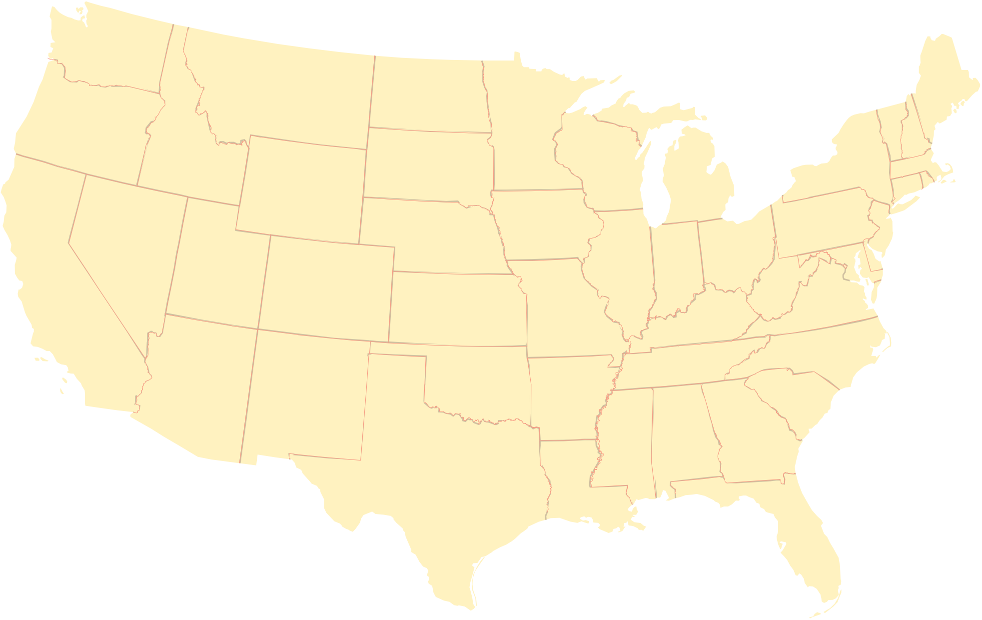
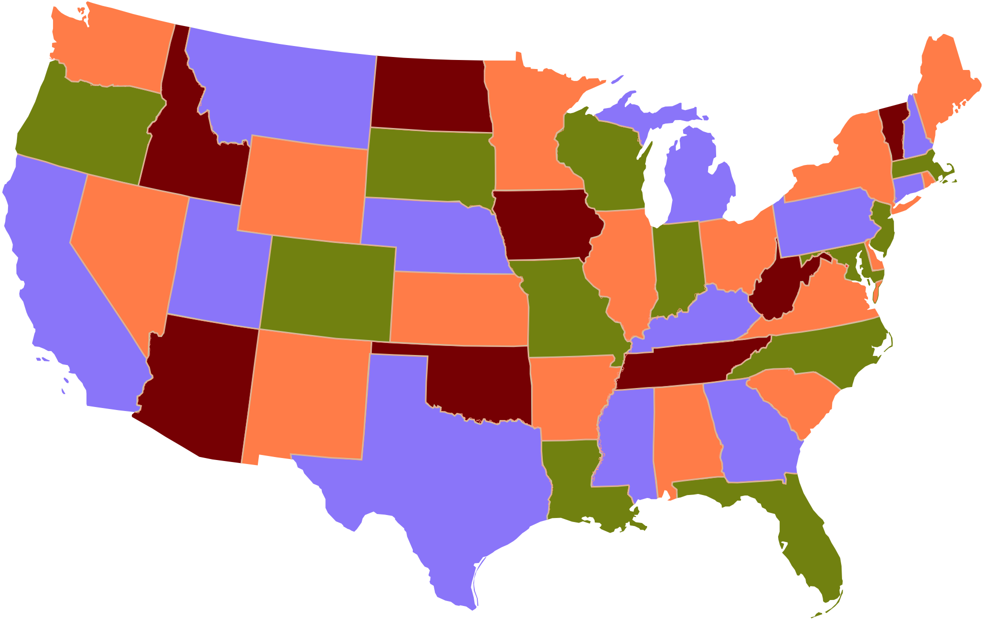
Conflict-Free Vertex Coloring of Planar Graphs

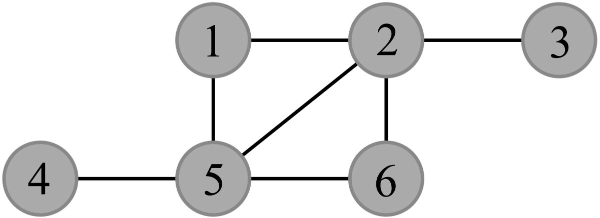
Shawn Seymour

April 15, 2017



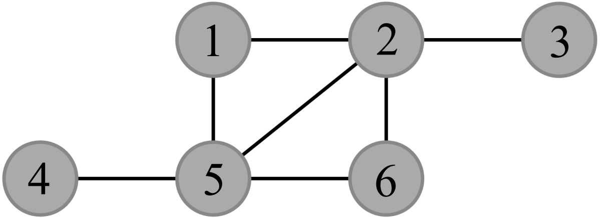


Overview

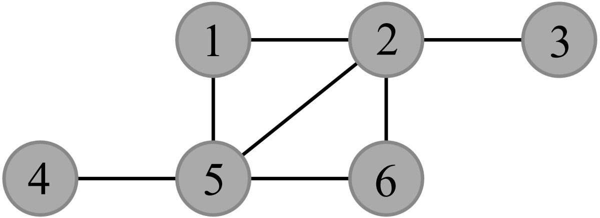


- **Simple graph**: undirected graph with no loops.

Graph Theory

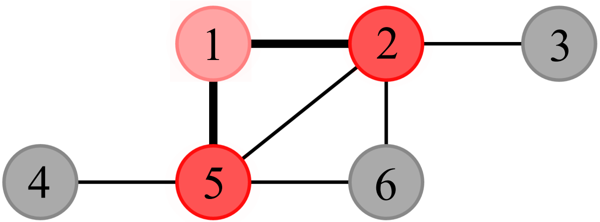


- **Simple graph**: undirected graph with no loops.
- **Planar graph**: no edges cross.



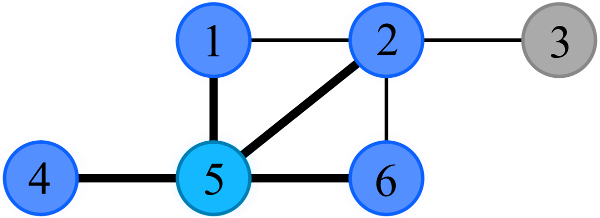
- **Neighborhood** of a vertex: a set of all adjacent vertices and the vertex itself.

Graph Theory



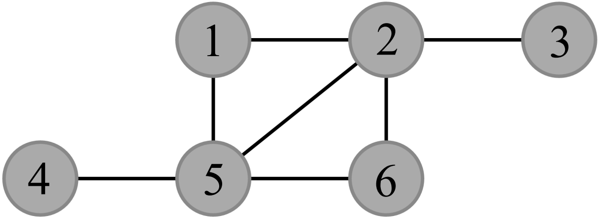
- **Neighborhood** of a vertex: a set of all adjacent vertices and the vertex itself.
- Example: The neighborhood of vertex 1: $\{1, 2, 5\}$.

Graph Theory



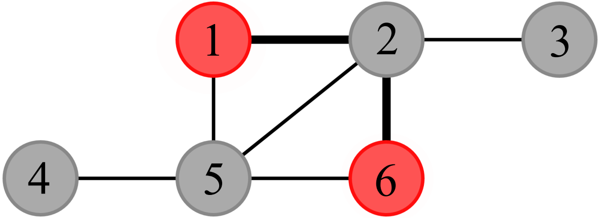
- **Neighborhood** of a vertex: a set of all adjacent vertices and the vertex itself.
- Example: The neighborhood of vertex 5: $\{1, 2, 4, 5, 6\}$.

Graph Theory

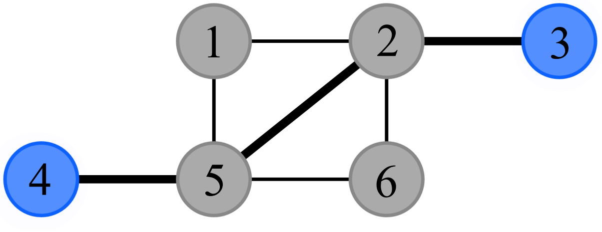


- **Distance**: smallest number of edges to get from one vertex to another.

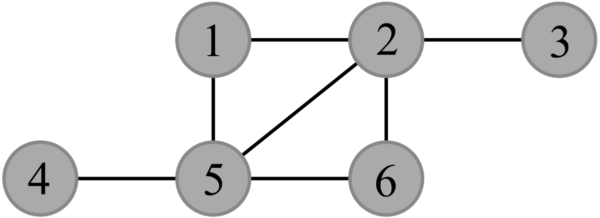
Graph Theory



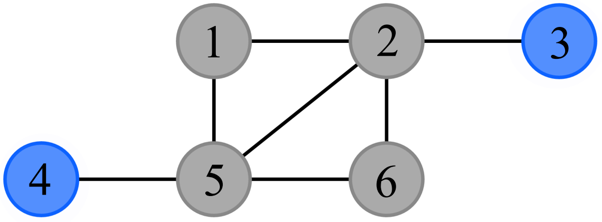
- **Distance**: smallest number of edges to get from one vertex to another.
- Example: distance from vertex 1 to 6: **2**.



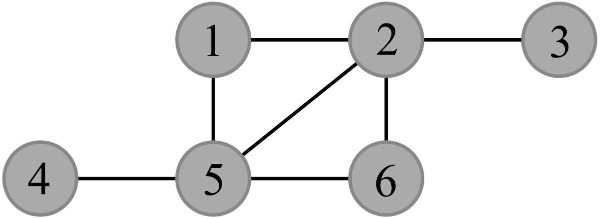
- **Distance**: smallest number of edges to get from one vertex to another.
- Example: distance from vertex 3 to 4: **3**.



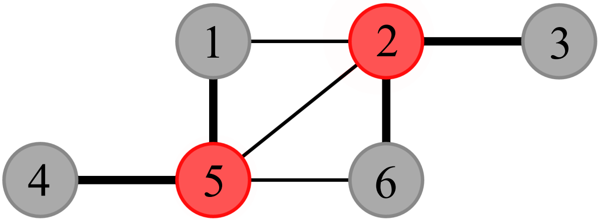
- **Distance-3-set**: contains all vertices with exactly distance 3 from each other.



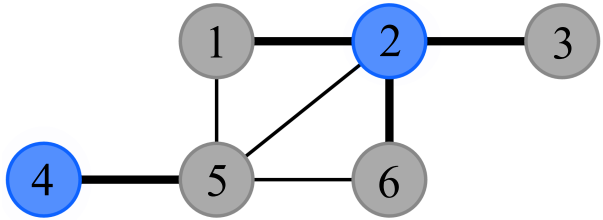
- **Distance-3-set**: contains all vertices with exactly distance 3 from each other.
- Example: The only possible distance-3-set: $\{3, 4\}$.



- **Dominating set**: all vertices not in the set must have distance 1 to some vertex within the set.

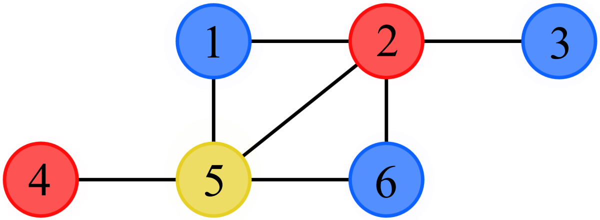


- **Dominating set**: all vertices not in the set must have distance 1 to some vertex within the set.
- Example: $\{2, 5\}$



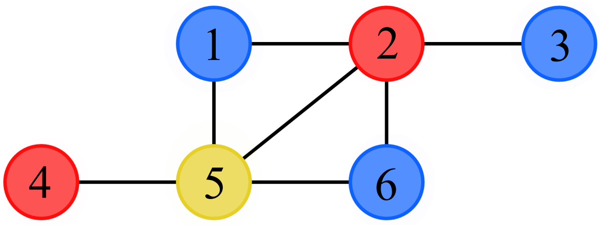
- **Dominating set**: all vertices not in the set must have distance 1 to some vertex within the set.
- Example: $\{2, 4\}$

Vertex Coloring



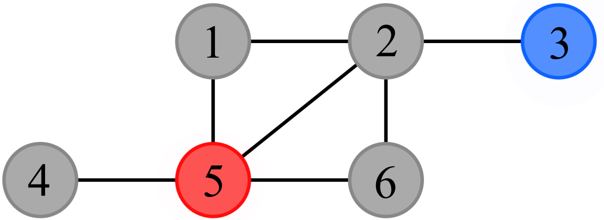
- A **proper vertex coloring** assigns colors to *every* vertex such that no two adjacent vertices share the same color.

Vertex Coloring



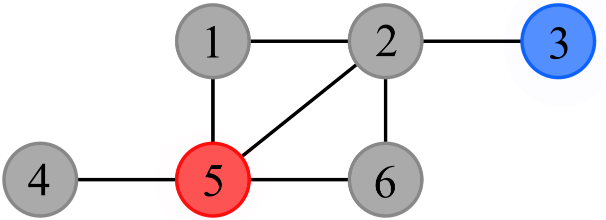
- A **proper vertex coloring** assigns colors to *every* vertex such that no two adjacent vertices share the same color.
- The **chromatic number** is the minimum number of colors needed to properly color a graph.

Conflict-Free Coloring



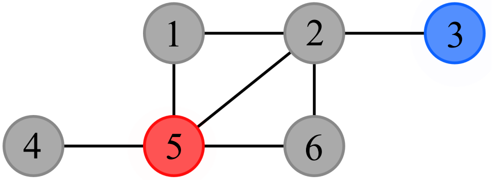
- A **conflict-free coloring** assigns colors to *some* vertices such that the neighborhood of every vertex contains at least one uniquely colored vertex.

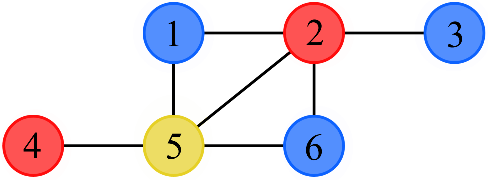
Conflict-Free Coloring



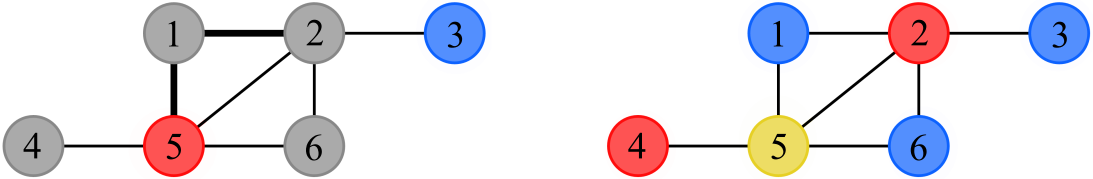
- A **conflict-free coloring** assigns colors to *some* vertices such that the neighborhood of every vertex contains at least one uniquely colored vertex.
- Proper vertex colorings are also conflict-free colorings.

Conflict-Free Coloring



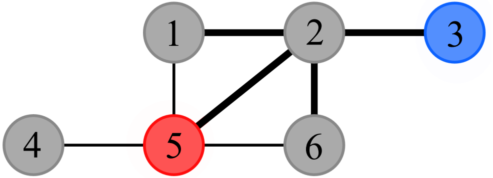


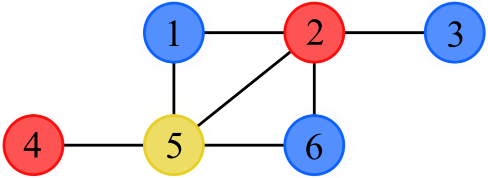
Conflict-Free Coloring



1: **red**

Conflict-Free Coloring

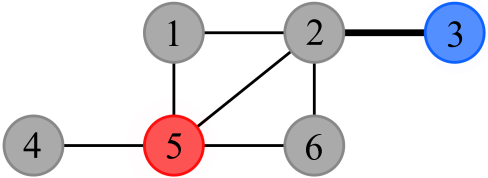


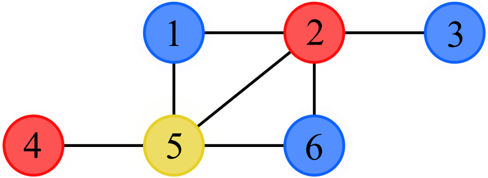


1: **red**

2: **blue**

Conflict-Free Coloring



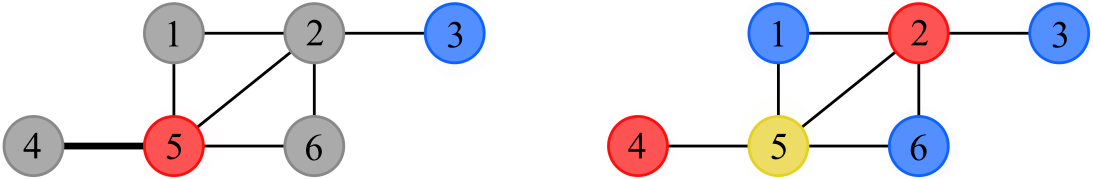


1: **red**

2: **blue**

3: **blue**

Conflict-Free Coloring



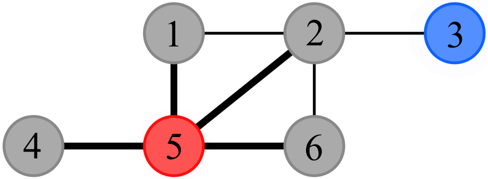
1: **red**

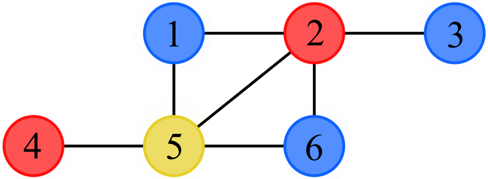
2: **blue**

3: **blue**

4: **red**

Conflict-Free Coloring





1: **red**

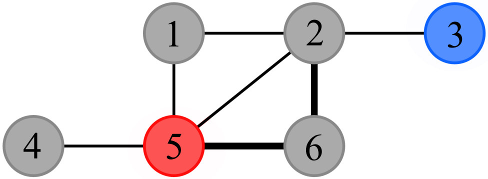
2: **blue**

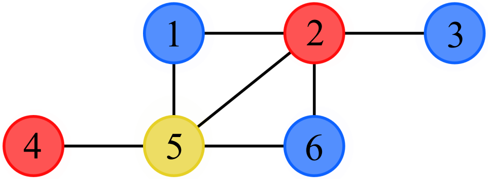
3: **blue**

4: **red**

5: **red**

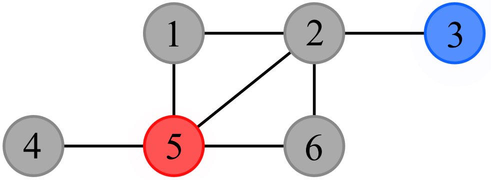
Conflict-Free Coloring

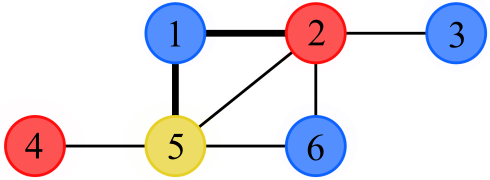




1: **red**	4: **red**
2: **blue**	5: **red**
3: **blue**	6: **red**

Conflict-Free Coloring





1: **red**

2: **blue**

3: **blue**

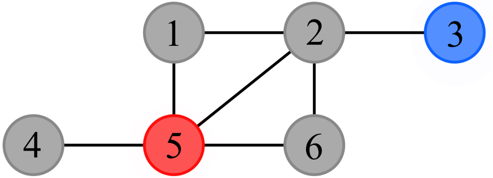
4: **red**

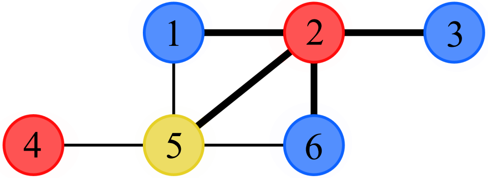
5: **red**

6: **red**

1: **blue**

Conflict-Free Coloring





1: **red**

2: **blue**

3: **blue**

4: **red**

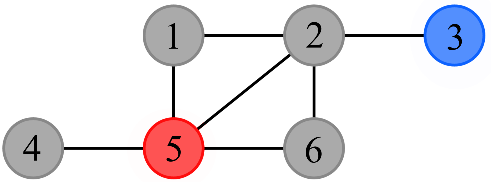
5: **red**

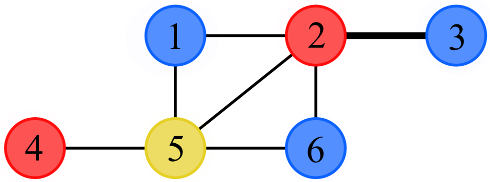
6: **red**

1: **blue**

2: **red**

Conflict-Free Coloring





1: **red**

2: **blue**

3: **blue**

4: **red**

5: **red**

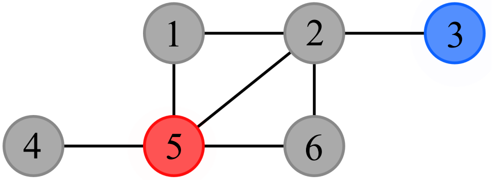
6: **red**

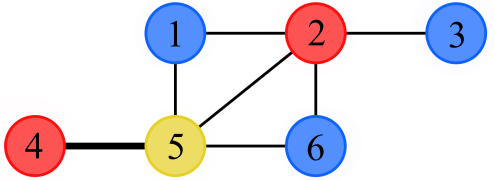
1: **blue**

2: **red**

3: **blue**

Conflict-Free Coloring

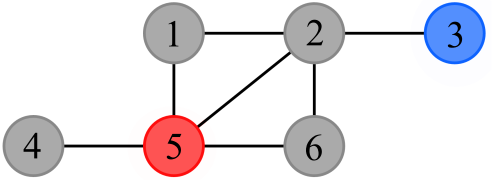


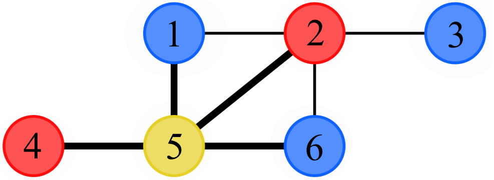


1: **red**
2: **blue**
3: **blue**
4: **red**
5: **red**
6: **red**

1: **blue**
2: **red**
3: **blue**
4: **red**

Conflict-Free Coloring





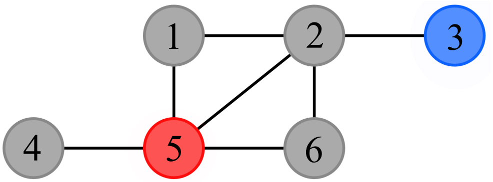
1: **red**
2: **blue**
3: **blue**

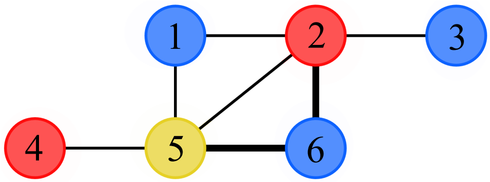
4: **red**
5: **red**
6: **red**

1: **blue**
2: **red**
3: **blue**

4: **red**
5: **yellow**

Conflict-Free Coloring





1: **red**	4: **red**	1: **blue**	4: **red**
2: **blue**	5: **red**	2: **red**	5: **yellow**
3: **blue**	6: **red**	3: **blue**	6: **blue**

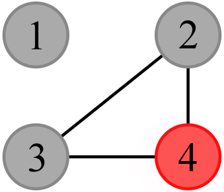
Conflict-Free Coloring Examples

Incorrect conflict-free colorings and their fixes:

Conflict-Free Coloring Examples

Incorrect conflict-free colorings and their fixes:

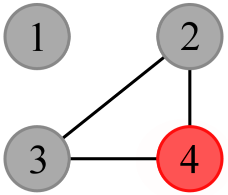
Isolated vertex

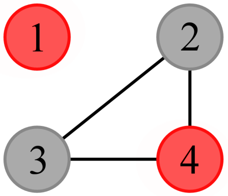


Conflict-Free Coloring Examples

Incorrect conflict-free colorings and their fixes:

Isolated vertex

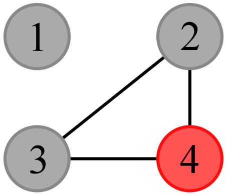


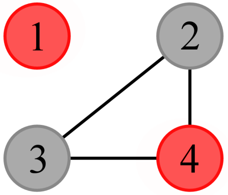


Conflict-Free Coloring Examples

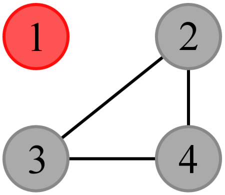
Incorrect conflict-free colorings and their fixes:

Isolated vertex





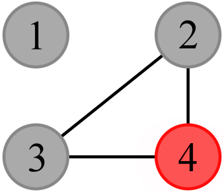
Triangle

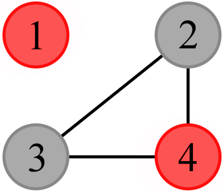


Conflict-Free Coloring Examples

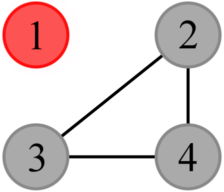
Incorrect conflict-free colorings and their fixes:

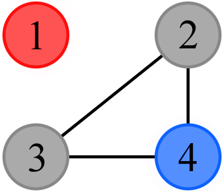
Isolated vertex





Triangle

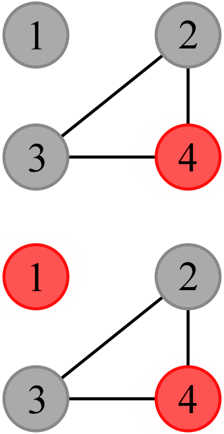




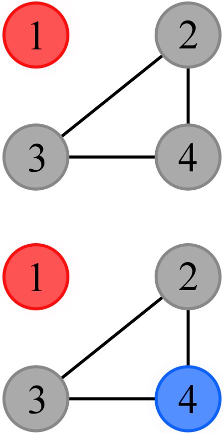
Conflict-Free Coloring Examples

Incorrect conflict-free colorings and their fixes:

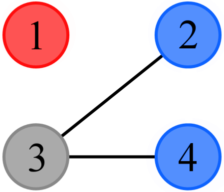
Isolated vertex



Triangle



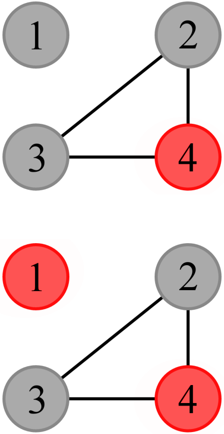
Missing unique



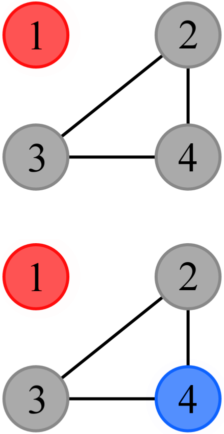
Conflict-Free Coloring Examples

Incorrect conflict-free colorings and their fixes:

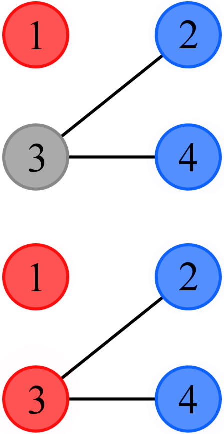
Isolated vertex



Triangle



Missing unique



Applications

- **Applications**: wireless networks, satellite communication systems, RFID networks.

Applications

- **Applications**: wireless networks, satellite communication systems, RFID networks.

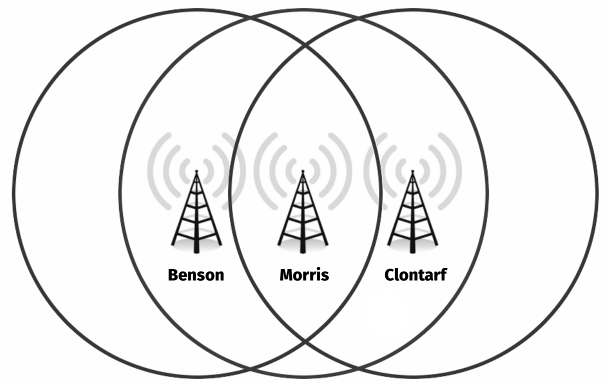
- **Cellular networks**: consist of towers and clients.

Applications

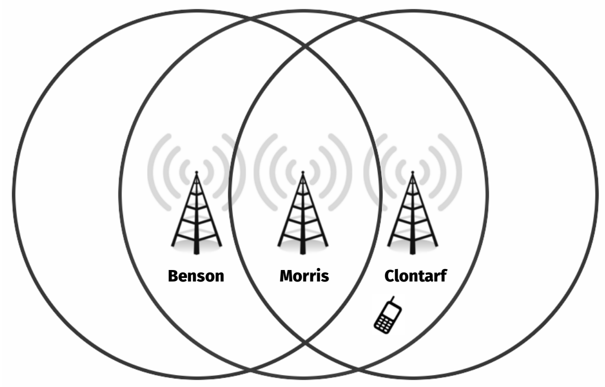
- **Applications**: wireless networks, satellite communication systems, RFID networks.

- **Cellular networks**: consist of towers and clients.
- The problem: **frequency assignment**.

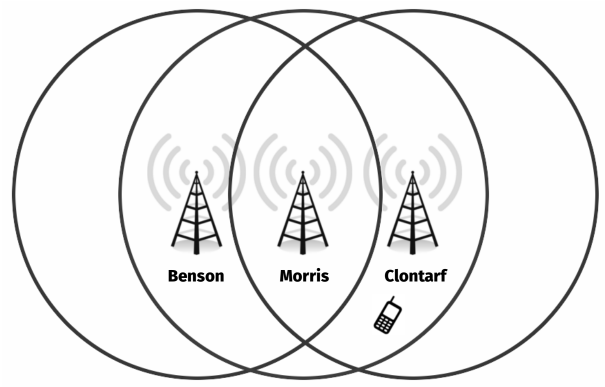
Cellular Networks



Cellular Networks

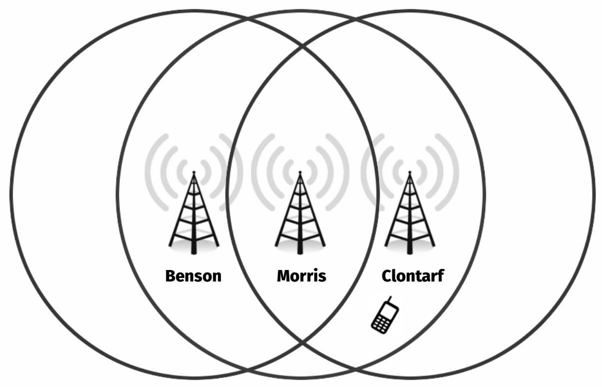


Cellular Networks

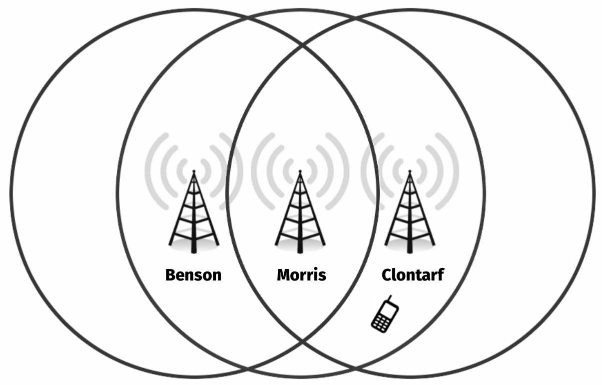


- Our goal is to assign frequencies such that:

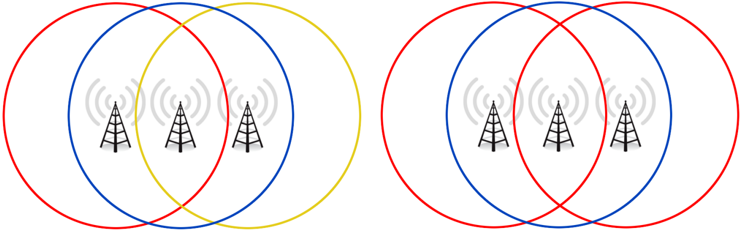
Cellular Networks



- Our goal is to assign frequencies such that:
  (1) Every client is served by a tower with a unique frequency.



- Our goal is to assign frequencies such that:
  (1) Every client is served by a tower with a unique frequency.
  (2) Minimize the number of frequencies used.



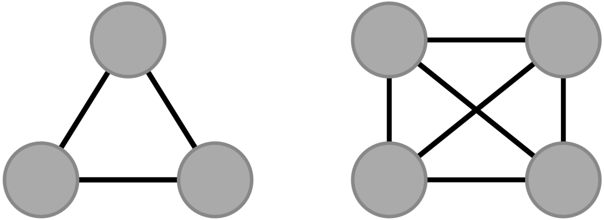
Vertex coloring and conflict-free coloring of cellular towers, respectively

Guaranteeing Conflict-Free k-Colorability

We want to guarantee a graph can be colored with k colors.

Guaranteeing Conflict-Free k-Colorability

We want to guarantee a graph can be colored with *k* colors.

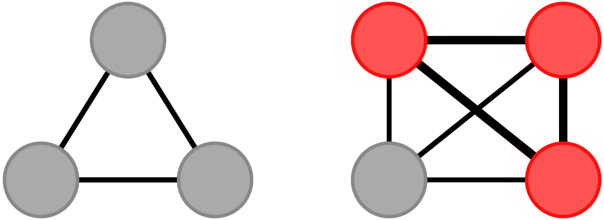


Complete 3 vertex and complete 4 vertex graphs

- **Complete graph**: every pair of distinct vertices is connected by an edge.

Guaranteeing Conflict-Free k-Colorability

We want to guarantee a graph can be colored with k colors.

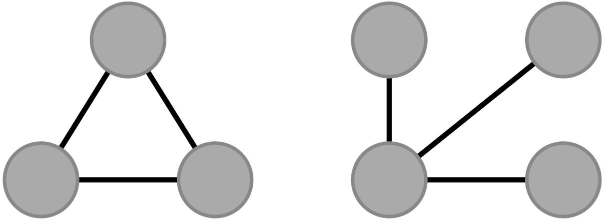


Complete 3 vertex and complete 4 vertex graphs

- **Complete graph**: every pair of distinct vertices is connected by an edge.
- **Tattered graph**: complete graph with a triangle removed.

Guaranteeing Conflict-Free k-Colorability

We want to guarantee a graph can be colored with k colors.

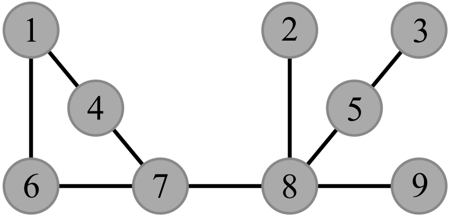


Complete 3 vertex and tattered 4 vertex graphs

- **Complete graph**: every pair of distinct vertices is connected by an edge.
- **Tattered graph**: complete graph with a triangle removed.

Guaranteeing Conflict-Free k-Colorability

Let's demonstrate meeting this criterion on a graph for the simplest case, 1 color.

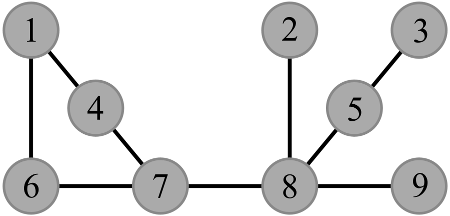


A simple graph

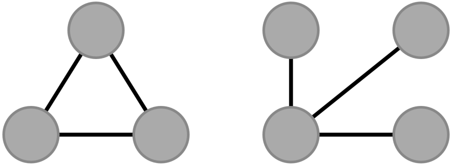
A graph is a **minor** if it can be formed by deleting and/or contracting edges from its parent graph.

Guaranteeing Conflict-Free k-Colorability

Let's demonstrate meeting this criterion on a graph for the simplest case, 1 color.



A simple graph

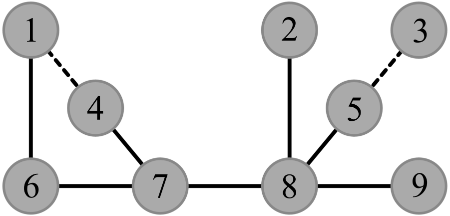


Complete 3 vertex and tattered 4 vertex graphs

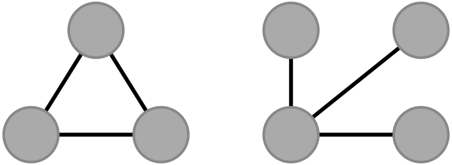
A graph is a **minor** if it can be formed by deleting and/or contracting edges from its parent graph.

Guaranteeing Conflict-Free k-Colorability

Let's demonstrate meeting this criterion on a graph for the simplest case, 1 color.



A simple graph



Complete 3 vertex and tattered 4 vertex graphs

A graph is a **minor** if it can be formed by deleting and/or contracting edges from its parent graph. G contains both graphs as a minor and thus **cannot** be conflict-free colored with 1 color.

Guaranteeing Conflict-Free k-Colorability

Let's demonstrate meeting this criterion on a graph for the simplest case, 1 color.

Theorem

A graph cannot be conflict-free colored with k colors if it contains a complete graph of $k+2$ vertices or a tattered graph of $k+3$ vertices as a minor.

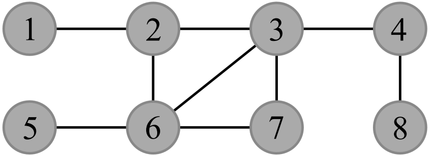
A graph is a **minor** if it can be formed by deleting and/or contracting edges from its parent graph. G contains both graphs as a minor and thus **cannot** be conflict-free colored with 1 color.

Iterated Elimination of Distance-3-Sets

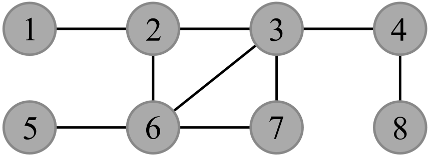
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
    $D \leftarrow \emptyset$
    for all components of G **do**
        Pick any vertex v
        $D \leftarrow D \cup \{v\}$
        while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
            Pick u at distance 3 from some vertex in D
            $D \leftarrow D \cup \{w\}$
        for all $u \in D$ **do**
            Color u with color i
        $i \leftarrow i + 1$
        for all $u \in D$ **do**
            Remove $N(u)$ from G
        Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



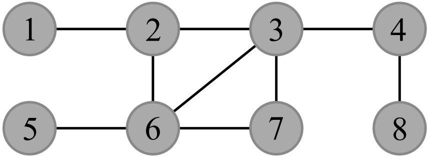
A simple graph G

Iterated Elimination of Distance-3-Sets

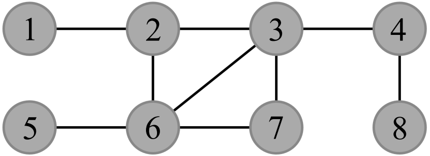
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
  $D \leftarrow \emptyset$
  for all components of G **do**
    Pick any vertex v
    $D \leftarrow D \cup \{v\}$
    while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
      Pick u at distance 3 from some vertex in D
      $D \leftarrow D \cup \{w\}$
    for all $u \in D$ **do**
      Color u with color i
    $i \leftarrow i + 1$
    for all $u \in D$ **do**
      Remove $N(u)$ from G
    Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



A simple graph G

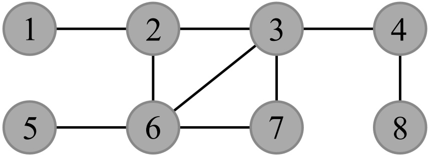
$i = 1$
$P = \{\}$

Iterated Elimination of Distance-3-Sets

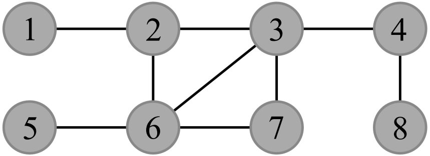
Algorithm IEDS

```
i ← 1,  P ← ∅
Remove all isolated paths from G
while G is not empty do
    D ← ∅
    for all components of G do
        Pick any vertex v
        D ← D ∪ {v}
        while ∃u at distance ≥ 3 ∀v ∈ D do
            Pick u at distance 3 from some vertex in D
            D ← D ∪ {w}
        for all u ∈ D do
            Color u with color i
        i ← i + 1
        for all u ∈ D do
            Remove N(u) from G
        Remove all isolated paths from G
Color all removed isolated paths using color i
```

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



A simple graph G

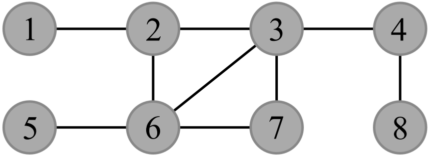
$$i = 1$$
$$P = \{\}$$
$$D = \{\}$$

Iterated Elimination of Distance-3-Sets

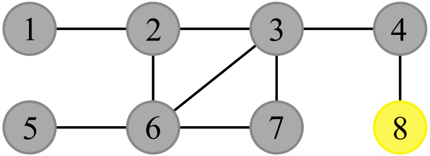
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
    $D \leftarrow \emptyset$
    for all components of G **do**
        Pick any vertex v
        $D \leftarrow D \cup \{v\}$
        while $\exists u$ at distance ≥ 3 $\forall v \in D$ **do**
            Pick u at distance 3 from some vertex in D
            $D \leftarrow D \cup \{w\}$
        for all $u \in D$ **do**
            Color u with color i
        $i \leftarrow i + 1$
        for all $u \in D$ **do**
            Remove $N(u)$ from G
        Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



A simple graph G

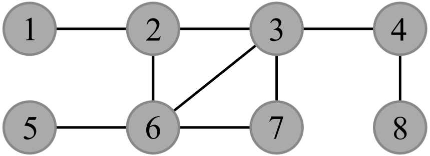
$i = 1$
$P = \{\}$
$D = \{8\}$

Iterated Elimination of Distance-3-Sets

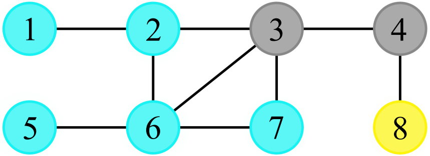
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
    $D \leftarrow \emptyset$
    for all components of G **do**
        Pick any vertex v
        $D \leftarrow D \cup \{v\}$
        while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
            Pick u at distance 3 from some vertex in D
            $D \leftarrow D \cup \{w\}$
        for all $u \in D$ **do**
            Color u with color i
        $i \leftarrow i + 1$
        for all $u \in D$ **do**
            Remove $N(u)$ from G
        Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



A simple graph G

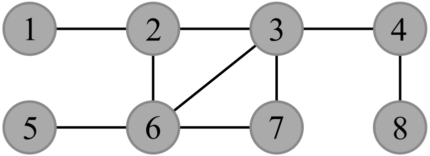
$i = 1$
$P = \{\}$
$D = \{8\}$

Iterated Elimination of Distance-3-Sets

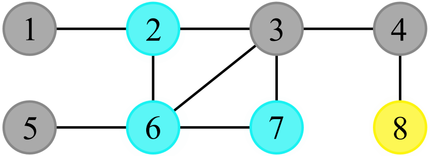
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
  $D \leftarrow \emptyset$
  for all components of G **do**
    Pick any vertex v
    $D \leftarrow D \cup \{v\}$
    while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
      Pick u at distance 3 from some vertex in D
      $D \leftarrow D \cup \{w\}$
    for all $u \in D$ **do**
      Color u with color i
    $i \leftarrow i + 1$
    for all $u \in D$ **do**
      Remove $N(u)$ from G
    Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



A simple graph G

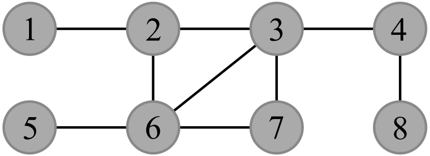
$i = 1$
$P = \{\}$
$D = \{8\}$

Iterated Elimination of Distance-3-Sets

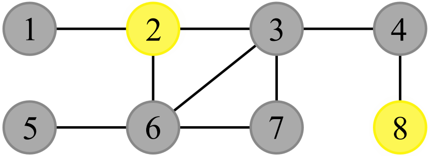
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
  $D \leftarrow \emptyset$
  for all components of G **do**
    Pick any vertex v
    $D \leftarrow D \cup \{v\}$
    while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
      Pick u at distance 3 from some vertex in D
      $D \leftarrow D \cup \{w\}$
    for all $u \in D$ **do**
      Color u with color i
    $i \leftarrow i + 1$
    for all $u \in D$ **do**
      Remove $N(u)$ from G
    Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



A simple graph G

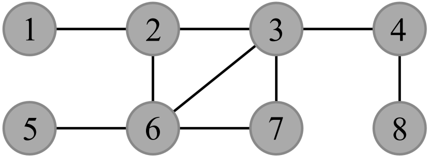
$i = 1$
$P = \{\}$
$D = \{2, 8\}$

Iterated Elimination of Distance-3-Sets

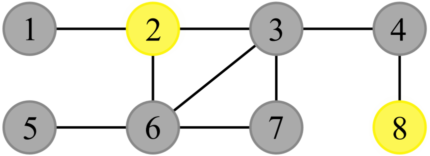
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
    $D \leftarrow \emptyset$
    for all components of G **do**
        Pick any vertex v
        $D \leftarrow D \cup \{v\}$
        while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
            Pick u at distance 3 from some vertex in D
            $D \leftarrow D \cup \{w\}$
        for all $u \in D$ **do**
            Color u with color i
        $i \leftarrow i + 1$
        for all $u \in D$ **do**
            Remove $N(u)$ from G
        Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



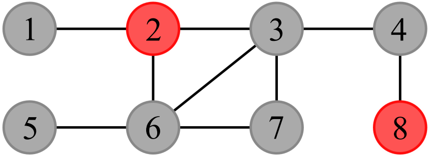
A simple graph G

$i = 1$
$P = \{\}$
$D = \{2, 8\}$

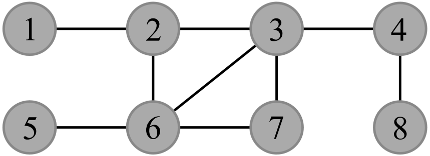
Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
    $D \leftarrow \emptyset$
    for all components of G **do**
        Pick any vertex v
        $D \leftarrow D \cup \{v\}$
        while $\exists u$ at distance ≥ 3 $\forall v \in D$ **do**
            Pick u at distance 3 from some vertex in D
            $D \leftarrow D \cup \{w\}$
        for all $u \in D$ **do**
            Color u with color i
        $i \leftarrow i + 1$
        for all $u \in D$ **do**
            Remove $N(u)$ from G
        Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far



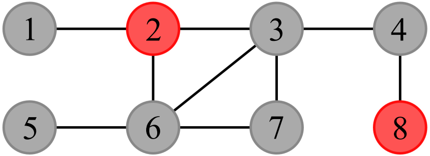
A simple graph G

$$i = 2$$
$$P = \{\}$$
$$D = \{2, 8\}$$

Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
  $D \leftarrow \emptyset$
  for all components of G **do**
    Pick any vertex v
    $D \leftarrow D \cup \{v\}$
    while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
      Pick u at distance 3 from some vertex in D
      $D \leftarrow D \cup \{w\}$
    for all $u \in D$ **do**
      Color u with color i
    $i \leftarrow i + 1$
    for all $u \in D$ **do**
      Remove $N(u)$ from G
    Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far

A simple graph G

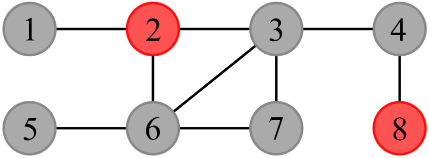
$$i = 2$$
$$P = \{\}$$
$$D = \{2, 8\}$$

Iterated Elimination of Distance-3-Sets

Algorithm IEDS

```
i ← 1,  P ← ∅
Remove all isolated paths from G
while G is not empty do
    D ← ∅
    for all components of G do
        Pick any vertex v
        D ← D ∪ {v}
        while ∃u at distance ≥ 3 ∀v ∈ D do
            Pick u at distance 3 from some vertex in D
            D ← D ∪ {w}
        for all u ∈ D do
            Color u with color i
        i ← i + 1
        for all u ∈ D do
            Remove N(u) from G
        Remove all isolated paths from G
Color all removed isolated paths using color i
```

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far

G is now empty

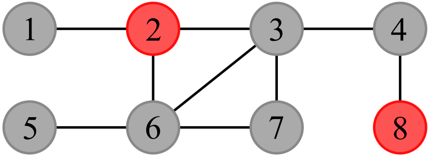
$$i = 2$$
$$P = \{\{5\}, \{7\}\}$$
$$D = \{2, 8\}$$

Iterated Elimination of Distance-3-Sets

Algorithm IEDS

```
i ← 1,  P ← ∅
Remove all isolated paths from G
while G is not empty do
    D ← ∅
    for all components of G do
        Pick any vertex v
        D ← D ∪ {v}
        while ∃u at distance ≥ 3 ∀v ∈ D do
            Pick u at distance 3 from some vertex in D
            D ← D ∪ {w}
        for all u ∈ D do
            Color u with color i
        i ← i + 1
        for all u ∈ D do
            Remove N(u) from G
        Remove all isolated paths from G
Color all removed isolated paths using color i
```

Colors: $\{1 : red,\ 2 : blue\}$



Coloring of G so far

Coloring removed isolated paths

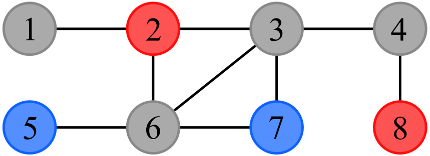
$$i = 2$$
$$P = \{\{5\}, \{7\}\}$$
$$D = \{2, 8\}$$

Iterated Elimination of Distance-3-Sets

Algorithm IEDS

$i \leftarrow 1,\ P \leftarrow \emptyset$
Remove all isolated paths from G
while G is not empty **do**
  $D \leftarrow \emptyset$
  for all components of G **do**
    Pick any vertex v
    $D \leftarrow D \cup \{v\}$
    while $\exists u$ at distance $\geq 3\ \forall v \in D$ **do**
      Pick u at distance 3 from some vertex in D
      $D \leftarrow D \cup \{w\}$
    for all $u \in D$ **do**
      Color u with color i
    $i \leftarrow i + 1$
    for all $u \in D$ **do**
      Remove $N(u)$ from G
    Remove all isolated paths from G
Color all removed isolated paths using color i

Colors: $\{1 : red,\ 2 : blue\}$



Final coloring of G

$$i = 2$$
$$P = \{\{5\}, \{7\}\}$$
$$D = \{2, 8\}$$

Theorem

Every loopless planar graph admits a proper **vertex coloring** *with at most four distinct colors.*

Bounds on Planar Graphs

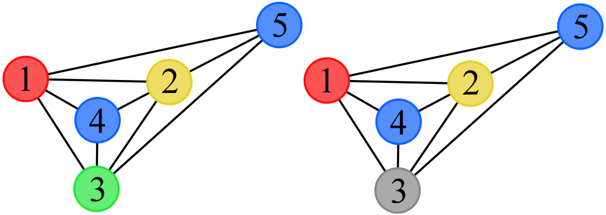
Theorem

Every loopless planar graph admits a proper ***vertex coloring*** *with at most four distinct colors.*

Theorem

Every loopless planar graph admits a ***conflict-free coloring*** *with at most three distinct colors.*

Bounds on Planar Graphs



A vertex and conflict-free coloring, respectively

Theorem

Every loopless planar graph admits a proper ***vertex coloring*** *with at most four distinct colors.*

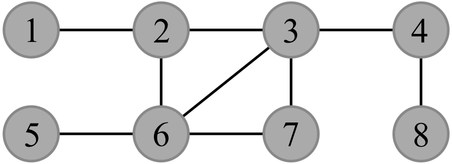
Theorem

Every loopless planar graph admits a ***conflict-free coloring*** *with at most three distinct colors.*

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

Find a dominating set, D, of G
for all $v \in V \setminus D$ **do**
    Pick a vertex $u \in D$ where $\{u, v\} \in E$
    Contract the edge $\{u, v\}$ towards u
Find a proper vertex coloring of G
Color the original G with the found coloring

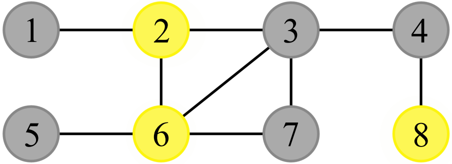


A simple, undirected graph $G = (V, E)$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

1: Find a dominating set, D, of G
2: **for all** $v \in V \setminus D$ **do**
3: Pick a vertex $u \in D$ where $\{u, v\} \in E$
4: Contract the edge $\{u, v\}$ towards u
5: Find a proper vertex coloring of G
6: Color the original G with the found coloring



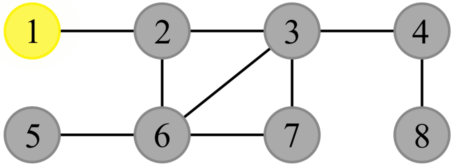
A dominating set of G

$$D = \{2, 6, 8\}$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

```
Find a dominating set, D, of G
for all v ∈ V \ D do
    Pick a vertex u ∈ D where {u, v} ∈ E
    Contract the edge {u, v} towards u
Find a proper vertex coloring of G
Color the original G with the found coloring
```



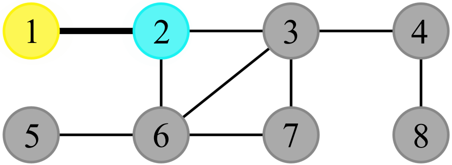
Pick v to be vertex 1

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{1, 3, 4, 5, 7\}$$
$$v \in V \setminus D = 1$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

Find a dominating set, D, of G
for all $v \in V \setminus D$ **do**
  Pick a vertex $u \in D$ where $\{u, v\} \in E$
  Contract the edge $\{u, v\}$ towards u
Find a proper vertex coloring of G
Color the original G with the found coloring



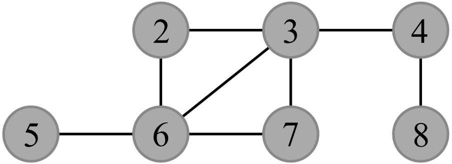
Pick u to be vertex 2

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{1, 3, 4, 5, 7\}$$
$$v \in V \setminus D = 1$$
$$u \in D = 2, \{1, 2\} \in E$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

1: Find a dominating set, D, of G
2: **for all** $v \in V \setminus D$ **do**
3: Pick a vertex $u \in D$ where $\{u, v\} \in E$
4: Contract the edge $\{u, v\}$ towards u
5: Find a proper vertex coloring of G
6: Color the original G with the found coloring



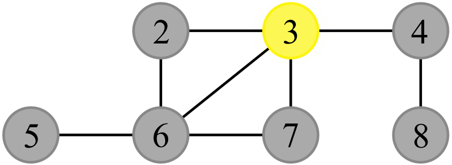
Contract $\{1, 2\}$ towards 2

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{3, 4, 5, 7\}$$
$$v \in V \setminus D = 1$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

Find a dominating set, D, of G
for all $v \in V \setminus D$ **do**
    Pick a vertex $u \in D$ where $\{u, v\} \in E$
    Contract the edge $\{u, v\}$ towards u
Find a proper vertex coloring of G
Color the original G with the found coloring



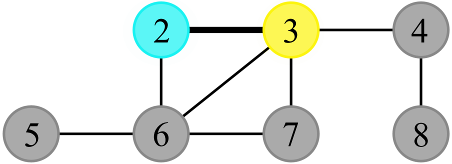
Pick v to be vertex 3

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{3, 4, 5, 7\}$$
$$v \in V \setminus D = 3$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

Find a dominating set, D, of G
for all $v \in V \setminus D$ **do**
    Pick a vertex $u \in D$ where $\{u, v\} \in E$
    Contract the edge $\{u, v\}$ towards u
Find a proper vertex coloring of G
Color the original G with the found coloring



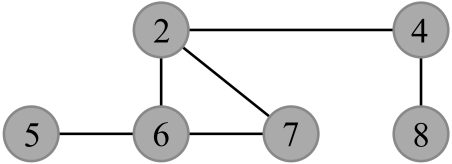
Pick u to be vertex 2

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{3, 4, 5, 7\}$$
$$v \in V \setminus D = 3$$
$$u \in D = 2, \{2, 3\} \in E$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

1: Find a dominating set, D, of G
2: **for all** $v \in V \setminus D$ **do**
3: Pick a vertex $u \in D$ where $\{u, v\} \in E$
4: Contract the edge $\{u, v\}$ towards u
5: Find a proper vertex coloring of G
6: Color the original G with the found coloring



Contract $\{2, 3\}$ towards 2

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{4, 5, 7\}$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

1: Find a dominating set, D, of G
2: **for all** $v \in V \setminus D$ **do**
3: Pick a vertex $u \in D$ where $\{u, v\} \in E$
4: Contract the edge $\{u, v\}$ towards u
5: Find a proper vertex coloring of G
6: Color the original G with the found coloring

G after lines 2-4

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{\}$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

```
Find a dominating set, D, of G
for all v ∈ V \ D do
    Pick a vertex u ∈ D where {u, v} ∈ E
    Contract the edge {u, v} towards u
Find a proper vertex coloring of G
Color the original G with the found coloring
```

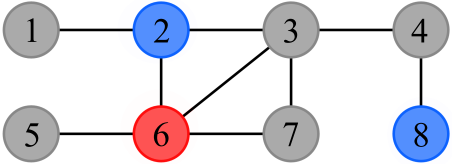
A proper vertex coloring

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{\}$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

1: Find a dominating set, D, of G
2: **for all** $v \in V \setminus D$ **do**
3: Pick a vertex $u \in D$ where $\{u, v\} \in E$
4: Contract the edge $\{u, v\}$ towards u
5: Find a proper vertex coloring of G
6: Color the original G with the found coloring



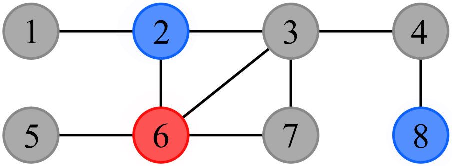
Original G colored

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{\}$$

Conflict-Free Coloring via Dominating Set

Algorithm Dominating Set

```
Find a dominating set, D, of G
for all v ∈ V \ D do
    Pick a vertex u ∈ D where {u, v} ∈ E
    Contract the edge {u, v} towards u
Find a proper vertex coloring of G
Color the original G with the found coloring
```



Original G colored

(1) Produces a valid conflict-free coloring

(2) Tries to minimize the number of colored vertices

$$D = \{2, 6, 8\}$$
$$V \setminus D = \{\}$$

Future Work

- Finding bounds and properties on specific graphs such as outerplanar graphs, interval graphs, hypergraphs, and more.

Future Work

- Finding bounds and properties on specific graphs such as outerplanar graphs, interval graphs, hypergraphs, and more.
    - Allows for accurate estimates when applying conflict-free coloring to real-world problems.

Future Work

- Finding bounds and properties on specific graphs such as outerplanar graphs, interval graphs, hypergraphs, and more.
    - Allows for accurate estimates when applying conflict-free coloring to real-world problems.
- Variations of conflict-free coloring such as requiring **another** vertex to have a unique color within the neighborhood of a selected vertex.

Future Work

- Finding bounds and properties on specific graphs such as outerplanar graphs, interval graphs, hypergraphs, and more.
    - Allows for accurate estimates when applying conflict-free coloring to real-world problems.
- Variations of conflict-free coloring such as requiring **another** vertex to have a unique color within the neighborhood of a selected vertex.
    - Guiding a robot (unique color 1) to a destination (unique color 2).

Thanks to Peter Dolan, Elena Machkasova, and Peh Ng for their advice and feedback.

github.com/devshawn/senior-seminar

References

[1] Z. Abel et al. "Three Colors Suffice: Conflict-Free Coloring of Planar Graphs". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 1951–1963.

[2] B. S. Baker. "Approximation algorithms for NP-complete problems on planar graphs". In: *Journal of the ACM (JACM)* 41.1 (1994), pp. 153–180.

[3] J. A. Bondy and U. S. R. Murty. *Graph theory with applications*. Vol. 290. Citeseer, 1976.

[4] P. Cheilaris et al. "Strong conflict-free coloring for intervals". In: *Algorithmica* 70.4 (2014), pp. 732–749.

[5] M. R. Garey and D. S. Johnson. *Computers and intractability*. Vol. 29. wh freeman New York, 2002.

References

[6] B. M. Moret. *The theory of computation*. Tech. rep. Addison-Wesley, Reading, Mass., 1998.

[7] N. Robertson et al. “The four-colour theorem”. In: *journal of combinatorial theory, Series B* 70.1 (1997), pp. 2–44.

[8] P. C. Sharma and N. S. Chaudhari. “A new reduction from 3-SAT to graph k-colorability for frequency assignment problem”. In: *Int. J. Comp. Applic* (2012), pp. 23–27.

[9] M. Sipser. *Introduction to the Theory of Computation*. Vol. 2. Thomson Course Technology Boston, 2006.

[10] S. Smorodinsky. “Conflict-free coloring and its applications”. In: *Geometry Intuitive, Discrete, and Convex*. Springer, 2013, pp. 331–389.

References

[11] D. B. West et al. *Introduction to graph theory*. Vol. 2. Prentice hall Upper Saddle River, 2001.