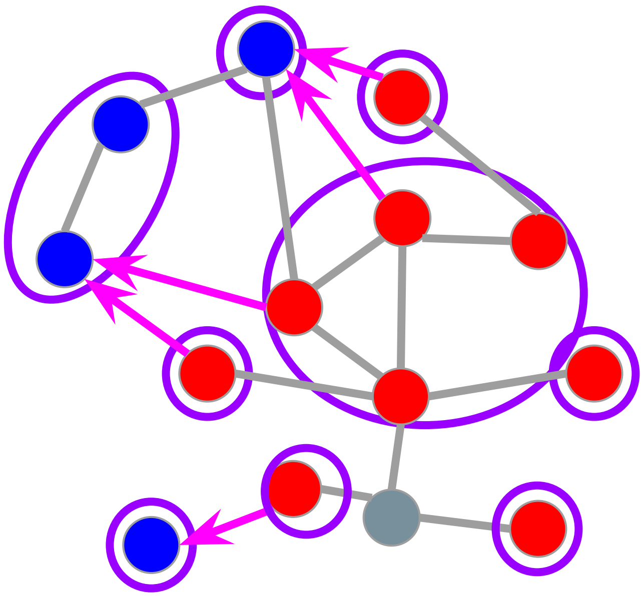


1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

Property 2:

The weak-diameter grows additively by ≤ 2 in each step.

Distributed ball carving

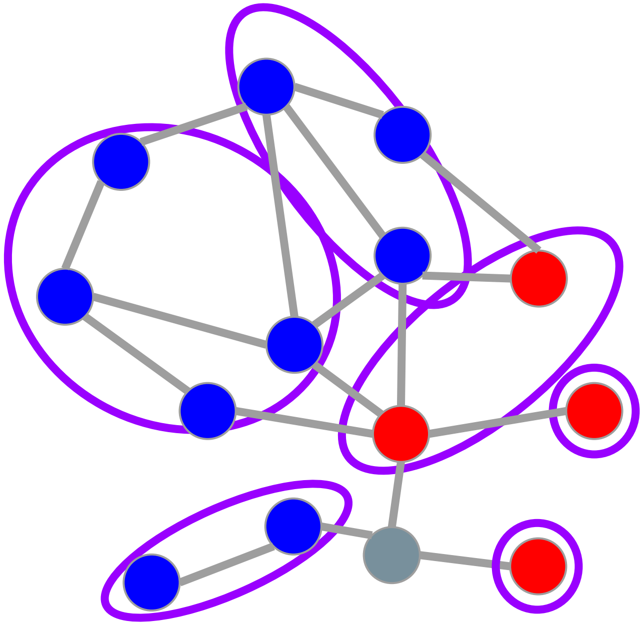


1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

Property 2:

The weak-diameter grows additively by ≤ 2 in each step.

Distributed ball carving

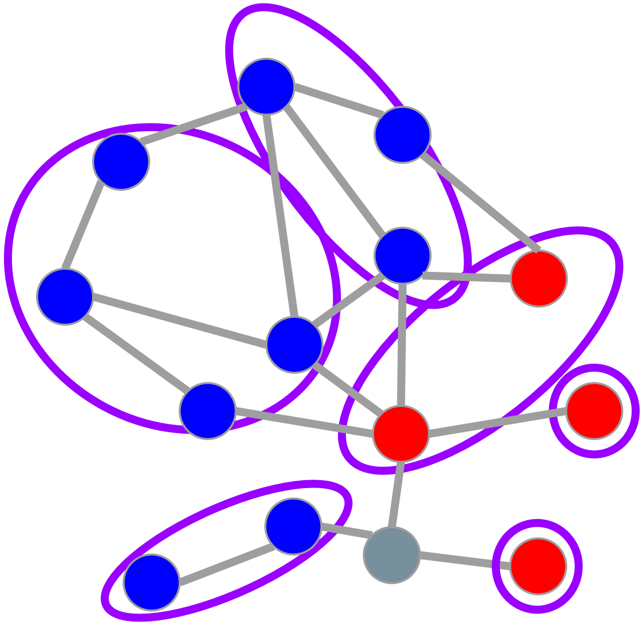


1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

Property 2:

The weak-diameter grows additively by ≤ 2 in each step.

Distributed ball carving



1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

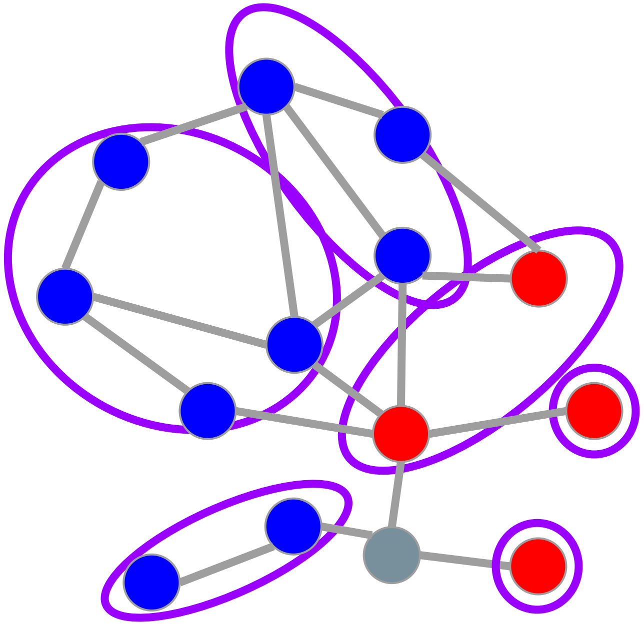
Property 2:

The weak-diameter grows additively by ≤ 2 in each step.

We have B phases and each phase has $4B \ln n$ steps.

Distributed ball carving

1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent



Property 2:

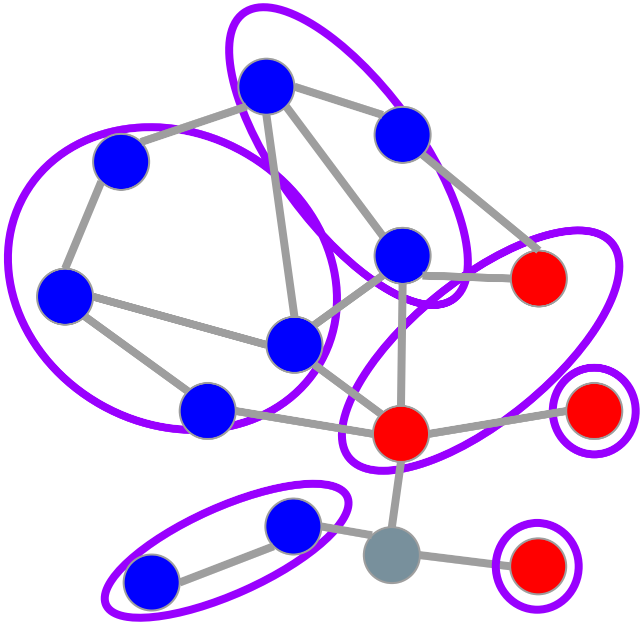
The weak-diameter grows additively by ≤ 2 in each step.

We have B phases and each phase has $4B \ln n$ steps.

Hence, the weak diameter is $O(B^2 \log n) = O(\log^3 n)$.

Distributed ball carving ✓

1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent



Property 2:

The weak-diameter grows additively by ≤ 2 in each step.

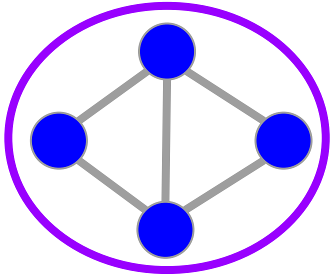
We have B phases and each phase has $4B \ln n$ steps.

Hence, the weak diameter is $O(B^2 \log n) = O(\log^3 n)$.

Distributed ball carving

Observation: If a blue cluster does not grow in some step, it does not have red neighbours in any future steps.

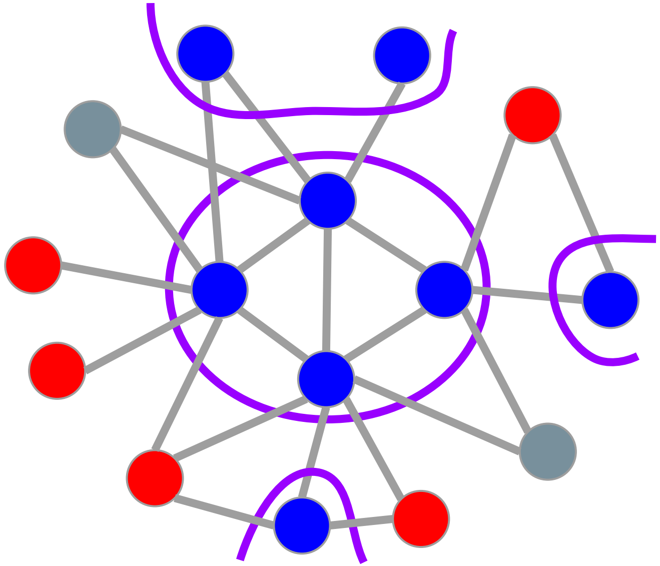
Hence, it does not grow any more during the phase.



Distributed ball carving

Observation: If a blue cluster does not grow in some step, it does not have red neighbours in any future steps.

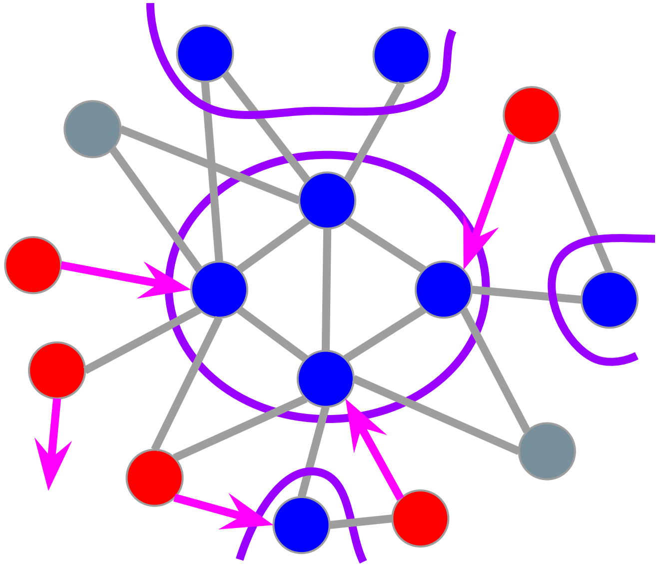
Hence, it does not grow any more during the phase.



Distributed ball carving

Observation: If a blue cluster does not grow in some step, it does not have red neighbours in any future steps.

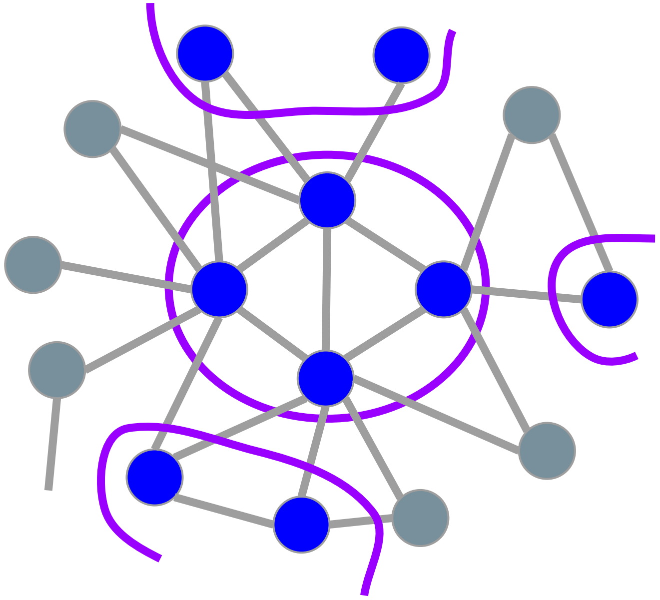
Hence, it does not grow any more during the phase.



Distributed ball carving

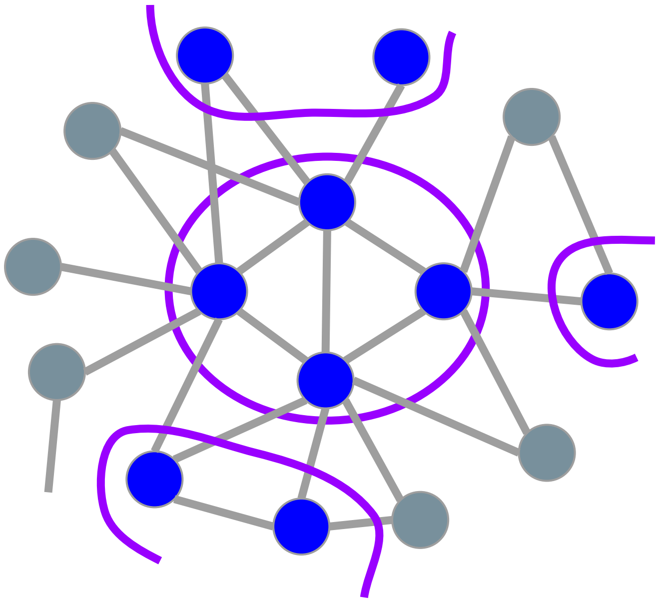
Observation: If a blue cluster does not grow in some step, it does not have red neighbours in any future steps.

Hence, it does not grow any more during the phase.



Distributed ball carving

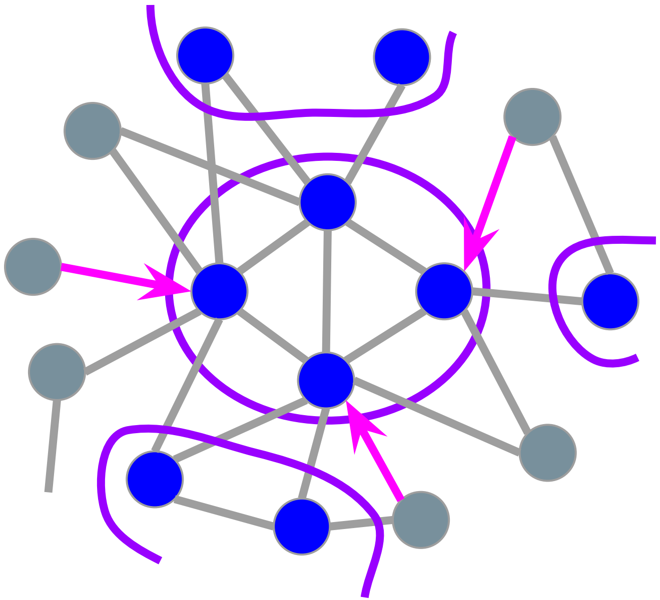
1. clusters at least ½ fraction of vertices
2. ✓ such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent



Property 1:

Distributed ball carving ✓

1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

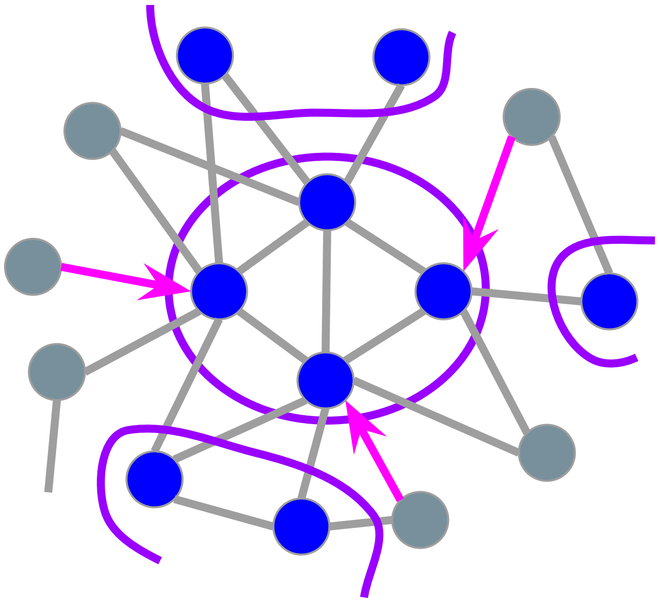


Property 1:

Each blue cluster C is at the end of a phase responsible for $|C|/(2B)$ deleted red vertices.

Distributed ball carving ✓

1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent



Property 1:

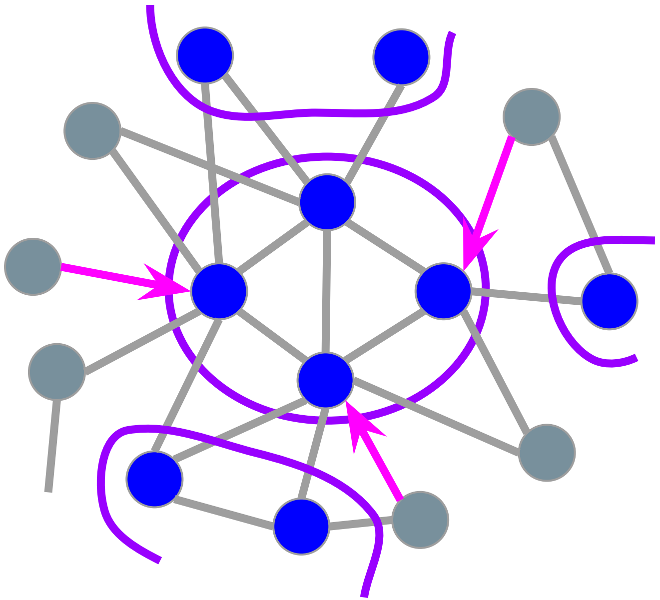
Each blue cluster C is at the end of a phase responsible for $|C|/(2B)$ deleted red vertices.

$\Rightarrow \leq 1/(2B)$ fraction of vertices deleted per phase

Distributed ball carving

✓

1. clusters at least ½ fraction of vertices
2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent



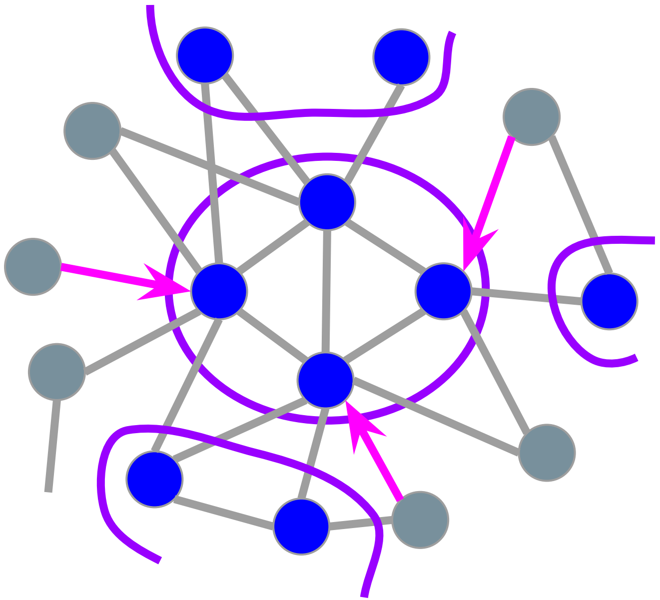
Property 1:

Each blue cluster C is at the end of a phase responsible for $|C|/(2B)$ deleted red vertices.

⇒ $\leq 1/(2B)$ fraction of vertices deleted per phase

⇒ $\leq B \cdot 1/(2B) = ½$ fraction of vertices deleted in total

Distributed ball carving



✓ 1. clusters at least ½ fraction of vertices
✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

Property 1:

Each blue cluster C is at the end of a phase responsible for $|C|/(2B)$ deleted red vertices.

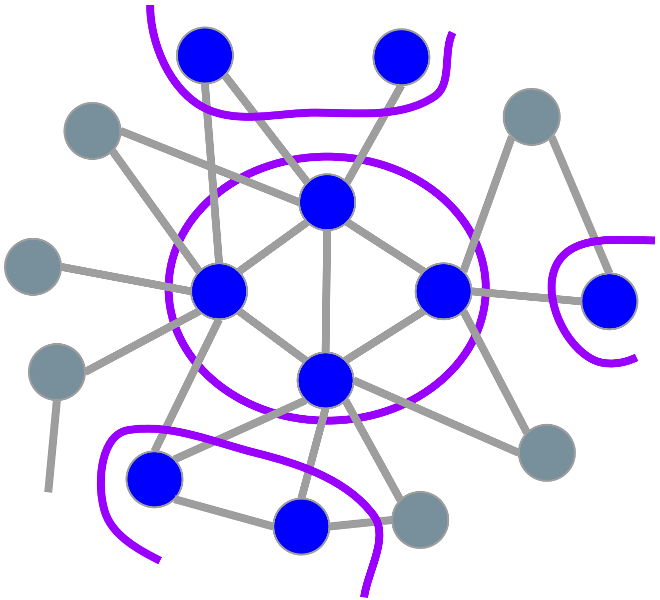
⇒ $\leq 1/(2B)$ fraction of vertices deleted per phase

⇒ $\leq B \cdot 1/(2B) = ½$ fraction of vertices deleted in total

Distributed ball carving

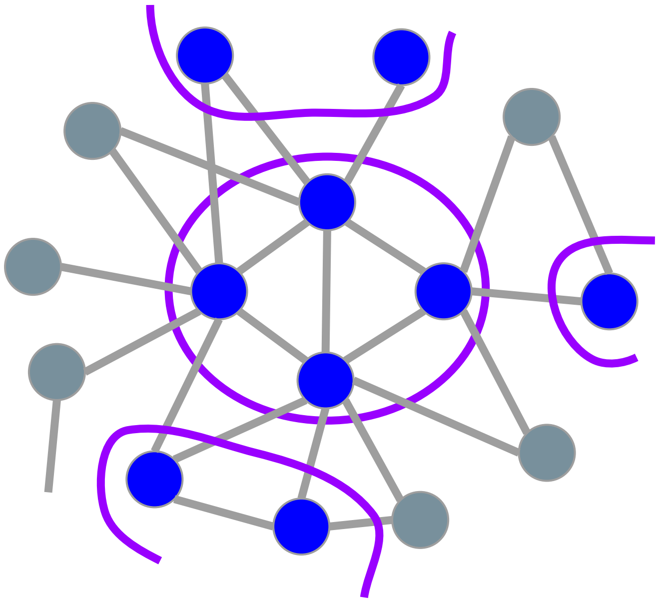
✓ 1. clusters at least ½ fraction of vertices
✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

Property 3:



2

Distributed ball carving

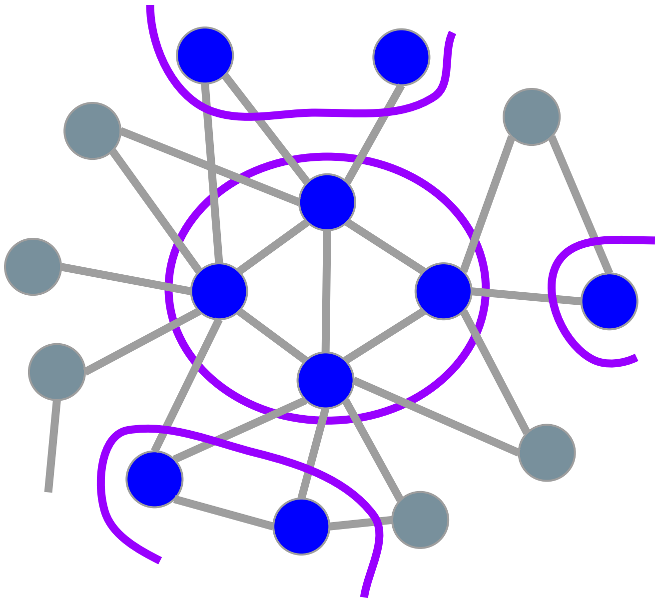


✓ 1. clusters at least ½ fraction of vertices
✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

Property 3:

At the end of each phase, there are no edges between red and blue nodes.

Distributed ball carving



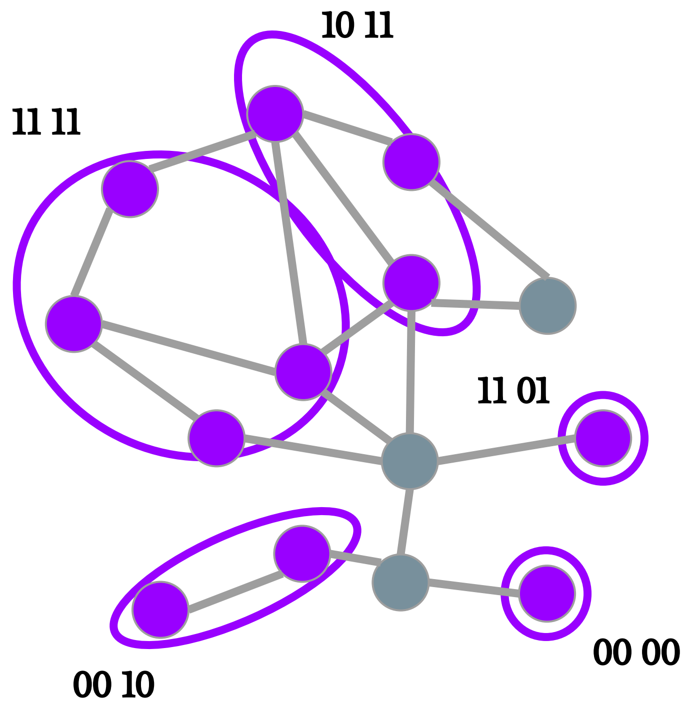
✓ 1. clusters at least ½ fraction of vertices
✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

Property 3:

At the end of each phase, there are no edges between red and blue nodes.

Otherwise there is a blue cluster of size $> (1+1/(2B))^{4B \ln n} > n$.

Distributed ball carving



✓ 1. clusters at least ½ fraction of vertices
✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$
3. clusters are non-adjacent

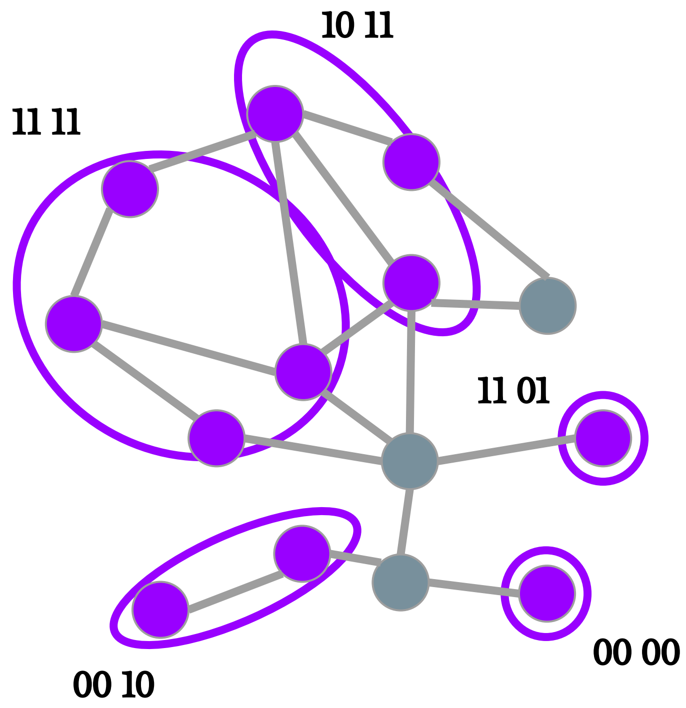
Property 3:

At the end of each phase, there are no edges between red and blue nodes.

Otherwise there is a blue cluster of size $> (1+1/(2B))^{4B \ln n} > n$.

After i-th phase, clusters in each connected component agree on their i-th bit, and this stays so during next phases.

Distributed ball carving



✓ 1. clusters at least ½ fraction of vertices
✓ 2. such that each cluster has weak-diameter $O(\log^3 n)$
✓ 3. clusters are non-adjacent

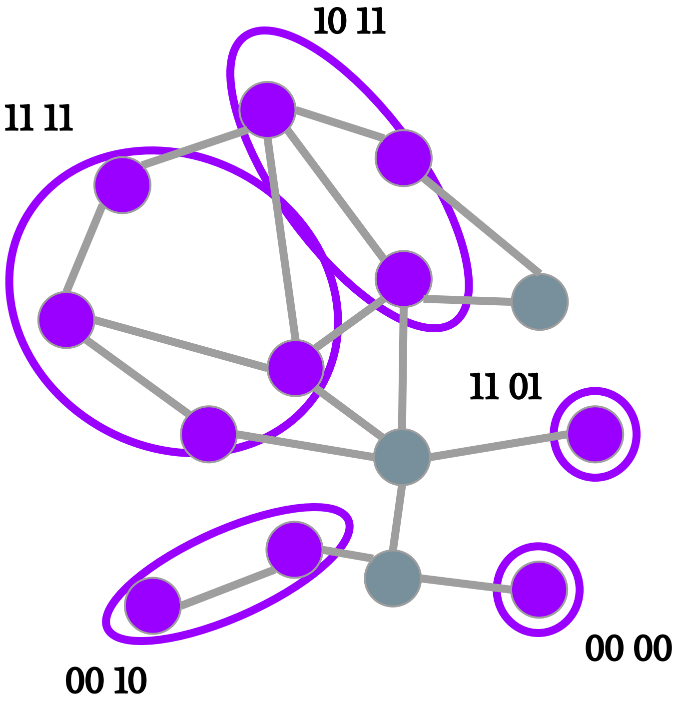
Property 3:

At the end of each phase, there are no edges between red and blue nodes.

Otherwise there is a blue cluster of size $> (1+1/(2B))^{4B \ln n} > n$.

After *i*-th phase, clusters in each connected component agree on their *i*-th bit, and this stays so during next phases.

Distributed ball carving



The running time of the whole algorithm is

$O(\log^7 n)$ =

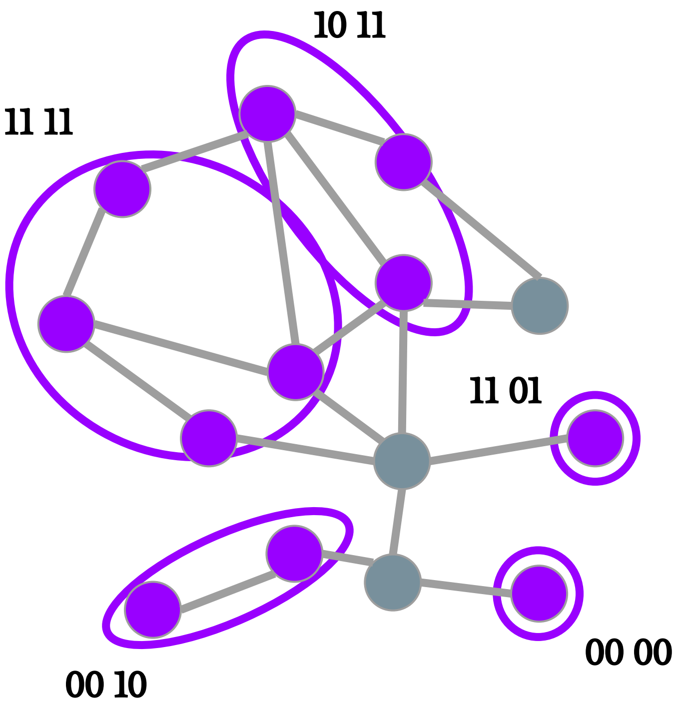
$O(\log n)$ # of colors of decomposition

· $O(\log n)$ # of phases

· $O(\log^2 n)$ steps per phase

· $O(\log^3 n)$ complexity of one step

Distributed ball carving



The running time of the whole algorithm is

$O(\log^7 n)$ =

$O(\log n)$	# of colors of decomposition
· $O(\log n)$	# of phases
· $O(\log^2 n)$	steps per phase
· $O(\log^3 n)$	complexity of one step

The whole algorithm works in the **CONGEST** model (see Section 2.2 in our paper).

Plan

1. More on **LOCAL** and **CONGEST** model
2. A deterministic algorithm for network decomposition.
   a. Sequential algorithm
   b. Distributed algorithm
3. Applications
   a. Derandomization and a bigger picture of the **LOCAL** model

General derandomization theorem

Theorem: [Ghaffari, Harris, Kuhn FOCS'18 + R., Ghaffari STOC'20]

P-LOCAL = P-RLOCAL.

P-LOCAL: problems* solvable by a deterministic poly($\log n$)-round algorithm in the **LOCAL** model

P-RLOCAL: problems* solvable by a randomized poly($\log n$)-round algorithm in the **LOCAL** model

*problems needs to be locally checkable = if a proposed solution is not correct, at least one node recognises that after looking at its poly($\log n$)-hop neighbourhood

General derandomization theorem

Theorem: [Ghaffari, Kuhn, Maus STOC'17 + Ghaffari, Harris, Kuhn FOCS'18 + R., Ghaffari STOC'20]

P-LOCAL = P-RLOCAL = P-SLOCAL = P-RSLOCAL.

P-LOCAL: problems* solvable by a deterministic poly(log n)-round algorithm in the **LOCAL** model

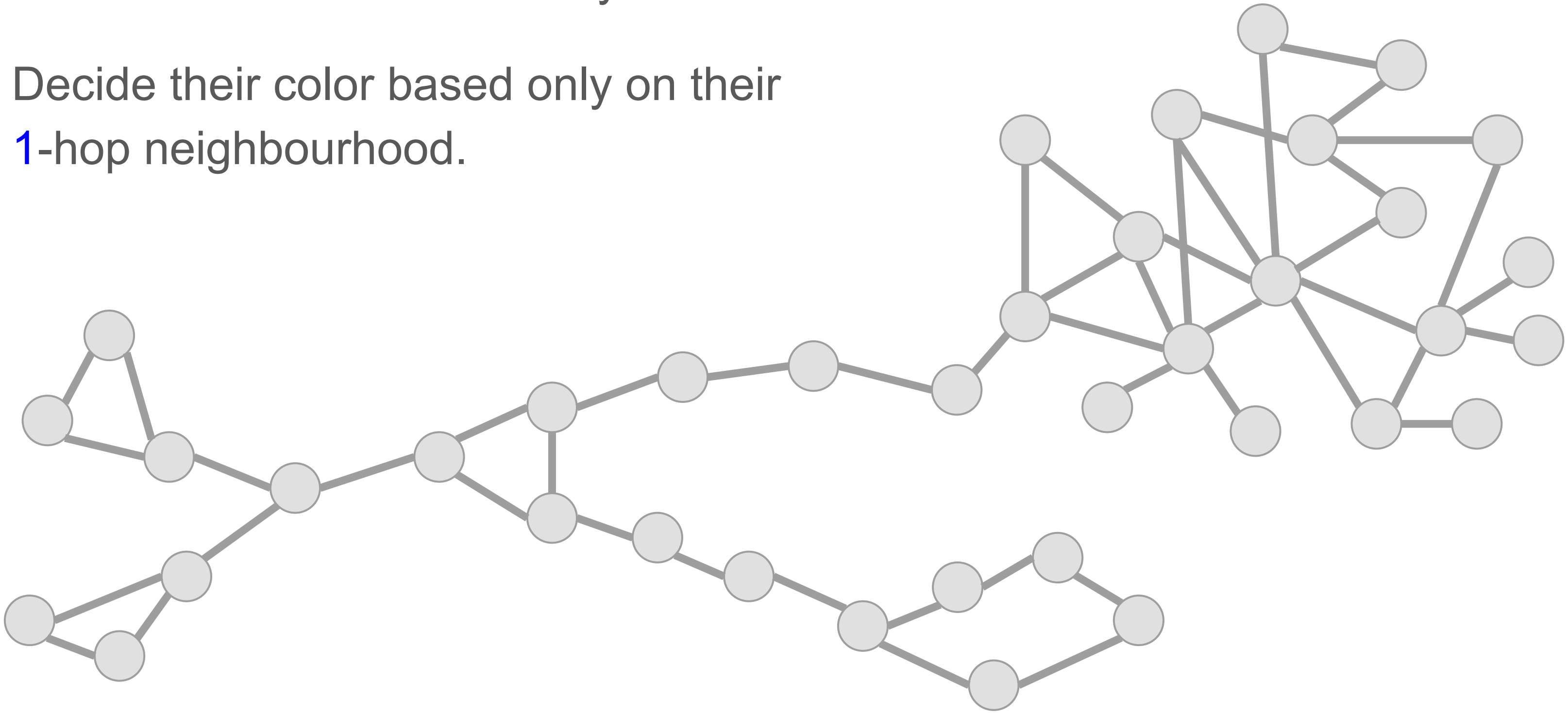
P-RLOCAL: problems* solvable by a randomized poly(log n)-round algorithm in the **LOCAL** model

*problems needs to be locally checkable = if a proposed solution is not correct, at least one node recognises that after looking at its poly(log n)-hop neighbourhood

Sequential Δ+1 coloring algorithm

Iterate over nodes in arbitrary order.

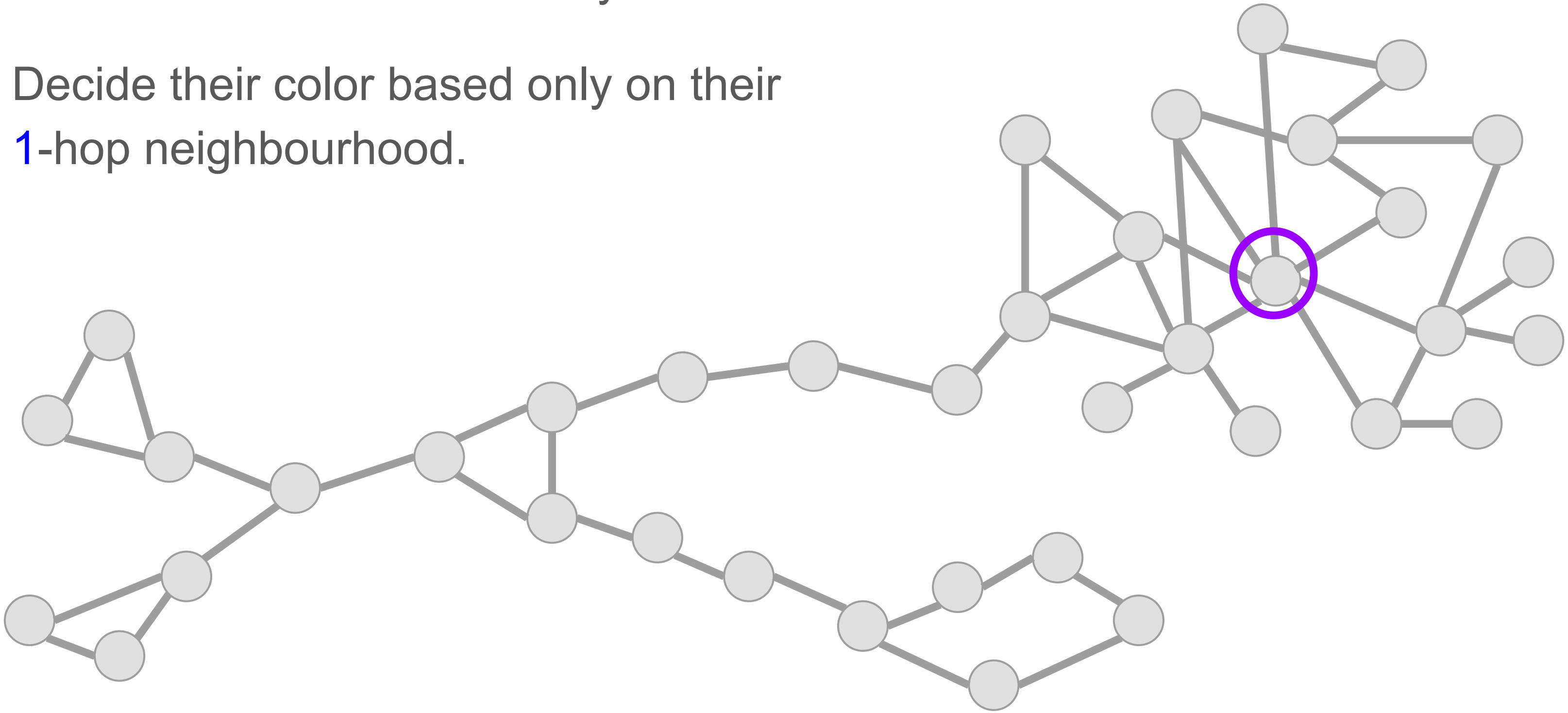
Decide their color based only on their 1-hop neighbourhood.



Sequential Δ+1 coloring algorithm

Iterate over nodes in arbitrary order.

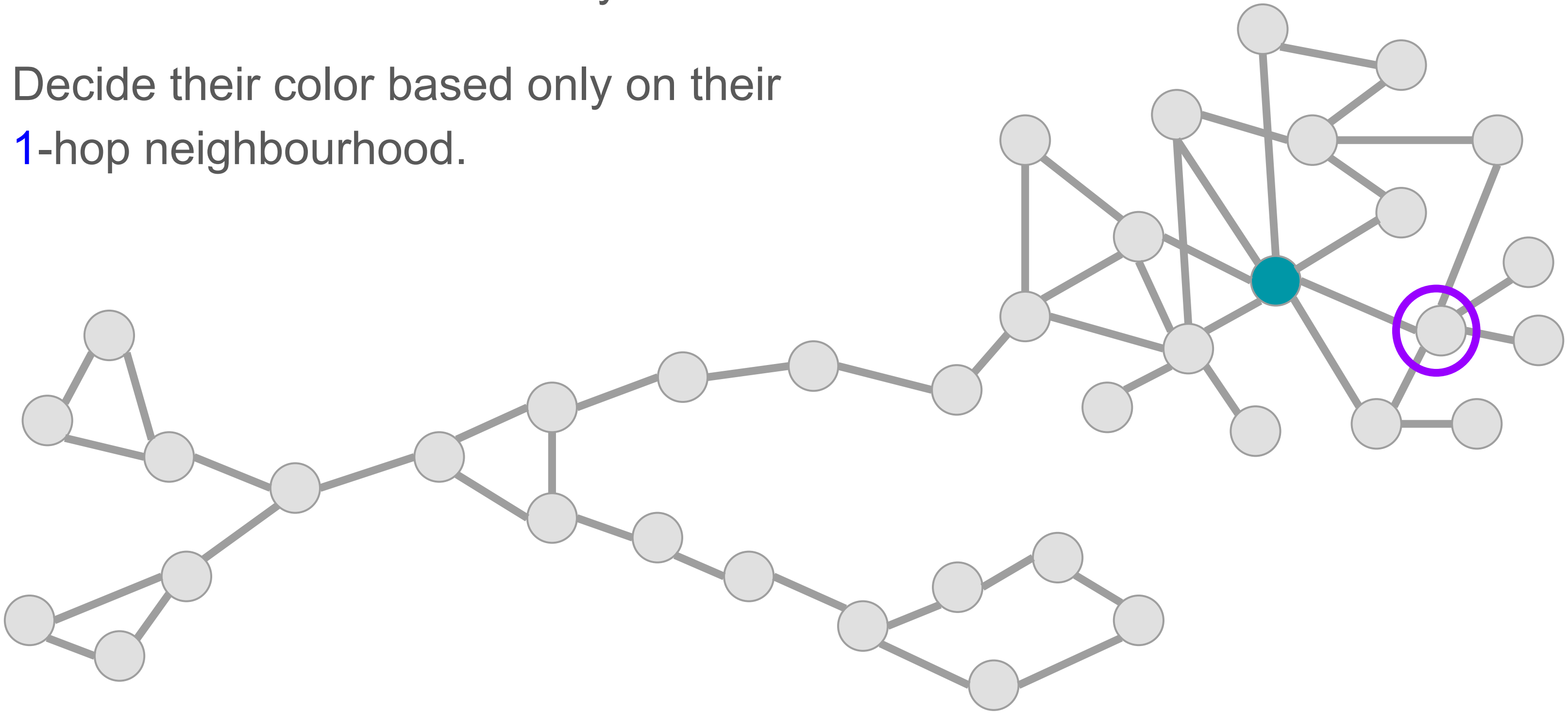
Decide their color based only on their 1-hop neighbourhood.



Sequential Δ+1 coloring algorithm

Iterate over nodes in arbitrary order.

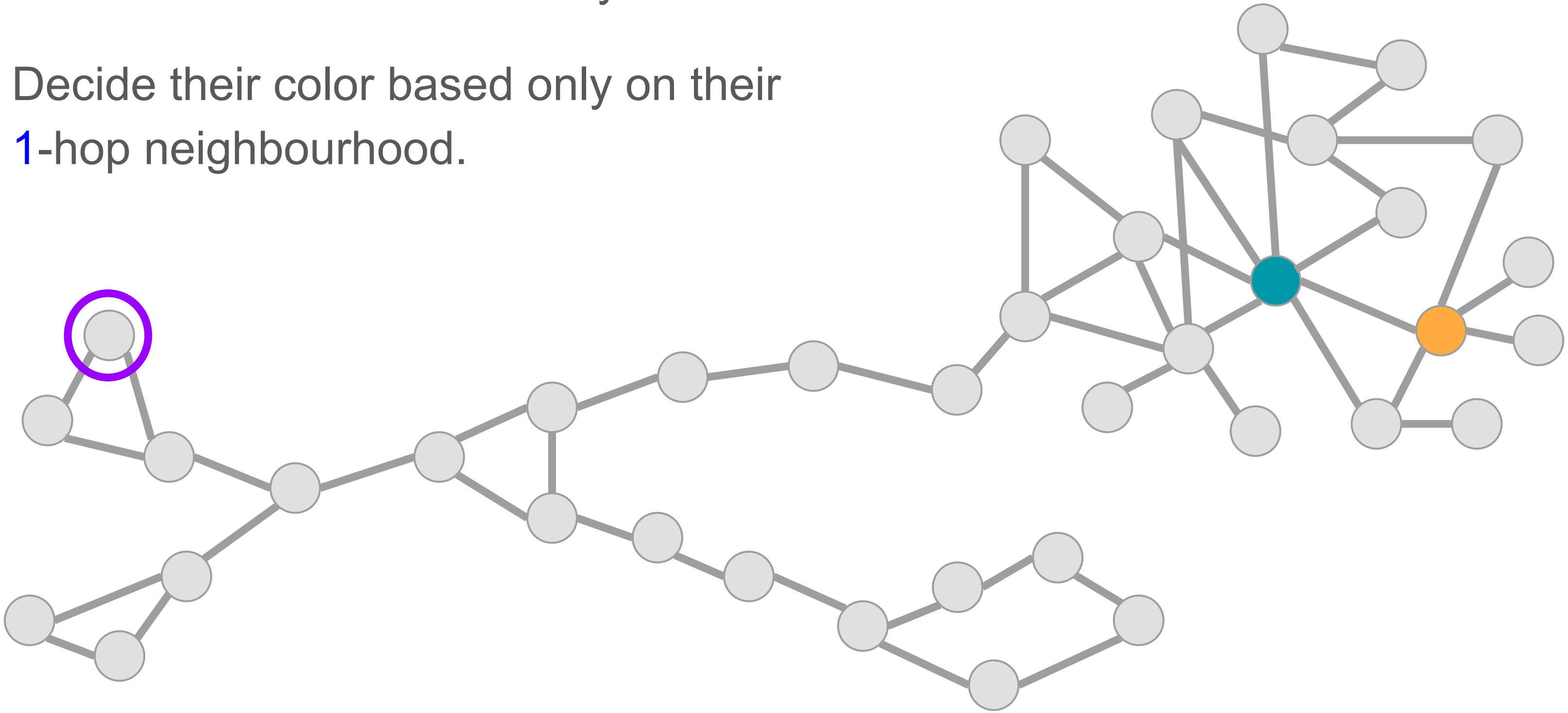
Decide their color based only on their 1-hop neighbourhood.



Sequential Δ+1 coloring algorithm

Iterate over nodes in arbitrary order.

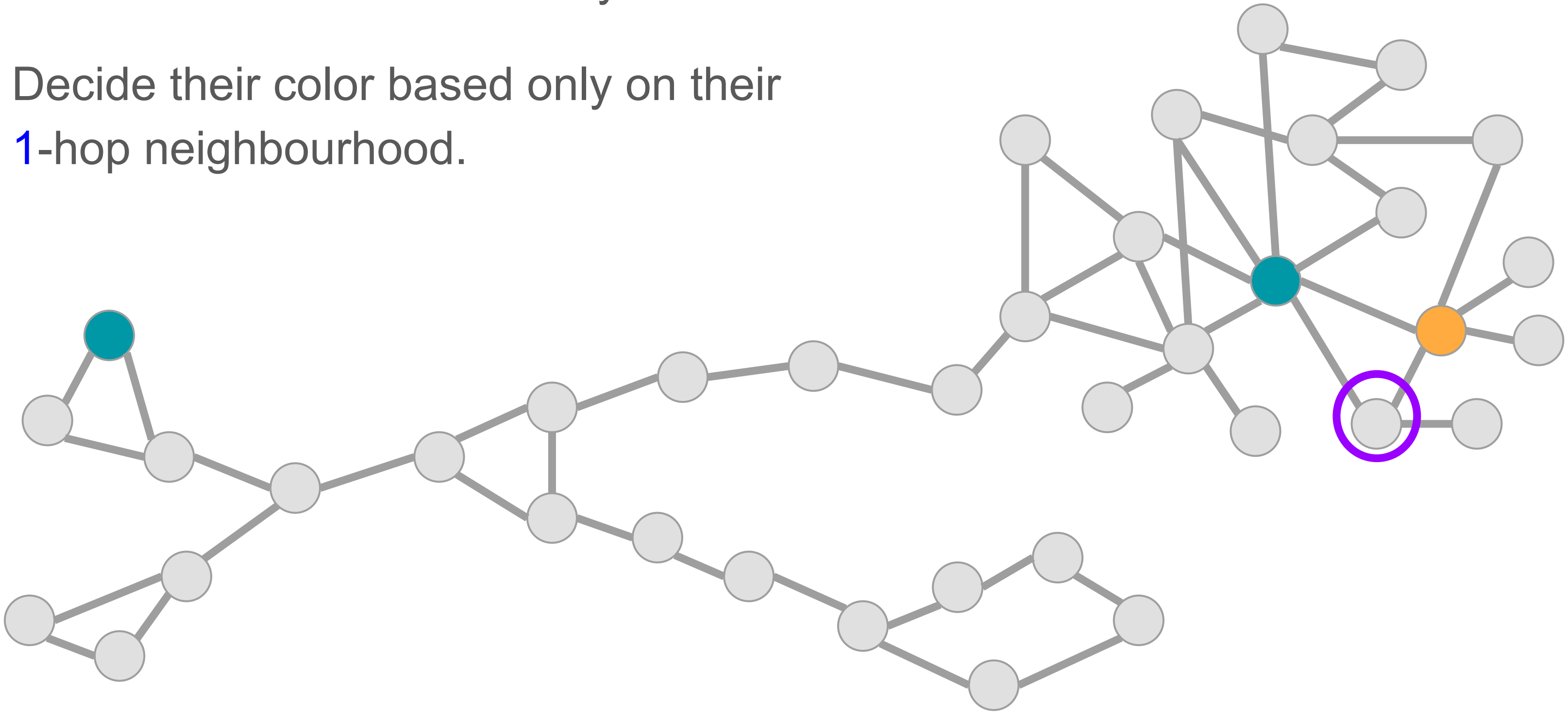
Decide their color based only on their 1-hop neighbourhood.



Sequential Δ+1 coloring algorithm

Iterate over nodes in arbitrary order.

Decide their color based only on their 1-hop neighbourhood.



SLOCAL - sequential variant of the **LOCAL** model

Iterate over nodes in adversarial order.

Decide their label based only on their r-hop neighbourhood.

Write the label to the node so that other nodes can see it.

P-SLOCAL: r = poly(log n), locally checkable

P-RSLOCAL: r = poly(log n), locally checkable, can use randomness

P-SLOCAL

“deterministic sequential”

P-RSLOCAL

“randomized sequential”

P-LOCAL

“deterministic distributed”

P-RLOCAL

“randomized distributed”

P-SLOCAL

“deterministic sequential”

↓ *deterministic* network decomposition [R., Ghaffari STOC’20]

↑ direct

P-LOCAL

“deterministic distributed”

P-SLOCAL

“deterministic sequential”

deterministic network decomposition

[R., Ghaffari STOC’20]

direct

P-LOCAL

“deterministic distributed”

Proof: for **P-SLOCAL** algorithm with locality r, construct network decomposition on G^r in $O(r \log^7 n)$ rounds with **C**=$O(\log n)$, **D**=$O(\log^3 n)$; iterate over color classes and simulate the sequential algorithm in $O(\log n \cdot r \log^3 n)$ rounds.

P-SLOCAL

"deterministic sequential"

deterministic network decomposition

[R., Ghaffari STOC'20]

direct

P-LOCAL

"deterministic distributed"

By the way: Network decomposition is a complete problem for this reduction.

P-SLOCAL

“deterministic sequential”

deterministic network decomposition

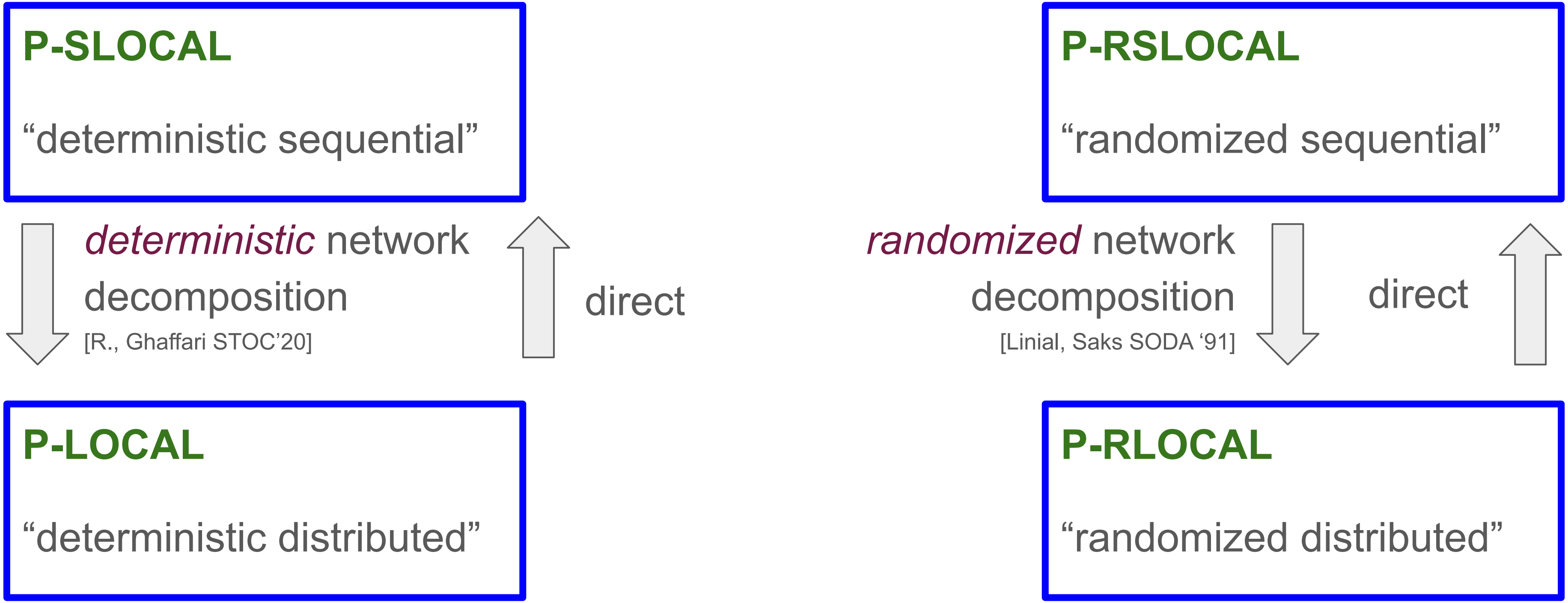
[R., Ghaffari STOC’20]

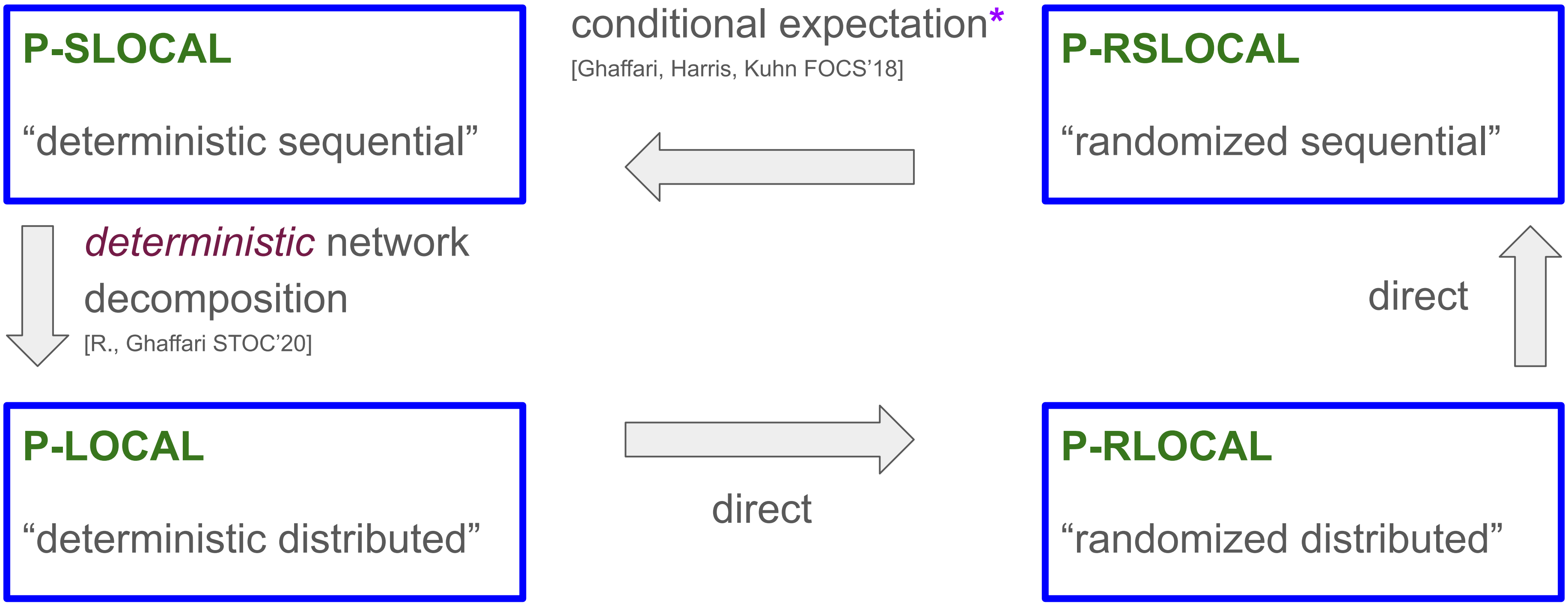
direct

P-LOCAL

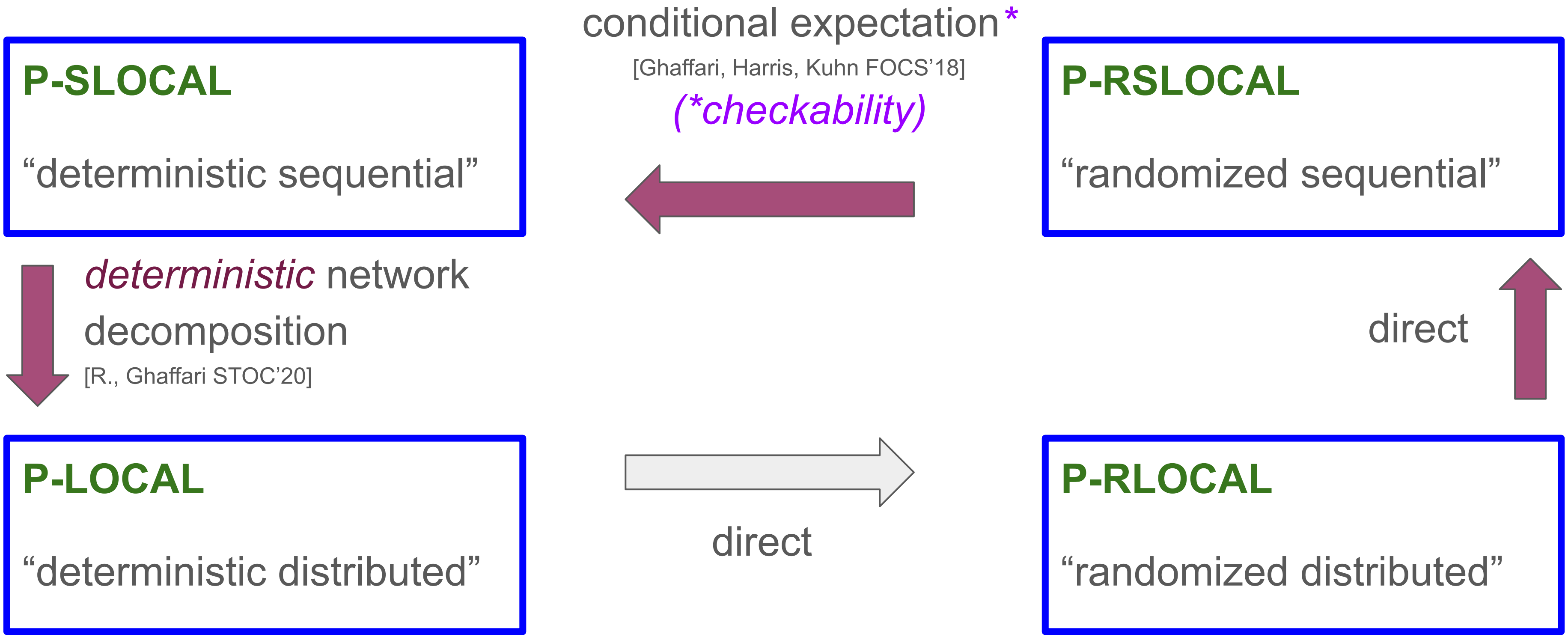
“deterministic distributed”

Corollary: There is an efficient deterministic algorithm for Δ+1 coloring, maximal independent set, strong diameter network decomposition

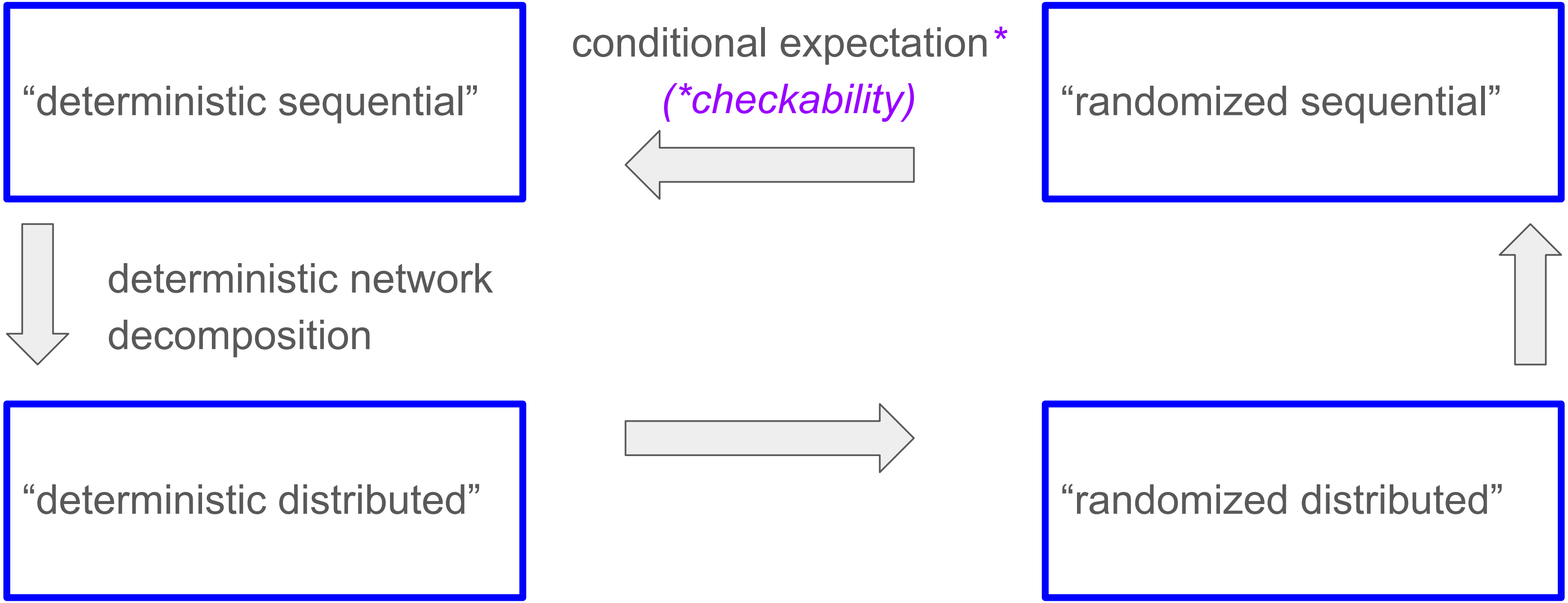

P-SLOCAL
“deterministic sequential”
deterministic network decomposition
[R., Ghaffari STOC’20]
direct
P-LOCAL
“deterministic distributed”
P-RSLOCAL
“randomized sequential”
randomized network decomposition
[Linial, Saks SODA ‘91]
direct
P-RLOCAL
“randomized distributed”



* for problems locally checkable in poly(log n) rounds



Corollary: There is an efficient deterministic algorithm for Lovász local lemma,...



We see a clean first-order theory of the LOCAL model.

Plan

1. More on **LOCAL** and **CONGEST** model
2. A deterministic algorithm for network decomposition.
   a. Sequential algorithm
   b. Distributed algorithm
3. Applications
   a. Derandomization and a bigger picture of the **LOCAL** model
   b. $\Delta+1$ coloring, MIS, Lovász local lemma

Δ+1 coloring

LOCAL, deterministic

LOCAL, randomized

MPC, randomized

Δ+1 coloring

LOCAL, deterministic

poly(log n)

LOCAL, randomized

MPC, randomized

Δ+1 coloring

LOCAL, deterministic

poly(log n)

LOCAL, randomized

poly(log log n)

MPC, randomized

shattering [Chang, Li, Pettie STOC'18]

+ network decomposition [R Ghaffari STOC'20]

Δ+1 coloring

amplification of success probability

[Chang, Kopelowitz, and Pettie FOCS'16]

LOCAL, deterministic

poly(log n)

LOCAL, randomized

poly(log log n)

MPC, randomized

shattering [Chang, Li, Pettie STOC'18]

+ network decomposition [R Ghaffari STOC'20]

Δ+1 coloring

amplification of success probability
[Chang, Kopelowitz, and Pettie FOCS'16]

LOCAL, deterministic
poly(log n)

LOCAL, randomized
poly(log log n)

MPC, randomized
O(log log log n)

shattering [Chang, Li, Pettie STOC'18]
+ network decomposition [R Ghaffari STOC'20]

graph exponentiation
[Chang, Fischer, Ghaffari, Uitto, Zheng PODC'19]

Δ+1 coloring

amplification of success probability
[Chang, Kopelowitz, and Pettie FOCS'16]

conditioned on hardness of connectivity in **MPC** [Ghaffari, Kuhn, Uitto FOCS'19]

LOCAL, deterministic

poly(log n)

LOCAL, randomized

poly(log log n)

MPC, randomized

O(log log log n)

shattering [Chang, Li, Pettie STOC'18]
+ network decomposition [R Ghaffari STOC'20]

graph exponentiation
[Chang, Fischer, Ghaffari, Uitto, Zheng PODC'19]

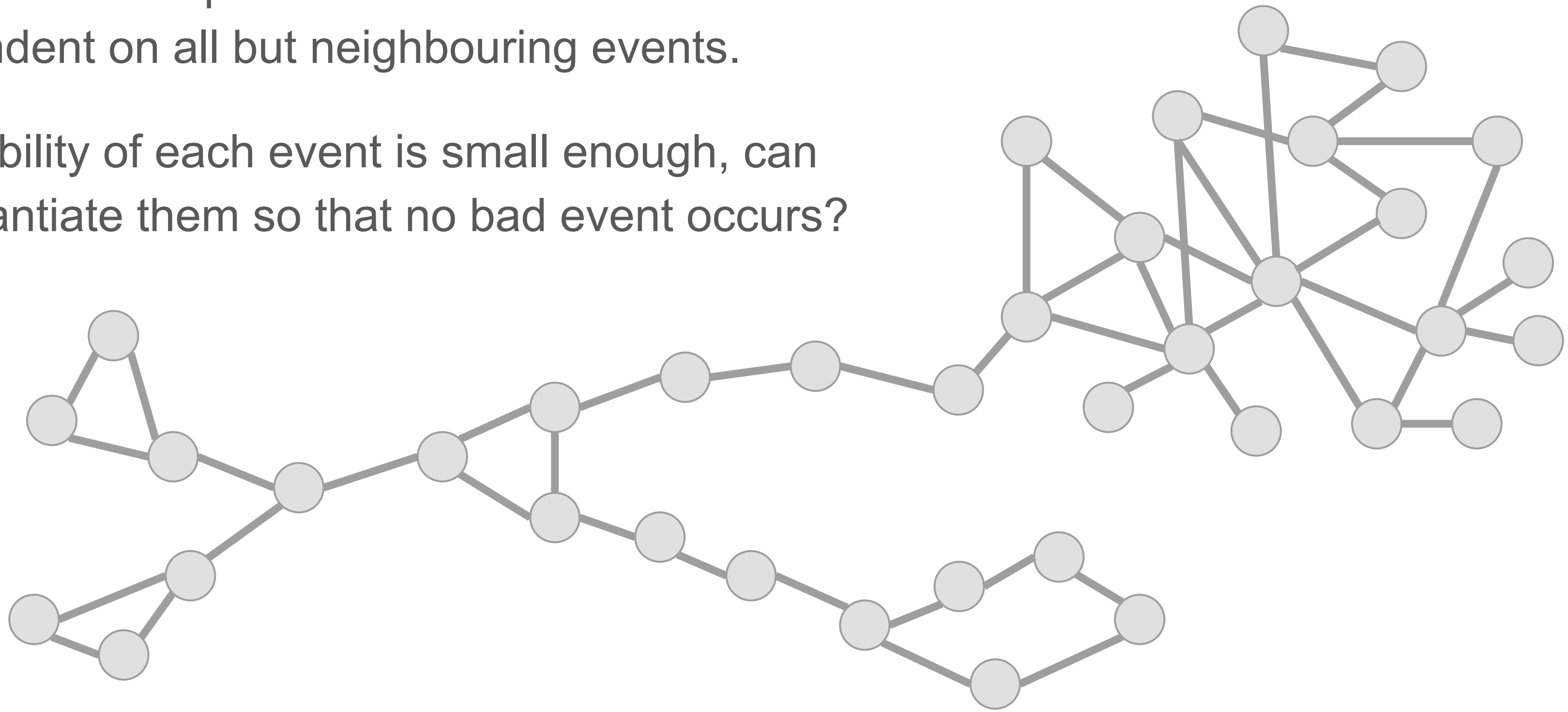
Maximal independent set (MIS)

	Upper bound	Lower bound
LOCAL, deterministic	$O(\log^7 n)$ [R. Ghaffari STOC'20]	$\Omega(\log n / \log\log n)$ [Balliu et al. FOCS'19]
LOCAL, randomized	$O(\log \Delta) + \mathrm{poly}(\log\log n)$ [Ghaffari SODA'16]	$\Omega(\log \Delta / \log\log \Delta)$ [Kuhn et al. J.ACM'16] $o(\Delta) + o(\log\log n / \log\log\log n)$ is impossible [Balliu et al. FOCS'19]

Lovász local lemma

Each node corresponds to “bad” event independent on all but neighbouring events.

If probability of each event is small enough, can we instantiate them so that no bad event occurs?



Lovász local lemma ($p = \Delta^{-10}$)

	Upper bound	Lower bound
LOCAL, deterministic	poly(log n) [Ghaffari, Harris, Kuhn FOCS'18 + R. Ghaffari STOC'20]	$\Omega(\log n)$ [Chang, Kopelowitz, Pettie FOCS'16]
LOCAL, randomized	$O(\log^2 n)$ [Moser, Tardos J.ACM'10] $O(\Delta^2)$ + poly(log log n) [Fischer, Ghaffari DISC'17]	$\Omega(\log \log n)$ [Brandt et al., SODA'16]

Lovász local lemma

Theorem [Chang, Pettie FOCS'17]:

In graphs of degree O(1), problems checkable with locality O(1) have randomized complexity of either $\Omega(\log n)$ or $O(T_{LLL})$.

Here, T_{LLL} is the randomized complexity of Lovász local lemma on constant degree graphs.

Plan

1. More on **LOCAL** and **CONGEST** model
2. A deterministic algorithm for network decomposition.
   a. Sequential algorithm
   b. Distributed algorithm
3. Applications
   a. Derandomization and a bigger picture of the **LOCAL** model
   b. Δ+1 coloring, MIS, Lovász local lemma
   c. **CONGEST** model and open problems

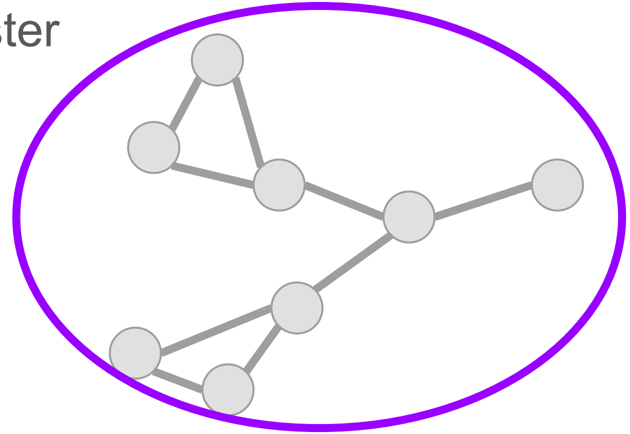
CONGEST model

Recall: In one round, only *$O(\log n)$* bits of information can be sent through an edge.

CONGEST model

Recall: In one round, only $O(\log n)$ bits of information can be sent through an edge.

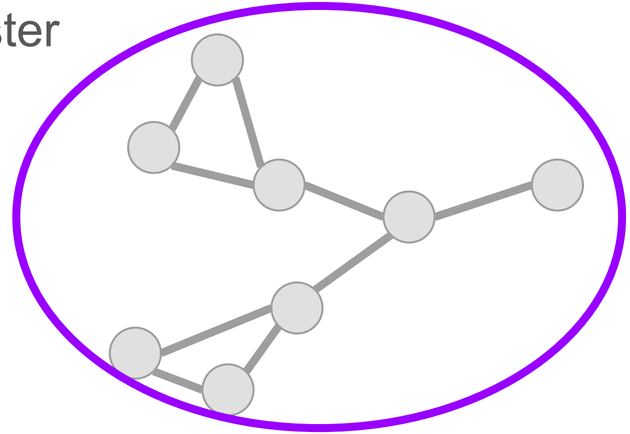
In general, we cannot collect the whole topology of a cluster



CONGEST model

Recall: In one round, only *O(log n)* bits of information can be sent through an edge.

In general, we cannot collect the whole topology of a cluster



Theorem: [Censor-Hillel, Parter, Schwartzman DISC'17; R. Ghaffari STOC'20]

There is poly(log n)-round **CONGEST** algorithm for MIS.

Theorem: [Bamberger, Kuhn, Maus PODC'20; R. Ghaffari STOC'20]

There is poly(log n)-round **CONGEST** algorithm for Δ+1 coloring.

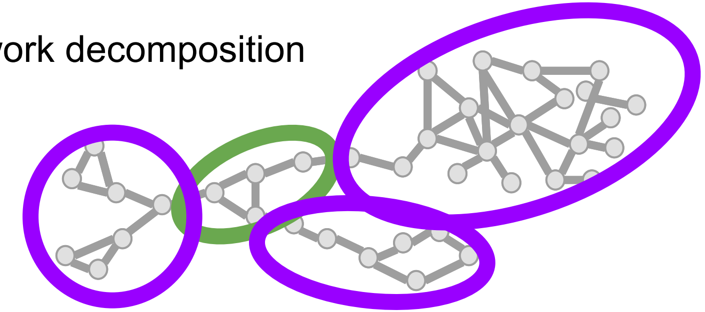
Open problems

Find deterministic algorithm for MIS, Δ+1 coloring, … in the **LOCAL** model faster than state-of-the-art algorithm for network decomposition.

Find a *combinatorial* deterministic poly(log n)-round algorithm for MIS, Δ+1 coloring, ... in the **CONGEST** model.

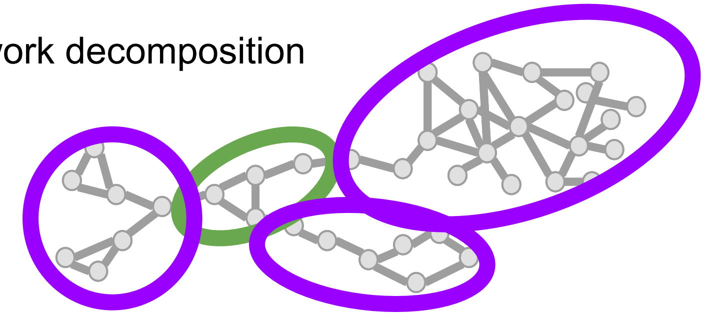
Summary

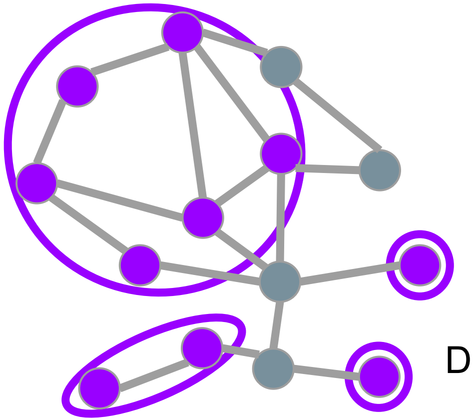
Network decomposition



Summary

Network decomposition

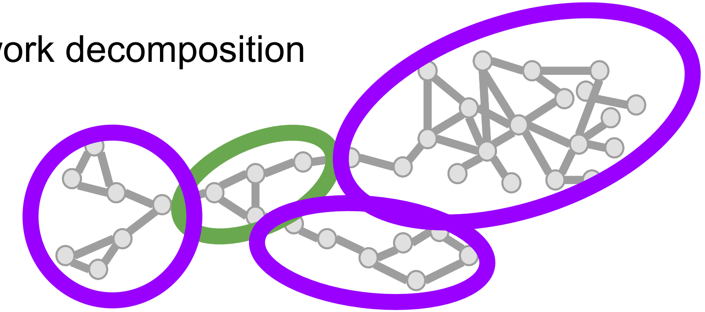


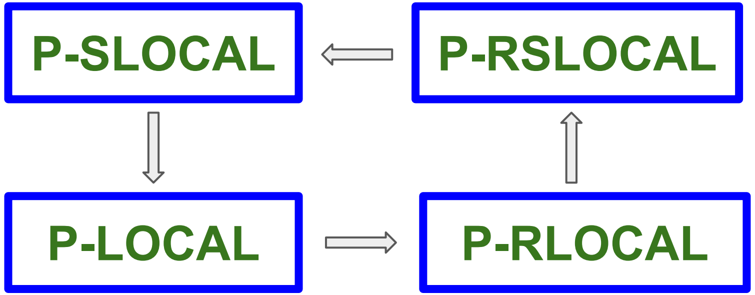


Distributed ball carving

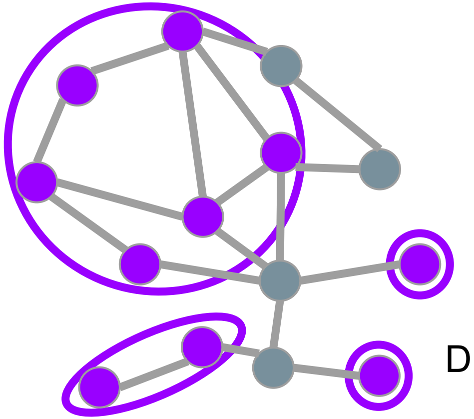
Summary

Network decomposition





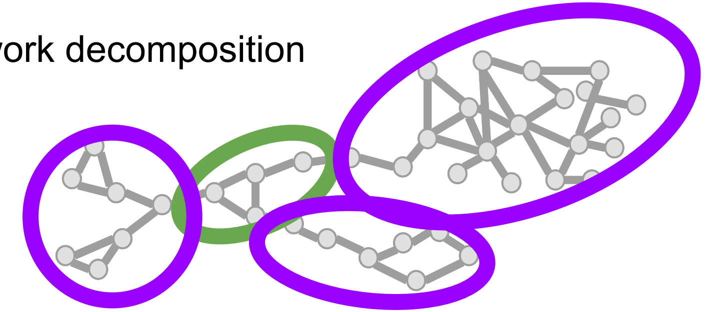
Big picture of the LOCAL model

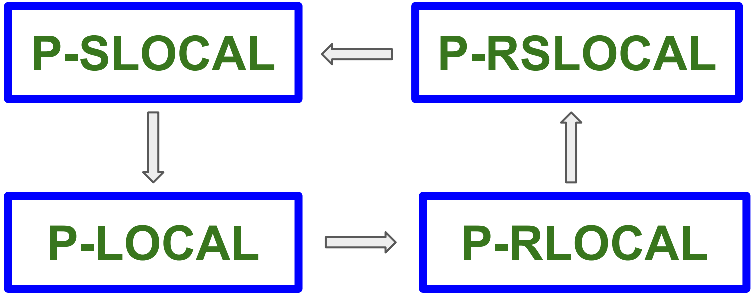


Distributed ball carving

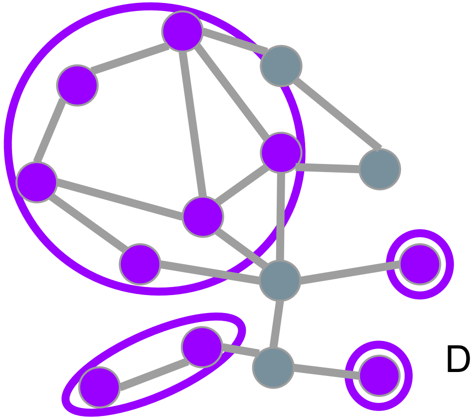
Summary

Network decomposition





Big picture of the LOCAL model



Distributed ball carving

Δ+1 coloring and connections across models