Contents I

1 Introduction
   - Line-Graph and Coloring
   - Edge-Colouring
   - Theorems
2 Hardness of the Edge-Colouring
   - Proof of Hoyer
3 Algorithms
   - Proof of König
   - Proof of Vizing
4 Colour with Greed
   - Simple Bounds
   - Algorithm
   - Examples
5 Theorem of Brooks
   - Statements
   - Proof
6 Girth
   - Statements
   - Proof
7 Colouring with known $\chi(G)$
   - Basics
   - Theorems
8 Complexity
   - Negative Theorems
   - Positive Theorems
**Definition of Coloring**

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
  - The mapping $f$ is called **coloring** of $G$.
  - $\chi(G)$ is the **chromatic number** $\chi(G)$ of $G$, iff
  - $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

**Definition**

Sei $G = (V, E)$ Graph.

\[
\begin{align*}
\alpha(G) &= \max \{ |V'| ; \ V' \subset V \ \land \ \forall a, b \in V' : (a, b) \notin E \} \\
\omega(G) &= \max \{ |V'| ; \ V' \subset V \ \land \ \forall a, b \in V' : (a, b) \in E \} \\
\chi(G) &= \min \{ k ; \ \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \ \land \\
&\quad \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}
\end{align*}
\]
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

**Definition**

Sei $G = (V, E)$ Graph.

- $\alpha(G) = \max\{ |V'| ; \ V' \subseteq V \land \forall a, b \in V' : (a, b) \notin E \}$
- $\omega(G) = \max\{ |V'| ; \ V' \subseteq V \land \forall a, b \in V' : (a, b) \in E \}$
- $\chi(G) = \min\{ k ; \ \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \land \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}$
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

Definition

Sei $G = (V, E)$ Graph.

\[
\alpha(G) = \max\{|V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \notin E\}
\]
\[
\omega(G) = \max\{|V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \in E\}
\]
\[
\chi(G) = \min\{k ; \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \land \\
\forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E\}
\]
**Definition of Coloring**

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.

- The mapping $f$ is called coloring of $G$.

- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff

- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

**Definition**

Sei $G = (V, E)$ Graph.

- $\alpha(G) = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \notin E \}$
- $\omega(G) = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \in E \}$
- $\chi(G) = \min\{ k ; \ \exists V_1, V_2, \ldots, V_k : \cup_{i=1}^{k} V_i = V \land \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}$
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

### Definition

Sei $G = (V, E)$ Graph.

\[
\begin{align*}
\alpha(G) & = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \notin E \} \\
\omega(G) & = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \in E \} \\
\chi(G) & = \min\{ k ; \ \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \land \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}
\end{align*}
\]
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

**Definition**

Sei $G = (V, E)$ Graph.

$$
\begin{align*}
\alpha(G) &= \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \notin E \} \\
\omega(G) &= \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \in E \} \\
\chi(G) &= \min\{ k ; \ \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^k V_i = V \land \\
&\quad \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}
\end{align*}
$$
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
- $\exists f : V \mapsto \{1, ..., k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

Definition

Sei $G = (V, E)$ Graph.

$$\alpha(G) = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \notin E \}$$

$$\omega(G) = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \in E \}$$

$$\chi(G) = \min\{ k ; \ \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \land \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}$$
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, ..., k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

Definition

Sei $G = (V, E)$ Graph.

$$\alpha(G) = \max\{ |V'| ; V' \subset V \land \forall a, b \in V' : (a, b) \notin E \}$$

$$\omega(G) = \max\{ |V'| ; V' \subset V \land \forall a, b \in V' : (a, b) \in E \}$$

$$\chi(G) = \min\{ k ; \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \land \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}$$
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called the coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

**Definition**

Sei $G = (V, E)$ Graph.

\[
\begin{align*}
\alpha(G) & = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \notin E \} \\
\omega(G) & = \max\{ |V'| ; \ V' \subset V \land \forall a, b \in V' : (a, b) \in E \} \\
\chi(G) & = \min\{ k ; \ \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \land \\
& \quad \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}
\end{align*}
\]
Definition of Coloring

- A graph $G = (V, E)$ is $k$-colorable iff:
  - $\exists f : V \mapsto \{1, \ldots, k\} : \forall (a, b) \in E, f(a) \neq f(b)$.
- The mapping $f$ is called coloring of $G$.
- $\chi(G)$ is the chromatic number $\chi(G)$ of $G$, iff
- $G$ is $\chi(G)$-colorable, but $G$ is not $(\chi(G) - 1)$-colorable.

Definition

Sei $G = (V, E)$ Graph.

\[
\alpha(G) = \max\{ |V'| \mid V' \subset V \land \forall a, b \in V' : (a, b) \notin E \} \\
\omega(G) = \max\{ |V'| \mid V' \subset V \land \forall a, b \in V' : (a, b) \in E \} \\
\chi(G) = \min\{ k \mid \exists V_1, V_2, \ldots, V_k : \bigcup_{i=1}^{k} V_i = V \land \forall i : 1 \leq i \leq k : \forall a, b \in V_i : (a, b) \notin E \}
\]
Line-Graphs

Definition (Line-Graphs)

Let $G = (V, E)$ be an undirected graph. $L(G) = (E, E')$ is called line-graph of $G$, iff

$$E' = \{(e, e') \mid e, e' \in E \land e \cap e' \neq \emptyset\}.$$ 

A graph $H$ is called line-graph, iff a graph $G$ exists, with $L(G) = H$. 

\[ a \quad b \quad c \]
Line-Graphs

Definition (Line-Graphs)

Let $G = (V, E)$ be an undirected graph. $L(G) = (E, E')$ is called line-graph of $G$, iff

$$E' = \{(e, e') \mid e, e' \in E \land e \cap e' \neq \emptyset\}.$$ 

A graph $H$ is called line-graph, iff a graph $G$ exists, with $L(G) = H$. 
Definition (Line-Graphs)

Let \( G = (V, E) \) be an undirected graph. \( L(G) = (E, E') \) is called line-graph of \( G \), iff

\[
E' = \{(e, e') \mid e, e' \in E \land e \cap e' \neq \emptyset\}.
\]

A graph \( H \) is called line-graph, iff a graph \( G \) exists, with \( L(G) = H \).
Example 1
Example 1
Example 1
Example 1
Example 1
Example 2

\[ \chi(G) \]

Line-Graph and Coloring (3:5.1)

Walter Unger 6.1.2015 17:05 WS2014/15
Example 2

\[
\begin{array}{c}
\text{a} & \text{b} & \text{c} & \text{d} \\
\text{ab} & \text{bc} & \text{cd} & \text{da}
\end{array}
\]
Example 2
Example 2
Example 2
Example 3
Example 3

\[ \chi(G) \]
Example 3
Example 3
Example 3
Edge-Colouring I

**Definition**

The Edge-Colouring-Problem for a graph $G$ corresponds to the node-colouring of $L(G)$:

$$\chi'(G) = \chi(L(G)).$$

**Theorem (Vizing 1965)**

$$\chi'(K_{2n}) = 2n - 1 \text{ and } \chi'(K_{2n+1}) = 2n + 1.$$ 

**Theorem**

$$\chi'(G) \geq \omega(L(G)) \geq \Delta(G).$$
Introduction

Hardness

Algorithms

Colour with Greed

Brooks

Girth

Colouring $\chi(G)$

Complexity

Edge-Colouring I

Definition

The Edge-Colouring-Problem for a graph $G$ corresponds to the node-colouring of $L(G)$:

$$\chi'(G) = \chi(L(G)).$$

Theorem (Vizing 1965)

$$\chi'(K_{2n}) = 2n - 1 \text{ and } \chi'(K_{2n+1}) = 2n + 1.$$ 

Theorem

$$\chi'(G) \geq \omega(L(G)) \geq \Delta(G).$$

$$\Delta(G) = \max_{v \in V(G)}\{\deg(v)\}$$
**Definition**

The Edge-Colouring-Problem for a graph $G$ corresponds to the node-colouring of $L(G)$:

$$
\chi'(G) = \chi(L(G)).
$$

**Theorem (Vizing 1965)**

$$
\chi'(K_{2n}) = 2n - 1 \text{ and } \chi'(K_{2n+1}) = 2n + 1.
$$

**Theorem**

$$
\chi'(G) \geq \omega(L(G)) \geq \Delta(G).
$$
Edge-Colouring II

Theorem (Holyer)

*The d-Edge-Colouring-Problem is NP-complete for \( d \geq 3 \).*

Theorem (König 1916)

*Any bipartite graph with degree \( \Delta \) is \( \Delta \) edge-colourable (Running-Time \( O(nm) \)).*

Theorem (Vizing 1964)

*Any graph with degree \( \Delta \) is \( \Delta + 1 \) edge-colourable (Running-Time \( O(nm) \)).*
**Theorem (Holyer)**

The d-Edge-Colouring-Problem is NP-complete for $d \geq 3$.

**Theorem (König 1916)**

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

**Theorem (Vizing 1964)**

Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).
**Theorem (Holyer)**

The $d$-Edge-Colouring-Problem is NP-complete for $d \geq 3$.

**Theorem (König 1916)**

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

**Theorem (Vizing 1964)**

Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).
Proof I (Holyer)

- This component assembles a negation.
  - W.l.o.g. \((a, b)\) and \((h, i)\) are coloured the same and
  - \((c, d), (j, k), (g, l)\) use three different colours.

- We will use this to represent variables and
- will use an odd cycle to represent the clauses.
This component assembles a negation.

- W.l.o.g. $(a, b)$ and $(h, i)$ are coloured the same and
- $(c, d), (j, k), (g, l)$ use three different colours.

We will use this to represent variables and
will use an odd cycle to represent the clauses.
Proof I (Holyer)

- This component assembles a negation.
  - W.l.o.g. \((a, b)\) and \((h, i)\) are coloured the same and
  - \((c, d), (j, k), (g, l)\) use three different colours.

- We will use this to represent variables and
- will use an odd cycle to represent the clauses.
Proof I (Holyer)

- This component assembles a negation.
  - W.l.o.g. \((a, b)\) and \((h, i)\) are coloured the same and
  - \((c, d), (j, k), (g, l)\) use three different colours.
- **We will use this to represent variables and**
- **will use an odd cycle to represent the clauses.**
This component assembles a negation.

- W.l.o.g. $(a, b)$ and $(h, i)$ are coloured the same and
- $(c, d), (j, k), (g, l)$ use three different colours.

We will use this to represent variables and
- will use an odd cycle to represent the clauses.
This component assembles a negation.

- W.l.o.g. \((a, b)\) and \((h, i)\) are coloured the same and
- \((c, d), (j, k), (g, l)\) use three different colours.

We will use this to represent variables and
will use an odd cycle to represent the clauses.
1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

- The colour \((i, e)\) and \((i, j)\) and show in the following:
- \((a, b)\) and \((h, i)\) are coloured the same and
- \((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

- In a same way we may proof:
- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
Proof II (Holyer)

1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

   The colour \((i, e)\) and \((i, j)\) and show in the following:

   \((a, b)\) and \((h, i)\) are coloured the same and

   \((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

   In a same way we may proof:

   \((c, d)\) and \((j, k)\) are coloured the same and

   \((a, b), (h, i), (g, l)\) use three different colours.
Proof II (Holyer)

1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

   The colour \((i, e)\) and \((i, j)\) and show in the following:

   \((a, b)\) and \((h, i)\) are coloured the same and

   \((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

   In a same way we may proof:
   \((c, d)\) and \((j, k)\) are coloured the same and

   \((a, b), (h, i), (g, l)\) use three different colours.
1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

The colour \((i, e)\) and \((i, j)\) and show in the following:

\((a, b)\) and \((h, i)\) are coloured the same and

\((c, d)\), \((j, k)\), \((g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

In a same way we may proof:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b)\), \((h, i)\), \((g, l)\) use three different colours.
Proof II (Holyer)

1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

The colour \((i, e)\) and \((i, j)\) and show in the following:

\((a, b)\) and \((h, i)\) are coloured the same and

\((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

In a same way we may proof:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b), (h, i), (g, l)\) use three different colours.
1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

The colour \((i, e)\) and \((i, j)\) and show in the following:

\((a, b)\) and \((h, i)\) are coloured the same and

\((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

In a same way we may proof:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b), (h, i), (g, l)\) use three different colours.
Proof II (Holyer)

1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

The colour \((i, e)\) and \((i, j)\) and show in the following:

\((a, b)\) and \((h, i)\) are coloured the same and

\((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

In a same way we may proof:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b), (h, i), (g, l)\) use three different colours.
Proof II (Holyer)

1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

The colour \((i, e)\) and \((i, j)\) and show in the following:

\((a, b)\) and \((h, i)\) are coloured the same and

\((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

In a same way we may proof:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b), (h, i), (g, l)\) use three different colours.
Proof II (Holyer)

1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

   The colour \((i, e)\) and \((i, j)\) and show in the following:

   \((a, b)\) and \((h, i)\) are coloured the same and

   \((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

   In a same way we may proof:

   \((c, d)\) and \((j, k)\) are coloured the same and

   \((a, b), (h, i), (g, l)\) use three different colours.
Proof II (Holyer)

1. Case: $(h, i)$ and $(l, g)$ are coloured equal.

The colour $(i, e)$ and $(i, j)$ and show in the following:

$(a, b)$ and $(h, i)$ are coloured the same and

$(c, d), (j, k), (g, l)$ use three different colours.

2. Case: $(j, k)$ and $(l, g)$ are coloured the same.

In a same way we may proof:

$(c, d)$ and $(j, k)$ are coloured the same and

$(a, b), (h, i), (g, l)$ use three different colours.
**Proof II (Holyer)**

1. **Case:** $(h, i)$ and $(l, g)$ are coloured equal.

   The colour $(i, e)$ and $(i, j)$ and show in the following:

   $(a, b)$ and $(h, i)$ are coloured the same and

   $(c, d), (j, k), (g, l)$ use three different colours.

2. **Case:** $(j, k)$ and $(l, g)$ are coloured the same.

   In a same way we may proof:

   $(c, d)$ and $(j, k)$ are coloured the same and

   $(a, b), (h, i), (g, l)$ use three different colours.
1. Case: \((h, i)\) and \((l, g)\) are coloured equal.

The colour \((i, e)\) and \((i, j)\) and show in the following:

\((a, b)\) and \((h, i)\) are coloured the same and

\((c, d), (j, k), (g, l)\) use three different colours.

2. Case: \((j, k)\) and \((l, g)\) are coloured the same.

In a same way we may proof:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b), (h, i), (g, l)\) use three different colours.
1. Case: $(h, i)$ and $(l, g)$ are coloured equal.

The colour $(i, e)$ and $(i, j)$ and show in the following:

$(a, b)$ and $(h, i)$ are coloured the same and

$(c, d), (j, k), (g, l)$ use three different colours.

2. Case: $(j, k)$ and $(l, g)$ are coloured the same.

In a same way we may proof:

$(c, d)$ and $(j, k)$ are coloured the same and

$(a, b), (h, i), (g, l)$ use three different colours.
3. **Case:** \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use an other colour.

**Case 3a:** \((i, j)\) has the same colour as \((l, g)\)

Show in the following:

This case does not happen.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use an other colour.

Case 3a: \((i, j)\) has the same colour as \((l, g)\)

Show in the following:

This case does not happen.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use an other colour.

Case 3a: \((i, j)\) has the same colour as \((l, g)\)

Show in the following:

This case does not happen.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use another colour.

Case 3a: \((i, j)\) has the same colour as \((l, g)\)

Show in the following:

This case does not happen.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use an other colour.

Case 3a: \((i, j)\) has the same colour as \((l, g)\)

Show in the following:

This case does not happen.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use an other colour.

Case 3a: \((i, j)\) has the same colour as \((l, g)\)

Show in the following:

This case does not happen.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use another colour.

Case 3b: \((i, j)\) use the third colour.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use another colour.

Case 3b: \((i, j)\) use the third colour.

Show in the following:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b), (h, i), (g, l)\) use three different colours.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use an other colour.

Case 3b: \((i, j)\) use the third colour.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use an other colour.

Case 3b: \((i, j)\) use the third colour.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h,i), (g, l)\) use three different colours.
Proof IV (Holyer)

3. Case: $(h, i)$ and $(j, k)$ are coloured the same and $(l, g)$ use an other colour.

Case 3b: $(i, j)$ use the third colour.

Show in the following:

$(c, d)$ and $(j, k)$ are coloured the same and

$(a, b), (h, i), (g, l)$ use three different colours.
Proof IV (Holyer)

3. Case: \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use another colour.

Case 3b: \((i, j)\) use the third colour.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
3. **Case:** \((h, i)\) and \((j, k)\) are coloured the same and \((l, g)\) use another colour.

**Case 3b:** \((i, j)\) use the third colour.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
3. Case: $(h, i)$ and $(j, k)$ are coloured the same and $(l, g)$ use an other colour.

Case 3b: $(i, j)$ use the third colour.

Show in the following:

$(c, d)$ and $(j, k)$ are coloured the same and

$(a, b), (h, i), (g, l)$ use three different colours.
4. Case: \((h, i), (j, k)\) and \((l, g)\) are coloured with three different colours.

Show in the following:

\((c, d)\) and \((j, k)\) are coloured the same and

\((a, b), (h, i), (g, l)\) use three different colours.
Proof V (Holyer)

- **4. Case:** \((h, i), (j, k)\) and \((l, g)\) are coloured with three different colours.

- **Show in the following:**
  - \((c, d)\) and \((j, k)\) are coloured the same and
  - \((a, b), (h, i), (g, l)\) use three different colours.
Proof V (Hoyer)

4. Case: \((h, i), (j, k)\) and \((l, g)\) are coloured with three different colours.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
4. Case: \((h, i), (j, k)\) and \((l, g)\) are coloured with three different colours.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
Proof V (Holyer)

4. Case: \((h, i), (j, k)\) and \((l, g)\) are coloured with three different colours.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
Proof V (Holyer)

- **4. Case:** \((h, i), (j, k)\) and \((l, g)\) are coloured with three different colours.

- Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and

- \((a, b), (h, i), (g, l)\) use three different colours.
Proof V (Hoyer)

4. Case: \((h, i), (j, k)\) and \((l, g)\) are coloured with three different colours.

Show in the following:

- \((c, d)\) and \((j, k)\) are coloured the same and
- \((a, b), (h, i), (g, l)\) use three different colours.
Proof VI (Holyer)

- We will now merge two of these construction to create a more powerful one.

- This new construction has three "Exits" (pairs of dedicated edges).

- An exit has the value "false" iff both edges are colours the same (otherwise "true").

- For this new component we have:
  - If the left [or right] exit is "false", then all exits are "false".
  - If the left [right] exit is "true", then the right [left] exit is "true".
We will now merge two of these construction to create a more powerful one.

This new construction has three “Exits” (pairs of dedicated edges).

An exit has the value “false” iff both edges are colours the same (otherwise “true”).

For this new component we have:

- If the left [or right] exit is “false”, then all exits are “false”.
- If the left [right] exit is “true”, then the right [left] exit is “true”.

\[ \chi(G) \]
Proof VI (Holyer)

- We will now merge two of these construction to create a more powerful one.

- This new construction has three “Exits” (pairs of dedicated edges).

- An exit has the value “false” iff both edges are colours the same (otherwise “true”).

- For this new component we have:
  - If the left [or right] exit is “false”, then all exits are “false”.
  - If the left [right] exit is “true”, then the right [left] exit is “true”.

![Diagram of the new construction with labeled vertices and edges representing the exits and their conditions.](insert_diagram)
Proof VI (Holyer)

- We will now merge two of these construction to create a more powerful one.

- This new construction has three “Exits” (pairs of dedicated edges).

- An exit has the value “false” iff both edges are colours the same (otherwise “true”).

- For this new component we have:
  
  - If the left [or right] exit is “false”, then all exits are “false”.
  
  - If the left [right] exit is “true”, then the right [left] exit is “true”.
Proof VI (Hoyer)

- We will now merge two of these construction to create a more powerful one.
- This new construction has three “Exits” (pairs of dedicated edges).
- An exit has the value “false” iff both edges are colours the same (otherwise “true”).
- For this new component we have:
  - If the left [or right] exit is “false”, then all exits are “false”.
  - If the left [right] exit is “true”, then the right [left] exit is “true”.

\[
\chi(G)
\]
We will now merge two of these constructions to create a more powerful one.

This new construction has three “Exits” (pairs of dedicated edges).

An exit has the value “false” iff both edges are colours the same (otherwise “true”).

For this new component we have:

- If the left [or right] exit is “false”, then all exits are “false”.
- If the left [right] exit is “true”, then the right [left] exit is “true”.

![Graph diagram with labeled vertices and edges]

\[
\chi(G) = \text{Complexity}
\]
Proof VI (Hoyer)

- We will now merge two of these construction to create a more powerful one.
- This new construction has three “Exits” (pairs of dedicated edges).
- An exit has the value “false” iff both edges are colours the same (otherwise “true”).
- For this new component we have:
  - If the left [or right] exit is “false”, then all exits are “false”.
  - If the left [right] exit is “true”, then the right [left] exit is “true”.

![Diagram of a graph with labeled nodes a to t and edges connecting them. The graph is structured with a central node connected to multiple branches, each with labeled nodes and edges indicating the exits with true and false values.](image-url)
We will now merge two of these construction to create a more powerful one.

This new construction has three “Exits” (pairs of dedicated edges).

An exit has the value “false” iff both edges are colours the same (otherwise “true”).

For this new component we have:

- If the left [or right] exit is “false”, then all exits are “false”.
- If the left [right] exit is “true”, then the right [left] exit is “true”.

![Diagram of the new construction with exits and edges labeled a to t.](image)
Proof VI (Holyer)

- We combine now at least three components in a cyclic way, to represent a variable.
- This component has at least three “Exits” (pairs of dedicated edges).
- For this component holds:
- All exits have the same logical value.
Proof VI (Holyer)

- We combine now at least three components in a cyclic way, to represent a variable.
- This component has at least three “Exits” (pairs of dedicated edges).
- For this component holds:
  - All exits have the same logical value.
Proof VI (Holyer)

- We combine now at least three components in a cyclic way, to represent a variable.
- This component has at least three “Exits” (pairs of dedicated edges).
- For this component holds:
  - All exits have the same logical value.
Proof VI (Holyer)

- We combine now at least three components in a cyclic way, to represent a variable.
- This component has at least three “Exits” (pairs of dedicated edges).
- For this component holds:
  - All exits have the same logical value.
Proof VI (Holyer)

- We combine now at least three components in a cyclic way, to represent a variable.
- This component has at least three “Exits” (pairs of dedicated edges).
- For this component holds:
- All exits have the same logical value.
Proof VII (Holey)

- To verify a clause the exits [may be after an additional negation] of the corresponding literals are joined with an odd cycle.
- For this component we have:
- If all exits have the value "false", then we need four colours.
Proof VII (Holyer)

- To verify a clause the exits [may be after an additional negation] of the corresponding literals are joined with an odd cycle.

- For this component we have:

- If all exits have the value “false”, then we need four colours.
To verify a clause the exits [may be after an additional negation] of the corresponding literals are joined with an odd cycle.

For this component we have:

- If all exits have the value “false”, then we need four colours.
Proof VII (Holyer)

- To verify a clause the exits [may be after an additional negation] of the corresponding literals are joined with an odd cycle.
- For this component we have:
- If all exits have the value “false”, then we need four colours.
Proof (König)

Theorem (König)

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
Proof (König)

**Theorem (König)**

*Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).*

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
Proof (König)

Theorem (König)

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
**Theorem (König)**

> Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.

- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
Proof (König)

Theorem (König)

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
Theorem (König)

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
Theorem (König)

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
Proof (König)

**Theorem (König)**

*Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).*

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- **Running-Time**: store for each node and colour the corresponding edge.
Proof (König)

**Theorem (König)**

Any bipartite graph with degree $\Delta$ is $\Delta$ edge-colourable (Running-Time $O(nm)$).

- Show how to colour an edge $(a, b)$ in $O(n)$ time.
- Let $c_a, c_b$ be the unused colours at the nodes $a, b$.
- If $c_a = c_b$, we may colour $(a, b)$ with $c_a$.
- Observe now the graph $H_{a,b}$, who consists only of edges coloured with $c_a, c_b$.
- $H_{a,b}$ consists of a disjoined set of paths and cycles.
- $a$ and $b$ are the endpoints of two different paths.
- Thus we may exchange the colours of one path.
- Running-Time: store for each node and colour the corresponding edge.
Proof (Vizing)

**Theorem (Vizing)**

Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).

- Proof by induction on the number of edges.
- Let $\Delta = \Delta(G)$ and $e = (x, y) \in E$.
- For $G - e$ exists an edge colouring $c : E \setminus \{e\} \mapsto \{1, 2, \cdots, \Delta + 1\}$.
- Note: At each node are $\Delta + 1 - \deg(v) \geq 1$ colours free.
- For $v \in V$ let $F_v$ be the set of free colours.
- If $F_x \cap F_y \neq \emptyset$ holds we may colour $(x, y)$.
- So assume for the following: $F_x \cap F_y = \emptyset$
Proof (Vizing)

**Theorem (Vizing)**

Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).

- Proof by induction on the number of edges.
- Let $\Delta = \Delta(G)$ and $e = (x, y) \in E$.
- For $G - e$ exists an edge colouring $c : E \setminus \{e\} \mapsto \{1, 2, \cdots, \Delta + 1\}$.
- Note: At each node are $\Delta + 1 - \deg(v) \geq 1$ colours free.
- For $v \in V$ let $F_v$ be the set of free colours.
- If $F_x \cap F_y \neq \emptyset$ holds we may colour $(x, y)$.
- So assume for the following: $F_x \cap F_y = \emptyset$
Proof (Vizing)

Theorem (Vizing)

Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).

Proof

- Proof by induction on the number of edges.
- Let $\Delta = \Delta(G)$ and $e = (x, y) \in E$.
- For $G - e$ exists an edge colouring $c : E \setminus \{e\} \mapsto \{1, 2, \cdots, \Delta + 1\}$.
- Note: At each node are $\Delta + 1 - \deg(v) \geq 1$ colours free.
- For $v \in V$ let $F_v$ be the set of free colours.
- If $F_x \cap F_y \neq \emptyset$ holds we may colour $(x, y)$.
- So assume for the following: $F_x \cap F_y = \emptyset$
Proof (Vizing)

**Theorem (Vizing)**

*Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).*

- Proof by induction on the number of edges.
- Let $\Delta = \Delta(G)$ and $e = (x, y) \in E$.
- For $G - e$ exists an edge colouring $c : E \setminus \{e\} \mapsto \{1, 2, \cdots, \Delta + 1\}$.
- **Note:** At each node are $\Delta + 1 - \deg(v) \geq 1$ colours free.
- For $v \in V$ let $F_v$ be the set of free colours.
- If $F_x \cap F_y \neq \emptyset$ holds we may colour $(x, y)$.
- So assume for the following: $F_x \cap F_y = \emptyset$. 
**Proof (Vizing)**

\[ \Delta(G) = \max_{v \in V(G)} \{\deg(v)\} \]

**Theorem (Vizing)**

*Any graph with degree \( \Delta \) is \( \Delta + 1 \) edge-colourable (Running-Time \( O(nm) \)).*

- Proof by induction on the number of edges.
- Let \( \Delta = \Delta(G) \) and \( e = (x, y) \in E \).
- For \( G - e \) exists an edge colouring \( c : E \setminus \{e\} \mapsto \{1, 2, \ldots, \Delta + 1\} \).
- Note: At each node are \( \Delta + 1 - \deg(v) \geq 1 \) colours free.
- For \( v \in V \) let \( F_v \) be the set of free colours.
- If \( F_x \cap F_y \neq \emptyset \) holds we may colour \((x, y)\).
- So assume for the following: \( F_x \cap F_y = \emptyset \)
Proof (Vizing)

Theorem (Vizing)

Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).

- Proof by induction on the number of edges.
- Let $\Delta = \Delta(G)$ and $e = (x, y) \in E$.
- For $G - e$ exists an edge colouring $c : E \setminus \{e\} \mapsto \{1, 2, \ldots, \Delta + 1\}$.
- Note: At each node are $\Delta + 1 - \deg(v) \geq 1$ colours free.
- For $v \in V$ let $F_v$ be the set of free colours.
- If $F_x \cap F_y \neq \emptyset$ holds we may colour $(x, y)$.
- So assume for the following: $F_x \cap F_y = \emptyset$
Proof (Vizing)

**Theorem (Vizing)**

*Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).*

- Proof by induction on the number of edges.
- Let $\Delta = \Delta(G)$ and $e = (x, y) \in E$.
- For $G - e$ exists an edge colouring $c : E \setminus \{e\} \mapsto \{1, 2, \ldots, \Delta + 1\}$.
- Note: At each node are $\Delta + 1 - \deg(v) \geq 1$ colours free.
- For $v \in V$ let $F_v$ be the set of free colours.
- If $F_x \cap F_y \neq \emptyset$ holds we may colour $(x, y)$.
- So assume for the following: $F_x \cap F_y = \emptyset$
Proof (Vizing)

**Theorem (Vizing)**

*Any graph with degree $\Delta$ is $\Delta + 1$ edge-colourable (Running-Time $O(nm)$).*

- Proof by induction on the number of edges.
- Let $\Delta = \Delta(G)$ and $e = (x, y) \in E$.
- For $G - e$ exists an edge colouring $c : E \setminus \{e\} \mapsto \{1, 2, \cdots, \Delta + 1\}$.
- Note: At each node are $\Delta + 1 - \deg(v) \geq 1$ colours free.
- For $v \in V$ let $F_v$ be the set of free colours.
- If $F_x \cap F_y \neq \emptyset$ holds we may colour $(x, y)$.
- So assume for the following: $F_x \cap F_y = \emptyset$
Proof I (Vizing)

- Construct a sequence \(\{y_1, y_2, \ldots, y_k\}\) of neighbours of \(x\) and \(\{b_1, b_2, \ldots, b_k\}\) of colours with:
  - \(y_1 = y\) and
  - \(b_j \in F_{y_j}\) and
  - \(c((x, y_{j+1})) = b_j\) and
  - \(\{y_1, y_2, \ldots, y_k\}\) are different.

- If in round \(k\) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \(f \in F_x \cap F_{y_k}\) with \(f \not\in \{b_1, b_2, \ldots, b_{k-1}\}\).

- Then do the following:
  - \(c((x, y_k)) = f\)
  - \(c((x, y_i)) = b_i\) for \(1 \leq i < k\).

- We call this operation \(\text{Shift}(k, f)\).
**Proof I (Vizing)**

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \( (x, y_k) \) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \ldots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \ldots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \ldots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \( (x, y_k) \) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \ldots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:

- \( y_1 = y \) and
- \( b_j \in F_{y_j} \) and
- \( c((x, y_{j+1})) = b_j \) and
- \( \{y_1, y_2, \cdots, y_k\} \) are different.

If in round \( k \) the following hold:

- The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

Then do the following:

- \( c((x, y_k)) = f \)
- \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \ldots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \ldots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \ldots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \ldots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \ldots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \ldots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \ldots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \( (x, y_k) \) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \ldots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \notin \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof of Vizing (3:21.9)

**Proof I (Vizing)**

- Construct a sequence \(\{y_1, y_2, \cdots, y_k\}\) of neighbours of \(x\) and \(\{b_1, b_2, \cdots, b_k\}\) of colours with:
  - \(y_1 = y\) and
  - \(b_j \in F_{y_j}\) and
  - \(c((x, y_{j+1})) = b_j\) and
  - \(\{y_1, y_2, \cdots, y_k\}\) are different.

- If in round \(k\) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \(f \in F_x \cap F_{y_k}\) with \(f \not\in \{b_1, b_2, \cdots, b_{k-1}\}\).

- Then do the following:
  - \(c((x, y_k)) = f\)
  - \(c((x, y_i)) = b_i\) for \(1 \leq i < k\).

We call this operation \(\text{Shift}(k, f)\).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c(((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c(((x, y_k)) = f \)
  - \( c(((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \notin \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
**Proof I (Vizing)**

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \ldots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \ldots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \ldots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \ldots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
**Proof I (Vizing)**

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:

- \( y_1 = y \) and
- \( b_j \in F_{y_j} \) and
- \( c((x, y_{j+1})) = b_j \) and
- \( \{y_1, y_2, \cdots, y_k\} \) are different.

If in round \( k \) the following hold:

- The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

Then do the following:

- \( c((x, y_k)) = f \)
- \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \(\{y_1, y_2, \ldots, y_k\}\) of neighbours of \(x\) and \(\{b_1, b_2, \ldots, b_k\}\) of colours with:
  - \(y_1 = y\) and
  - \(b_j \in F_{y_j}\) and
  - \(c((x, y_{j+1})) = b_j\) and
  - \(\{y_1, y_2, \ldots, y_k\}\) are different.

- If in round \(k\) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \(f \in F_x \cap F_{y_k}\) with \(f \not\in \{b_1, b_2, \ldots, b_{k-1}\}\).

- Then do the following:
  - \(c((x, y_k)) = f\)
  - \(c((x, y_i)) = b_i\) for \(1 \leq i < k\).

- We call this operation \(\text{Shift}(k, f)\).
Proof I (Vizing)

Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
- \( y_1 = y \) and
- \( b_j \in F_{y_j} \) and
- \( c((x, y_{j+1})) = b_j \) and
- \( \{y_1, y_2, \cdots, y_k\} \) are different.

If in round \( k \) the following hold:

- The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

Then do the following:
- \( c((x, y_k)) = f \)
- \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \(\{y_1, y_2, \cdots, y_k\}\) of neighbours of \(x\) and \(\{b_1, b_2, \cdots, b_k\}\) of colours with:
  - \(y_1 = y\) and
  - \(b_j \in F_{y_j}\) and
  - \(c((x, y_{j+1})) = b_j\) and
  - \(\{y_1, y_2, \cdots, y_k\}\) are different.

- If in round \(k\) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \(f \in F_x \cap F_{y_k}\) with \(f \not\in \{b_1, b_2, \cdots, b_{k-1}\}\).

- Then do the following:
  - \(c((x, y_k)) = f\)
  - \(c((x, y_i)) = b_i\) for \(1 \leq i < k\).

- We call this operation \(\text{Shift}(k, f)\).
Proof I (Vizing)

Construct a sequence \( \{y_1, y_2, \ldots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \ldots, b_k\} \) of colours with:

- \( y_1 = y \) and
- \( b_j \in F_{y_j} \) and
- \( c(\langle x, y_{j+1} \rangle) = b_j \) and
- \( \{y_1, y_2, \ldots, y_k\} \) are different.

If in round \( k \) the following hold:

- The edge \( \langle x, y_k \rangle \) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \ldots, b_{k-1}\} \).

Then do the following:

- \( c(\langle x, y_k \rangle) = f \)
- \( c(\langle x, y_i \rangle) = b_i \) for \( 1 \leq i < k \).

We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:

- \( y_1 = y \) and
- \( b_j \in F_{y_j} \) and
- \( c((x, y_{j+1})) = b_j \) and
- \( \{y_1, y_2, \cdots, y_k\} \) are different.

If in round \( k \) the following hold:

- The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

Then do the following:

- \( c((x, y_k)) = f \)
- \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

  Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

  We call this operation \textit{Shift}(k, f).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \ldots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \ldots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \ldots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \( (x, y_k) \) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \ldots, b_{k-1}\} \).
  
  - Then do the following:
    - \( c((x, y_k)) = f \)
    - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof 1 (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x,y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x,y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x,y_k)) = f \)
  - \( c((x,y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \textit{Shift}(k, f).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \cdots, b_{k-1}\} \).

- Then do the following:
  - \( c((x, y_k)) = f \)
  - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

- We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

- Construct a sequence \( \{y_1, y_2, \cdots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \cdots, b_k\} \) of colours with:
  - \( y_1 = y \) and
  - \( b_j \in F_{y_j} \) and
  - \( c((x, y_{j+1})) = b_j \) and
  - \( \{y_1, y_2, \cdots, y_k\} \) are different.

- If in round \( k \) the following hold:
  - The edge \( (x, y_k) \) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \notin \{b_1, b_2, \cdots, b_{k-1}\} \).
  - Then do the following:
    - \( c((x, y_k)) = f \)
    - \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).
  - We call this operation \( \text{Shift}(k, f) \).
Proof I (Vizing)

Construct a sequence \( \{y_1, y_2, \ldots, y_k\} \) of neighbours of \( x \) and \( \{b_1, b_2, \ldots, b_k\} \) of colours with:

- \( y_1 = y \) and
- \( b_j \in F_{y_j} \) and
- \( c((x, y_{j+1})) = b_j \) and
- \( \{y_1, y_2, \ldots, y_k\} \) are different.

If in round \( k \) the following hold:

- The edge \((x, y_k)\) could be recoloured to colour \( f \in F_x \cap F_{y_k} \) with \( f \not\in \{b_1, b_2, \ldots, b_{k-1}\} \).

Then do the following:

- \( c((x, y_k)) = f \)
- \( c((x, y_i)) = b_i \) for \( 1 \leq i < k \).

We call this operation \( \text{Shift}(k, f) \).
Proof II (Vizing)

- We will now construct such a sequence.
- What happens if the recolouring is not possible.
- Then we have: \( y_{k+1} \in \{y_1, y_2, \ldots, y_k\} \),
- I.e. \( y_{k+1} = y_i \) and \( b_k = b_{i-1} \).
- Then we have \( i \neq 1 \) and \( i \neq k \).
- Let \( a \in F_x \).
- Consider \( H(a, b_k) \); the subgraph using the colours \( a \) and \( b_k \).
- In each component of \( H(a, b_k) \) the colours may be exchanged.
- At the node \( y_k \) starts a path \( P \) of \( H(a, b_k) \).
- Let \( z \) be the other endpoint of path \( P \).
We will now construct such a sequence.

What happens if the recolouring is not possible.

Then we have: $y_{k+1} \in \{y_1, y_2, \ldots, y_k\}$,
I.e. $y_{k+1} = y_i$ and $b_k = b_{i-1}$.
Then we have $i \neq 1$ and $i \neq k$.
Let $a \in F_x$.
Consider $H(a, b_k)$; the subgraph using the colours $a$ and $b_k$.
In each component of $H(a, b_k)$ the colours may be exchanged.
At the node $y_k$ starts a path $P$ of $H(a, b_k)$.
Let $z$ be the other endpoint of path $P$. 

edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
Proof II (Vizing)

- We will now construct such a sequence.
- What happens if the recolouring is not possible.
- Then we have: \( y_{k+1} \in \{ y_1, y_2, \ldots, y_k \} \),
- I.e. \( y_{k+1} = y_i \) and \( b_k = b_{i-1} \).
- Then we have \( i \neq 1 \) and \( i \neq k \).
- Let \( a \in F_x \).
- Consider \( H(a, b_k) \); the subgraph using the colours \( a \) and \( b_k \).
- In each component of \( H(a, b_k) \) the colours may be exchanged.
- At the node \( y_k \) starts a path \( P \) of \( H(a, b_k) \).
- Let \( z \) be the other endpoint of path \( P \).
Proof II (Vizing)

We will now construct such a sequence.

What happens if the recolouring is not possible.

Then we have: $y_{k+1} \in \{y_1, y_2, \ldots, y_k\}$,

l.e. $y_{k+1} = y_i$ and $b_k = b_{i-1}$.

Then we have $i \neq 1$ and $i \neq k$.

Let $a \in F_x$.

Consider $H(a, b_k)$; the subgraph using the colours $a$ and $b_k$.

In each component of $H(a, b_k)$ the colours may be exchanged.

At the node $y_k$ starts a path $P$ of $H(a, b_k)$.

Let $z$ be the other endpoint of path $P$. 

$$\text{edge-sequence } (y_1, \ldots, y_k) \ y_1 = y, \ b_j \in F_{y_j}, \ c((x, y_{j+1})) = b_j$$
Proof II (Vizing)

We will now construct such a sequence.

What happens if the recolouring is not possible.

Then we have: \( y_{k+1} \in \{y_1, y_2, \ldots, y_k\} \),

I.e. \( y_{k+1} = y_i \) and \( b_k = b_{i-1} \).

Then we have \( i \neq 1 \) and \( i \neq k \).

Let \( a \in F_x \).

Consider \( H(a, b_k) \); the subgraph using the colours \( a \) and \( b_k \).

In each component of \( H(a, b_k) \) the colours may be exchanged.

At the node \( y_k \) starts a path \( P \) of \( H(a, b_k) \).

Let \( z \) be the other endpoint of path \( P \).
We will now construct such a sequence.

What happens if the recolouring is not possible.

Then we have: $y_{k+1} \in \{y_1, y_2, \ldots, y_k\}$,

i.e. $y_{k+1} = y_i$ and $b_k = b_{i-1}$.

Then we have $i \neq 1$ and $i \neq k$.

Let $a \in F_x$.

Consider $H(a, b_k)$; the subgraph using the colours $a$ and $b_k$.

In each component of $H(a, b_k)$ the colours may be exchanged.

At the node $y_k$ starts a path $P$ of $H(a, b_k)$.

Let $z$ be the other endpoint of path $P$. 

edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
We will now construct such a sequence.

What happens if the recolouring is not possible.

Then we have: \( y_{k+1} \in \{y_1, y_2, \ldots, y_k\} \),

i.e. \( y_{k+1} = y_i \) and \( b_k = b_{i-1} \).

Then we have \( i \neq 1 \) and \( i \neq k \).

Let \( a \in F_x \).

Consider \( H(a, b_k) \); the subgraph using the colours \( a \) and \( b_k \).

In each component of \( H(a, b_k) \) the colours may be exchanged.

At the node \( y_k \) starts a path \( P \) of \( H(a, b_k) \).

Let \( z \) be the other endpoint of path \( P \).
Proof II (Vizing)

edge-sequence \((y_1, \ldots, y_k)\) \(y_1 = y\), \(b_j \in F_{y_j}\), \(c((x, y_{j+1})) = b_j\)

- We will now construct such a sequence.
- What happens if the recolouring is not possible.
- Then we have: \(y_{k+1} \in \{y_1, y_2, \ldots, y_k\}\),
- i.e. \(y_{k+1} = y_i\) and \(b_k = b_{i-1}\).
- Then we have \(i \neq 1\) and \(i \neq k\).
- Let \(a \in F_x\).
- Consider \(H(a, b_k)\); the subgraph using the colours \(a\) and \(b_k\).
- In each component of \(H(a, b_k)\) the colours may be exchanged.
- At the node \(y_k\) starts a path \(P\) of \(H(a, b_k)\).
- Let \(z\) be the other endpoint of path \(P\).
We will now construct such a sequence.

What happens if the recolouring is not possible.

Then we have: \( y_{k+1} \in \{y_1, y_2, \ldots, y_k \} \),

i.e. \( y_{k+1} = y_i \) and \( b_k = b_{i-1} \).

Then we have \( i \neq 1 \) and \( i \neq k \).

Let \( a \in F_x \).

Consider \( H(a, b_k) \); the subgraph using the colours \( a \) and \( b_k \).

In each component of \( H(a, b_k) \) the colours may be exchanged.

At the node \( y_k \) starts a path \( P \) of \( H(a, b_k) \).

Let \( z \) be the other endpoint of path \( P \).
Proof II (Vizing)

- We will now construct such a sequence.
- What happens if the recolouring is not possible.
- Then we have: \( y_{k+1} \in \{y_1, y_2, \ldots, y_k\} \),
- I.e. \( y_{k+1} = y_i \) and \( b_k = b_{i-1} \).
- Then we have \( i \neq 1 \) and \( i \neq k \).
- Let \( a \in F_x \).
- Consider \( H(a, b_k) \); the subgraph using the colours \( a \) and \( b_k \).
- In each component of \( H(a, b_k) \) the colours may be exchanged.
- At the node \( y_k \) starts a path \( P \) of \( H(a, b_k) \).
- Let \( z \) be the other endpoint of path \( P \).
We will now construct such a sequence.

What happens if the recolouring is not possible.

Then we have: \( y_{k+1} \in \{y_1, y_2, \ldots, y_k\} \),

I.e. \( y_{k+1} = y_i \) and \( b_k = b_{i-1} \).

Then we have \( i \neq 1 \) and \( i \neq k \).

Let \( a \in F_x \).

Consider \( H(a, b_k) \); the subgraph using the colours \( a \) and \( b_k \).

In each component of \( H(a, b_k) \) the colours may be exchanged.

At the node \( y_k \) starts a path \( P \) of \( H(a, b_k) \).

Let \( z \) be the other endpoint of path \( P \).
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ $(1 \leq j \leq k)$ with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \notin (x, y_{i-1})$. i.e. $z \notin \{y_1, y_2, \ldots, y_k\}$

\[
\text{edge-sequence } (y_1, \ldots, y_k) \ y_1 = y, \ b_j \in F_{y_j}, \ c((x, y_{j+1})) = b_j
\]
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ ($1 \leq j \leq k$) with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \notin (x, y_{i-1})$. I.e. $z \notin \{y_1, y_2, \ldots, y_k\}$

(edge-sequence $(y_1, \ldots, y_k)$, $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$)
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ ($1 \leq j \leq k$) with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \not\in (x, y_{i-1})$. I.e. $z \not\in \{y_1, y_2, \ldots, y_k\}$

edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ $(1 \leq j \leq k)$ with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \not\in (x, y_{i-1})$. I.e. $z \not\in \{y_1, y_2, \ldots, y_k\}$
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ $(1 \leq j \leq k)$
- with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \not\in (x, y_{i-1})$. I.e. $z \not\in \{y_1, y_2, \ldots, y_k\}$

---

**Graph**

- Edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
Proof III (Vizing)

Recall $a \in F_x$.

Recall $b_k \in F_{y_{i-1}}$.

Note $P$ contains no edges of the form $(x, y_j)$ ($1 \leq j \leq k$) with the exception of $(x, y_i)$.

If $z = x$ holds, we also have $(x, y_i)$ in $P$.

We will now consider the following cases:

- $z = y_{i-1}$
- $z = x$
- $z \notin (x, y_{i-1})$. I.e. $z \notin \{y_1, y_2, \ldots, y_k\}$

edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ $(1 \leq j \leq k)$
  with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \notin (x, y_{i-1})$. I.e. $z \notin \{y_1, y_2, \ldots, y_k\}$

edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ ($1 \leq j \leq k$) with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \notin (x, y_{i-1})$. I.e. $z \notin \{y_1, y_2, \ldots, y_k\}$

- edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ ($1 \leq j \leq k$) with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.

We will now consider the following cases:

- $z = y_{i-1}$
- $z = x$
- $z \notin (x, y_{i-1})$. I.e. $z \notin \{y_1, y_2, \ldots, y_k\}$

edge-sequence $(y_1, \ldots, y_k)$ $y_1 = y$, $b_j \in F_{y_j}$, $c((x, y_{j+1})) = b_j$
Proof III (Vizing)

- Recall $a \in F_x$.
- Recall $b_k \in F_{y_{i-1}}$.
- Note $P$ contains no edges of the form $(x, y_j)$ $(1 \leq j \leq k)$ with the exception of $(x, y_i)$.
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- We will now consider the following cases:
  - $z = y_{i-1}$
  - $z = x$
  - $z \not\in (x, y_{i-1})$. I.e. $z \not\in \{y_1, y_2, \ldots, y_k\}$

\[
\text{edge-sequence } (y_1, \ldots, y_k) \ y_1 = y, \ b_j \in F_{y_j}, \ c((x, y_{j+1})) = b_j
\]
Proof IIIa (Vizing)

- Note: \( a \in F_x, b_k \in F_{y_{i-1}} \) and
- \( P \) contains no edges of the form \((x, y_j)\) \((j \in \{1, \ldots, k\} \setminus \{i\})\)
- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

Case: \( z = y_{i-1} \)

- Both edges at the ends of \( P \) are coloured with \( a \).
- Exchange the colours on \( P \).
- After this, the colour \( a \) is not used at \( y_{i-1} \).
- Do \textit{Shift}(i - 1, a) as the final step.
Proof IIIa (Vizing)

- Note: $a \in F_x$, $b_k \in F_{y_{i-1}}$ and
- $P$ contains no edges of the form $(x, y_j)$ ($j \in \{1, \ldots, k\} \setminus \{i\}$)
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.

Case: $z = y_{i-1}$

- Both edges at the ends of $P$ are coloured with $a$.
- Exchange the colours on $P$.
- After this, the colour $a$ is not used at $y_{i-1}$.
- Do $\text{Shift}(i - 1, a)$ as the final step.

\[\text{edge-sequence } (y_1, \ldots, y_k) \ y_1 = y, \ b_j \in F_{y_j}, \ c((x, y_{j+1})) = b_j\]
Proof IIIa (Vizing)

- Note: \( a \in F_x, b_k \in F_{y_{i-1}} \) and
- \( P \) contains no edges of the form \((x, y_j)\) \((j \in \{1, \ldots, k\} \setminus \{i\})\)
- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

Case: \( z = y_{i-1} \)

- Both edges at the ends of \( P \) are coloured with \( a \).
- Exchange the colours on \( P \).
- After this, the colour \( a \) is not used at \( y_{i-1} \).
- Do \textit{Shift}(i - 1, a) as the final step.
Proof IIIa (Vizing)

- Note: $a \in F_x$, $b_k \in F_{y_{i-1}}$ and
- $P$ contains no edges of the form $(x, y_j)$ ($j \in \{1, \ldots, k\} \setminus \{i\}$)
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.
- Case: $z = y_{i-1}$
  - Both edges at the ends of $P$ are coloured with $a$.
  - Exchange the colours on $P$.
  - After this, the colour $a$ is not used at $y_{i-1}$.
  - Do $\text{Shift}(i-1, a)$ as the final step.
Proof IIIa (Vizing)

- **Note:** $a \in F_x$, $b_k \in F_{y_{i-1}}$ and $P$ contains no edges of the form $(x, y_j)$ ($j \in \{1, \ldots, k\} \setminus \{i\}$).
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.

**Case:** $z = y_{i-1}$

- Both edges at the ends of $P$ are coloured with $a$.
- Exchange the colours on $P$.
- After this, the colour $a$ is not used at $y_{i-1}$.
- Do $\text{Shift}(i - 1, a)$ as the final step.
Proof IIIa (Vizing)

- Note: \( a \in F_x, b_k \in F_{y_i-1} \) and
- \( P \) contains no edges of the form \((x, y_j)\) \((j \in \{1, \ldots, k\} \setminus \{i\})\)
- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

Case: \( z = y_i-1 \)

- Both edges at the ends of \( P \) are coloured with \( a \).
- Exchange the colours on \( P \).
- After this, the colour \( a \) is not used at \( y_i-1 \).
- Do \( \text{Shift}(i-1, a) \) as the final step.
Proof IIIb (Vizing)

- Note: \( a \in F_x, b_k \in F_{y_{i-1}} \) and
- \( P \) contains no edges of the form \((x, y_j)\) \((j \in \{1, \ldots, k\} \setminus \{i\})\)
- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

- Case: \( z = x \)
  - Exchange the colour on \( P \).
  - Then the colour \( b_k = b_{i-1} \) is not used at \( x \).
  - Do \( \text{Shift}(i - 1, b_{i-1}) \) as the final step.
Proof IIIb (Vizing)

- Note: $a \in F_x$, $b_k \in F_{y_{i-1}}$ and
  - $P$ contains no edges of the form $(x, y_j)$ $(j \in \{1, \ldots, k\} \setminus \{i\}$)
  - If $z = x$ holds, we also have $(x, y_i)$ in $P$.

- Case: $z = x$
  - Exchange the colour on $P$.
  - Then the colour $b_k = b_{i-1}$ is not used at $x$.
  - Do $\text{Shift}(i - 1, b_{i-1})$ as the final step.
Proof IIIb (Vizing)

- Note: \( a \in F_x, b_k \in F_{y_{i-1}} \) and
- \( P \) contains no edges of the form \((x, y_j)\) \((j \in \{1, \ldots, k\} \setminus \{i\})\)
- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

- Case: \( z = x \)
  - Exchange the colour on \( P \).
  - Then the colour \( b_k = b_{i-1} \) is not used at \( x \).
  - Do \( \text{Shift}(i - 1, b_{i-1}) \) as the final step.
Proof IIIb (Vizing)

- Note: \( a \in F_x, b_k \in F_{y_{i-1}} \) and
- \( P \) contains no edges of the form \((x, y_j)\) \((j \in \{1, \ldots, k\})\)
- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

Case: \( z = x \)
- Exchange the colour on \( P \).
- Then the colour \( b_k = b_{i-1} \) is not used at \( x \).
- Do \textit{Shift}(i - 1, b_{i-1}) as the final step.
Proof IIIb (Vizing)

Note: $a \in F_x$, $b_k \in F_{y_i-1}$ and

$P$ contains no edges of the form $(x, y_j)$ ($j \in \{1, \ldots, k\}$)

If $z = x$ holds, we also have $(x, y_i)$ in $P$.

Case: $z = x$

- Exchange the colour on $P$.
- Then the colour $b_k = b_{i-1}$ is not used at $x$.
- Do $\text{Shift}(i - 1, b_{i-1})$ as the final step.
Proof IIIc (Vizing)

- Note: $a \in F_x$, $b_k \in F_{y_{i-1}}$ and
- $P$ contains no edges of the form $(x, y_j)$ ($j \in \{1, \ldots, k\} \setminus \{i\}$)
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.

Case: $z \notin (x, y_{i-1})$

- Exchange the colours on the path $P$ (if there are edges).
- Then the colour $a$ is not used at $y_k$.
- Do $\text{Shift}(k, a)$ as the last step.
Proof IIIc (Vizing)

- Note: $a \in F_x$, $b_k \in F_{y_{i-1}}$ and
- $P$ contains no edges of the form $(x, y_j)$ $(j \in \{1, \ldots, k\} \setminus \{i\})$
- If $z = x$ holds, we also have $(x, y_i)$ in $P$.

Case: $z \notin (x, y_{i-1})$

- Exchange the colours on the path $P$ (if there are edges).
- Then the colour $a$ is not used at $y_k$.
- Do $Shift(k, a)$ as the last step.
Proof IIIc (Vizing)

Note: $a \in F_x$, $b_k \in F_{y_{i-1}}$ and

$P$ contains no edges of the form $(x, y_j)$ ($j \in \{1, \ldots, k\} \setminus \{i\}$)

If $z = x$ holds, we also have $(x, y_i)$ in $P$.

Case: $z \notin (x, y_{i-1})$

- Exchange the colours on the path $P$ (if there are edges).
- Then the colour $a$ is not used at $y_k$.
- Do $Shift(k, a)$ as the last step.
Note: \( a \in F_x, b_k \in F_{y_{i-1}} \) and

- \( P \) contains no edges of the form \((x, y_j)\) 
  \((j \in \{1, \ldots, k\} \setminus \{i\})\)

- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

Case: \( z \notin (x, y_{i-1}) \)

- Exchange the colours on the path \( P \) (if there are edges).
- Then the colour \( a \) is not used at \( y_k \).
- Do \( \text{Shift}(k, a) \) as the last step.
**Proof Ilc (Vizing)**

- **Note:** \( a \in F_x, \ b_k \in F_{y_{i-1}} \) and
- \( P \) contains no edges of the form \((x, y_j)\) \((j \in \{1, \ldots, k\} \setminus \{i\})\)
- If \( z = x \) holds, we also have \((x, y_i)\) in \( P \).

- **Case:** \( z \not\in (x, y_{i-1}) \)
  - Exchange the colours on the path \( P \) (if there are edges).
  - Then the colour \( a \) is not used at \( y_k \).
  - Do \textit{Shift}(k, a) as the last step.
Some Bounds

Note

Let $G = (V, E)$ be a graph. Then the following hold: $\chi(G) \geq \omega(G)$.

Note

Let $G = (V, E)$ be a graph with $|V| = n$. Then we have: $\chi(G) \geq n/\alpha(G)$.

Theorem

Let $G = (V, E)$ be a graph with $|E| = m$. Then: $\chi(G)(\chi(G) - 1) \leq 2m$.

- Let $k = \chi(G)$.
- There exist $k$ independent sets $I_i$ with $i \in \{1, \ldots, k\}$.
- Between $I_i$ and $I_j$ ($i \neq j$) exists at least one edge.
- From which we get $k \cdot (k - 1)/2$ edges in total.
Some Bounds

Note

Let $G = (V, E)$ be a graph. Then the following hold: $\chi(G) \geq \omega(G)$.

Note

Let $G = (V, E)$ be a graph with $|V| = n$. Then we have: $\chi(G) \geq n/\alpha(G)$.

Theorem

Let $G = (V, E)$ be a graph with $|E| = m$. Then: $\chi(G)(\chi(G) - 1) \leq 2m$.

- Let $k = \chi(G)$.
- There exist $k$ independent sets $I_i$ with $i \in \{1, \ldots, k\}$.
- Between $I_i$ and $I_j$ ($i \neq j$) exists at least one edge.
- From which we get $k \cdot (k - 1)/2$ edges in total.
Some Bounds

**Note**

Let $G = (V, E)$ be a graph. Then the following hold: $\chi(G) \geq \omega(G)$.

**Note**

Let $G = (V, E)$ be a graph with $|V| = n$. Then we have: $\chi(G) \geq n/\alpha(G)$.

**Theorem**

Let $G = (V, E)$ be a graph with $|E| = m$. Then: $\chi(G)(\chi(G) - 1) \leq 2m$.

- Let $k = \chi(G)$.
- There exist $k$ independent sets $I_i$ with $i \in \{1, \ldots, k\}$.
- Between $I_i$ and $I_j$ ($i \neq j$) exists at least one edge.
- From which we get $k \cdot (k - 1)/2$ edges in total.
Some Bounds

Note

Let $G = (V, E)$ be a graph. Then the following hold: $\chi(G) \geq \omega(G)$.

Note

Let $G = (V, E)$ be a graph with $|V| = n$. Then we have: $\chi(G) \geq n/\alpha(G)$.

Theorem

Let $G = (V, E)$ be a graph with $|E| = m$. Then: $\chi(G)(\chi(G) - 1) \leq 2m$.

- Let $k = \chi(G)$.
- There exist $k$ independent sets $I_i$ with $i \in \{1, \ldots, k\}$.
- Between $I_i$ and $I_j$ ($i \neq j$) exists at least one edge.
- From which we get $k \cdot (k - 1)/2$ edges in total.
Some Bounds

**Note**

Let $G = (V, E)$ be a graph. Then the following hold: $\chi(G) \geq \omega(G)$.

**Note**

Let $G = (V, E)$ be a graph with $|V| = n$. Then we have: $\chi(G) \geq n/\alpha(G)$.

**Theorem**

Let $G = (V, E)$ be a graph with $|E| = m$. Then: $\chi(G)(\chi(G) - 1) \leq 2m$.

- Let $k = \chi(G)$.
- There exist $k$ independent sets $l_i$ with $i \in \{1, \ldots, k\}$.
- Between $l_i$ and $l_j$ ($i \neq j$) exists at least one edge.
- From which we get $k \cdot (k - 1)/2$ edges in total.
Some Bounds

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $G = (V, E)$ be a graph. Then the following hold: $\chi(G) \geq \omega(G)$.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $G = (V, E)$ be a graph with $</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let $G = (V, E)$ be a graph with $</td>
</tr>
</tbody>
</table>

- Let $k = \chi(G)$.
- There exist $k$ independent sets $I_i$ with $i \in \{1, \ldots, k\}$.
- Between $I_i$ and $I_j$ ($i \neq j$) exists at least one edge.
- From which we get $k \cdot (k - 1)/2$ edges in total.
Some Bounds

Note

Let \( G = (V, E) \) be a graph. Then the following hold: \( \chi(G) \geq \omega(G) \).

Note

Let \( G = (V, E) \) be a graph with \( |V| = n \). Then we have: \( \chi(G) \geq n/\alpha(G) \).

Theorem

Let \( G = (V, E) \) be a graph with \( |E| = m \). Then: \( \chi(G)(\chi(G) - 1) \leq 2m \).

- Let \( k = \chi(G) \).
- There exist \( k \) independent sets \( I_i \) with \( i \in \{1, \ldots, k\} \).
- Between \( I_i \) and \( I_j \) (\( i \neq j \)) exists at least one edge.
- From which we get \( k \cdot (k - 1)/2 \) edges in total.
Let $G = (V, E)$ be a Graph.

Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.

Algorithm: $GreedyColour(G, \sigma)$.

Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.

Colour: $c(v_1) := 1$.

Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$

Number of colours: $GreedyColour(G, \sigma) := |\{c(v) \mid v \in V\}|$.

We have: $\chi(G) \leq GreedyColour(G, \sigma) \leq \Delta(G) + 1$.

For odd cycles and cliques holds:

$\chi(G) = GreedyColour(G, \sigma) = \Delta(G) + 1$.

Running time: $O(|V| + |E|)$
Colour with Greed

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- Algorithm: $GreedyColour(G, \sigma)$.
- Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
- Colour: $c(v_1) := 1$.
- Colour: $c(v_i) := \min \{k \in \mathbb{N} \mid k \neq c(u) \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
- Number of colours: $GreedyColour(G, \sigma) := |\{c(v) \mid v \in V\}|$.
- We have: $\chi(G) \leq GreedyColour(G, \sigma) \leq \Delta(G) + 1$.
- For odd cycles and cliques holds:
  - $\chi(G) = GreedyColour(G, \sigma) = \Delta(G) + 1$.
- Running time: $O(|V| + |E|)$.
Colour with Greed

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- **Algorithm:** $\text{GreedyColour}(G, \sigma)$.
  - Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
  - Colour: $c(v_1) := 1$.
  - Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
  - Number of colours: $\text{GreedyColour}(G, \sigma) := |\{c(v) \mid v \in V\}|$.
  - We have: $\chi(G) \leq \text{GreedyColour}(G, \sigma) \leq \Delta(G) + 1$.
  - For odd cycles and cliques holds:
    - $\chi(G) = \text{GreedyColour}(G, \sigma) = \Delta(G) + 1$.
  - Running time: $O(|V| + |E|)$.
Colour with Greed

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- Algorithm: $\text{GreedyColour}(G, \sigma)$.
- Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
- Colour: $c(v_1) := 1$.
- Colour: $c(v_i) := \min \{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
- Number of colours: $\text{GreedyColour}(G, \sigma) := |\{c(v) \mid v \in V\}|$.
- We have: $\chi(G) \leq \text{GreedyColour}(G, \sigma) \leq \Delta(G) + 1$.
- For odd cycles and cliques holds:
  - $\chi(G) = \text{GreedyColour}(G, \sigma) = \Delta(G) + 1$.
- Running time: $O(|V| + |E|)$
Colour with Greed

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- Algorithm: $\text{GreedyColour}(G, \sigma)$.
- Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
  - Colour: $c(v_1) := 1$.
  - Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \; \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
  - Number of colours: $\text{GreedyColour}(G, \sigma) := |\{c(v) \mid v \in V\}|$.
  - We have: $\chi(G) \leq \text{GreedyColour}(G, \sigma) \leq \Delta(G) + 1$.
  - For odd cycles and cliques holds:
    - $\chi(G) = \text{GreedyColour}(G, \sigma) = \Delta(G) + 1$.
  - Running time: $O(|V| + |E|)$

$$G[W] = (W, \{(a, b) \in E(G) \mid a, b \in W\})$$
Colour with Greed

Let $G = (V, E)$ be a Graph.

Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.

Algorithm: $GreedyColour(G, \sigma)$.

Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.

Colour: $c(v_1) := 1$.

Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$

Number of colours: $GreedyColour(G, \sigma) := |\{c(v) \mid v \in V\}|$.

We have: $\chi(G) \leq GreedyColour(G, \sigma) \leq \Delta(G) + 1$.

For odd cycles and cliques holds:

$\chi(G) = GreedyColour(G, \sigma) = \Delta(G) + 1$.

Running time: $O(|V| + |E|)$
Colour with Greed

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- Algorithm: $\text{GreedyColour}(G, \sigma)$.
- Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
- Colour: $c(v_1) := 1$.
- Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
- Number of colours: $\text{GreedyColour}(G, \sigma) := |\{c(v) \mid v \in V\}|$.
- We have: $\chi(G) \leq \text{GreedyColour}(G, \sigma) \leq \Delta(G) + 1$.
- For odd cycles and cliques holds:
  - $\chi(G) = \text{GreedyColour}(G, \sigma) = \Delta(G) + 1$.
- Running time: $O(|V| + |E|)$
Colour with Greed

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- Algorithm: $GreedyColour(G, \sigma)$.
- Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
- Colour: $c(v_1) := 1$.
- Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
- Number of colours: $GreedyColour(G, \sigma) := |\{c(v) \mid v \in V\}|$.
- We have: $\chi(G) \leq GreedyColour(G, \sigma) \leq \Delta(G) + 1$.
- For odd cycles and cliques holds:
  - $\chi(G) = GreedyColour(G, \sigma) = \Delta(G) + 1$.
- Running time: $O(|V| + |E|)$
Colour with Greed

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- Algorithm: $\text{GreedyColour}(G, \sigma)$.
- Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
- Colour: $c(v_1) := 1$.
- Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
- Number of colours: $\text{GreedyColour}(G, \sigma) := |\{c(v) \mid v \in V\}|$.
- We have: $\chi(G) \leq \text{GreedyColour}(G, \sigma) \leq \Delta(G) + 1$.
- For odd cycles and cliques holds:
  - $\chi(G) = \text{GreedyColour}(G, \sigma) = \Delta(G) + 1$.
- Running time: $O(|V| + |E|)$
Colour with Greed

Let $G = (V, E)$ be a Graph.

Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.

Algorithm: $\text{GreedyColour}(G, \sigma)$.

Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.

Colour: $c(v_1) := 1$.

Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$

Number of colours: $\text{GreedyColour}(G, \sigma) := |\{c(v) \mid v \in V\}|$.

We have: $\chi(G) \leq \text{GreedyColour}(G, \sigma) \leq \Delta(G) + 1$.

For odd cycles and cliques holds:

- $\chi(G) = \text{GreedyColour}(G, \sigma) = \Delta(G) + 1$.

Running time: $O(|V| + |E|)$
**Colour with Greed**

- Let $G = (V, E)$ be a Graph.
- Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.
- Algorithm: $\text{GreedyColour}(G, \sigma)$.
- Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.
- Colour: $c(v_1) := 1$.
- Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$
- Number of colours: $\text{GreedyColour}(G, \sigma) := |\{c(v) \mid v \in V\}|$.
- We have: $\chi(G) \leq \text{GreedyColour}(G, \sigma) \leq \Delta(G) + 1$.
- For odd cycles and cliques holds:
  - $\chi(G) = \text{GreedyColour}(G, \sigma) = \Delta(G) + 1$.

Running time: $O(|V| + |E|)$
Colour with Greed

Let $G = (V, E)$ be a Graph.

Choose an ordering of the nodes: $\sigma = (v_1, v_2, \ldots, v_n)$.

Algorithm: $GreedyColour(G, \sigma)$.

Let $V_i = \{v_1, v_2, \ldots, v_i\}$ and $G_i = G[V_i]$.

Colour: $c(v_1) := 1$.

Colour: $c(v_i) := \min\{k \in \mathbb{N} \mid k \neq c(u) \ \forall u \in \Gamma(v_i) \cap V_{i-1}\}$

Number of colours: $GreedyColour(G, \sigma) := |\{c(v) \mid v \in V\}|$.

We have: $\chi(G) \leq GreedyColour(G, \sigma) \leq \Delta(G) + 1$.

For odd cycles and cliques holds:

$\chi(G) = GreedyColour(G, \sigma) = \Delta(G) + 1$.

Running time: $O(|V| + |E|)$
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} | v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{graph.png}
\end{figure}
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.
2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

   Note:
   - $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
   - $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
   - But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.
2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $GreedyColour(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $GreedyColour(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

   Note:
   - $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
   - $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
   - But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} | v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.
2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:

- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 
Analysis of the Error

1. Extreme case: $K_{1,\Delta}$.

2. Extreme case: $B_n$:
   - $B_n = (V_n, W_n, E_n)$
   - $V_n = \{v_1, v_2, v_3, \ldots, v_n\}$
   - $W_n = \{w_1, w_2, w_3, \ldots, w_n\}$
   - $E_n = \{\{v_i, w_j\} \mid v_i \in V_n, w_j \in W_n, i \neq j\}$

Note:
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n))$.
- $\text{GreedyColour}(B_n, (v_1, w_1, v_2, w_2, v_3, w_3, \ldots, v_n, w_n)) = n$.
- But $\chi(B_n) = 2$. 

\[ \chi(G) \]
**Theorem**

Let \( \varepsilon, \delta > 0 \) and \( c < 1 \).

- For large enough \( n \) exists graphs \( G_n \) with:
  - \( \chi(G_n) \leq n^\varepsilon \) and
  - on \( o(n^{-\delta}) \) orderings Greedy will use \( c \cdot n / \log n \) colours.

**Lemma**

There is an ordering \( \sigma^* \) with: \( \text{GreedyColour}(G, \sigma^*) = \chi(G) \).

**Lemma**

\[
\min_{\sigma \in S_n} \text{GreedyColour}(G, \sigma) = \chi(G) \text{ hold.}
\]
**Error-Analysis**

### Theorem
- *Let* $\varepsilon, \delta > 0$ *and* $c < 1$.
- *For large enough* $n$ *exists graphs* $G_n$ *with:*
  - $\chi(G_n) \leq n^\varepsilon$ *and*
  - *on* $o(n^{-\delta})$ *orderings Greedy will use* $c \cdot n / \log n$ *colours.*

### Lemma
*There is an ordering* $\sigma^*$ *with:* $\text{GreedyColour}(G, \sigma^*) = \chi(G)$.

### Lemma
$\min_{\sigma \in S_n} \text{GreedyColour}(G, \sigma) = \chi(G)$ *hold.*
**Theorem**

- Let $\varepsilon, \delta > 0$ and $c < 1$.
- For large enough $n$ exists graphs $G_n$ with:
  - $\chi(G_n) \leq n^\varepsilon$ and
  - on $o(n^{-\delta})$ orderings Greedy will use $c \cdot n / \log n$ colours.

**Lemma**

There is an ordering $\sigma^*$ with: $\text{GreedyColour}(G, \sigma^*) = \chi(G)$.

**Lemma**

$\min_{\sigma \in S_n} \text{GreedyColour}(G, \sigma) = \chi(G)$ hold.
Theorem

- Let $\varepsilon, \delta > 0$ and $c < 1$.
- For large enough $n$ exists graphs $G_n$ with:
  - $\chi(G_n) \leq n^\varepsilon$ and
  - on $o(n^{-\delta})$ orderings Greedy will use $c \cdot n / \log n$ colours.

Lemma

There is an ordering $\sigma^*$ with: $\text{GreedyColour}(G, \sigma^*) = \chi(G)$.

Lemma

$\min_{\sigma \in S_n} \text{GreedyColour}(G, \sigma) = \chi(G)$ hold.
Error-Analysis

**Theorem**

- Let $\varepsilon, \delta > 0$ and $c < 1$.
- For large enough $n$ exists graphs $G_n$ with:
  - $\chi(G_n) \leq n^\varepsilon$ and
  - on $o(n^{-\delta})$ orderings Greedy will use $c \cdot n / \log n$ colours.

**Lemma**

There is an ordering $\sigma^*$ with: $\text{GreedyColour}(G, \sigma^*) = \chi(G)$.  

**Lemma**

$\min_{\sigma \in S_n} \text{GreedyColour}(G, \sigma) = \chi(G)$ hold.
### Error-Analysis

#### Theorem
- Let $\varepsilon, \delta > 0$ and $c < 1$.
- For large enough $n$ exists graphs $G_n$ with:
  - $\chi(G_n) \leq n^\varepsilon$ and
  - on $o(n^{-\delta})$ orderings Greedy will use $c \cdot n / \log n$ colours.

#### Lemma
There is an ordering $\sigma^*$ with: $\text{GreedyColour}(G, \sigma^*) = \chi(G)$.

#### Lemma
$\min_{\sigma \in S_n} \text{GreedyColour}(G, \sigma) = \chi(G)$ hold.
Introduction

Theorems (3:30.7)

Hardness

Algorithms

Colour with Greed

Brooks

Girth

Colouring χ(G)

Complexity

Error-Analysis

Theorem

- Let ε, δ > 0 and c < 1.
- For large enough n exists graphs G_n with:
  - χ(G_n) ≤ n^ε and
  - on o(n^{-δ}) orderings Greedy will use c · n/ log n colours.

Lemma

There is an ordering σ^* with: GreedyColour(G, σ^*) = χ(G).

Lemma

min_{σ ∈ S_n} GreedyColour(G, σ) = χ(G) hold.
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.
- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.
- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$
- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.
- Such an ordering is called: “smallest-last”
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.
- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.
- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$
- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.
- Such an ordering is called: “smallest-last”
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.
- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.
- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$
- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.
- Such an ordering is called: “smallest-last”
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.

- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.

- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$

- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.

- Such an ordering is called: “smallest-last”
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.
- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.
- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$
- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.
- Such an ordering is called: “smallest-last”
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.
- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.
- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$
- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.

Such an ordering is called: “smallest-last”
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.
- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.
- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$
- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.
- Such an ordering is called: “smallest-last”
Improvements

- Note: for $v_i$ are at most $d_{G_i}(v_i)$ colours unusable.
- Let $b(\sigma) = \max_{1 \leq i \leq n} d_{G_i}(v_i)$ with $\sigma = (v_1, v_2, \ldots, v_n)$.
- $\chi(G) \leq \min_{\sigma \in S_n} b(\sigma)$
- The ordering $\sigma$ which gives the minimum is constructable.
  - Choose $v_n$ with the minimal degree.
  - Recursively compute the ordering on $G - v_n$.
- Such an ordering is called: “smallest-last”
Lemma

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subseteq G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

Proof

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subseteq G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subseteq G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subseteq G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j)$
  - Furthermore: $\max_{H \subseteq G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 

Application

Lemma

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subset G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

Proof

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subset G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subset G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subset G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j)$
  - Furthermore: $\max_{H \subset G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 
Lemma

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subset G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

Proof

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subset G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subset G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subset G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j)$
  - Furthermore: $\max_{H \subset G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 
Lemma

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subseteq G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

Proof

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subseteq G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subseteq G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subseteq G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j)$
  - Furthermore: $\max_{H \subseteq G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 
Application

Lemma

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subseteq G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

Proof

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subseteq G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subseteq G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subseteq G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j)$
  - Furthermore: $\max_{H \subseteq G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 
Application

Lemma

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subseteq G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

Proof

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subseteq G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subseteq G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subseteq G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j) \leq d_{G_j}(v_j)$
  - Furthermore: $\max_{H \subseteq G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 

**Lemma**

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subseteq G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

**Proof**

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subseteq G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subseteq G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subseteq G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j) \leq d_{G_j}(v_j) \leq b(\sigma)$
  - Furthermore: $\max_{H \subseteq G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$.
**Application**

**Lemma**

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subseteq G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

**Proof**

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subseteq G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subseteq G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subseteq G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j) \leq d_{G_j}(v_j) \leq b(\sigma)$
  - Furthermore: $\max_{H \subseteq G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 
**Lemma**

Let $\sigma_{sl}$ be a smallest-last ordering. Then we have:

$$b(\sigma_{sl}) = \max_{H \subseteq G} \delta(H) = \min_{\sigma \in S_n} b(\sigma)$$

**Proof**

- $b(\sigma_{sl}) \leq \max_i \delta(G_i) \leq \max_{H \subseteq G} \delta(H)$
- Let $H^*$ be a subgraph of $G$ with: $\delta(H^*) = \max_{H \subseteq G} \delta(H)$.
- Let $j$ be the smallest index with: $H^*$ is a subgraph of $G_j$ for some permutation $\sigma$. Then we get:
  - $\max_{H \subseteq G} \delta(H) = \delta(H^*) \leq d_{H^*}(v_j) \leq d_{G_j}(v_j) \leq b(\sigma)$
  - Furthermore: $\max_{H \subseteq G} \delta(H) \leq \min_{\sigma \in S_n} b(\sigma)$.
  - The claim follows by: $\min_{\sigma \in S_n} b(\sigma) \leq b(\sigma_{sl})$. 
Lemma

Let $G = (V, E)$ and $\sigma_{sl}$ smallest-last ordering. Then the following hold:

$$\chi(G) \leq \text{GreedyColour}(G, \sigma_{sl}) \leq 1 + \max_{H \subset G} \delta(H)$$

Running Time: $O(|V| + |E|)$. 
**Lemma**

Let $G = (V, E)$ and $\sigma_{sl}$ smallest-last ordering. Then the following hold:

$$
\chi(G) \leq \text{GreedyColour}(G, \sigma_{sl}) \leq 1 + \max_{H \subseteq G} \delta(H)
$$

Running Time: $O(|V| + |E|)$.
Implications II

Lemma

Let $G = (V, E)$ connected and not $\Delta(G)$-regular. Then $\chi(G) \leq \Delta(G)$ holds.

- Let $v_1$ a node with $d(v_1) < \Delta(G)$.
- Choose ordering $\sigma = (v_1, v_2, v_3, \ldots, v_n)$ by breadth-first-search from $v_1$.
- Call $\text{GreedyColour}(G, \sigma^{-1})$. Then the following hold:
  - $d(v_1) < \Delta(G)$, d.h. $c(v_1) \leq \Delta(G)$
  - $v_i$ has a non-coloured neighbour, thus $c(v_i) \leq \Delta(G)$ holds.
Lemma

Let $G = (V, E)$ connected and not $\Delta(G)$-regular. Then $\chi(G) \leq \Delta(G)$ holds.

- Let $v_1$ a node with $d(v_1) < \Delta(G)$.
- Choose ordering $\sigma = (v_1, v_2, v_3, \ldots, v_n)$ by breadth-first-search from $v_1$.
- Call $\text{GreedyColour}(G, \sigma^{-1})$. Then the following hold:
  - $d(v_1) < \Delta(G)$, d.h. $c(v_1) \leq \Delta(G)$
  - $v_i$ has a non-coloured neighbour, thus $c(v_i) \leq \Delta(G)$ holds.
Implications II

Lemma

Let \( G = (V, E) \) connected and not \( \Delta(G) \)-regular. Then \( \chi(G) \leq \Delta(G) \) holds.

- Let \( v_1 \) a node with \( d(v_1) < \Delta(G) \).
- Choose ordering \( \sigma = (v_1, v_2, v_3, \ldots, v_n) \) by breadth-first-search from \( v_1 \).
- Call \textit{GreedyColour}(G, \sigma^{-1})\). Then the following hold:
  - \( d(v_1) < \Delta(G) \), d.h. \( c(v_1) \leq \Delta(G) \)
  - \( v_i \) has a non-coloured neighbour, thus \( c(v_i) \leq \Delta(G) \) holds.
Implications II

Lemma

Let $G = (V, E)$ connected and not $\Delta(G)$-regular. Then $\chi(G) \leq \Delta(G)$ holds.

- Let $v_1$ a node with $d(v_1) < \Delta(G)$.
- Choose ordering $\sigma = (v_1, v_2, v_3, \ldots, v_n)$ by breadth-first-search from $v_1$.
- Call GreedyColour($G, \sigma^{-1}$). Then the following hold:
  - $d(v_1) < \Delta(G)$, d.h. $c(v_1) \leq \Delta(G)$
  - $v_i$ has a non-coloured neighbour, thus $c(v_i) \leq \Delta(G)$ holds.
Implications II

**Lemma**

Let $G = (V, E)$ connected and not $\Delta(G)$-regular. Then $\chi(G) \leq \Delta(G)$ holds.

- Let $v_1$ a node with $d(v_1) < \Delta(G)$.
- Choose ordering $\sigma = (v_1, v_2, v_3, \ldots, v_n)$ by breadth-first-search from $v_1$.
- Call $\text{GreedyColour}(G, \sigma^{-1})$. Then the following hold:
  - $d(v_1) < \Delta(G)$, d.h. $c(v_1) \leq \Delta(G)$
  - $v_i$ has a non-coloured neighbour, thus $c(v_i) \leq \Delta(G)$ holds.
Lemma

Let $G = (V, E)$ connected and not $\Delta(G)$-regular. Then $\chi(G) \leq \Delta(G)$ holds.

- Let $v_1$ a node with $d(v_1) < \Delta(G)$.
- Choose ordering $\sigma = (v_1, v_2, v_3, \ldots, v_n)$ by breadth-first-search from $v_1$.
- Call $\text{GreedyColour}(G, \sigma^{-1})$. Then the following hold:
  - $d(v_1) < \Delta(G)$, d.h. $c(v_1) \leq \Delta(G)$
  - $v_i$ has a non-coloured neighbour, thus $c(v_i) \leq \Delta(G)$ holds.
Implications II

**Lemma**

*Let $G = (V, E)$ connected and not $\Delta(G)$-regular. Then $\chi(G) \leq \Delta(G)$ holds.*

- Let $v_1$ a node with $d(v_1) < \Delta(G)$.
- Choose ordering $\sigma = (v_1, v_2, v_3, \ldots, v_n)$ by breadth-first-search from $v_1$.
- Call *GreedyColour*$(G, \sigma^{-1})$. Then the following hold:
  - $d(v_1) < \Delta(G)$, d.h. $c(v_1) \leq \Delta(G)$
  - $v_i$ has a non-coloured neighbour, thus $c(v_i) \leq \Delta(G)$ holds.
Theorem (Brooks 1941)

Let \( G = (V, E) \) be a connected Graph with at least three nodes. Let \( G \) be no clique nor an odd cycle. Then the following holds:

\[
\chi(G) \leq \Delta(G)
\]

- If \( G \) is not two-connected, consider block \( B \):
  - If \( B \) is regular, then \( B \) is not \( \Delta(G) \)-regular.
  - If \( B \) is not regular, colour the graph using the above algorithm.
  - In both cases we use at most \( \Delta(G) \) colours.

- If \( G \) two-connected and not regular, then colour again using the above algorithm

- If \( G \) two-connected and regular, continue as follows:
Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected, consider block $B$:
  - If $B$ is regular, then $B$ is not $\Delta(G)$-regular.
  - If $B$ is not regular, colour the graph using the above algorithm.
  - In both cases we use at most $\Delta(G)$ colours.

- If $G$ two-connected and not regular, then colour again using the above algorithm.

- If $G$ two-connected and regular, continue as follows:
Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected, consider block $B$:
  - If $B$ is regular, then $B$ is not $\Delta(G)$-regular.
  - If $B$ is not regular, colour the graph using the above algorithm.
  - In both cases we use at most $\Delta(G)$ colours.

- If $G$ two-connected and not regular, then colour again using the above algorithm.

- If $G$ two-connected and regular, continue as follows:
Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected, consider block $B$:
  - If $B$ is regular, then $B$ is not $\Delta(G)$-regular.
  - If $B$ is not regular, colour the graph using the above algorithm.
  - In both cases we use at most $\Delta(G)$ colours.

- If $G$ two-connected and not regular, then colour again using the above algorithm

- If $G$ two-connected and regular, continue as follows:
Theorem (Brooks 1941)

Let \( G = (V, E) \) be a connected Graph with at least three nodes. Let \( G \) be no clique nor an odd cycle. Then the following holds:

\[
\chi(G) \leq \Delta(G)
\]

- If \( G \) is not two-connected, consider block \( B \):
  - If \( B \) is regular, then \( B \) is not \( \Delta(G) \)-regular.
  - If \( B \) is not regular, colour the graph using the above algorithm.
  - In both cases we use at most \( \Delta(G) \) colours.

- If \( G \) two-connected and not regular, then colour again using the above algorithm

- If \( G \) two-connected and regular, continue as follows:
Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected, consider block $B$:
  - If $B$ is regular, then $B$ is not $\Delta(G)$-regular.
  - If $B$ is not regular, colour the graph using the above algorithm.
  - In both cases we use at most $\Delta(G)$ colours.

- If $G$ two-connected and not regular, then colour again using the above algorithm

- If $G$ two-connected and regular, continue as follows:
Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected, consider block $B$:
  - If $B$ is regular, then $B$ is not $\Delta(G)$-regular.
  - If $B$ is not regular, colour the graph using the above algorithm.
  - In both cases we use at most $\Delta(G)$ colours.

- If $G$ two-connected and not regular, then colour again using the above algorithm

- If $G$ two-connected and regular, continue as follows:
Theorem (Brooks 1941)

Let \( G = (V, E) \) be a connected Graph with at least three nodes. Let \( G \) be no clique nor an odd cycle. Then the following holds:

\[
\chi(G) \leq \Delta(G)
\]

- If \( G \) is not two-connected, consider block \( B \):
  - If \( B \) is regular, then \( B \) is not \( \Delta(G) \)-regular.
  - If \( B \) is not regular, colour the graph using the above algorithm.
  - In both cases we use at most \( \Delta(G) \) colours.

- If \( G \) two-connected and not regular, then colour again using the above algorithm

- If \( G \) two-connected and regular, continue as follows:
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours, such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $\text{GreedyColour}(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours,
  - such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $GreedyColour(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1.$
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours, such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $GreedyColour(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours, such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $\text{GreedyColour}(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

1. If $G$ is not two-connected (done)
2. If $G$ is two-connected and not regular: (done)
3. If $G$ is two-connected and regular, then continue:
   - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours,
   - such that $G - \{v_{n-1}, v_n\}$ is still connected.
   - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
   - Colour with $\text{GreedyColour}(G, \sigma^{-1})$.
   - $v_{n-1}$ and $v_n$ get the same colour.
   - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours, such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $GreedyColour(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours,
  - such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $\text{GreedyColour}(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

**Theorem (Brooks 1941)**

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours,
  - such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $\text{GreedyColour}(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours,
  - such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $GreedyColour(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Proof

Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$
\chi(G) \leq \Delta(G)
$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours,
  - such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $\text{GreedyColour}(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Theorem (Brooks 1941)

Let $G = (V, E)$ be a connected Graph with at least three nodes. Let $G$ be no clique nor an odd cycle. Then the following holds:

$$\chi(G) \leq \Delta(G)$$

- If $G$ is not two-connected (done)
- If $G$ is two-connected and not regular: (done)
- If $G$ is two-connected and regular, then continue:
  - Choose $v_1$ with neighbours $v_{n-1}$ and $v_n$, who are neighbours,
  - such that $G - \{v_{n-1}, v_n\}$ is still connected.
  - Compute $v_2, v_3, \ldots, v_{n-2}$ using breadth-first-search from $v_1$ on $G - \{v_{n-1}, v_n\}$.
  - Colour with $\text{GreedyColour}(G, \sigma^{-1})$.
  - $v_{n-1}$ and $v_n$ get the same colour.
  - Thus at most $\Delta(G) - 1$ colours are not usable for $v_1$. 
Implications

Lemma

Let $G = (V, E)$ two-connected, regular with at least three nodes. Let $G$ be no clique nor a cycle. Then there exists $x, y \in V$ with $\text{dist}(x, y) = 2$ and $G - x - y$ is connected.

- Let $v \in V$ with $d(v) = \Delta(G)$.
- Then is $H := G[\{v\} \cup \Gamma(v)]$ not complete.
- Thus there exists $x', y'$ in $\Gamma(v)$ with $\text{dist}(x', y') = 2$.
- If $G - \{x', y'\}$ is connected, we are done!
- If not, is $x', y'$ a minimal separator.
- We have $\Delta(G) \geq 3$ and $d(v) \geq 3$.
- Let $C$ be the component in $G - \{x', y'\}$, which contains $v$. 
Implications

Lemma

Let \( G = (V, E) \) two-connected, regular with at least three nodes. Let \( G \) be no clique nor a cycle. Then there exists \( x, y \in V \) with \( \text{dist}(x, y) = 2 \) and \( G - x - y \) is connected.

- Let \( v \in V \) with \( d(v) = \Delta(G) \).
- Then is \( H := G[\{v\} \cup \Gamma(v)] \) not complete.
- Thus there exists \( x', y' \) in \( \Gamma(v) \) with \( \text{dist}(x', y') = 2 \).
- If \( G - \{x', y'\} \) is connected, we are done!
- If not, is \( x', y' \) a minimal separator.
- We have \( \Delta(G) \geq 3 \) and \( d(v) \geq 3 \).
- Let \( C \) be the component in \( G - \{x', y'\} \), which contains \( v \).
Implications

Lemma

Let $G = (V, E)$ two-connected, regular with at least three nodes. Let $G$ be no clique nor a cycle. Then there exists $x, y \in V$ with $\text{dist}(x, y) = 2$ and $G - x - y$ is connected.

- Let $v \in V$ with $d(v) = \Delta(G)$.
- Then is $H := G[\{v\} \cup \Gamma(v)]$ not complete.
- Thus there exists $x', y'$ in $\Gamma(v)$ with $\text{dist}(x', y') = 2$.
- If $G - \{x', y'\}$ is connected, we are done!
- If not, is $x', y'$ a minimal separator.
- We have $\Delta(G) \geq 3$ and $d(v) \geq 3$.
- Let $C$ be the component in $G - \{x', y'\}$, which contains $v$. 
Implications

Lemma

Let $G = (V, E)$ be two-connected, regular with at least three nodes. Let $G$ be no clique nor a cycle. Then there exists $x, y \in V$ with $\text{dist}(x, y) = 2$ and $G - x - y$ is connected.

- Let $v \in V$ with $d(v) = \Delta(G)$.
- Then is $H := G[\{v\} \cup \Gamma(v)]$ not complete.
- Thus there exists $x', y'$ in $\Gamma(v)$ with $\text{dist}(x', y') = 2$.
- If $G - \{x', y'\}$ is connected, we are done!
- If not, is $x', y'$ a minimal separator.
- We have $\Delta(G) \geq 3$ and $d(v) \geq 3$.
- Let $C$ be the component in $G - \{x', y'\}$, which contains $v$. 
Implications

Lemma

Let $G = (V, E)$ two-connected, regular with at least three nodes. Let $G$ be no clique nor a cycle. Then there exists $x, y \in V$ with $\text{dist}(x, y) = 2$ and $G - x - y$ is connected.

- Let $v \in V$ with $d(v) = \Delta(G)$.
- Then is $H := G\left[\{v\} \cup \Gamma(v)\right]$ not complete.
- Thus there exists $x', y'$ in $\Gamma(v)$ with $\text{dist}(x', y') = 2$.
- If $G - \{x', y'\}$ is connected, we are done!
- If not, is $x', y'$ a minimal separator.
- We have $\Delta(G) \geq 3$ and $d(v) \geq 3$.
- Let $C$ be the component in $G - \{x', y'\}$, which contains $v$. 
**Lemma**

Let \( G = (V, E) \) two-connected, regular with at least three nodes. Let \( G \) be no clique nor a cycle. Then there exists \( x, y \in V \) with \( \text{dist}(x, y) = 2 \) and \( G - x - y \) is connected.

- Let \( v \in V \) with \( d(v) = \Delta(G) \).
- Then is \( H := G[\{v\} \cup \Gamma(v)] \) not complete.
- Thus there exists \( x', y' \) in \( \Gamma(v) \) with \( \text{dist}(x', y') = 2 \).
- If \( G - \{x', y'\} \) is connected, we are done!
- If not, is \( x', y' \) a minimal separator.
- We have \( \Delta(G) \geq 3 \) and \( d(v) \geq 3 \).
- Let \( C \) be the component in \( G - \{x', y'\} \), which contains \( v \).
Implications

Lemma

Let $G = (V, E)$ two-connected, regular with at least three nodes. Let $G$ be no clique nor a cycle. Then there exists $x, y \in V$ with $\text{dist}(x, y) = 2$ and $G - x - y$ is connected.

- Let $v \in V$ with $d(v) = \Delta(G)$.
- Then is $H := G[\{v\} \cup \Gamma(v)]$ not complete.
- Thus there exists $x', y'$ in $\Gamma(v)$ with $\text{dist}(x', y') = 2$.
- If $G - \{x', y'\}$ is connected, we are done!
- If not, is $x', y'$ a minimal separator.
- We have $\Delta(G) \geq 3$ and $d(v) \geq 3$.
- Let $C$ be the component in $G - \{x', y'\}$, which contains $v$. 
Implications

Lemma

Let $G = (V, E)$ two-connected, regular with at least three nodes. Let $G$ be no clique nor a cycle. Then there exists $x, y \in V$ with $\text{dist}(x, y) = 2$ and $G - x - y$ is connected.

- Let $v \in V$ with $d(v) = \Delta(G)$.
- Then is $H := G[{v} \cup \Gamma(v)]$ not complete.
- Thus there exists $x', y'$ in $\Gamma(v)$ with $\text{dist}(x', y') = 2$.
- If $G - \{x', y'\}$ is connected, we are done!
- If not, is $x', y'$ a minimal separator.
- We have $\Delta(G) \geq 3$ and $d(v) \geq 3$.
- Let $C$ be the component in $G - \{x', y'\}$, which contains $v$. 
**Lemma**

Let $G = (V, E)$ two-connected, regular with at least three nodes. Let $G$ be no clique nor a cycle. Then there exists $x, y \in V$ with $\text{dist}(x, y) = 2$ and $G - x - y$ is connected.

- Let $v \in V$ with $d(v) = \Delta(G)$.
- Then is $H := G[\{v\} \cup \Gamma(v)]$ not complete.
- Thus there exists $x', y'$ in $\Gamma(v)$ with $\text{dist}(x', y') = 2$.
- If $G - \{x', y'\}$ is connected, we are done!
- If not, is $x', y'$ a minimal separator.
- We have $\Delta(G) \geq 3$ and $d(v) \geq 3$.
- Let $C$ be the component in $G - \{x', y'\}$, which contains $v$. 
Implications

- There exists $x$ in $C$ with $x$ is neighboured to $x'$ or $y'$.
- This hold for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - Show: Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
- Running time: $O(|V| + |E|)$. 
Implications

- There exists $x$ in $C$ with $x$ is neighboured to $x'$ or $y'$.
- This holds for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - Show: Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
- Running time: $O(|V| + |E|)$. 
Implications

- There exists \( x \) in \( C \) with \( x \) is neighboured to \( x' \) or \( y' \).
- This hold for each component in \( G - \{x', y'\} \).
- Thus there exists \( y \) from some other component with \( \text{dist}(x, y) = 2 \).

We will now show that \( G - \{x, y\} \) is connected.

- \( x' \) and \( y' \) are in \( G - \{x, y\} \) connected.
- Show: Each node in \( G - \{x, y\} \) is connected with \( x' \) or \( y' \).
- \( G - x \) is connected.
- Each node from \( C - x \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( y \).
- \( G - y \) is connected.
- Each node from \( (V \setminus C) - y \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( x \).
- Running time: \( O(|V| + |E|) \).
Implications

- There exists $x$ in $C$ with $x$ is neighboured to $x'$ or $y'$.
- This hold for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - Show: Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
- Running time: $O(|V| + |E|)$.
Implications

- There exists \( x \) in \( C \) with \( x \) is neighboured to \( x' \) or \( y' \).
- This hold for each component in \( G - \{x', y'\} \).
- Thus there exists \( y \) from some other component with \( \text{dist}(x, y) = 2 \).
- We will now show that \( G - \{x, y\} \) is connected.
  - \( x' \) and \( y' \) are in \( G - \{x, y\} \) connected.
  - Show: Each node in \( G - \{x, y\} \) is connected with \( x' \) or \( y' \).
  - \( G - x \) is connected.
  - Each node from \( C - x \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( y \).
  - \( G - y \) is connected.
  - Each node from \( (V \setminus C) - y \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( x \).
- Running time: \( O(|V| + |E|) \).
Implications

- There exists $x$ in $C$ with $x$ is neighboured to $x'$ or $y'$.
- This hold for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - **Show:** Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
  - Running time: $O(|V| + |E|)$. 
Implications

- There exists $x$ in $C$ with $x$ is neighoured to $x'$ or $y'$.
- This hold for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - Show: Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
  - Running time: $O(|V| + |E|)$. 
Implications

- There exists \( x \) in \( C \) with \( x \) is neighboured to \( x' \) or \( y' \).
- This hold for each component in \( G \setminus \{x', y'\} \).
- Thus there exists \( y \) from some other component with \( \text{dist}(x, y) = 2 \).
- We will now show that \( G \setminus \{x, y\} \) is connected.
  - \( x' \) and \( y' \) are in \( G \setminus \{x, y\} \) connected.
  - Show: Each node in \( G \setminus \{x, y\} \) is connected with \( x' \) or \( y' \).
  - \( G \setminus x \) is connected.
  - Each node from \( C \setminus x \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( y \).
  - \( G \setminus y \) is connected.
  - Each node from \( (V \setminus C) \setminus y \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( x \).
- Running time: \( O(|V| + |E|) \).
Implications

- There exists $x$ in $C$ with $x$ is neighboured to $x'$ or $y'$.
- This hold for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - Show: Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
  - Running time: $O(|V| + |E|)$. 

\[ \chi(G) \]
Implications

- There exists \( x \) in \( C \) with \( x \) is neighboured to \( x' \) or \( y' \).
- This hold for each component in \( G - \{x', y'\} \).
- Thus there exists \( y \) from some other component with \( \text{dist}(x, y) = 2 \).
- We will now show that \( G - \{x, y\} \) is connected.
  - \( x' \) and \( y' \) are in \( G - \{x, y\} \) connected.
  - Show: Each node in \( G - \{x, y\} \) is connected with \( x' \) or \( y' \).
  - \( G - x \) is connected.
  - Each node from \( C - x \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( y \).
  - \( G - y \) is connected.
  - Each node from \( (V \setminus C) - y \) is connected by a path \( P \) with \( x' \) or \( y' \), without using \( x \).
- Running time: \( O(|V| + |E|) \).
Implications

- There exists $x$ in $C$ with $x$ is neighboured to $x'$ or $y'$.
- This hold for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - Show: Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
- **Running time:** $O(|V| + |E|)$. 
Implications

- There exists $x$ in $C$ with $x$ is neighboured to $x'$ or $y'$.
- This hold for each component in $G - \{x', y'\}$.
- Thus there exists $y$ from some other component with $\text{dist}(x, y) = 2$.
- We will now show that $G - \{x, y\}$ is connected.
  - $x'$ and $y'$ are in $G - \{x, y\}$ connected.
  - Show: Each node in $G - \{x, y\}$ is connected with $x'$ or $y'$.
  - $G - x$ is connected.
  - Each node from $C - x$ is connected by a path $P$ with $x'$ or $y'$, without using $y$.
  - $G - y$ is connected.
  - Each node from $(V \setminus C) - y$ is connected by a path $P$ with $x'$ or $y'$, without using $x$.
- Running time: $O(|V| + |E|)$. 
Theorems

Theorem (Mycielski’s)

For each number \( k \) there is a graph \( G \) with:

1. \( \chi(G) = k \) and
2. \( \omega(G) = 2 \).

Theorem (Erdös)

For each numbers \( k, l \) there is a graph \( G \) with:

1. \( \chi(G) = k \) and
2. The shortest cycle has length \( l \).

We will show only the first theorem:

- \( M_i \) has no triangles.
- \( \chi(M_i) = i \).
Theorems

Theorem (Mycielski’s)

For each number $k$ there is a graph $G$ with:
1. $\chi(G) = k$ and
2. $\omega(G) = 2$.

Theorem (Erdös)

For each numbers $k, l$ there is a graph $G$ with:
1. $\chi(G) = k$ and
2. The shortest cycle has length $l$.

We will show only the first theorem:
- $M_i$ has no triangles.
- $\chi(M_i) = i.$
Theorems

Theorem (Mycielski’s)

For each number $k$ there is a graph $G$ with:

1. $\chi(G) = k$ and
2. $\omega(G) = 2$.

Theorem (Erdös)

For each numbers $k, l$ there is a graph $G$ with:

1. $\chi(G) = k$ and
2. The shortest cycle has length $l$.

We will show only the first theorem:

- $M_i$ has no triangles.
- $\chi(M_i) = i$. 
Proof (Construction)

- $M_3 = C_5$
- Let $v_1, v_2, \ldots, v_n$ be the nodes of $M_k$.
- $M_{k+1}$ has the following additional nodes $u_1, u_2, \ldots, u_n$ and $w$.
- Add the following edges:
  - $\{w, u_i\}$ for $1 \leq i \leq n$ and
  - $\{u_i, x\}$ iff $\{v_i, x\} \in E(M_k)$.
Proof (Construction)

- $M_3 = C_5$
- Let $v_1, v_2, \ldots, v_n$ be the nodes of $M_k$.
- $M_{k+1}$ has the following additional nodes $u_1, u_2, \ldots, u_n$ and $w$.
- Add the following edges:
  - $\{w, u_i\}$ for $1 \leq i \leq n$ and
  - $\{u_i, x\}$ iff $\{v_i, x\} \in E(M_k)$.
Proof (Construction)

- $M_3 = C_5$
- Let $v_1, v_2, \ldots, v_n$ be the nodes of $M_k$.
- $M_{k+1}$ has the following additional nodes $u_1, u_2, \ldots, u_n$ and $w$.
- Add the following edges:
  - $\{w, u_i\}$ for $1 \leq i \leq n$ and
  - $\{u_i, x\}$ iff $\{v_i, x\} \in E(M_k)$. 
Proof (Construction)

- $M_3 = C_5$

- Let $v_1, v_2, \ldots, v_n$ be the nodes of $M_k$.

- $M_{k+1}$ has the following additional nodes $u_1, u_2, \ldots, u_n$ and $w$.

- Add the following edges:
  - $\{w, u_i\}$ for $1 \leq i \leq n$ and
  - $\{u_i, x\}$ iff $\{v_i, x\} \in E(M_k)$.
Proof (Construction)

- $M_3 = C_5$
- Let $v_1, v_2, \ldots, v_n$ be the nodes of $M_k$.
- $M_{k+1}$ has the following additional nodes $u_1, u_2, \ldots, u_n$ and $w$.
- Add the following edges:
  - $\{w, u_i\}$ for $1 \leq i \leq n$ and
  - $\{u_i, x\}$ iff $\{v_i, x\} \in E(M_k)$. 

Diagram:

- $v_1, v_2, v_3, v_4, v_5$ with edges $v_1v_2, v_2v_3, v_3v_4, v_4v_5, v_5v_1$.
- $w, u_1, u_2, u_3, u_4$ with edges $wv_1, wv_2, wv_3, wv_4$ and $u_1u_2, u_2u_3, u_3u_4, u_4u_1$. 
- Additional edges between $v_i$ and $u_i$ for $i = 1, 2, 3, 4$. 

Proof (Construction)

- $M_3 = C_5$
- Let $v_1, v_2, \ldots, v_n$ be the nodes of $M_k$.
- $M_{k+1}$ has the following additional nodes $u_1, u_2, \ldots, u_n$ and $w$.
- Add the following edges:
  - $\{w, u_i\}$ for $1 \leq i \leq n$ and
  - $\{u_i, x\}$ iff $\{v_i, x\} \in E(M_k)$. 
Proof (Construction)

- $M_3 = C_5$
- Let $v_1, v_2, \ldots, v_n$ be the nodes of $M_k$.
- $M_{k+1}$ has the following additional nodes $u_1, u_2, \ldots, u_n$ and $w$.
- Add the following edges:
  - $\{w, u_i\}$ for $1 \leq i \leq n$ and
  - $\{u_i, x\}$ iff $\{v_i, x\} \in E(M_k)$. 
Proof (Construction)

Note:

- \{u_1, u_2, \ldots, u_n\} is a stable set.
- \Gamma(v_i) is a stable set.
- Thus there are no triangles in \(M_{k+1}\).

- \(\chi(M_{k+1}) \leq k + 1\):
  - \(c(w) = k + 1\) and
  - \(c(u_i) = c(v_i)\).
Note:

1. \( \{u_1, u_2, \ldots, u_n\} \) is a stable set.
2. \( \Gamma(v_i) \) is a stable set.
3. Thus there are no triangles in \( M_{k+1} \).

\( \chi(M_{k+1}) \leq k + 1 \):

1. \( c(w) = k + 1 \) and
2. \( c(u_i) = c(v_i) \).
Proof (Construction)

- **Note:**
  - \(\{u_1, u_2, \ldots, u_n\}\) is a stable set.
  - \(\Gamma(v_i)\) is a stable set.
  - Thus there are no triangles in \(M_{k+1}\).

- \(\chi(M_{k+1}) \leq k + 1:\)
  - \(c(w) = k + 1\) and
  - \(c(u_i) = c(v_i)\).
Proof (Construction)

Note:
- \(\{u_1, u_2, \ldots, u_n\}\) is a stable set.
- \(\Gamma(v_i)\) is a stable set.
- Thus there are no triangles in \(M_{k+1}\).

\(\chi(M_{k+1}) \leq k + 1:\)
- \(c(w) = k + 1\) and
- \(c(u_i) = c(v_i)\).
Proof (Construction)

- Note:
  - \{u_1, u_2, \ldots, u_n\} is a stable set.
  - \Gamma(v_i) is a stable set.
  - Thus there are no triangles in \(M_{k+1}\).

- \(\chi(M_{k+1}) \leq k + 1\):
  - \(c(w) = k + 1\) and
  - \(c(u_i) = c(v_i)\).
Proof (Construction)

- Note:
  - \( \{u_1, u_2, \ldots, u_n\} \) is a stable set.
  - \( \Gamma(v_i) \) is a stable set.
  - Thus there are no triangles in \( M_{k+1} \).

- \( \chi(M_{k+1}) \leq k + 1 \):
  - \( c(w) = k + 1 \) and
  - \( c(u_i) = c(v_i) \).
Proof (Construction)

- Note:
  - \( \{u_1, u_2, \ldots, u_n\} \) is a stable set.
  - \( \Gamma(v_i) \) is a stable set.
  - Thus there are no triangles in \( M_{k+1} \).

- \( \chi(M_{k+1}) \leq k + 1 \):
  - \( c(w) = k + 1 \) and
  - \( c(u_i) = c(v_i) \).
Proof (Construction)

Note:

- \{u_1, u_2, \ldots, u_n\} is a stable set.
- \Gamma(v_i) is a stable set.
- Thus there are no triangles in \(M_{k+1}\).

- \(\chi(M_{k+1}) \leq k + 1\):
  - \(c(w) = k + 1\) and
  - \(c(u_i) = c(v_i)\).
**Proof (Construction)**

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
    - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
    - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,
  - Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.
  - If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
    - change the colours: $c(u_i) := c(v_i)$.
  - Let $v_j$ be a node with $c(v_j) = k$.
  - Then we have:
    - $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k - 1\}$
    - $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$
  - Contradiction!
Proof (Construction)

If \( \chi(M_{k+1}) = k \), we have:

- w.l.o.g.: \( c(w) = k \) and therefore:
  - \( \{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\} \),
  - \( \{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\} \).

Choose a colouring \( c \) with \(|\{i \mid c(v_i) = k\}| \) minimal.

If \( k \neq c(v_i) \neq c(u_i) \) for some \( i \),

- change the colours: \( c(u_i) := c(v_i) \).

Let \( v_j \) be a node with \( c(v_j) = k \).

Then we have:

- \( \{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\} \)
- \( \{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\} \)

Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
  - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
  - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,

- Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.

- If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
  - change the colours: $c(u_i) := c(v_i)$.

- Let $v_j$ be a node with $c(v_j) = k$.

- Then we have:
  - $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
  - $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

- Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
  - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
  - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$.

- Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.

- If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
  - change the colours: $c(u_i) := c(v_i)$.

- Let $v_j$ be a node with $c(v_j) = k$.

- Then we have:
  - $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
  - $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

- Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
    - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
    - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,
  - Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.
  - If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
    - change the colours: $c(u_i) := c(v_i)$.
  - Let $v_j$ be a node with $c(v_j) = k$.
  - Then we have:
    - $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k - 1\}$
    - $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$
  - Contradiction!
Proof (Construction)

If $\chi(M_{k+1}) = k$, we have:

- w.l.o.g.: $c(w) = k$ and therefore:
  - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
  - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,

Choose a colouring $c$ with $\{|i \mid c(v_i) = k\}$ minimal.

If $k \neq c(v_i) \neq c(u_i)$ for some $i$,

- change the colours: $c(u_i) := c(v_i)$.

Let $v_j$ be a node with $c(v_j) = k$.

Then we have:

- $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
- $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
  - $\{c(v_i) | 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
  - $\{c(u_i) | 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,

- Choose a colouring $c$ with $\{|i | c(v_i) = k\}$ minimal.

- If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
  - change the colours: $c(u_i) := c(v_i)$.

- Let $v_j$ be a node with $c(v_j) = k$.

- Then we have:
  - $\{c(a) | a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
  - $\{c(a) | a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

- Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
  - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
  - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,

Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.

- If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
  - change the colours: $c(u_i) := c(v_i)$.

- Let $v_j$ be a node with $c(v_j) = k$.

Then we have:

- $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
- $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
    - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
    - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,
  - Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.
  - If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
    - change the colours: $c(u_i) := c(v_i)$.
  - Let $v_j$ be a node with $c(v_j) = k$.
  - Then we have:
    - $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k - 1\}$
    - $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$
  - Contradiction!
Proof (Construction)

If $\chi(M_{k+1}) = k$, we have:
- w.l.o.g.: $c(w) = k$ and therefore:
  - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
  - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,

Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.

If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
- change the colours: $c(u_i) := c(v_i)$.

Let $v_j$ be a node with $c(v_j) = k$.

Then we have:
- $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
- $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

Contradiction!
Proof (Construction)

If $\chi(M_{k+1}) = k$, we have:

- w.l.o.g.: $c(w) = k$ and therefore:
  - $\{c(v_i) | 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
  - $\{c(u_i) | 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,

Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.

If $k \neq c(v_i) \neq c(u_i)$ for some $i$,

- change the colours: $c(u_i) := c(v_i)$.

Let $v_j$ be a node with $c(v_j) = k$.

Then we have:

- $\{c(a) | a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
- $\{c(a) | a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
    - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
    - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,
  - Choose a colouring $c$ with $\{|i \mid c(v_i) = k\}$ minimal.
  - If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
    - change the colours: $c(u_i) := c(v_i)$.
  - Let $v_j$ be a node with $c(v_j) = k$.
  - Then we have:
    - $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
    - $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$
  - Contradiction!
Proof (Construction)

- If $\chi(M_{k+1}) = k$, we have:
  - w.l.o.g.: $c(w) = k$ and therefore:
    - $\{c(v_i) \mid 1 \leq i \leq n\} = \{1, 2, \ldots, k\}$,
    - $\{c(u_i) \mid 1 \leq i \leq n\} = \{1, \ldots, k-1\}$,
  - Choose a colouring $c$ with $|\{i \mid c(v_i) = k\}|$ minimal.
  
- If $k \neq c(v_i) \neq c(u_i)$ for some $i$,
  - change the colours: $c(u_i) := c(v_i)$.

- Let $v_j$ be a node with $c(v_j) = k$.
- Then we have:
  - $\{c(a) \mid a \in \Gamma(v_j)\} = \{1, \ldots, k-1\}$
  - $\{c(a) \mid a \in \Gamma(u_j)\} = \{1, \ldots, k\}$

- Contradiction!
Computing the Colouring

**Theorem (Widgerson 1983)**

*Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.*

**Proof:**

- If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
- We colour the nodes by checking their degree:
- As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours
- After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
- Colour this subgraph with new colours.
- The number of colours is at most: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
- Detailed analysis show: $\sqrt{8 \cdot n}$. 
Computing the Colouring

**Theorem (Widgerson 1983)**

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.

**Proof:**

- If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
- We colour the nodes by checking their degree:
  - As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours.
  - After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
- Colour this subgraph with new colours.
- The number of colours is at most: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
- Detailed analysis show: $\sqrt{8 \cdot n}$.
Computing the Colouring

**Theorem (Widgerson 1983)**

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.

Proof:

- If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
- We colour the nodes by checking their degree:
  - As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours
  - After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
  - Colour this subgraph with new colours.
- The number of colours is at most: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
- Detailed analysis show: $\sqrt{8 \cdot n}$. 
Computing the Colouring

**Theorem (Widgerson 1983)**

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.

**Proof:**

- If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
- We colour the nodes by checking their degree:
- As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours
- After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
- Colour this subgraph with new colours.
- The number of colours is at most: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
- Detailed analysis show: $\sqrt{8 \cdot n}$. 

Computing the Colouring

Theorem (Widgerson 1983)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.

Proof:

- If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
- We colour the nodes by checking their degree:
- As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours
- After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
- Colour this subgraph with new colours.
- The number of colours is at most: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
- Detailed analysis show: $\sqrt{8 \cdot n}$. 
Computing the Colouring

Theorem (Widgerson 1983)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.

Proof:

1. If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
2. We colour the nodes by checking their degree:
3. As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours
4. After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
5. Colour this subgraph with new colours.
6. **The number of colours is at most**: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
7. Detailed analysis show: $\sqrt{8 \cdot n}$. 
Introduction

Hardness

Algorithms

Colour with Greed

Brooks

Girth

Complexity

Basics (3:43.7)

Computing the Colouring

Theorem (Widgerson 1983)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.

Proof:

- If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
- We colour the nodes by checking their degree:
- As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours
- After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
- Colour this subgraph with new colours.
- The number of colours is at most: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
- Detailed analysis show: $\sqrt{8 \cdot n}$. 

Computing the Colouring

Theorem (Widgerson 1983)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(\sqrt{n})$ colouring.

Proof:

- If $\chi(G) = 3$ holds, $\chi(G[\Gamma(v)]) \leq 2$ is true.
- We colour the nodes by checking their degree:
- As long as there is a node $v$ with $\deg_G(v) \geq \sqrt{n}$ colour $\Gamma(v)$ using two colours
- After at most $\sqrt{n}$ steps we get a subgraph with at most $\sqrt{n}$ nodes.
- Colour this subgraph with new colours.
- The number of colours is at most: $2 \cdot \sqrt{n} + \sqrt{n} = 3 \cdot \sqrt{n}$.
- Detailed analysis show: $\sqrt{8 \cdot n}$. 
Computing the Colouring

Theorem (Blum 1994)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{3/8})$ colouring.

Theorem (Karger, Motwani, Sudan 1994)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{1/4})$ colouring.

Theorem (Blum, Karger 1996)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{3/14})$ colouring.
Computing the Colouring

**Theorem (Blum 1994)**

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{3/8})$ colouring.

**Theorem (Karger, Motwani, Sudan 1994)**

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{1/4})$ colouring.

**Theorem (Blum, Karger 1996)**

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{3/14})$ colouring.
Computing the Colouring

Theorem (Blum 1994)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{3/8})$ colouring.

Theorem (Karger, Motwani, Sudan 1994)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{1/4})$ colouring.

Theorem (Blum, Karger 1996)

Let $G = (V, E)$ be a graph with $\chi(G) = 3$. Then we may efficiently compute a $O(n^{3/14})$ colouring.
Introduction

Hardness

Algorithms

Colour with Greed

Brooks

Girth

Colouring $\chi(G)$

Complexity

Negative Theorems (3:45.1)

<>

Walter Unger 6.1.2015 17:05
WS2014/15

Theorems

**Theorem**

*The 3-colouring-problem is for graphs of degree $\leq 4$ NP-complete.* The $k$-colouring-problem is NP-complete.

**Theorem**

*Let $k \geq 3$ and $c = 1/(2 + 3 \cdot \log(k + 1))$. Then the $k$-colouring-problem on graphs with girth $\lceil c \log c \rceil$ is NP-complete.*

**Theorem**

*The colouring-problem could not be approximated by a constant factor (Assuming $\mathbb{P} \neq \mathbb{NP}$).*

**Theorem**

*To compute a 4-colouring for a 3-colourable graph is NP-hard.*
**Theorem**

The 3-colouring-problem is for graphs of degree \( \leq 4 \) NP-complete. The \( k \)-colouring-problem is NP-complete.

**Theorem**

Let \( k \geq 3 \) and \( c = 1/(2 + 3 \cdot \log(k + 1)) \). Then the \( k \)-colouring-problem on graphs with girth \( \lceil c \log c \rceil \) is NP-complete.

**Theorem**

The colouring-problem could not be approximated by a constant factor (Assuming \( \mathcal{P} \neq \mathcal{NP} \)).

**Theorem**

To compute a 4-colouring for a 3-colourable graph is NP-hard.
Theorem

*The 3-colouring-problem is for graphs of degree $\leq 4$ NP-complete. The $k$-colouring-problem is NP-complete.*

Theorem

*Let $k \geq 3$ and $c = 1/(2 + 3 \cdot \log(k + 1))$. Then the $k$-colouring-problem on graphs with girth $\lceil c \log c \rceil$ is NP-complete.*

Theorem

*The colouring-problem could not be approximated by a constant factor (Assuming $P \neq NP$).*

Theorem

*To compute a 4-colouring for a 3-colourable graph is NP-hard.*
Theorems

Theorem

The 3-colouring-problem is for graphs of degree $\leq 4$ NP-complete. The $k$-colouring-problem is NP-complete.

Theorem

Let $k \geq 3$ and $c = 1/(2 + 3 \cdot \log(k + 1))$. Then the $k$-colouring-problem on graphs with girth $\lceil c \log c \rceil$ is NP-complete.

Theorem

The colouring-problem could not be approximated by a constant factor (Assuming $\mathcal{P} \neq \mathcal{NP}$).

Theorem

To compute a 4-colouring for a 3-colourable graph is NP-hard.
Theorems

**Theorem**

The 3-colouring-problem is for graphs of degree \(\leq 4\) NP-complete. The \(k\)-colouring-problem is NP-complete.

**Theorem**

Let \(k \geq 3\) and \(c = 1/(2 + 3 \cdot \log(k + 1))\). Then the \(k\)-colouring-problem on graphs with girth \([c \log c]\) is NP-complete.

**Theorem**

The colouring-problem could not be approximated by a constant factor (Assuming \(\mathcal{P} \neq \mathcal{NP}\)).

**Theorem**

To compute a 4-colouring for a 3-colourable graph is NP-hard.
### Theorems

**Theorem**

*The 3-colouring-problem is for graphs of degree $\leq 4$ NP-complete. The $k$-colouring-problem is NP-complete.*

**Theorem**

*Let $k \geq 3$ and $c = 1/(2 + 3 \cdot \log(k + 1))$. Then the $k$-colouring-problem on graphs with girth $\lceil c \log c \rceil$ is NP-complete.*

**Theorem**

*The colouring-problem could not be approximated by a constant factor (Assuming $\mathcal{P} \neq \mathcal{NP}$).*

**Theorem**

*To compute a 4-colouring for a 3-colourable graph is NP-hard.*
Lemma

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of $4/3$ for the colouring-problem.

Theorem (Garry, Johnson 1976)

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of $2$ for the colouring-problem.

Theorem (Land, Jannakakis 1993)

If $\mathcal{P} \neq \mathcal{NP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^\varepsilon$ for the colouring-problem.

Theorem (Feige, Kilian 1996)

If $\mathcal{P} \neq \mathcal{ZPP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^{1-\varepsilon}$ for the colouring-problem.
Theorems

Lemma

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of $4/3$ for the colouring-problem.

Theorem (Garry, Johnson 1976)

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of 2 for the colouring-problem.

Theorem (Land, Jannakakis 1993)

If $\mathcal{P} \neq \mathcal{NP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^\varepsilon$ for the colouring-problem.

Theorem (Feige, Kilian 1996)

If $\mathcal{P} \neq \mathcal{ZPP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^{1-\varepsilon}$ for the colouring-problem.
Theorems

Lemma

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of $4/3$ for the colouring-problem.

Theorem (Garry, Johnson 1976)

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of 2 for the colouring-problem.

Theorem (Land, Jannakakis 1993)

If $\mathcal{P} \neq \mathcal{NP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^\varepsilon$ for the colouring-problem.

Theorem (Feige, Kilian 1996)

If $\mathcal{P} \neq \mathcal{ZPP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^{1-\varepsilon}$ for the colouring-problem.
### Lemma

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of $4/3$ for the colouring-problem.

### Theorem (Garry, Johnson 1976)

If $\mathcal{P} \neq \mathcal{NP}$, then there is no polynomial time algorithm with an approximation-factor of $2$ for the colouring-problem.

### Theorem (Land, Jannakakis 1993)

If $\mathcal{P} \neq \mathcal{NP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^\varepsilon$ for the colouring-problem.

### Theorem (Feige, Kilian 1996)

If $\mathcal{P} \neq \mathcal{ZPP}$, then there is for any $\varepsilon > 0$ no polynomial time algorithm with an approximation-factor of $n^{1-\varepsilon}$ for the colouring-problem.
**Theorems**

**Lemma**

*Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.*

- **If** $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \left(\frac{2}{c}\right)!))$.
  - Running time: $O(1)$ and error factor 1.

- **If** $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\lfloor n/\lfloor c \cdot n \rfloor \rfloor$ or $\lceil n/\lfloor c \cdot n \rfloor \rceil$.
  - Each part has size $\leq \frac{n}{c n - 1} + 1 \leq 2 = \tilde{O}(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
Theorems

Lemma

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \left(\frac{2}{c}\right)!))$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\lfloor n/\lfloor c \cdot n \rfloor \rfloor$ or $\lceil n/\lfloor c \cdot n \rfloor \rceil$.
  - Each part has size $\leq \frac{n}{c n - 1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
**Theorems**

**Lemma**

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - **Running time**: $O((2/c)! \cdot \binom{(2/c)!}{2})$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor cn \rfloor$ Parts of size $\lfloor n/(c \cdot n) \rfloor$ or $\lceil n/(c \cdot n) \rceil$.
  - Each part has size $\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
Lemma

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \binom{2/c}{2})$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\left\lfloor \frac{n}{\lfloor c \cdot n \rfloor} \right\rfloor$ or $\left\lceil \frac{n}{\lfloor c \cdot n \rfloor} \right\rceil$.
  - Each part has size $\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
Lemma

Let \(0 < c \leq 1\) be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of \(\max(1, c \cdot n)\).

- If \(|V| \leq 2/c\) then just colour \(G\):
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: \(O((2/c)! \cdot \binom{2/c}{2})\).
  - Running time: \(O(1)\) and error factor 1.

- If \(|V| > 2/c\) then colour \(G\):
  - Split \(V(G)\) in \(\lfloor c \cdot n \rfloor\) Parts of size \(\lfloor n/\lfloor c \cdot n \rfloor \rfloor\) or \(\lceil n/\lfloor c \cdot n \rfloor \rceil\).
  - Each part has size \(\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)\).
  - Each part may be coloured optimal in constant time.
  - Total number of colours: \(\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn\).
Lemma

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \binom{2/c}{2})$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\lfloor n / (c \cdot n) \rfloor$ or $\lceil n / (c \cdot n) \rceil$.
  - Each part has size $\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
Lemma

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \binom{(2/c)!}{2})$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\lfloor n/\lfloor c \cdot n \rfloor \rfloor$ or $\lceil n/\lfloor c \cdot n \rfloor \rceil$.
  - Each part has size $\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
**Lemma**

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \left(\frac{2}{c}\right)!)$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\left\lfloor \frac{n}{\lfloor c \cdot n \rfloor} \right\rfloor$ or $\left\lceil \frac{n}{\lfloor c \cdot n \rfloor} \right\rceil$.
  - Each part has size $\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$.
Lemma

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \binom{2/c}{2})$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\lfloor n/\lfloor c \cdot n \rfloor \rfloor$ or $\lceil n/\lfloor c \cdot n \rfloor \rceil$.
  - Each part has size $\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
**Theorems**

**Lemma**

Let $0 < c \leq 1$ be a constant. There is a linear Algorithm, which approximates the colouring-problem with a factor of $\max(1, c \cdot n)$.

- If $|V| \leq 2/c$ then just colour $G$:
  - Colour the graph by greedy algorithm using all permutations of the nodes.
  - Running time: $O((2/c)! \cdot \binom{2/c}{2})$.
  - Running time: $O(1)$ and error factor 1.

- If $|V| > 2/c$ then colour $G$:
  - Split $V(G)$ in $\lfloor c \cdot n \rfloor$ Parts of size $\lfloor n/\lfloor c \cdot n \rfloor \rfloor$ or $\lceil n/\lfloor c \cdot n \rfloor \rceil$.
  - Each part has size $\leq \frac{n}{cn-1} + 1 \leq \frac{2}{c} = O(1)$.
  - Each part may be coloured optimal in constant time.
  - Total number of colours: $\frac{\lfloor cn \rfloor \cdot \chi(G)}{\chi(G)} \leq cn$. 
Theorem (Johnson 1974)

The colouring-problem could be approximated within a factor of $O(n / \log n)$ in time $O(nm)$.

Theorem

The colouring-problem could be efficiently approximated within a factor of $O(n(\log n) - 3(\log \log n)/2)$.
Theorems

**Theorem (Johnson 1974)**

*The colouring-problem could be approximated within a factor of $O(n / \log n)$ in time $O(nm)$.*

**Theorem**

*The colouring-problem could be efficiently approximated within a factor of $O(n(\log n) - 3(\log \log n)/2)$.*
Theorems

Theorem (Johnson 1974)

The colouring-problem could be approximated within a factor of $O(n/\log n)$ in time $O(nm)$.

Theorem

The colouring-problem could be efficiently approximated within a factor of $O(n(\log n) - 3(\log \log n)/2)$.
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy-colouring?
Questions

- How hard is the edge-colouring-problem?
- How many colours needed to colour the edges of a clique?
- How could the edges of a bipartite graph be coloured?
- What is the upper bound for the number of colours for the edge-colouring?
- What is the idea of the proof of Vizing?
- How hard is the node-colouring-problem?
- What bounds are known?
- What error is possible when using greedy colouring?
Legend

- : Not of relevance
- : implicitly used basics
- : idea of proof or algorithm
- : structure of proof or algorithm
- : Full knowledge