1.5-Approximation Algorithm for the 2-Convex Recoloring Problem

Reuven Bar-Yehuda [1] **Gilad Kutiel** [1] Dror Rawitz [2]

[1]Computer Science, Technion, Israel

[2]Engineering, Bar Ilan University, Israel

October 6, 2015

Convex Coloring

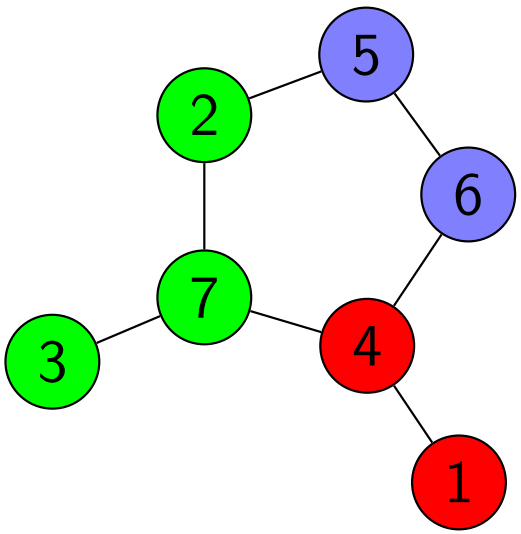
Definition

A Convex Coloring is a vertex coloring of a graph such that every color induces a **connected component**

Convex Coloring

Definition

A Convex Coloring is a vertex coloring of a graph such that every color induces a **connected component**

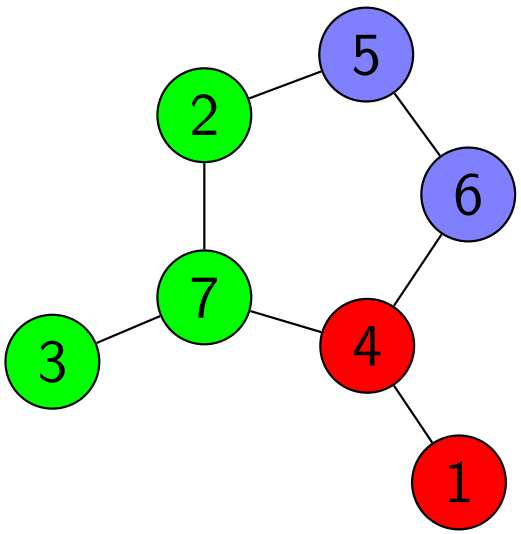


Convex

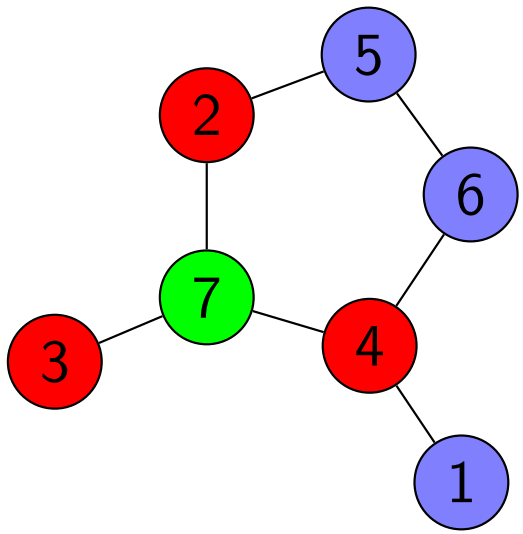
Convex Coloring

Definition

A Convex Coloring is a vertex coloring of a graph such that every color induces a **connected component**



Convex



Non-Convex

Convex Recoloring Problem

- **Instance**: a colored graph

Convex Recoloring Problem

- **Instance**: a colored graph
- **Solution**: convex coloring (recoloring)

Convex Recoloring Problem

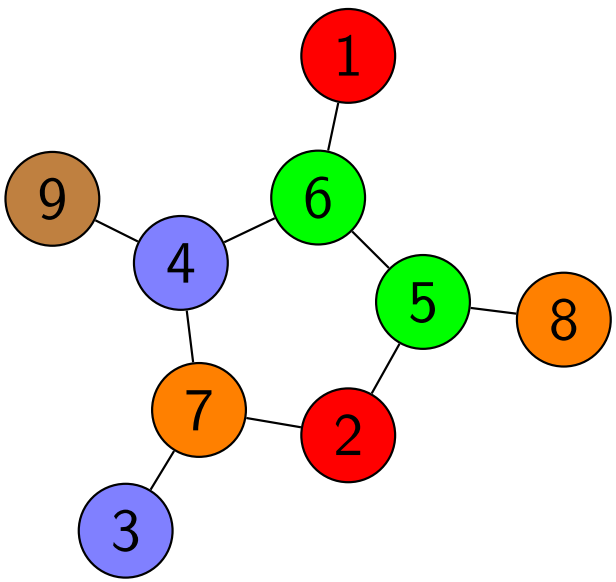
- **Instance**: a colored graph
- **Solution**: convex coloring (recoloring)
- **Cost**: # of vertices (sum of weights) that change color

Convex Recoloring Problem

- **Instance**: a colored graph
- **Solution**: convex coloring (recoloring)
- **Cost**: # of vertices (sum of weights) that change color
- **Goal**: minimize the cost

Convex Recoloring Problem

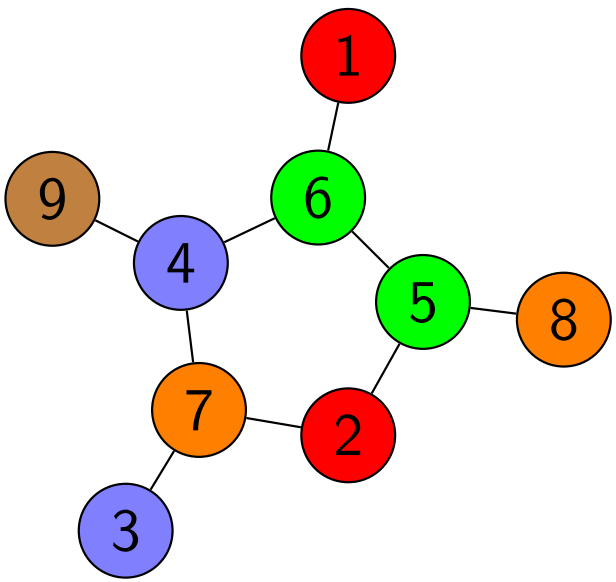
- **Instance**: a colored graph
- **Solution**: convex coloring (recoloring)
- **Cost**: # of vertices (sum of weights) that change color
- **Goal**: minimize the cost



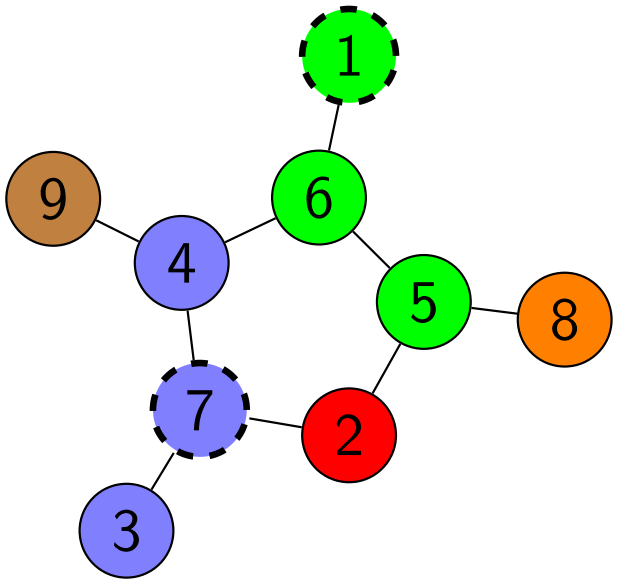
Input

Convex Recoloring Problem

- **Instance**: a colored graph
- **Solution**: convex coloring (recoloring)
- **Cost**: # of vertices (sum of weights) that change color
- **Goal**: minimize the cost



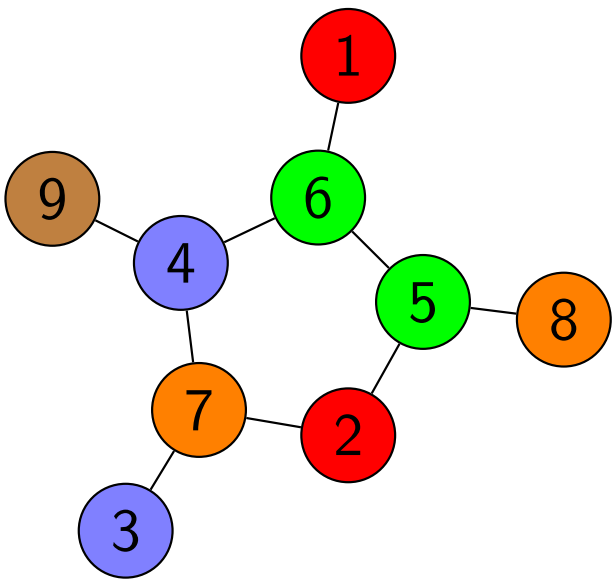
Input



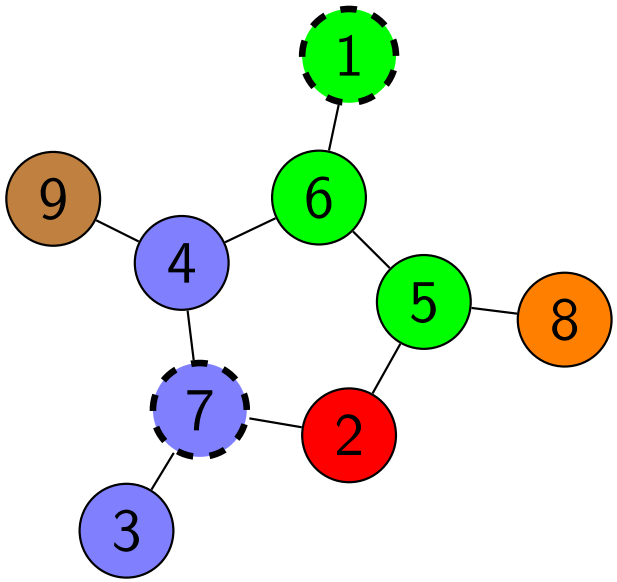
Output

Convex Recoloring Problem

- **Instance**: a colored graph
- **Solution**: convex coloring (recoloring)
- **Cost**: # of vertices (sum of weights) that change color
- **Goal**: minimize the cost



Input



Output

- **2-convex recoloring**: at most 2 vertices with the same color (input)

Previous Work (non exhaustive)

Convex Recoloring

- NP-hard on paths [Moran and Snir, 2005]

Previous Work (non exhaustive)

Convex Recoloring

- NP-hard on paths [Moran and Snir, 2005]
- 2-approx. alg. on weighted paths [Moran and Snir, 2005]

Previous Work (non exhaustive)

Convex Recoloring

- NP-hard on paths [Moran and Snir, 2005]
- 2-approx. alg. on weighted paths [Moran and Snir, 2005]
- $(2+\epsilon)$-approx. alg. on weighted trees [Bar-Yehuda et al., 2006]

Previous Work (non exhaustive)

Convex Recoloring

- NP-hard on paths [Moran and Snir, 2005]
- 2-approx. alg. on weighted paths [Moran and Snir, 2005]
- $(2+\epsilon)$-approx. alg. on weighted trees [Bar-Yehuda et al., 2006]
- $O(nk2^k)$ FP alg. on trees [Bar-Yehuda et al., 2006]

* k is the cost of the optimal solution

Previous Work (non exhaustive)

Convex Recoloring

- NP-hard on paths [Moran and Snir, 2005]
- 2-approx. alg. on weighted paths [Moran and Snir, 2005]
- $(2+\epsilon)$-approx. alg. on weighted trees [Bar-Yehuda et al., 2006]
- $O(nk2^k)$ FP alg. on trees [Bar-Yehuda et al., 2006]
- $O(k^2)$ kernel size on trees [Bodlaender et al., 2007]

* k is the cost of the optimal solution

Previous Work (non exhaustive)

Convex Recoloring

- NP-hard on paths [Moran and Snir, 2005]
- 2-approx. alg. on weighted paths [Moran and Snir, 2005]
- $(2+\epsilon)$-approx. alg. on weighted trees [Bar-Yehuda et al., 2006]
- $O(nk2^k)$ FP alg. on trees [Bar-Yehuda et al., 2006]
- $O(k^2)$ kernel size on trees [Bodlaender et al., 2007]
- $\Omega(\ln n)$ approx. bound on bipartite graphs [Campêlo et al., 2013]

* k is the cost of the optimal solution

** assuming $P \neq NP$

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]
- $\Omega(\ln \ln n)$ approx. bound on (binary) weighted graphs [Kammer et al., 2008]

* assuming $P \neq NP$

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]
- $\Omega(\ln \ln n)$ approx. bound on (binary) weighted graphs [Kammer et al., 2008]
- 3/2-approx. alg. on paths [Lima et al., 2011]

* assuming $P \neq NP$

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]
- $\Omega(\ln \ln n)$ approx. bound on (binary) weighted graphs [Kammer et al., 2008]
- 3/2-approx. alg. on paths [Lima et al., 2011]
- No approximation for (binary) weighted graphs

* assuming $P \neq NP$

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]
- $\Omega(\ln \ln n)$ approx. bound on (binary) weighted graphs [Kammer et al., 2008]
- 3/2-approx. alg. on paths [Lima et al., 2011]
- No approximation for (binary) weighted graphs
- 3/2-approx. alg.

* assuming $P \neq NP$

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]
- $\Omega(\ln \ln n)$ approx. bound on (binary) weighted graphs [Kammer et al., 2008]
- 3/2-approx. alg. on paths [Lima et al., 2011]
- No approximation for (binary) weighted graphs
- 3/2-approx. alg.
- 5/4-approx. alg. on paths

* assuming $P \neq NP$

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]
- $\Omega(\ln \ln n)$ approx. bound on (binary) weighted graphs [Kammer et al., 2008]
- 3/2-approx. alg. on paths [Lima et al., 2011]
- No approximation for (binary) weighted graphs
- 3/2-approx. alg.
- 5/4-approx. alg. on paths
- $O(k)$ kernel size

* assuming $P \neq NP$

** k is the cost of the optimal solution

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- NP-hard on paths [Kanj and Kratsch. 2008]
- $\Omega(\ln \ln n)$ approx. bound on (binary) weighted graphs [Kammer et al., 2008]
- 3/2-approx. alg. on paths [Lima et al., 2011]
- No approximation for (binary) weighted graphs
- 3/2-approx. alg.
- 5/4-approx. alg. on paths
- $O(k)$ kernel size
- $O(n + 2^{k \log k})$ FP alg.

* assuming $P \neq NP$

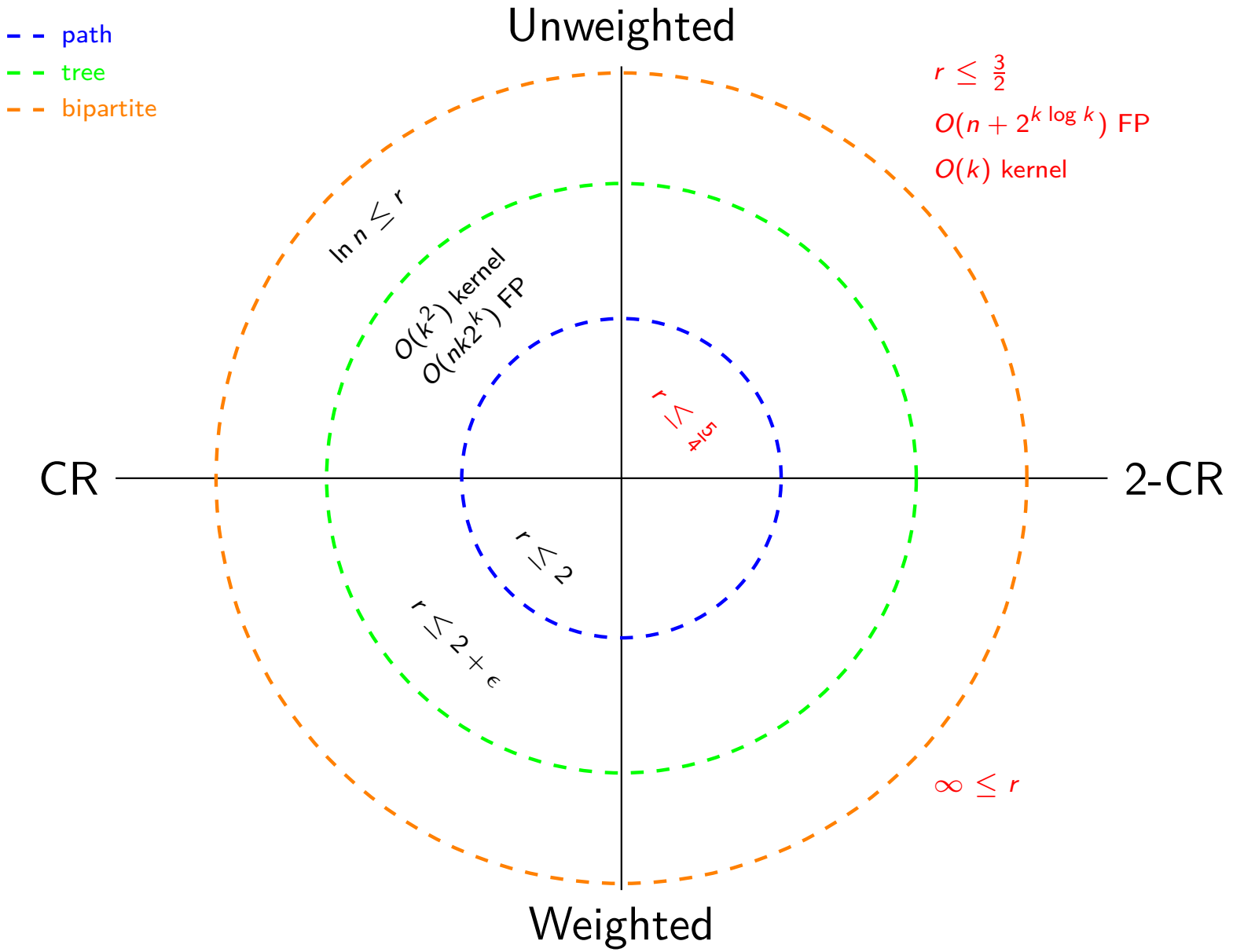
** k is the cost of the optimal solution

Previous/Our Work (non exhaustive)

2-Convex Recoloring (2-CR)

- 3/2-approx. alg.

Previous/Our Work (non exhaustive)



Partial Convex Coloring

Definition

Partial Coloring: coloring of a subset of the vertices

Partial Convex Coloring

Definition

Partial Coloring: coloring of a subset of the vertices

- Same definition of convexity

Partial Convex Coloring

Definition

Partial Coloring: coloring of a subset of the vertices

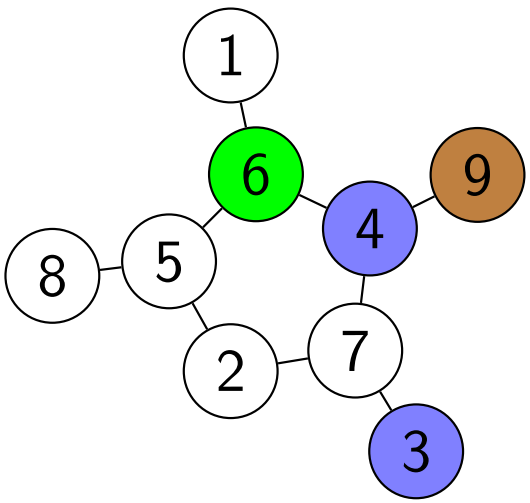
- Same definition of convexity
- There are no constraints on the uncolored vertices

Partial Convex Coloring

Definition

Partial Coloring: coloring of a subset of the vertices

- Same definition of convexity
- There are no constraints on the uncolored vertices



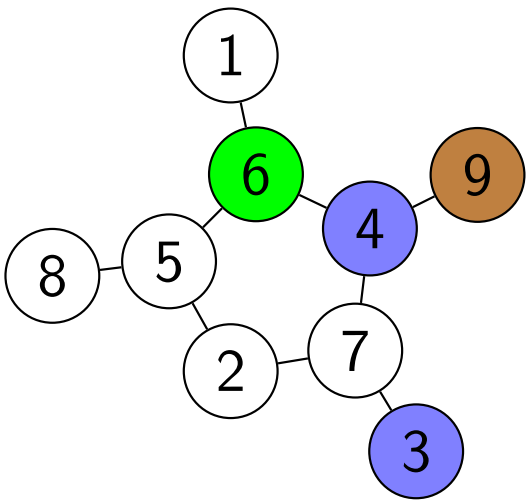
Non-Convex

Partial Convex Coloring

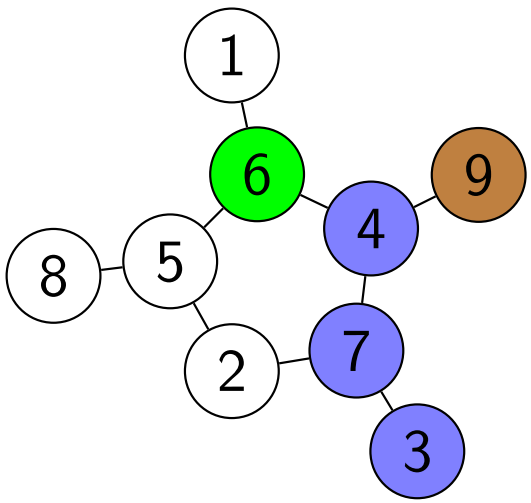
Definition

Partial Coloring: coloring of a subset of the vertices

- Same definition of convexity
- There are no constraints on the uncolored vertices



Non-Convex



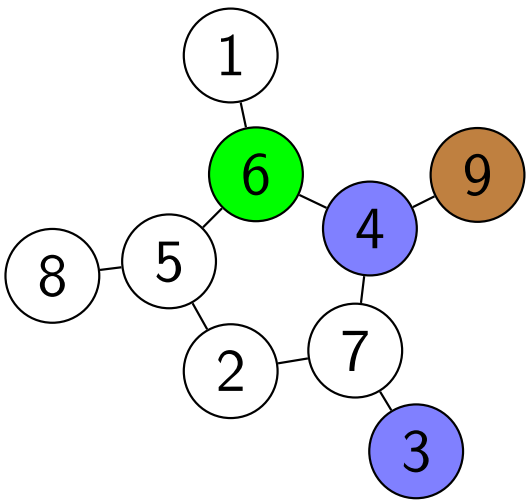
Convex

Partial Convex Coloring

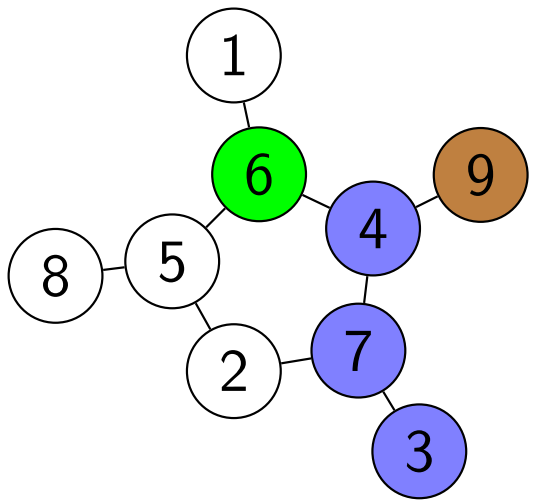
Definition

Partial Coloring: coloring of a subset of the vertices

- Same definition of convexity
- There are no constraints on the uncolored vertices



Non-Convex



Convex

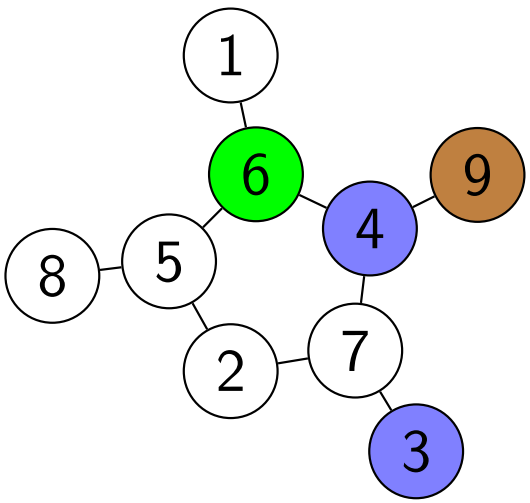
- Can be completed to a (total) convex coloring

Partial Convex Coloring

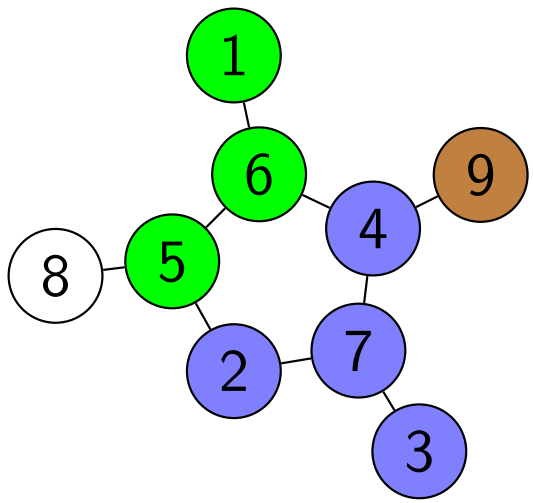
Definition

Partial Coloring: coloring of a subset of the vertices

- Same definition of convexity
- There are no constraints on the uncolored vertices



Non-Convex



Convex

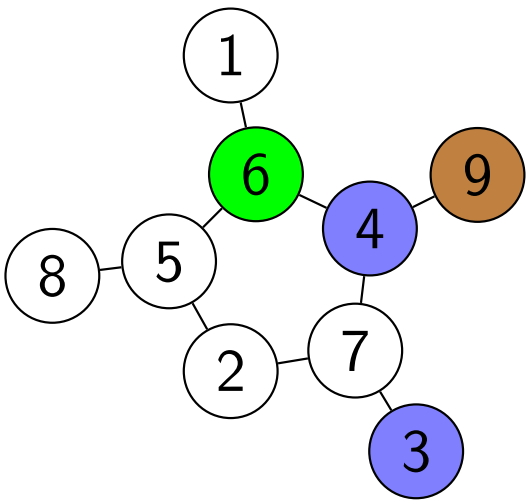
- Can be completed to a (total) convex coloring

Partial Convex Coloring

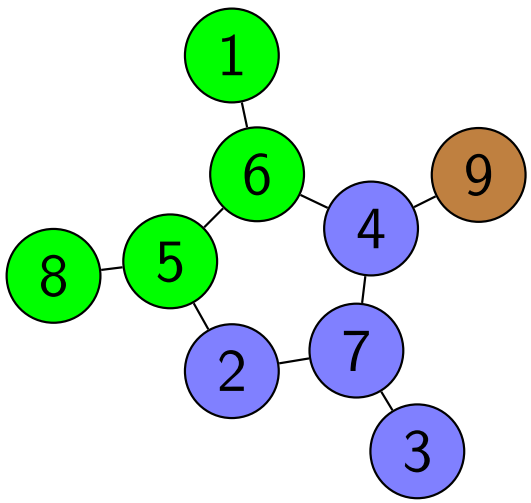
Definition

Partial Coloring: coloring of a subset of the vertices

- Same definition of convexity
- There are no constraints on the uncolored vertices



Non-Convex



Convex

- Can be completed to a (total) convex coloring

Partial Convex Recoloring

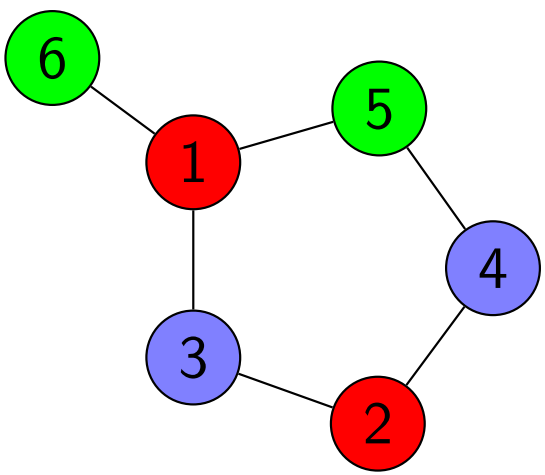
- We accept a partial convex recoloring as the output for 2-CR

Partial Convex Recoloring

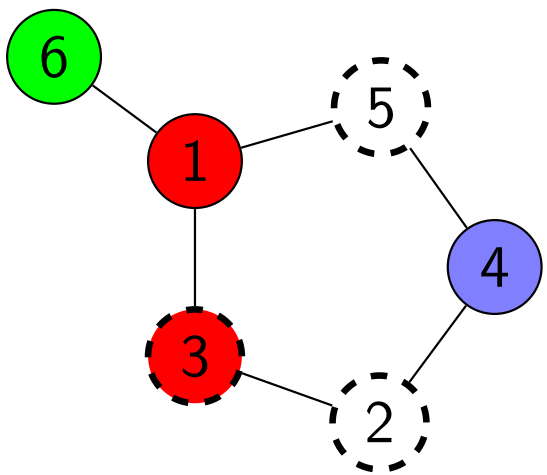
- We accept a partial convex recoloring as the output for 2-CR
- The cost of a partial recoloring is:
  # of vertices that **change** color + # of **uncolored** vertices

Partial Convex Recoloring

- We accept a partial convex recoloring as the output for 2-CR
- The cost of a partial recoloring is:
  # of vertices that **change** color + # of **uncolored** vertices



Input



Output

Retains-All (Partial) Convex Recoloring

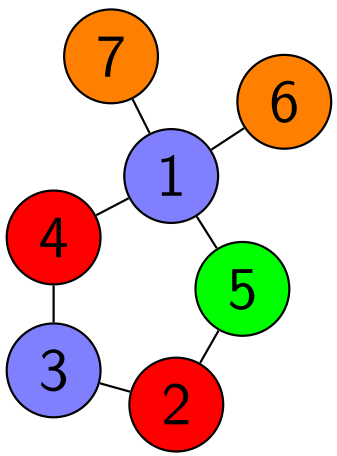
Definition

A Retains-All Recoloring is a convex recoloring in which at least one vertex of every color survives

Retains-All (Partial) Convex Recoloring

Definition

A Retains-All Recoloring is a convex recoloring in which at least one vertex of every color survives

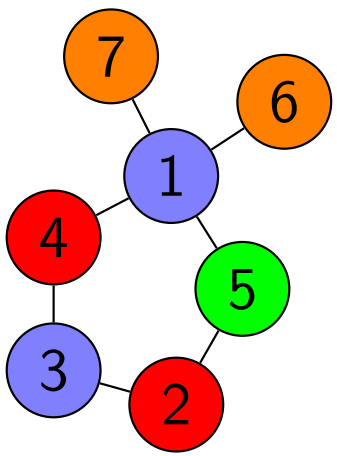


Input

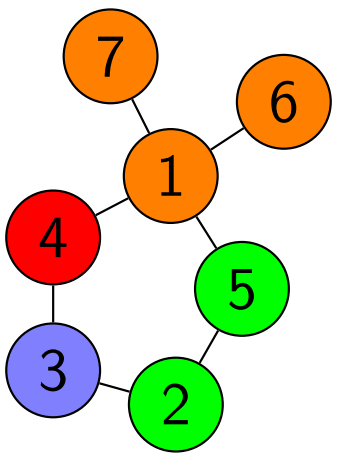
Retains-All (Partial) Convex Recoloring

Definition

A Retains-All Recoloring is a convex recoloring in which at least one vertex of every color survives



Input

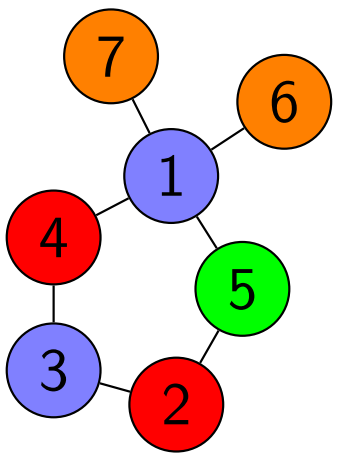


Retains All

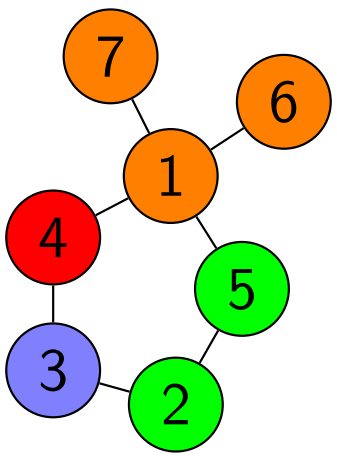
Retains-All (Partial) Convex Recoloring

Definition

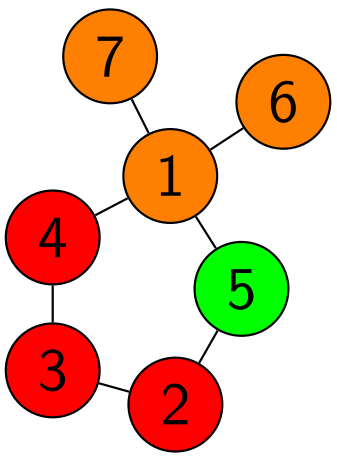
A Retains-All Recoloring is a convex recoloring in which at least one vertex of every color survives



Input



Retains All



Not Retains All

Optimal Retains-All Recoloring

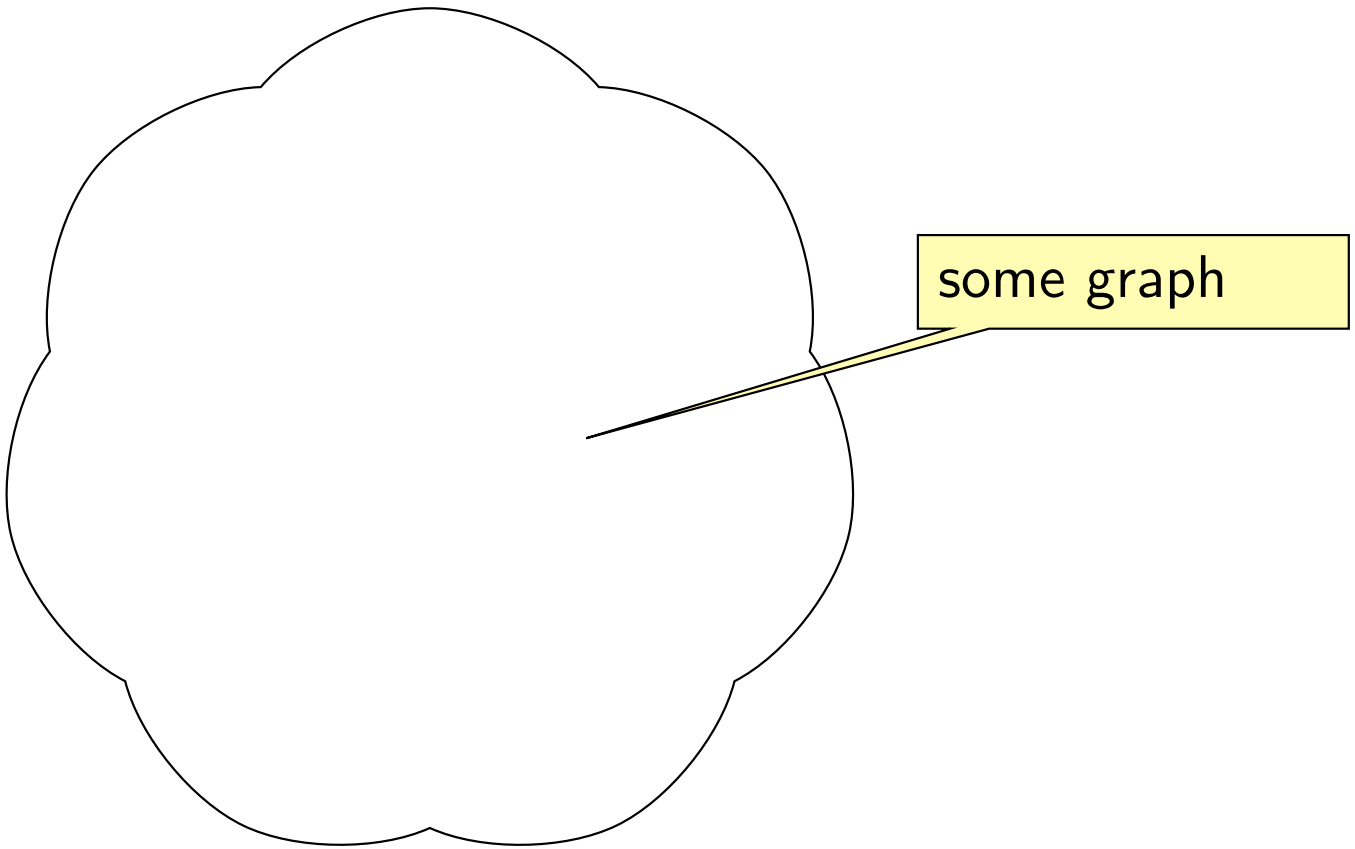
Lemma

There is an optimal retains-all recoloring

Optimal Retains-All Recoloring

Lemma

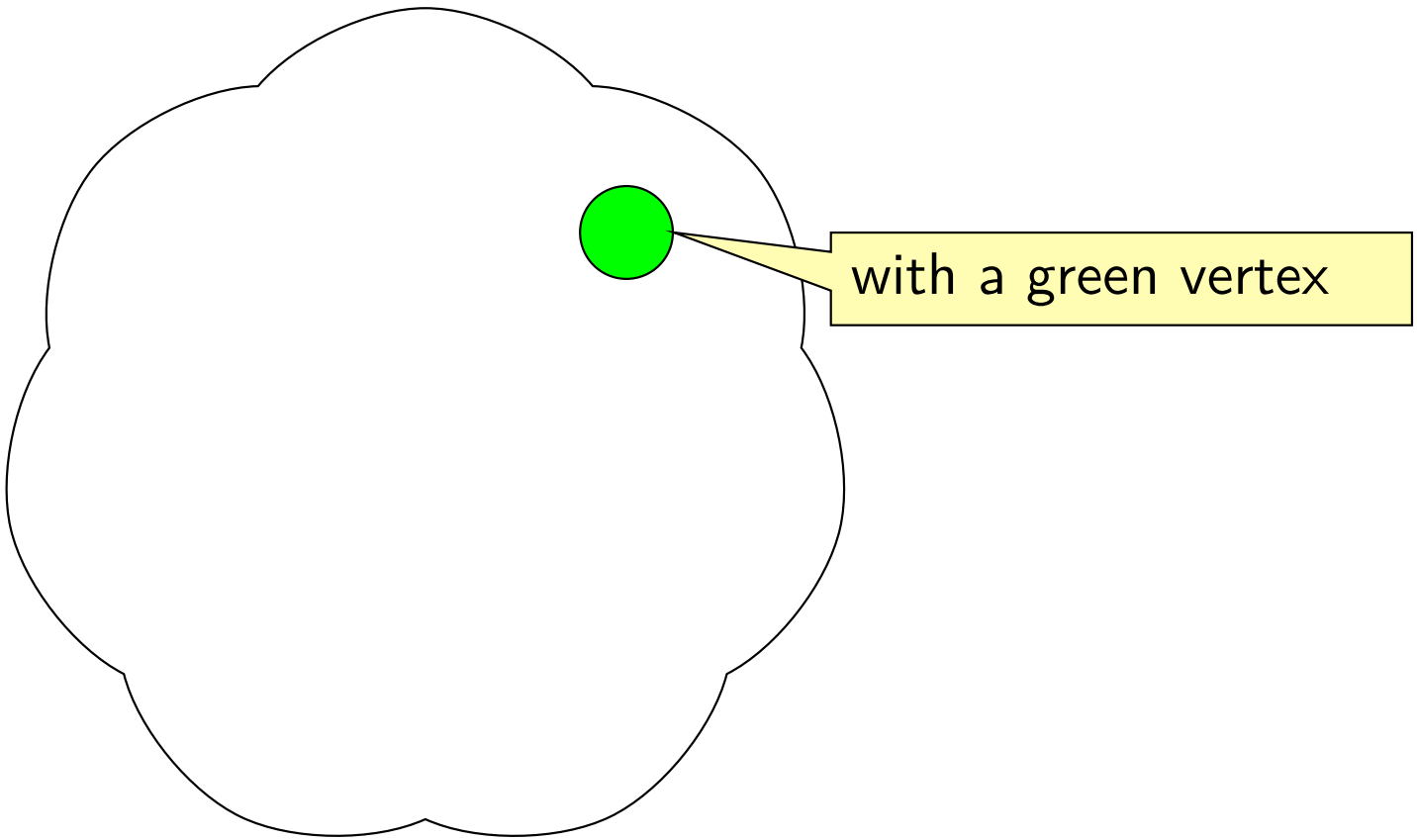
There is an optimal retains-all recoloring



Optimal Retains-All Recoloring

Lemma

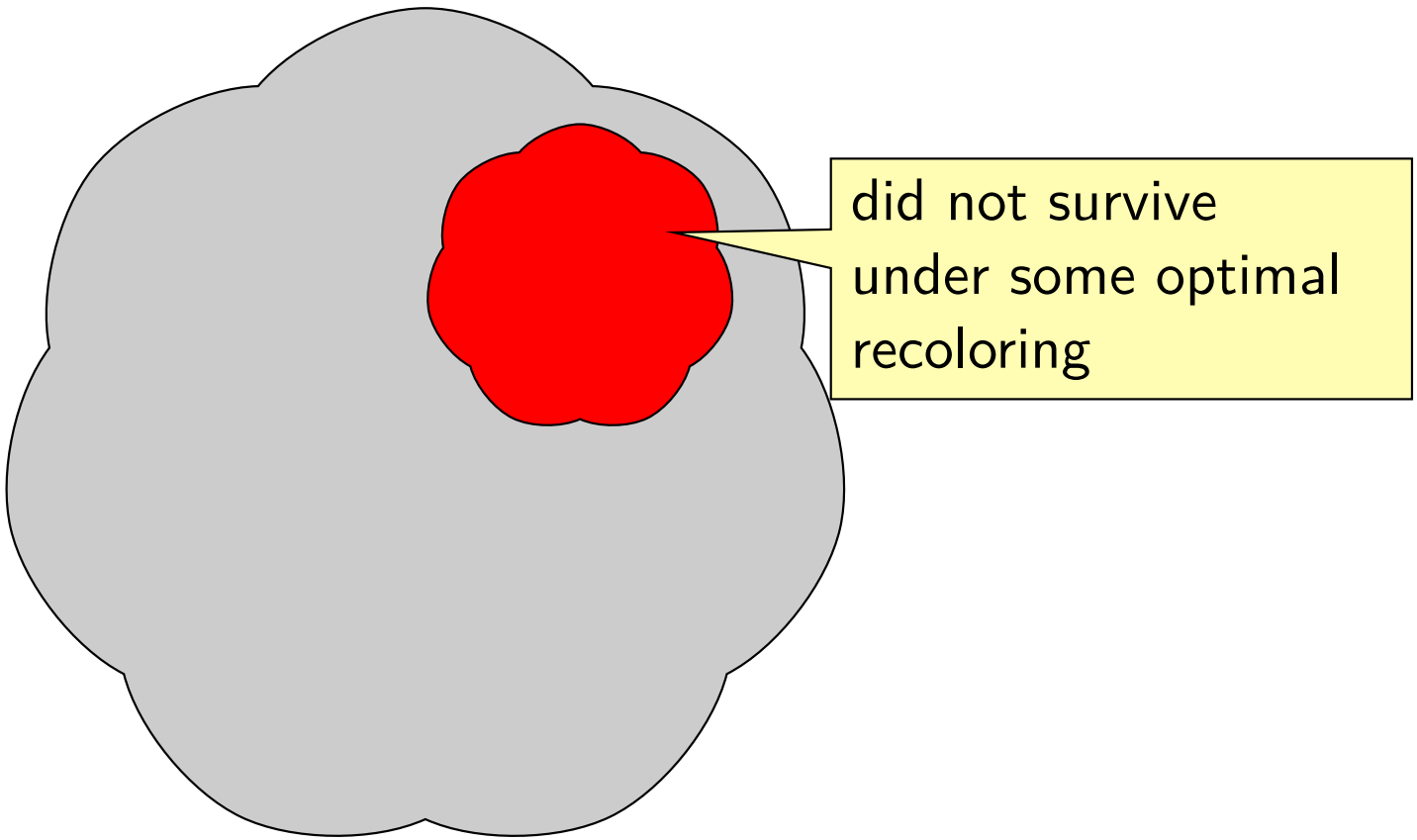
There is an optimal retains-all recoloring



Optimal Retains-All Recoloring

Lemma

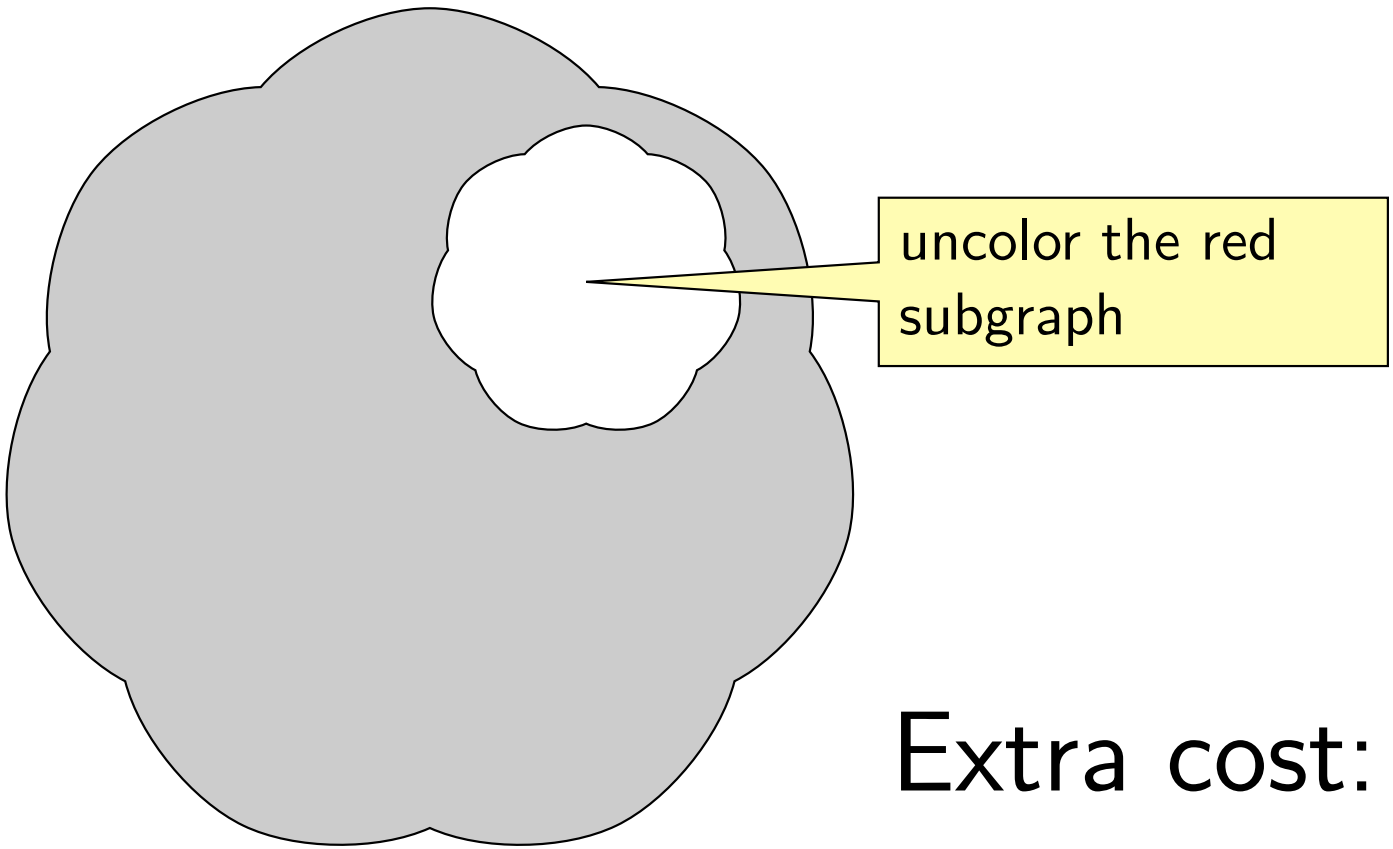
There is an optimal retains-all recoloring



Optimal Retains-All Recoloring

Lemma

There is an optimal retains-all recoloring

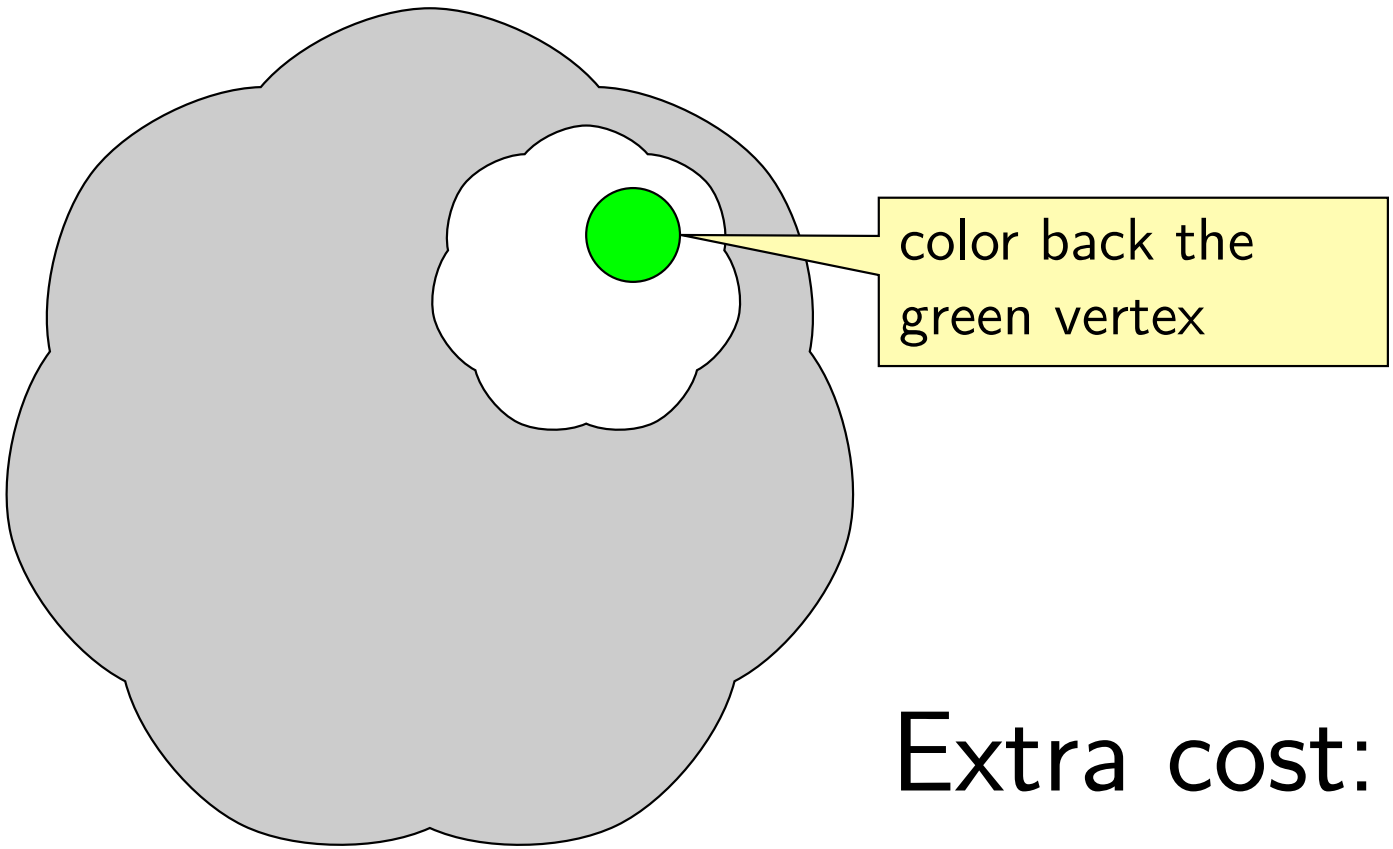


Extra cost: 2$

Optimal Retains-All Recoloring

Lemma

There is an optimal retains-all recoloring

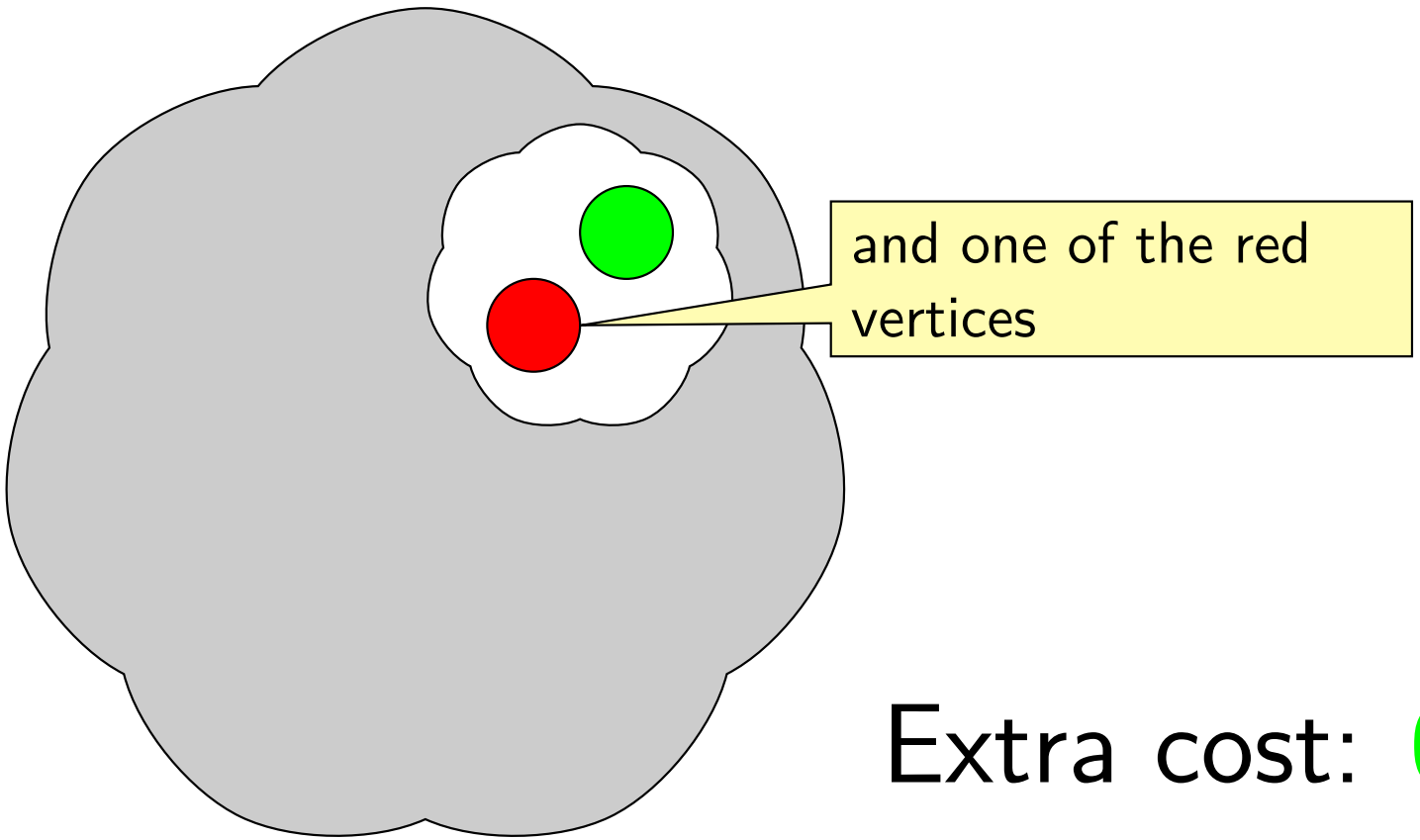


Extra cost: 1$

Optimal Retains-All Recoloring

Lemma

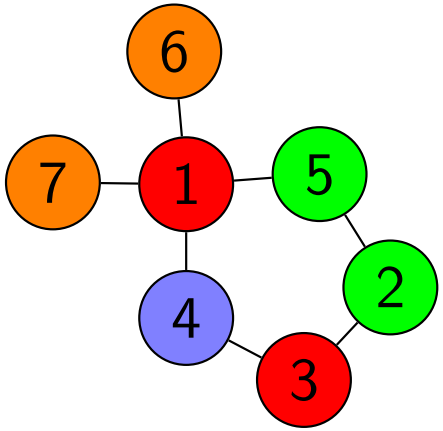
There is an optimal retains-all recoloring



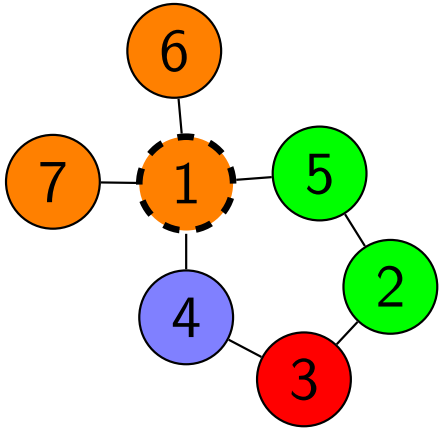
Extra cost: 0$

The Cost of Retains-All Recoloring

Input:



Input

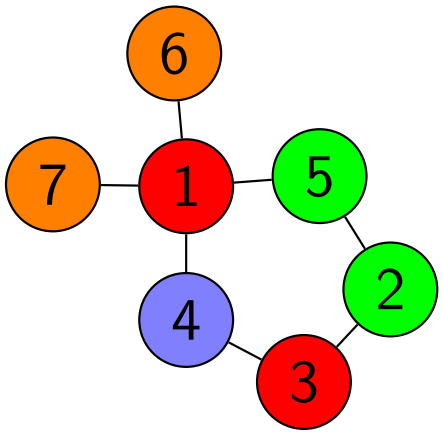


Output

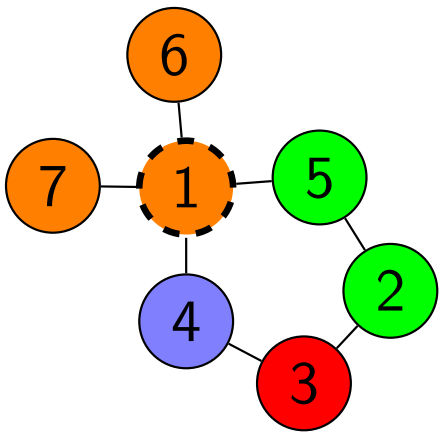
The Cost of Retains-All Recoloring

Input:

- Pair: {1, 3}



Input

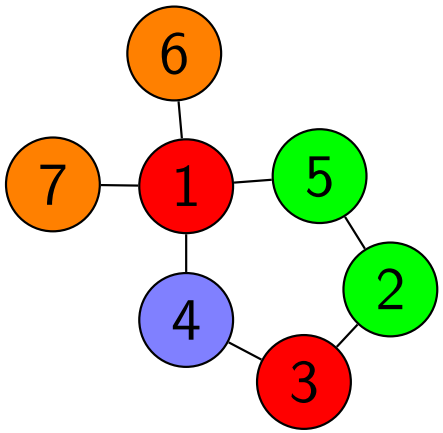


Output

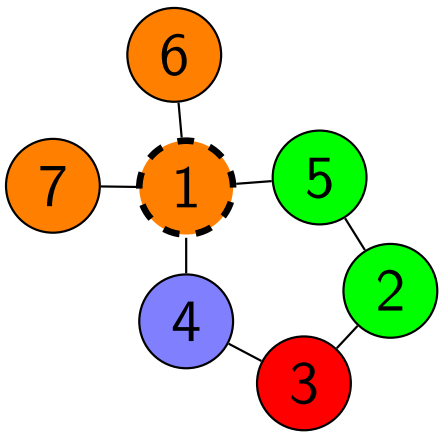
The Cost of Retains-All Recoloring

Input:

- Pair: {1, 3}
- Singleton: 4



Input

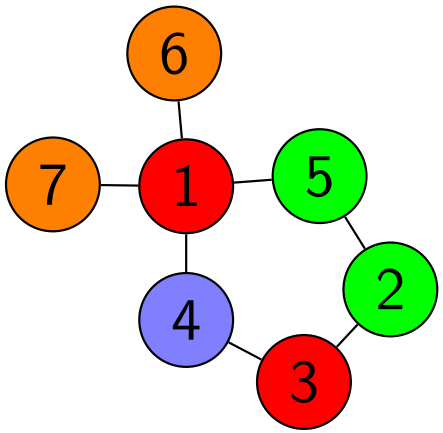


Output

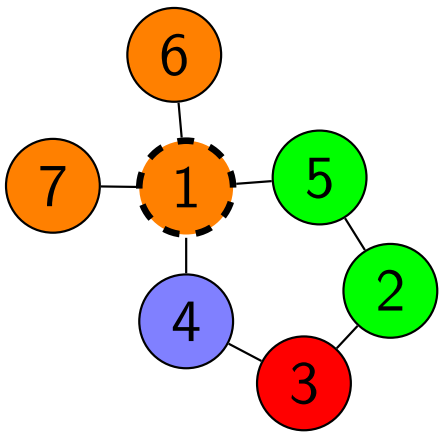
The Cost of Retains-All Recoloring

Input:

- Pair: {1, 3}
- Singleton: 4
- P:{{1, 3}, {2, 5}, {6, 7}}



Input



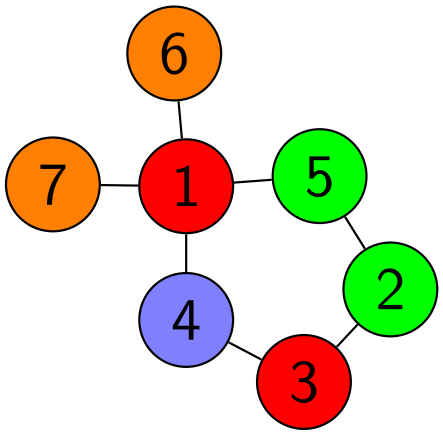
Output

The Cost of Retains-All Recoloring

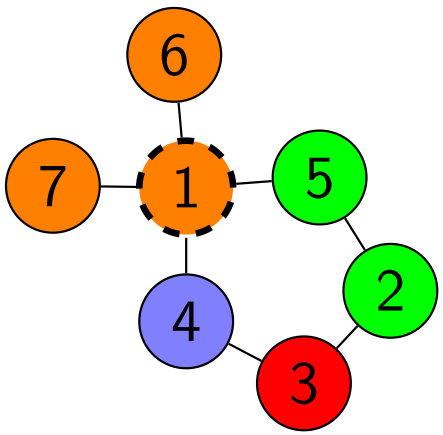
Input:

- Pair: {1, 3}
- Singleton: 4
- P:{{1, 3}, {2, 5}, {6, 7}}

Output:



Input



Output

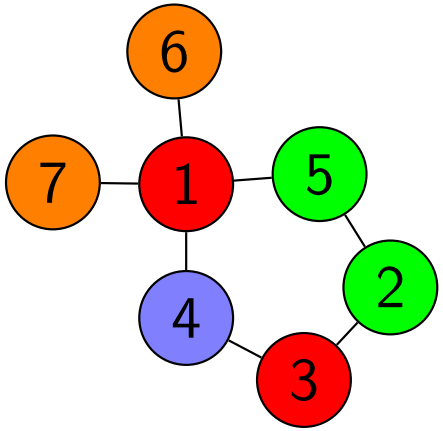
The Cost of Retains-All Recoloring

Input:

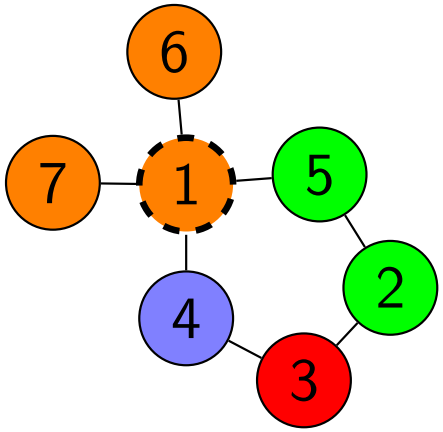
- Pair: {1, 3}
- Singleton: 4
- P:{{1, 3}, {2, 5}, {6, 7}}

Output:

- {6, 7} survives



Input



Output

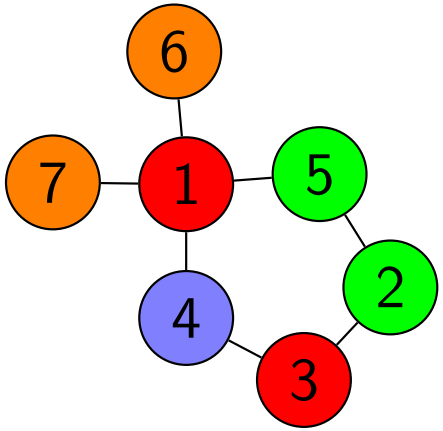
The Cost of Retains-All Recoloring

Input:

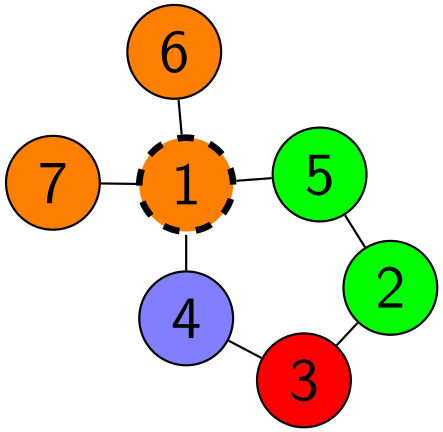
- Pair: {1, 3}
- Singleton: 4
- P:{{1, 3}, {2, 5}, {6, 7}}

Output:

- {6, 7} survives
- S: {{2, 5}, {6, 7}}



Input



Output

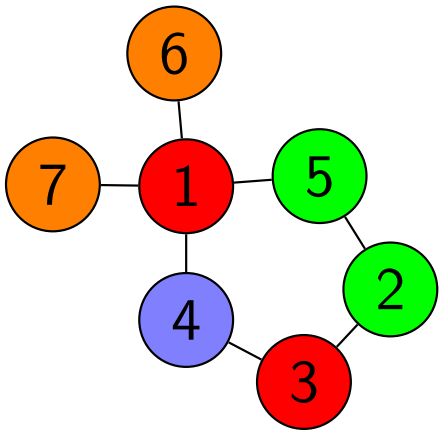
The Cost of Retains-All Recoloring

Input:

- Pair: {1, 3}
- Singleton: 4
- P:{{1, 3}, {2, 5}, {6, 7}}

Output:

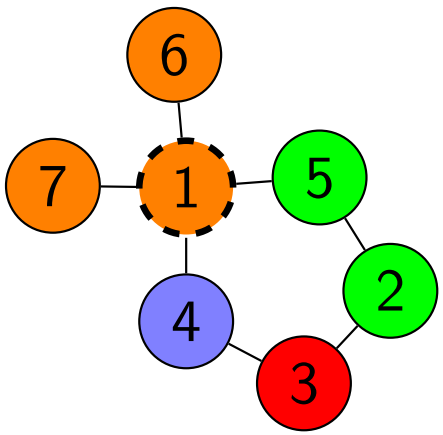
- {6, 7} survives
- S: {{2, 5}, {6, 7}}



Input

Observation

The cost is $|P| - |S|$



Output

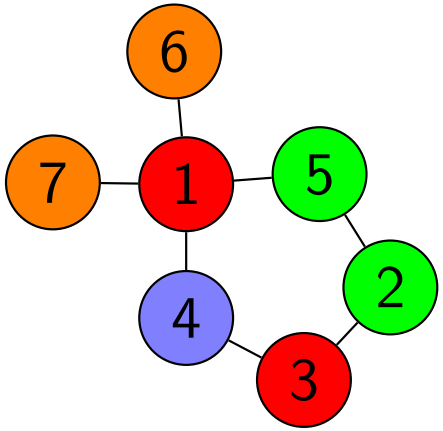
The Cost of Retains-All Recoloring

Input:

- Pair: {1, 3}
- Singleton: 4
- P:{{1, 3}, {2, 5}, {6, 7}}

Output:

- {6, 7} survives
- S: {{2, 5}, {6, 7}}

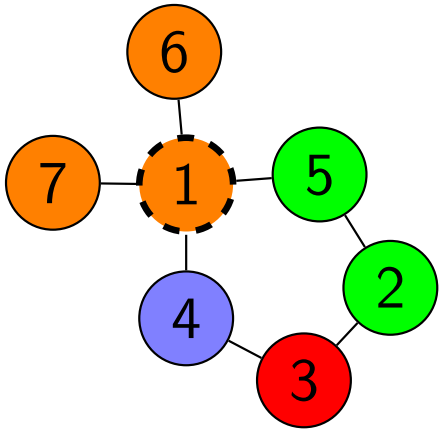


Input

Observation

The cost is $|P| - |S|$

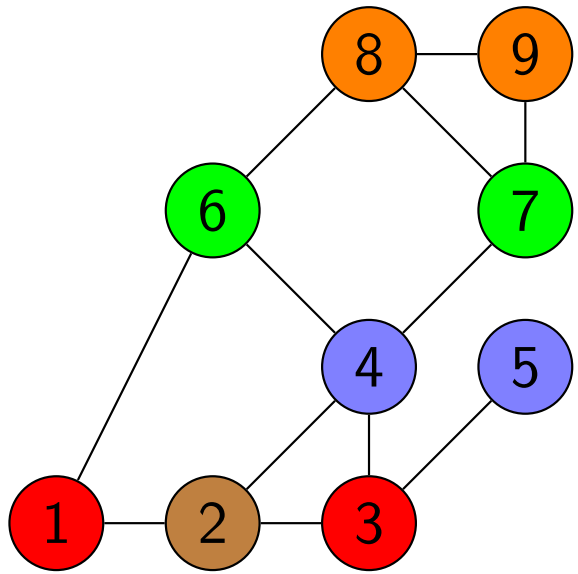
Goal: maximize $|S|$



Output

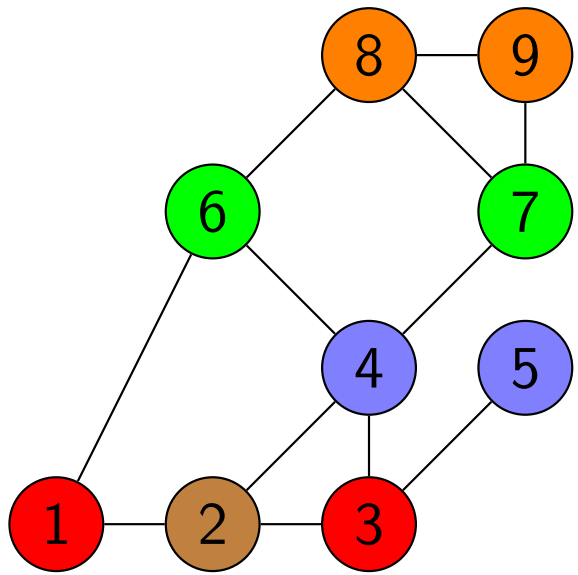
Pair Packing

- Component



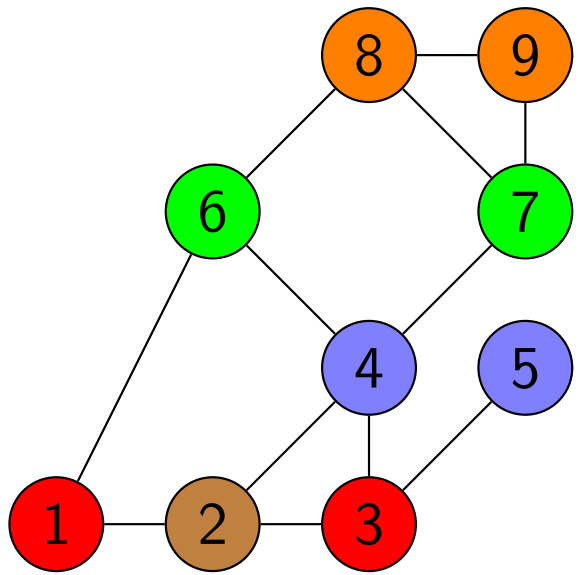
Pair Packing

- Component
  - **connected** subgraph



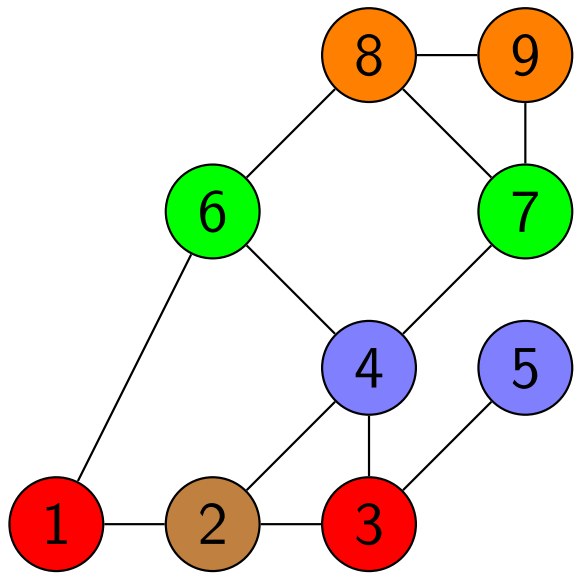
Pair Packing

- Component
  - **connected** subgraph
  - exactly **one** pair



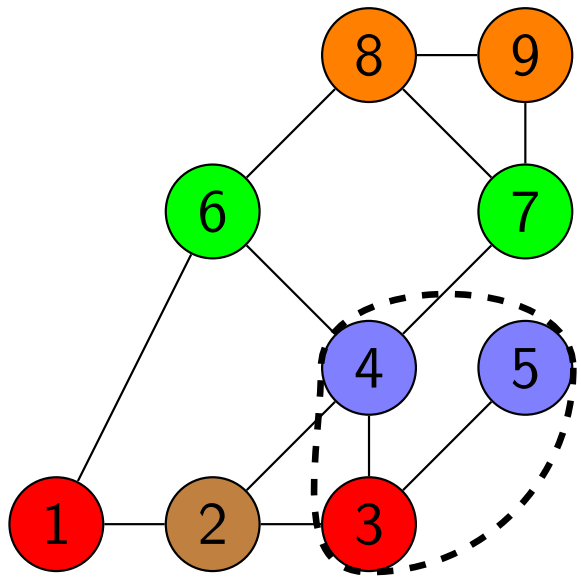
Pair Packing

- Component
  - **connected** subgraph
  - exactly **one** pair
  - no singleton



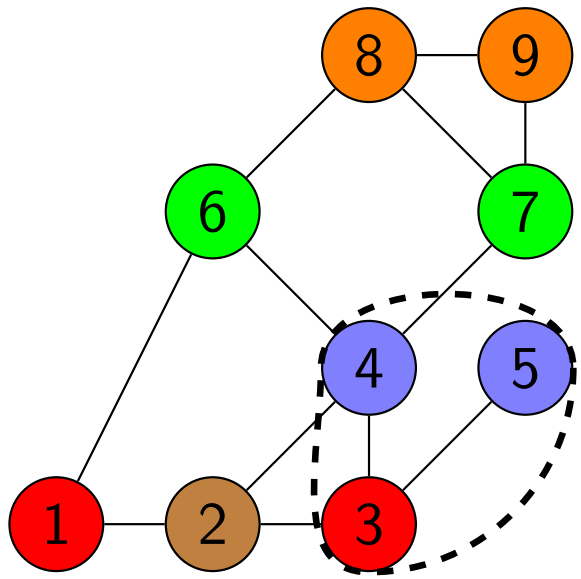
Pair Packing

- Component
  - **connected** subgraph
  - exactly **one** pair
  - no singleton



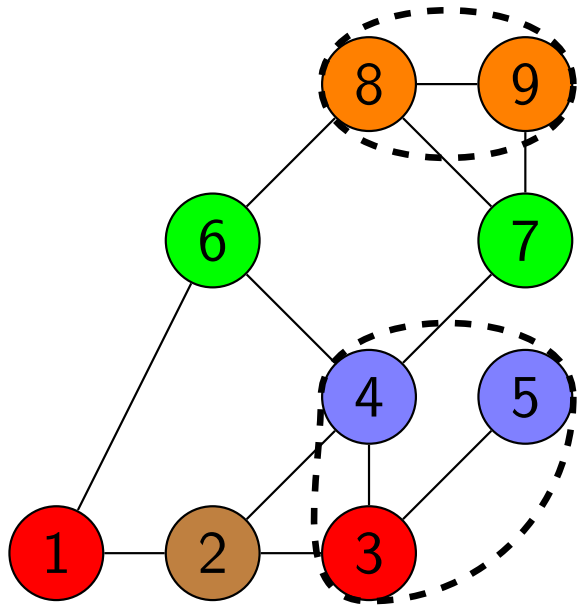
Pair Packing

- Component
  - **connected** subgraph
  - exactly **one** pair
  - no singleton
- Packing: **color disjoint** components



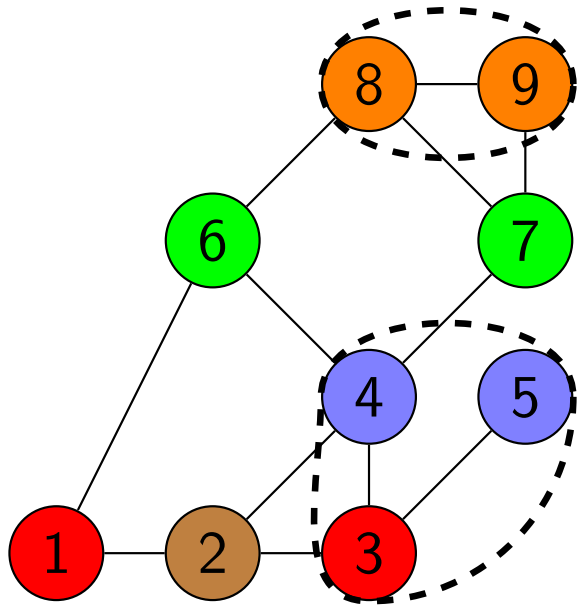
Pair Packing

- Component
  - **connected** subgraph
  - exactly **one** pair
  - no singleton
- Packing: **color disjoint** components



Pair Packing

- Component
  - **connected** subgraph
  - exactly **one** pair
  - no singleton
- Packing: **color disjoint** components
- **Goal**: maximum packing



Pair Packing and 2-CR

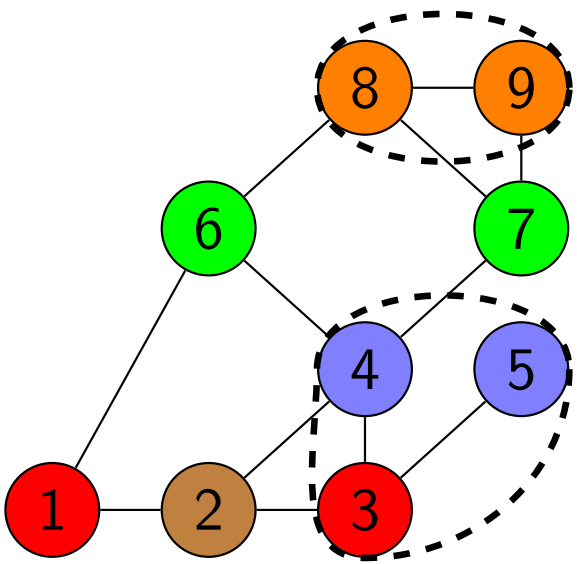
Observation

*A **packing** $\mathcal{I}$ can be transformed to a retains-all **recoloring** that costs $|P| - |\mathcal{I}|$ (and vice-versa)*

Pair Packing and 2-CR

Observation

*A **packing** $\mathcal{I}$ can be transformed to a retains-all **recoloring** that costs $|P| - |\mathcal{I}|$ (and vice-versa)*

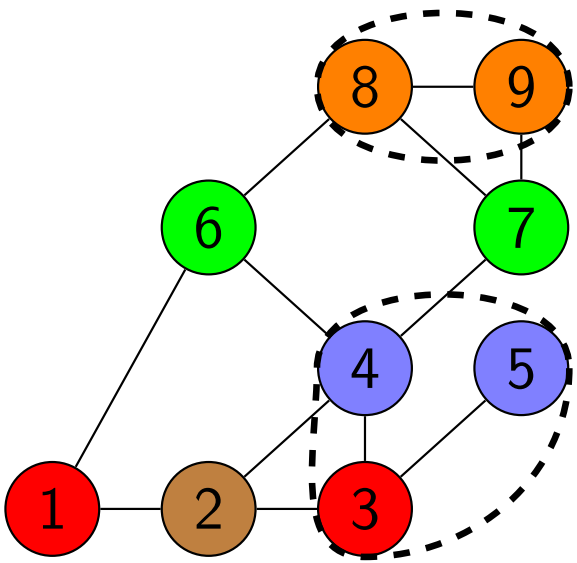


Pair Packing

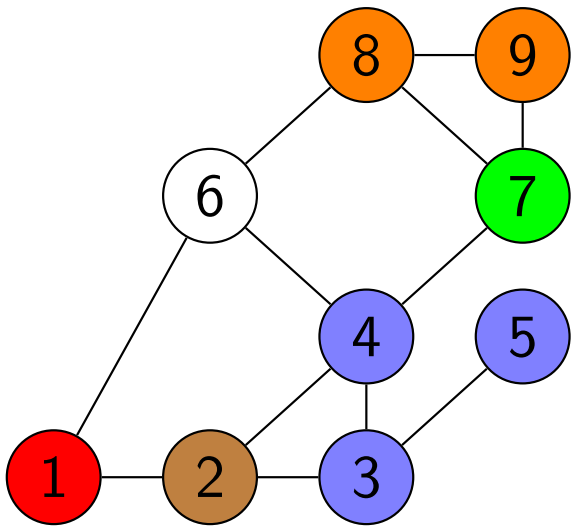
Pair Packing and 2-CR

Observation

*A **packing** $\mathcal{I}$ can be transformed to a retains-all **recoloring** that costs $|P| - |\mathcal{I}|$ (and vice-versa)*



Pair Packing

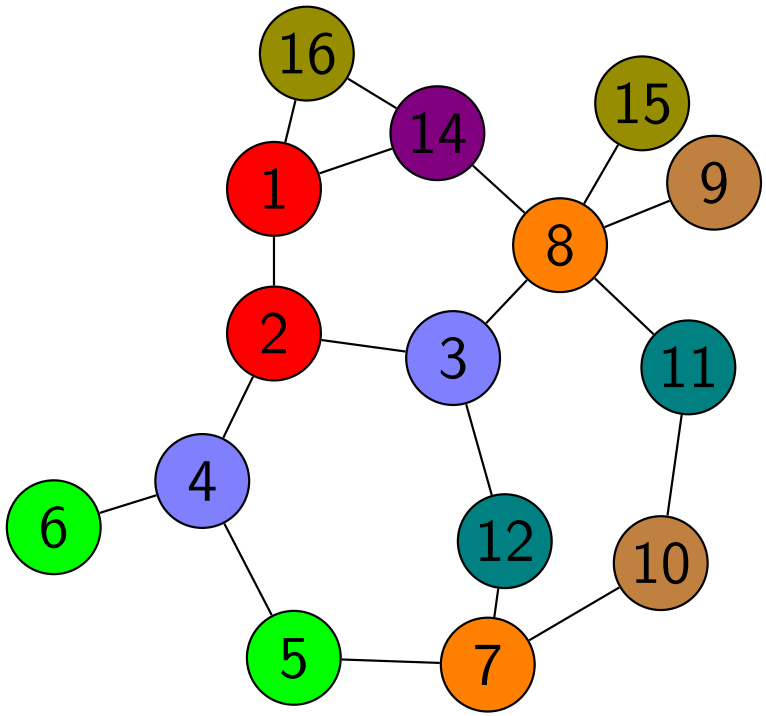


Convex Recoloring

Greedy Algorithm for Pair Packing

Algorithm 1 Greedy Algorithm

1: $\mathcal{I} \leftarrow \emptyset$
2: **while** $\mathcal{I}$ is not maximal **do**
3: $\quad C_{min} \leftarrow \arg\min_{C:\mathcal{I}\cup\{C\} \text{ is a packing}} |C|$
4: $\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{C_{min}\}$
5: **end while**
6: **return** $\mathcal{I}$

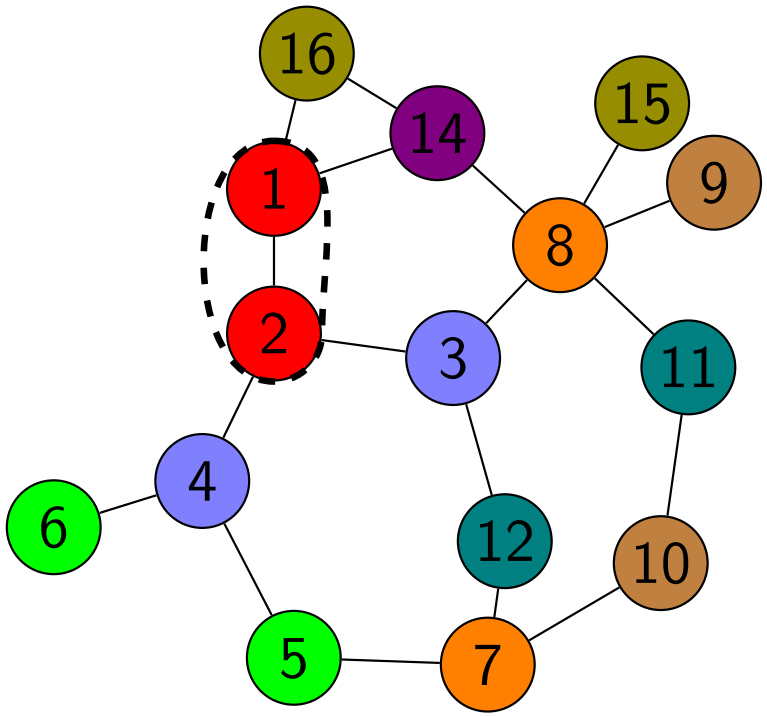


* $|C|$ is the number of vertices in C

Greedy Algorithm for Pair Packing

Algorithm 1 Greedy Algorithm

1: $\mathcal{I} \leftarrow \emptyset$
2: **while** $\mathcal{I}$ is not maximal **do**
3: $\quad C_{min} \leftarrow \arg\min_{C:\mathcal{I}\cup\{C\} \text{ is a packing}} |C|$
4: $\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{C_{min}\}$
5: **end while**
6: **return** $\mathcal{I}$

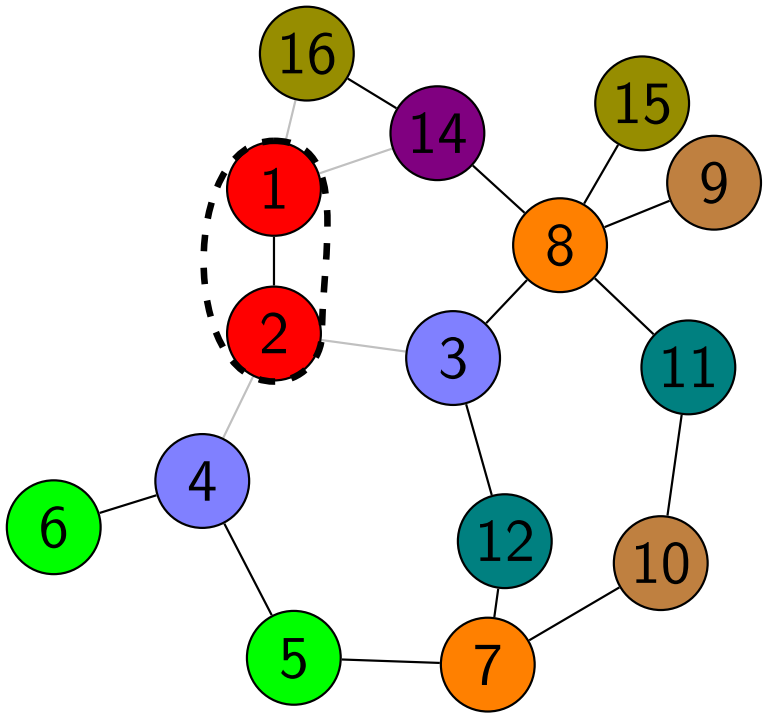


* $|C|$ is the number of vertices in C

Greedy Algorithm for Pair Packing

Algorithm 1 Greedy Algorithm

1: $\mathcal{I} \leftarrow \emptyset$
2: **while** $\mathcal{I}$ is not maximal **do**
3: $\quad C_{min} \leftarrow \arg\min_{C:\mathcal{I}\cup\{C\} \text{ is a packing}} |C|$
4: $\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{C_{min}\}$
5: **end while**
6: **return** $\mathcal{I}$

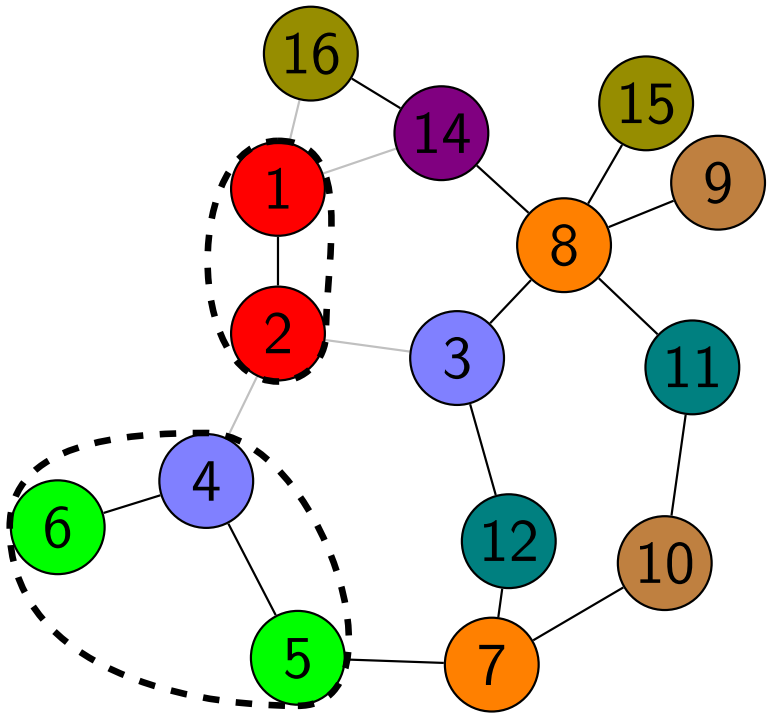


* $|C|$ is the number of vertices in C

Greedy Algorithm for Pair Packing

Algorithm 1 Greedy Algorithm

1: $\mathcal{I} \leftarrow \emptyset$
2: **while** $\mathcal{I}$ is not maximal **do**
3: $\quad C_{min} \leftarrow \arg\min_{C:\mathcal{I}\cup\{C\} \text{ is a packing}} |C|$
4: $\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{C_{min}\}$
5: **end while**
6: **return** $\mathcal{I}$

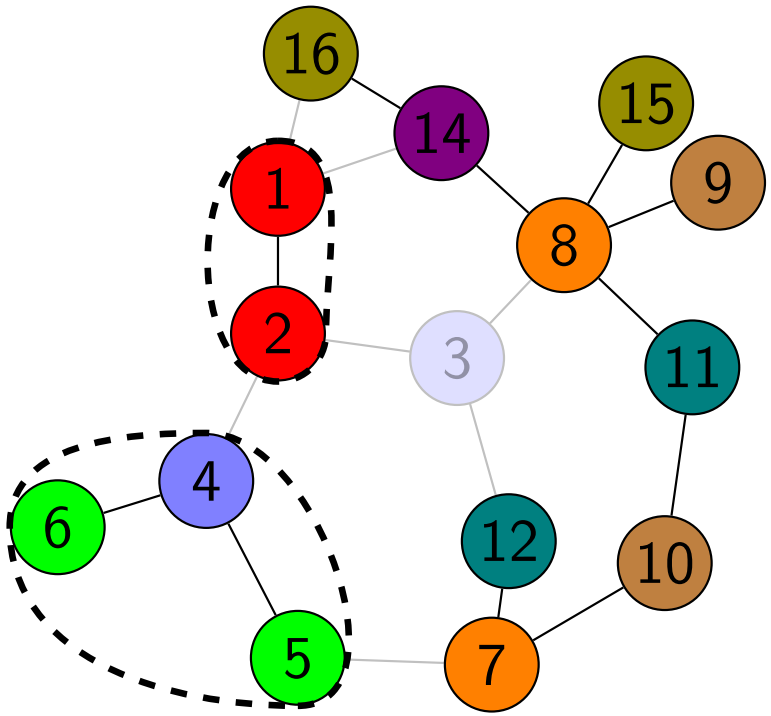


* $|C|$ is the number of vertices in C

Greedy Algorithm for Pair Packing

Algorithm 1 Greedy Algorithm

1: $\mathcal{I} \leftarrow \emptyset$
2: **while** $\mathcal{I}$ is not maximal **do**
3: $\quad C_{min} \leftarrow \arg\min_{C:\mathcal{I}\cup\{C\} \text{ is a packing}} |C|$
4: $\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{C_{min}\}$
5: **end while**
6: **return** $\mathcal{I}$

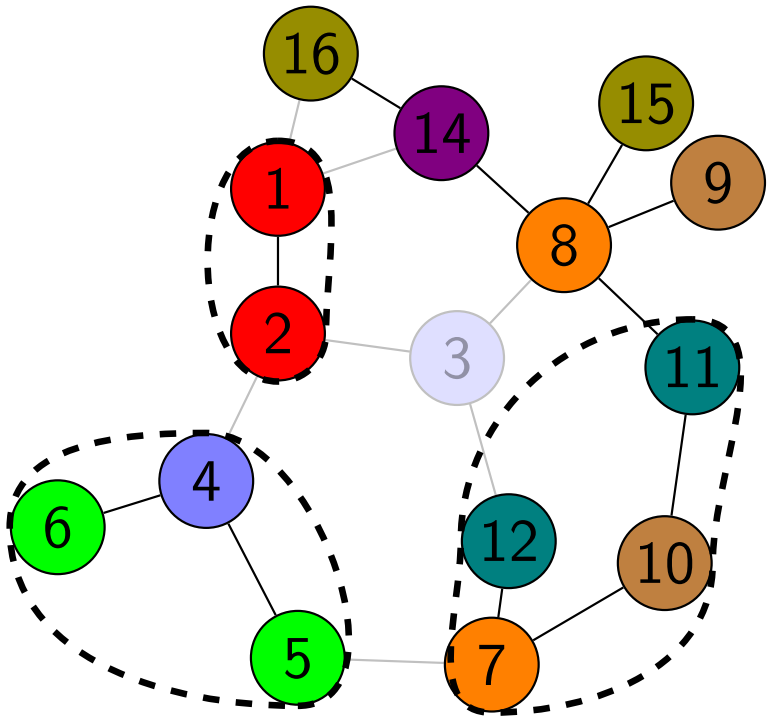


* $|C|$ is the number of vertices in C

Greedy Algorithm for Pair Packing

Algorithm 1 Greedy Algorithm

1: $\mathcal{I} \leftarrow \emptyset$
2: **while** $\mathcal{I}$ is not maximal **do**
3: $\quad C_{min} \leftarrow \arg\min_{C:\mathcal{I}\cup\{C\} \text{ is a packing}} |C|$
4: $\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{C_{min}\}$
5: **end while**
6: **return** $\mathcal{I}$

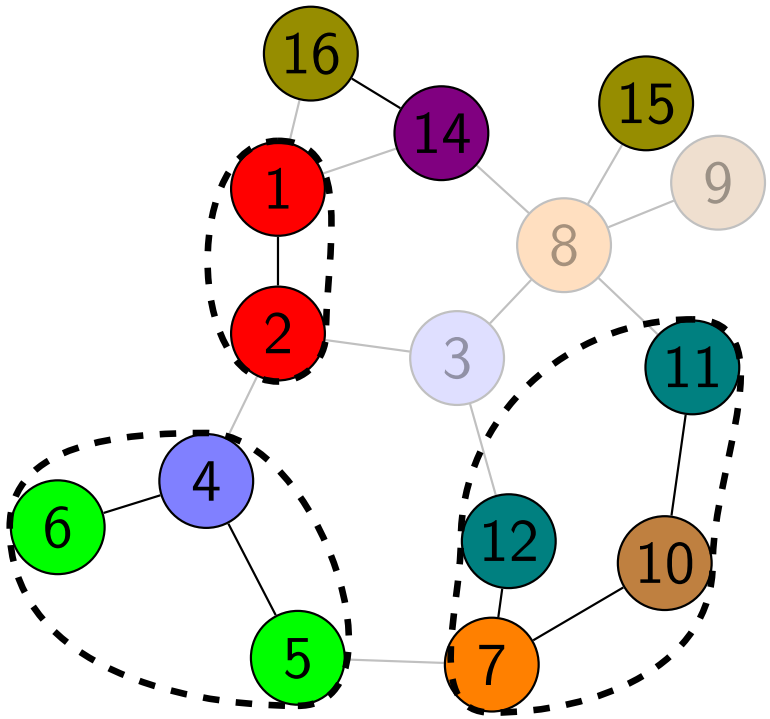


* $|C|$ is the number of vertices in C

Greedy Algorithm for Pair Packing

Algorithm 1 Greedy Algorithm

1: $\mathcal{I} \leftarrow \emptyset$
2: **while** $\mathcal{I}$ is not maximal **do**
3: $\quad C_{min} \leftarrow \arg\min_{C:\mathcal{I}\cup\{C\} \text{ is a packing}} |C|$
4: $\quad \mathcal{I} \leftarrow \mathcal{I} \cup \{C_{min}\}$
5: **end while**
6: **return** $\mathcal{I}$



* $|C|$ is the number of vertices in C

Greedy vs Optimal

Notation:

Greedy vs Optimal

Notation:

- $\mathcal{I}$: greedy packing

Greedy vs Optimal

Notation:

- $\mathcal{I}$: greedy packing
- $\mathcal{I}^*$: optimal packing

Greedy vs Optimal

Notation:

- $\mathcal{I}$: greedy packing
- $\mathcal{I}^*$: optimal packing
- α: $\frac{|\mathcal{I}^*|}{|\mathcal{I}|}$

Greedy vs Optimal

Notation:

- $\mathcal{I}$: greedy packing
- $\mathcal{I}^*$: optimal packing
- α: $\frac{|\mathcal{I}^*|}{|\mathcal{I}|}$

Observation

The approximation ratio for 2-CR is

$$r := \frac{|P| - |\mathcal{I}|}{|P| - |\mathcal{I}^*|} = \frac{\alpha|P| - |\mathcal{I}^*|}{\alpha(|P| - |\mathcal{I}^*|)}$$

Greedy vs Optimal

Notation:

- $\mathcal{I}$: greedy packing
- $\mathcal{I}^*$: optimal packing
- α: $\frac{|\mathcal{I}^*|}{|\mathcal{I}|}$

Observation

The approximation ratio for 2-CR is

$$r := \frac{|P| - |\mathcal{I}|}{|P| - |\mathcal{I}^*|} = \frac{\alpha|P| - |\mathcal{I}^*|}{\alpha(|P| - |\mathcal{I}^*|)}$$

- We want to **upper bound** r

Greedy vs Optimal

Notation:

- $\mathcal{I}$: greedy packing
- $\mathcal{I}^*$: optimal packing
- α: $\frac{|\mathcal{I}^*|}{|\mathcal{I}|}$

Observation

The approximation ratio for 2-CR is

$$r := \frac{|P| - |\mathcal{I}|}{|P| - |\mathcal{I}^*|} = \frac{\alpha|P| - |\mathcal{I}^*|}{\alpha(|P| - |\mathcal{I}^*|)}$$

- We want to **upper bound** r
- Can be done by **upper bounding** $|\mathcal{I}^*|$

First Bound

Simplification: no edge between two vertices of the same pair

First Bound

Simplification: no edge between two vertices of the same pair

Observation

$$|\mathcal{I}^*| \leq \frac{|P|}{2}$$

First Bound

Simplification: no edge between two vertices of the same pair

Observation

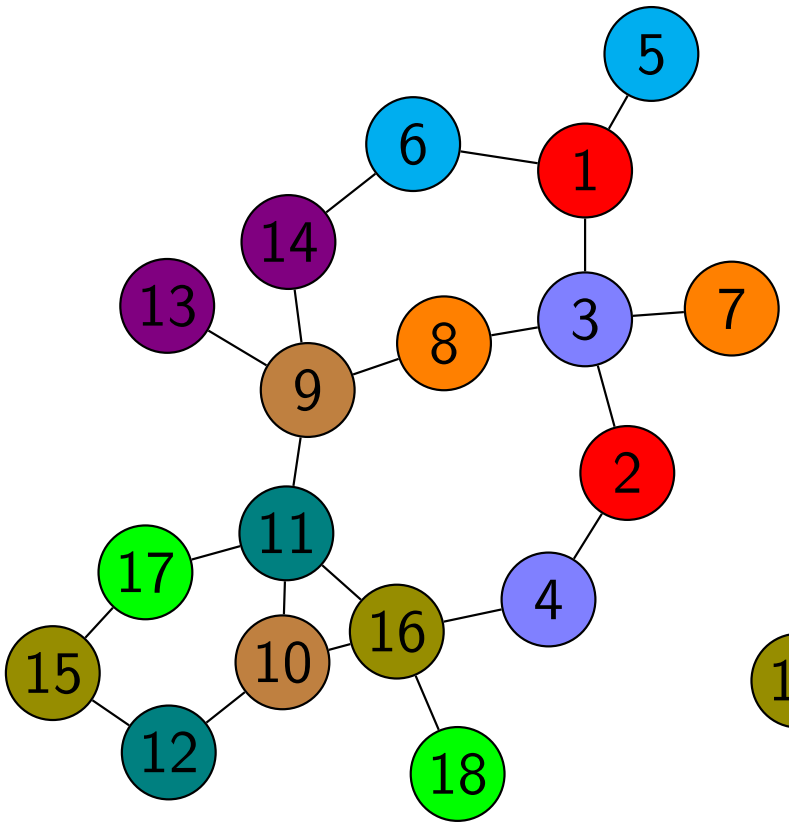
$$|\mathcal{I}^*| \leq \frac{|P|}{2}$$

Corollary

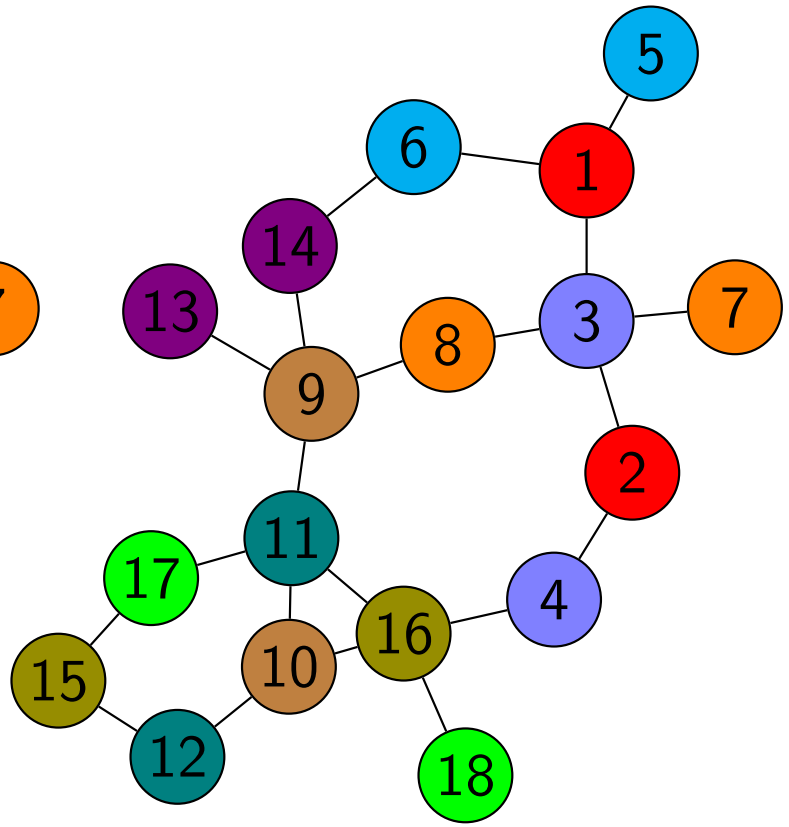
$$r := \frac{\alpha|P| - |\mathcal{I}^*|}{\alpha(|P| - |\mathcal{I}^*|)} \leq 2 - \frac{1}{\alpha} \leq 2$$

Toward a Second Bound

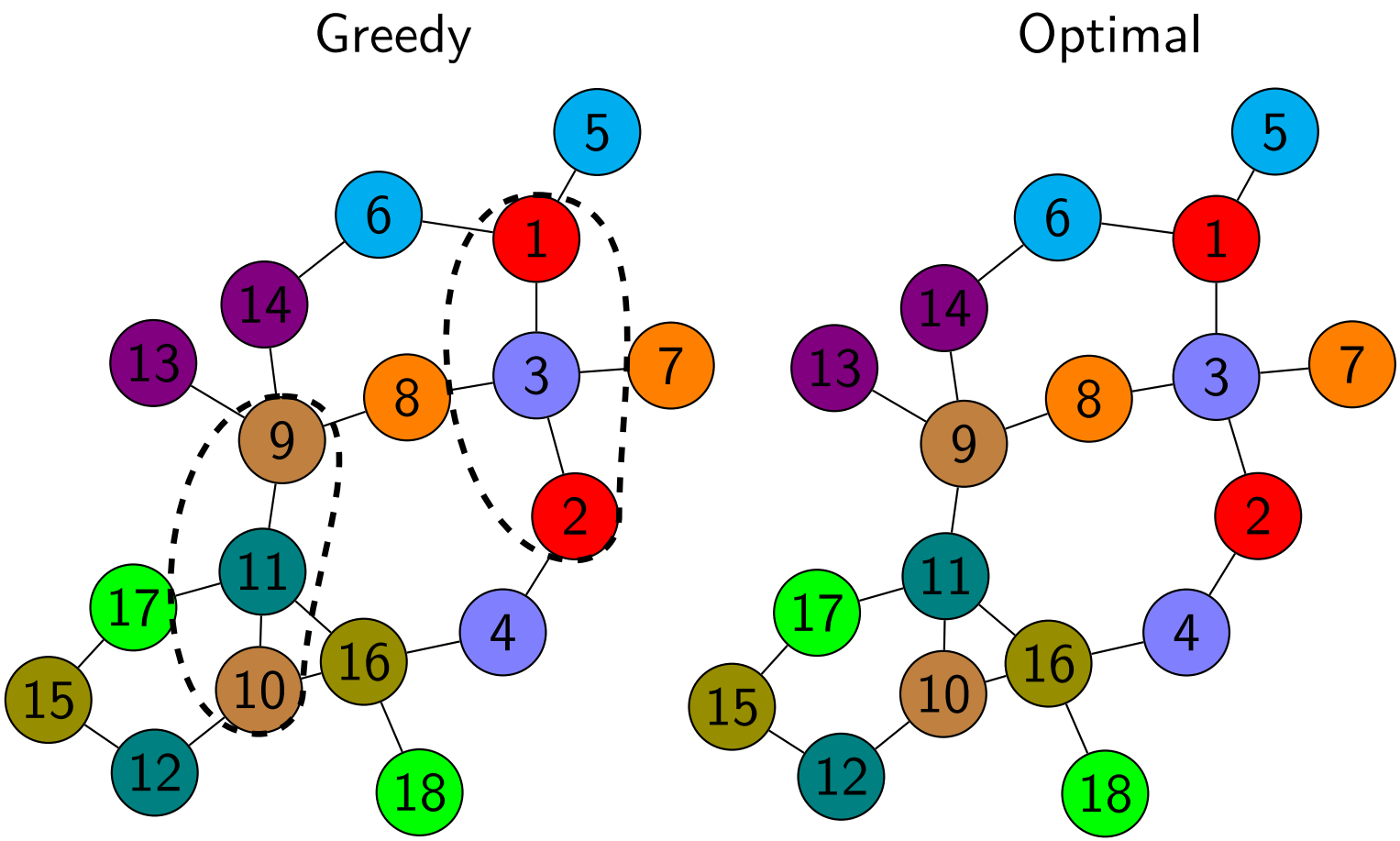
Greedy



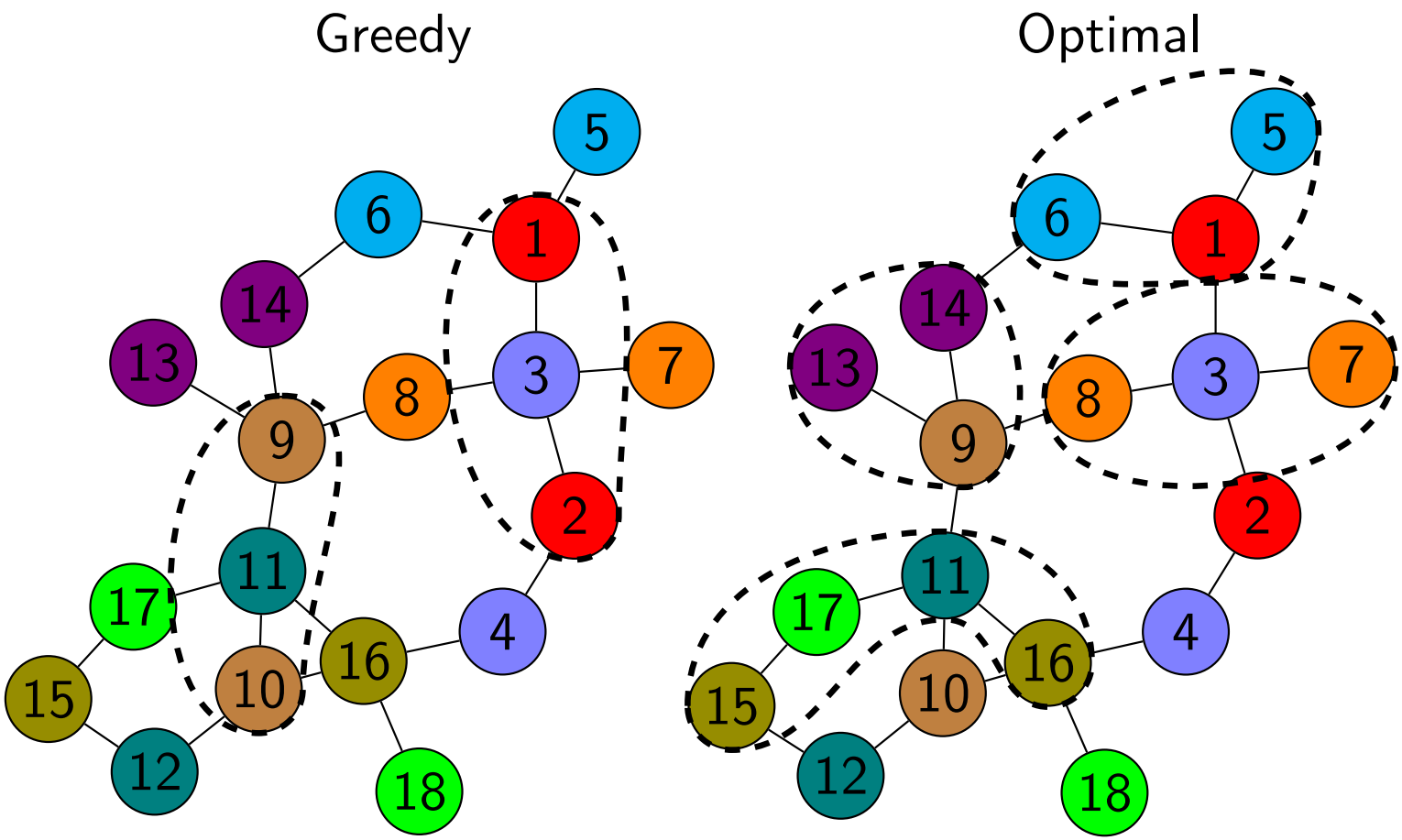
Optimal



Toward a Second Bound



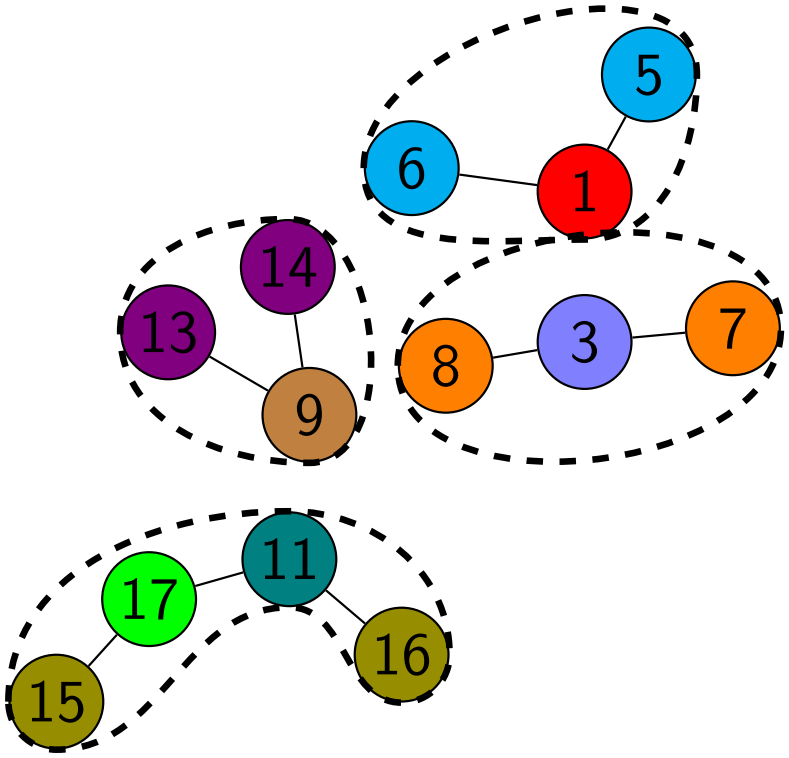
Toward a Second Bound



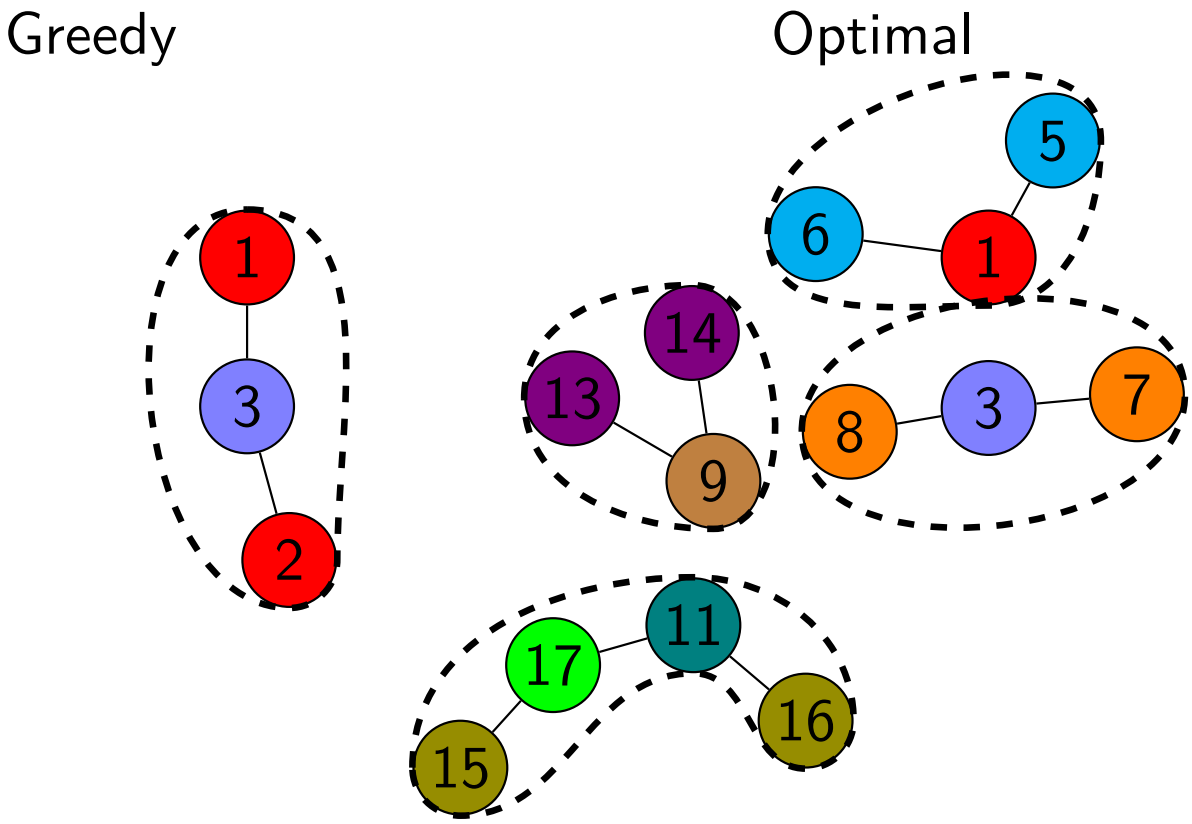
Toward a Second Bound

Greedy

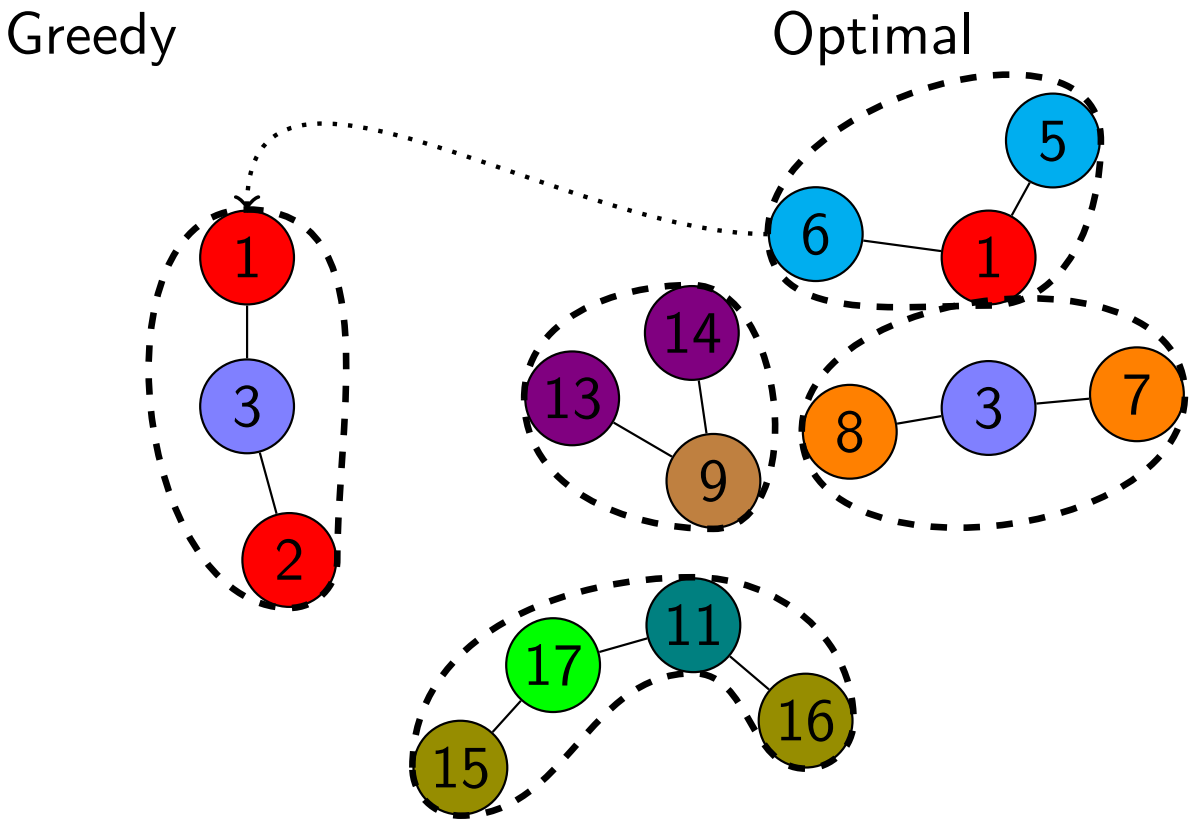
Optimal



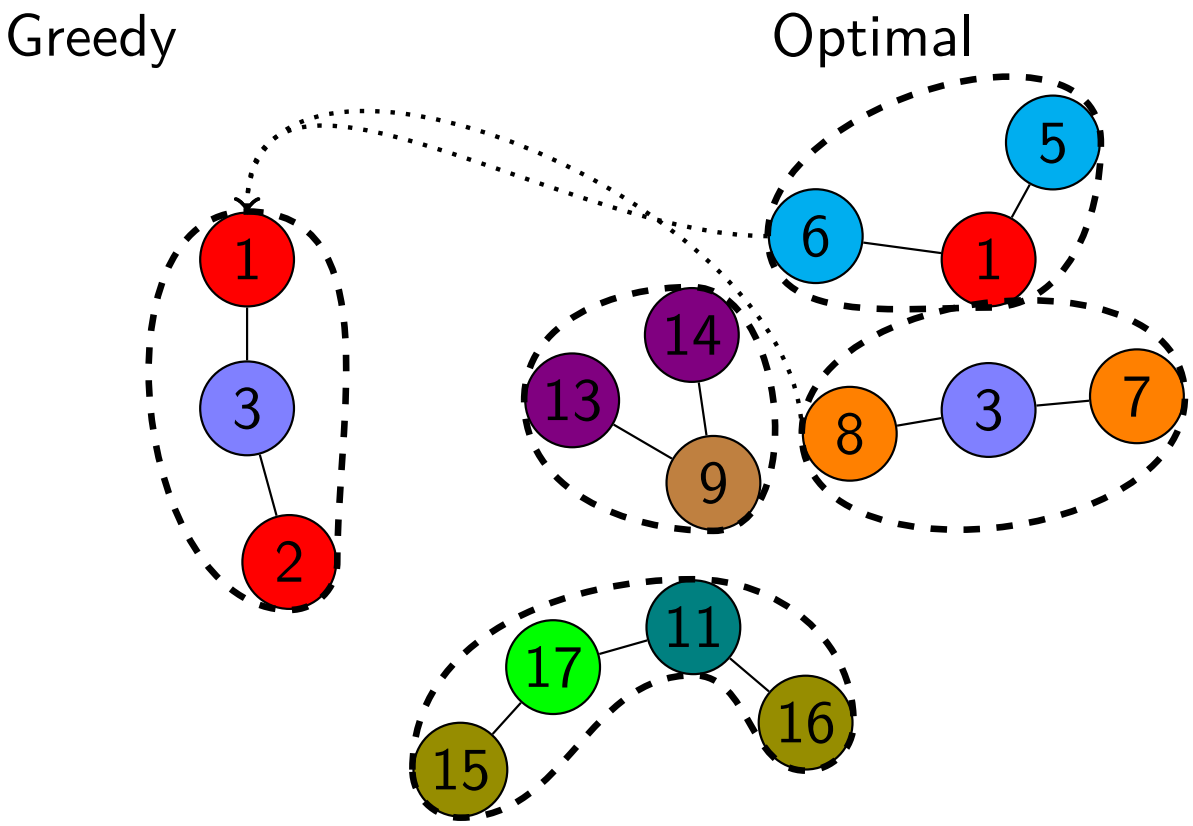
Toward a Second Bound



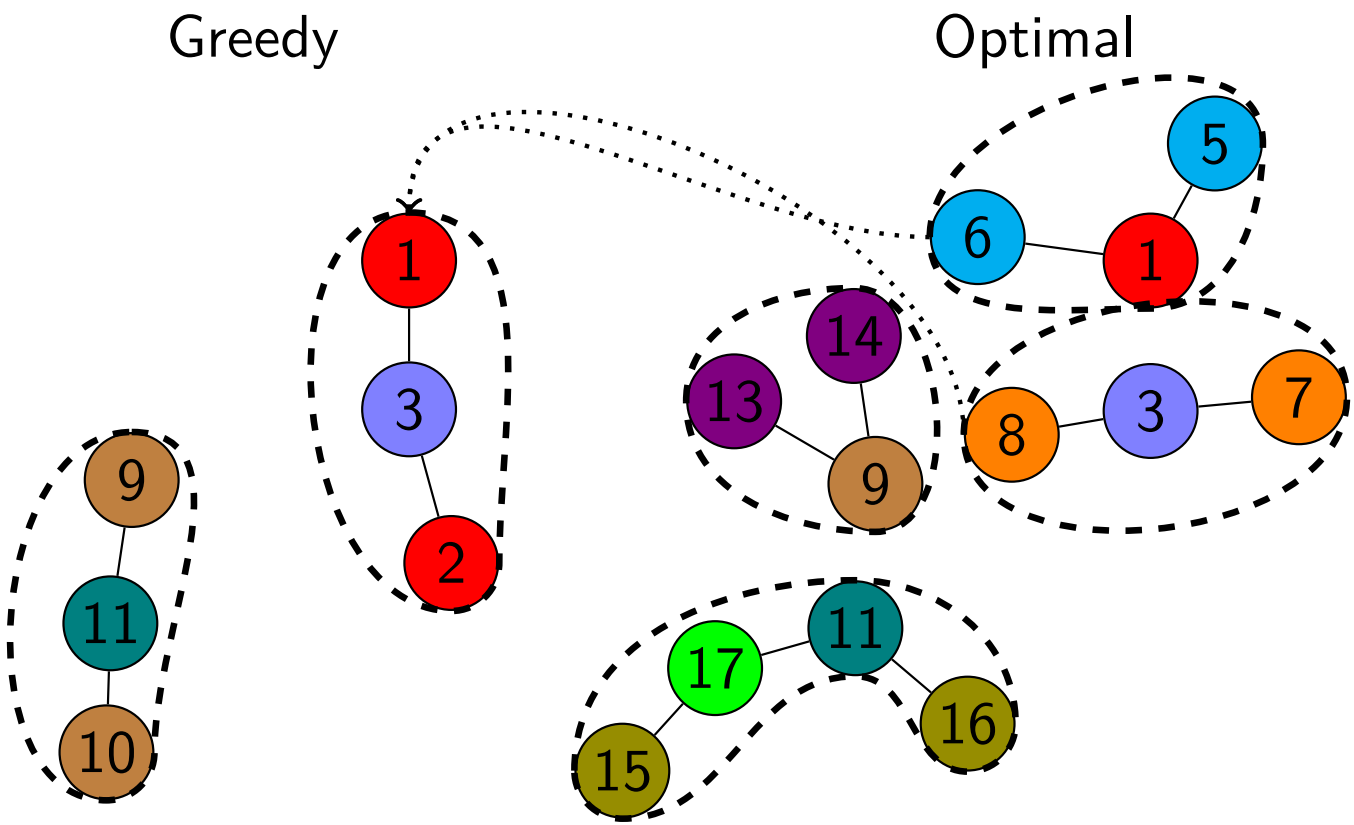
Toward a Second Bound



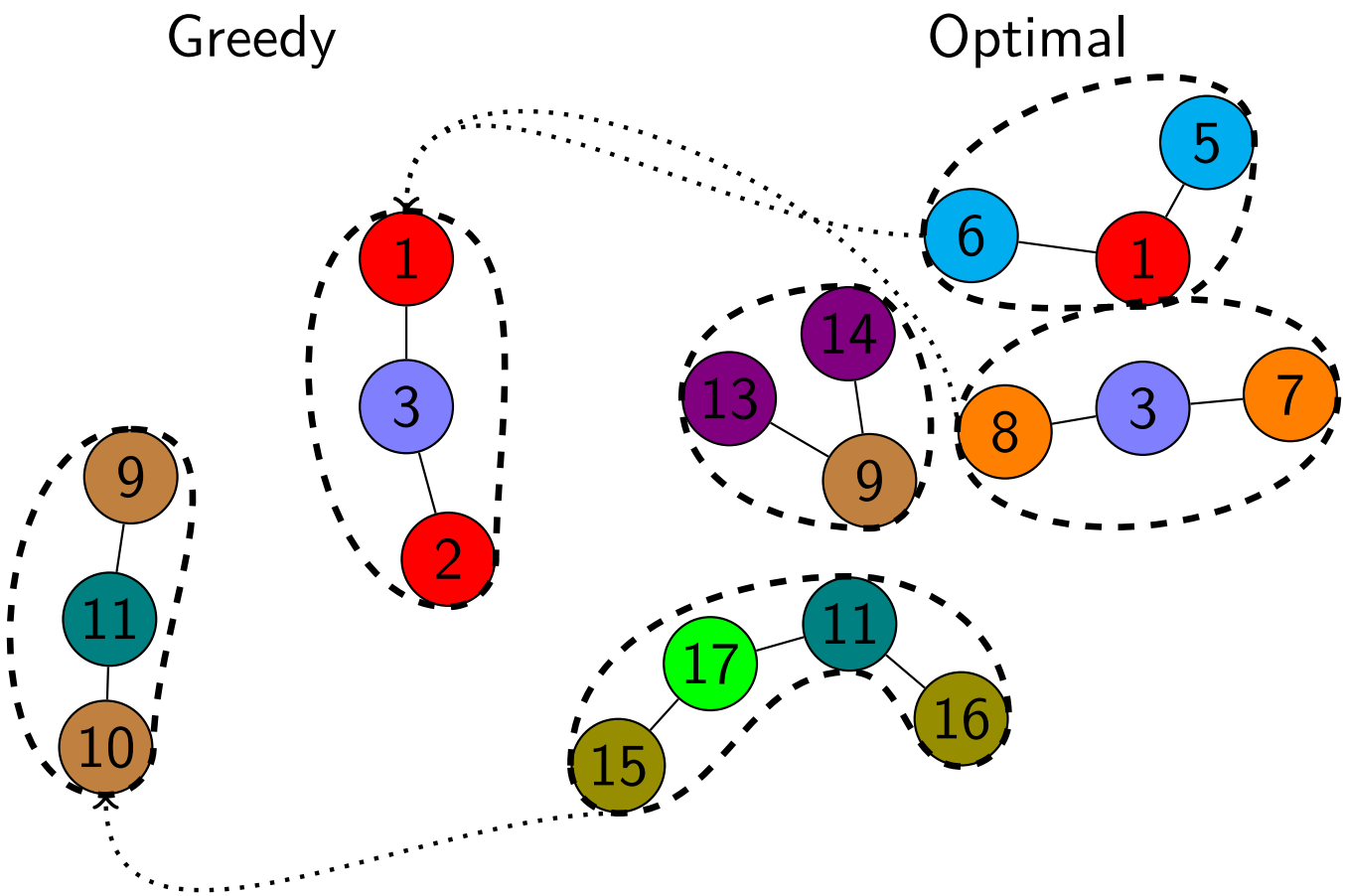
Toward a Second Bound



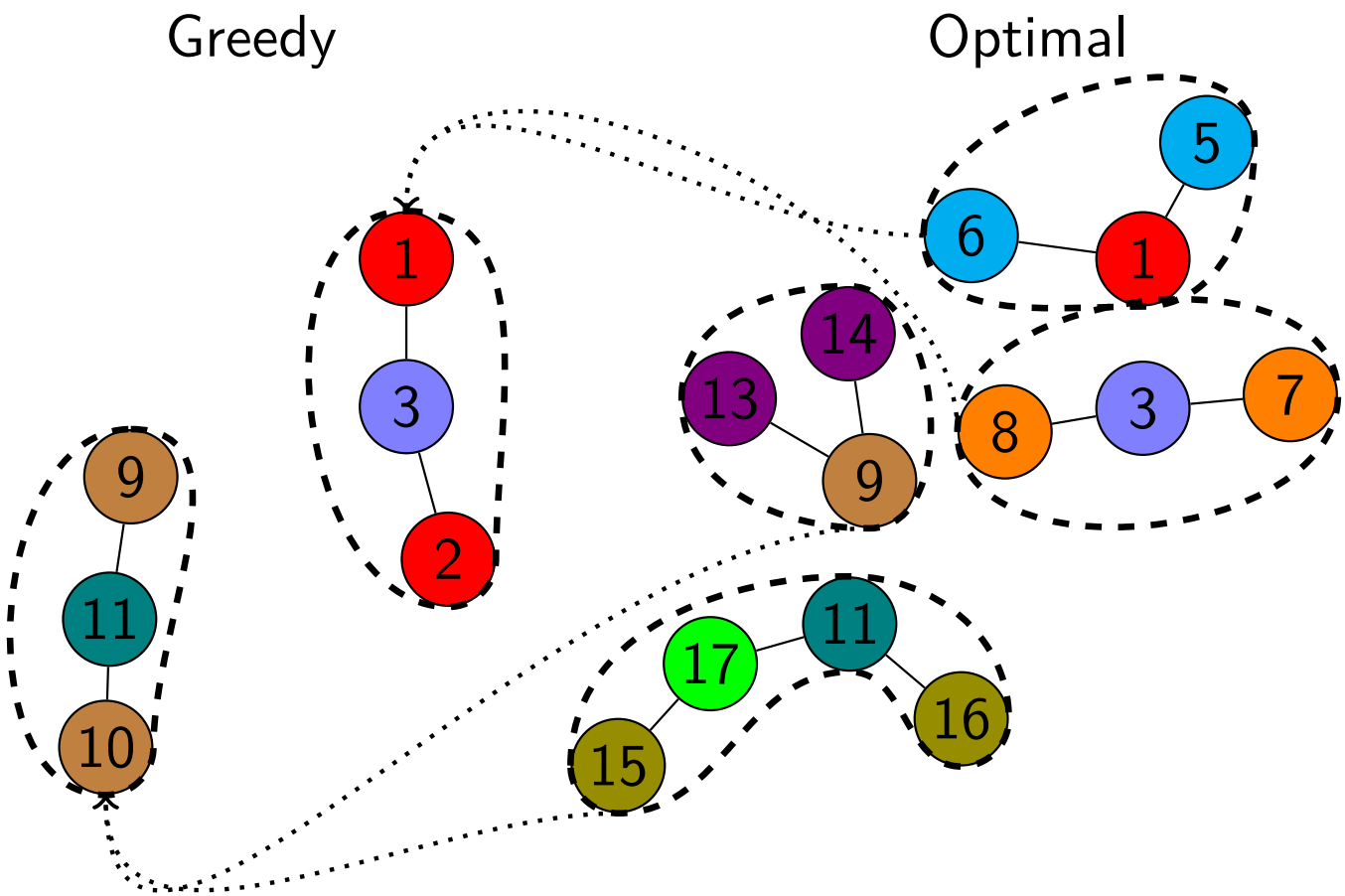
Toward a Second Bound



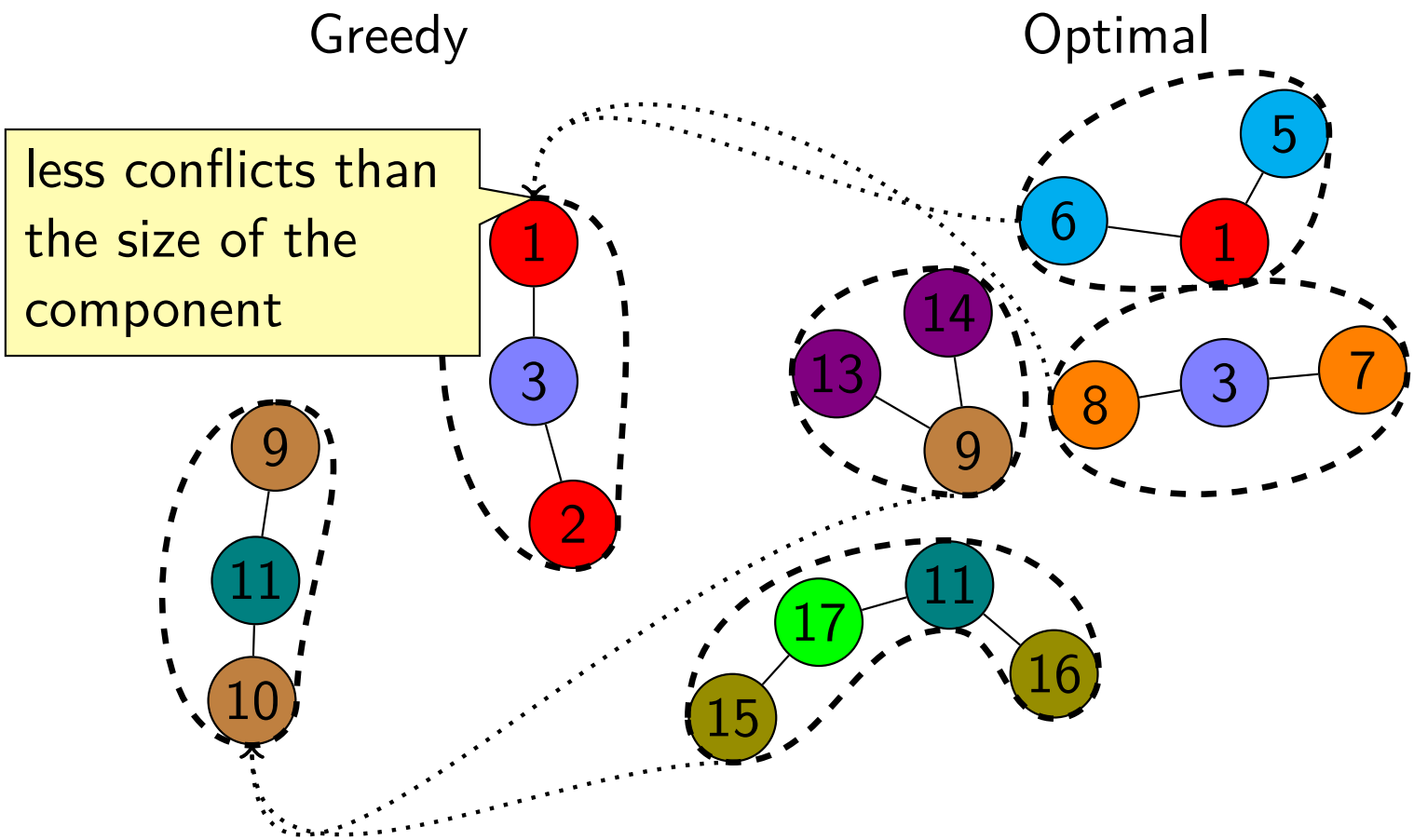
Toward a Second Bound



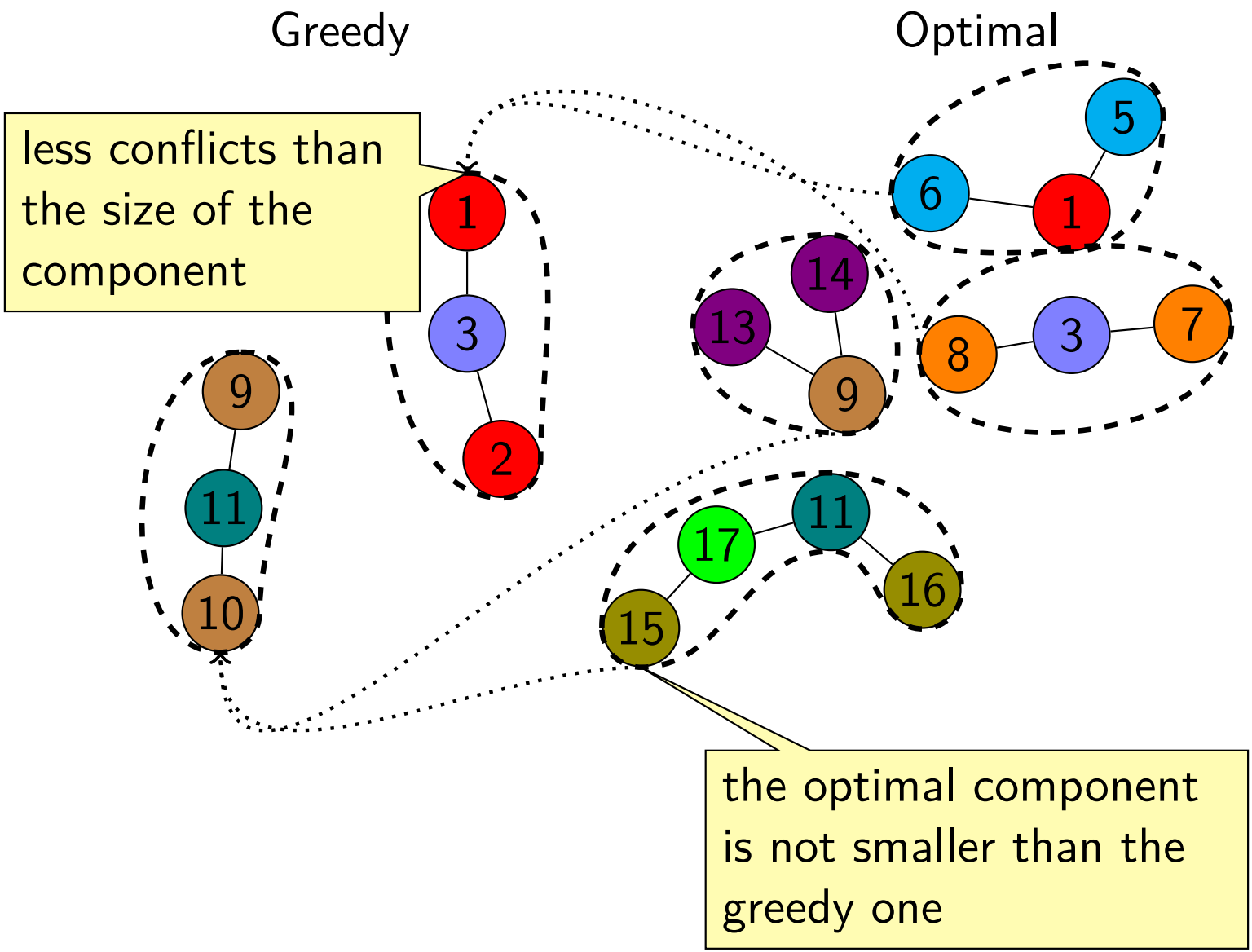
Toward a Second Bound



Toward a Second Bound



Toward a Second Bound



Second Bound

We can use these properties to prove the following lemma

Lemma

$$|\mathcal{I}^*| \leq \frac{|P|}{\alpha}$$

Second Bound

We can use these properties to prove the following lemma

Lemma

$$|\mathcal{I}^*| \leq \frac{|P|}{\alpha}$$

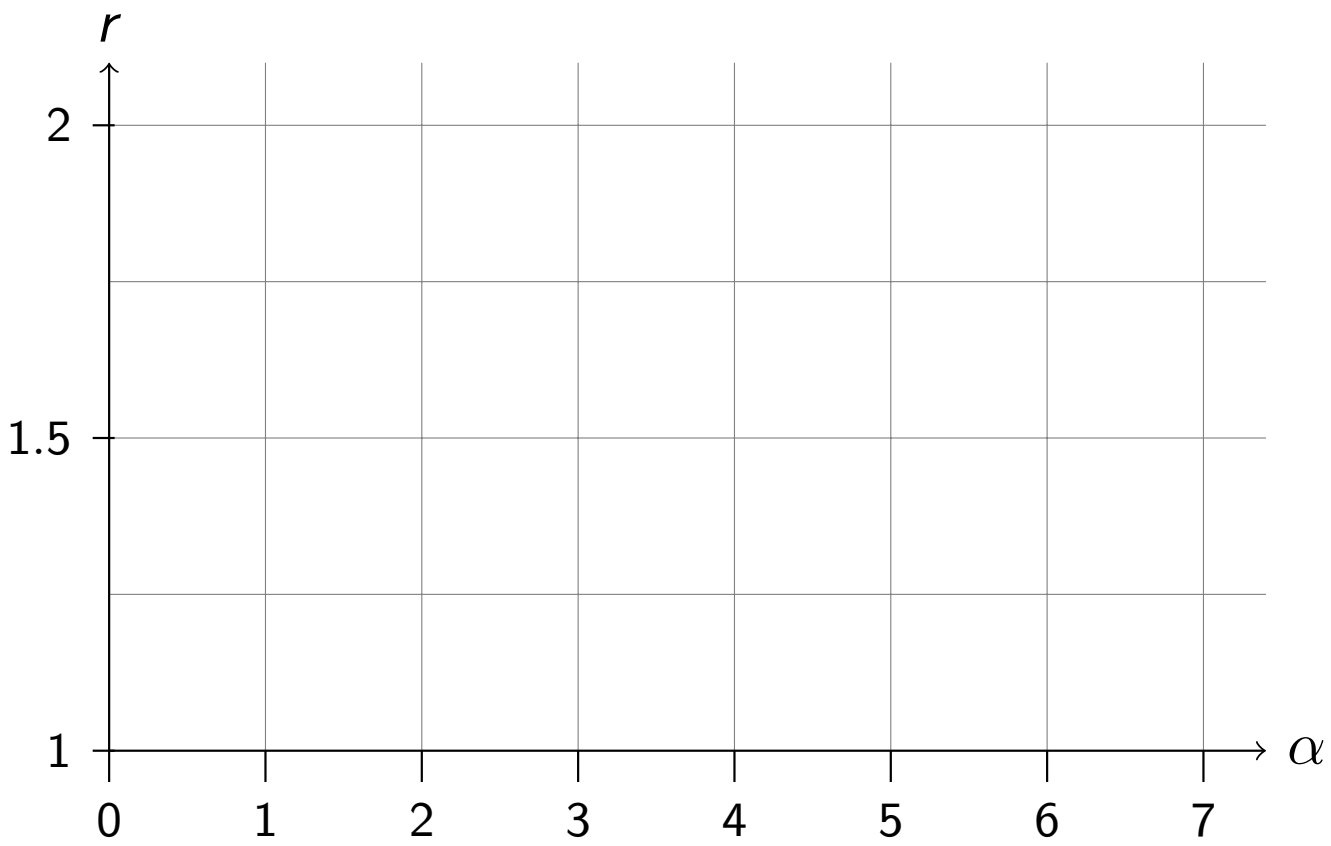
Corollary

$$r := \frac{\alpha|P| - |\mathcal{I}^*|}{\alpha(|P| - |\mathcal{I}^*|)} \leq 1 + \frac{1}{\alpha} \leq 2$$

$(\alpha \geq 1)$

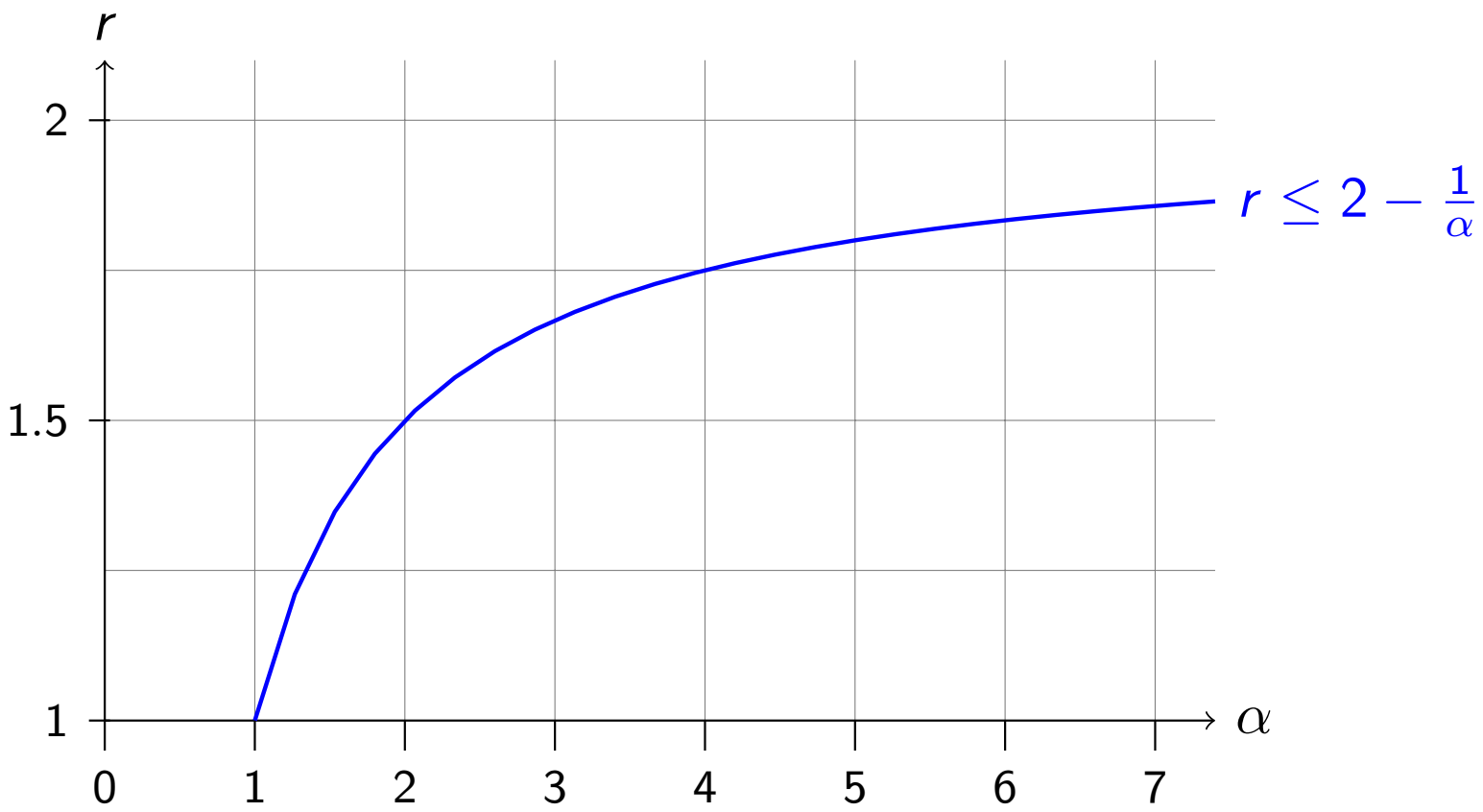
Approximation Ratio

Combining the two bounds we get



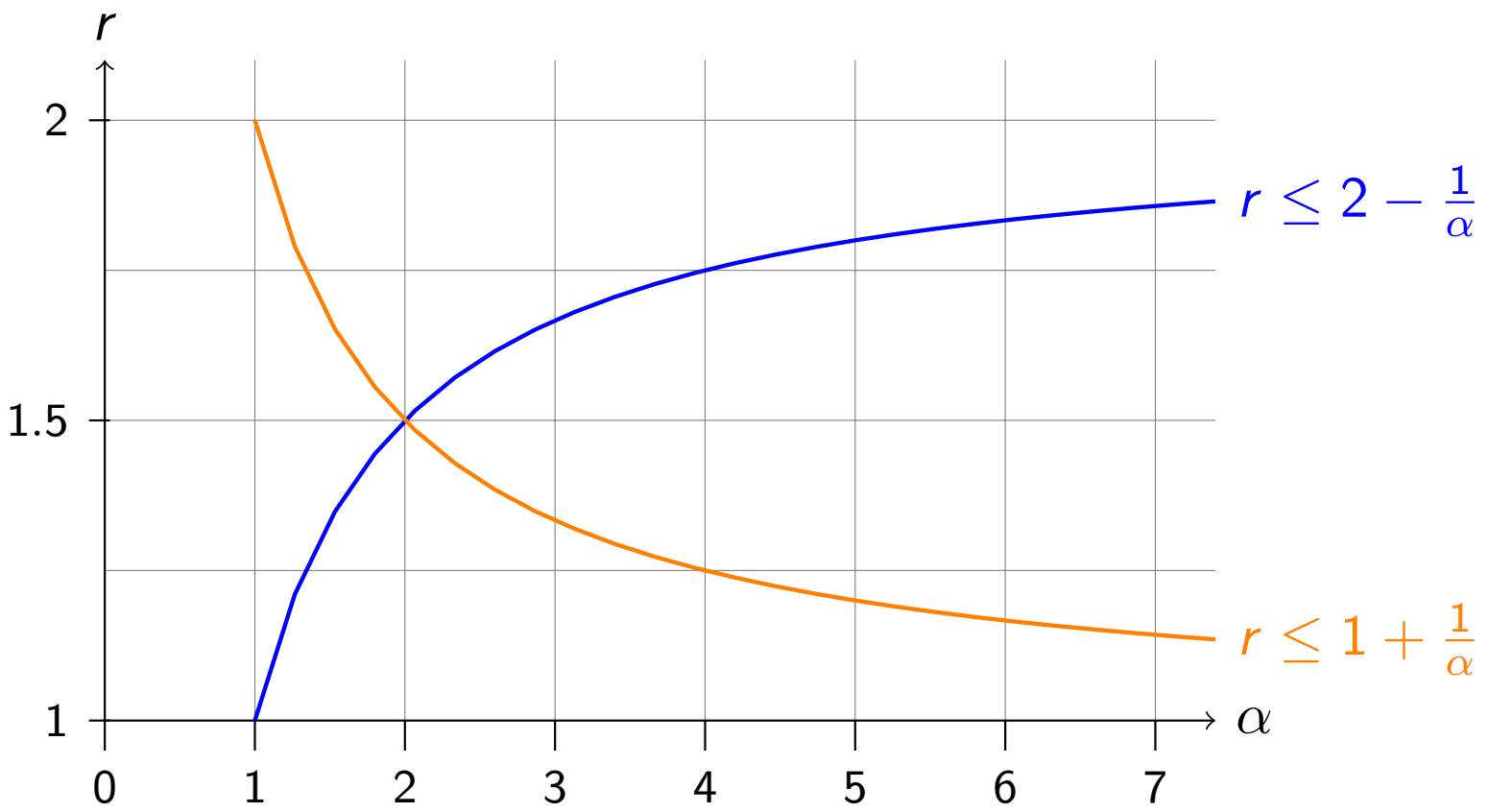
Approximation Ratio

Combining the two bounds we get



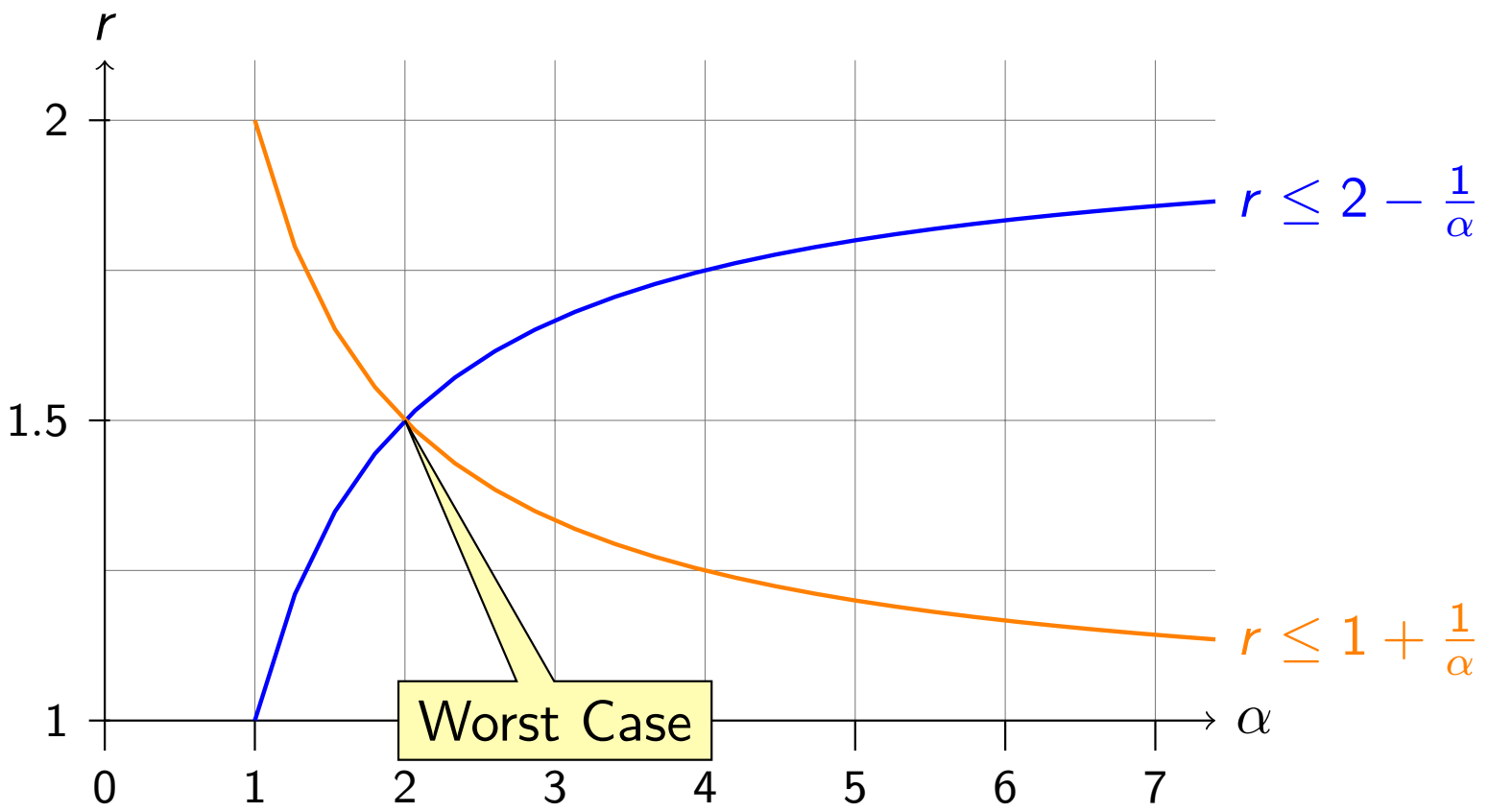
Approximation Ratio

Combining the two bounds we get



Approximation Ratio

Combining the two bounds we get



Conclusion

Our Result

Conclusion

Our Result

- First approximation algorithm for graphs

Conclusion

Our Result

- First approximation algorithm for graphs
  - Improved approximation ratio for paths

Conclusion

Our Result

- First approximation algorithm for graphs
  - Improved approximation ratio for paths
- Linear size kernel

Conclusion

Our Result

- First approximation algorithm for graphs
  - Improved approximation ratio for paths
- Linear size kernel
- FP algorithm for graphs

Conclusion

Our Result

- First approximation algorithm for graphs
  - Improved approximation ratio for paths
- Linear size kernel
- FP algorithm for graphs

Open Questions

Conclusion

Our Result

- First approximation algorithm for graphs
  - Improved approximation ratio for paths
- Linear size kernel
- FP algorithm for graphs

Open Questions

- Is 2-CR APX-hard ?

Conclusion

Our Result

- First approximation algorithm for graphs
  - Improved approximation ratio for paths
- Linear size kernel
- FP algorithm for graphs

Open Questions

- Is 2-CR APX-hard ?
- Hardness of k-CR, when $k > 2$, say 3 ?

Conclusion

Our Result

- First approximation algorithm for graphs
  - Improved approximation ratio for paths
- Linear size kernel
- FP algorithm for graphs

Open Questions

- Is 2-CR APX-hard ?
- Hardness of k-CR, when $k > 2$, say 3 ?

Thank You !