# Graph Coloring

Alessandro Hill

rev. 1.0(AH) – 2024

# The Graph Coloring Problem

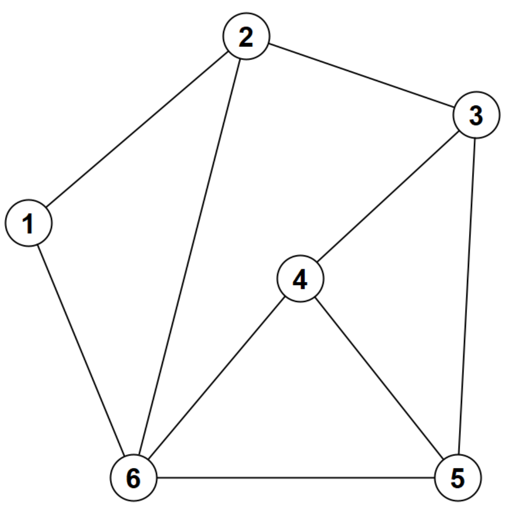
**Colorazione dei grafi** (Due nodi collegati da un arco devono avere colori diversi)

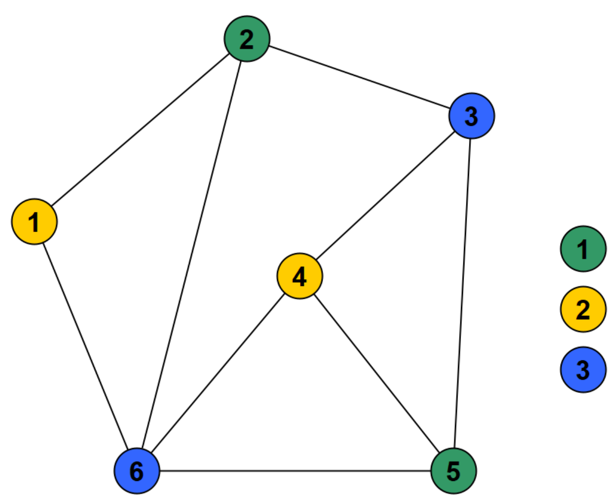
We are given an undirected graph **G** with node set **V** and edge set **E**. Edges in **E** are also called conflicts. The set **C** contains numbers that correspond to colors.

A feasible coloring of **G** is an assignment of its nodes to colors from **C** such that two adjacent nodes have different colors.

The Graph Coloring Problem asks for a feasible coloring of **G** using the minimum number of colors from **C**.*(Trovare una colorazione fattibile usando il minor numero possibile di colori)

*NP-hard (extremely difficult to optimize)




**Applications:**

- Frequency assignment
- Seating assignment
- Scheduling
- Experiment design
- Computer science, etc.

# The Graph Coloring Problem

È il numero minimo di colori necessari per una colorazione fattibile del grafo G.

chi

**Definition:** The chromatic number $\chi(G)$ of graph **G** is the minimum number of colors needed for a feasible coloring of **G**.

omega

**Definition:** The clique number $\omega(G)$ of graph **G** is the size of a maximum clique.

È la dimensione della clique massima in G, ovvero il numero massimo di nodi che sono tutti collegati fra loro.

**Proposition:** $\chi(G) \geq \omega(G)$

perché per colorare una clique di dimensione k servono almeno k colori distinti.

**Observation:** Nodes of same color form an independent set.

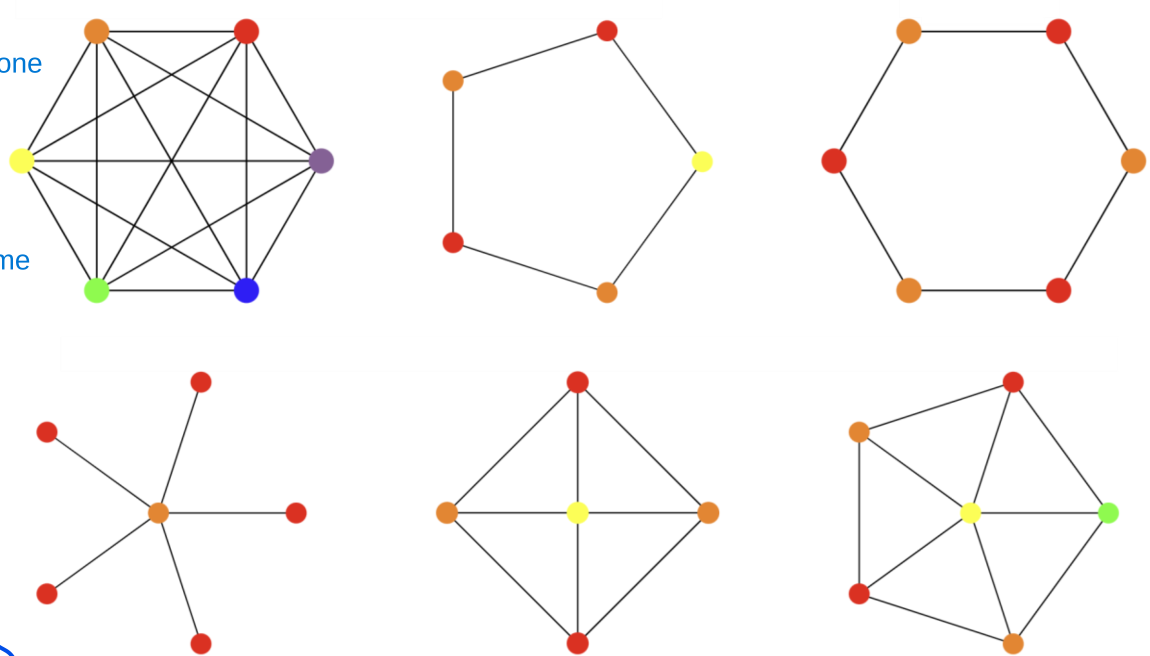
I nodi dello stesso colore formano un insieme indipendente (nessuno è adiacente).

**In General:** Given a clique Q and a feasible coloring with k colors. Then:

$$|Q| \leq \chi(G) \leq k$$

lower bound (LB) upper bound (UB)

massimo numero di colori usati in una colorazione fattibile



# The Graph Coloring Problem

## IP Model

**Binary color assignment variable for each node and color:**

$$x_{v,c} = \begin{cases} 1 & \text{if node } v \text{ will be colored with color } c, \\ 0 & \text{otherwise.} \end{cases}$$

**Binary color usage variable for each color:**

$$y_c = \begin{cases} 1 & \text{if color } c \text{ is used}, \\ 0 & \text{otherwise.} \end{cases}$$

**Minimize** $\sum_{c \in C} y_c$

**Idea:** Two connected nodes cannot be colored with the same color.

$$x_{u,c} + x_{v,c} \leq 1 \qquad \forall \{u,v\} \in E,\ c \in C$$

Due nodi adiacenti (u e v) non possono essere colorati con lo stesso colore c. Questo vincolo garantisce che la colorazione sia fattibile.

collega le variabili xv,c e yc.

$$\frac{1}{|V|} \sum_{v \in V} x_{v,c} \leq y_c \qquad \forall c \in C \qquad \text{(Variable Linking)}$$

yc indica se colore attivato: se non attivato, non ho nodi colorati da colore C ; se attivo, grazie a 1/V avrò <= 1

$$\sum_{c \in C} x_{v,c} = 1 \qquad \forall v \in V \qquad \text{(Color Assignment)}$$

Ogni nodo deve essere colorato esattamente con un solo colore.

# The Graph Coloring Problem

## Greedy Heuristic

Scelgo un nodo non colorato e lo coloro in modo corretto (coloro con un colore diverso dagli adiacenti (uso colore già usato ma se non posso, uso un colore nuovo))

**Input:** Undirected graph G=(V,E); Colors C={1,…,|V|}.

Essendo che non ho una regola precisa nella scelta dei nodi, nell'esempio, ho usato il metodo di scelta del nodo con indice più basso disponibile, per evitare di scegliare il prossimo nodo in modo random

1. Pick an uncolored node v in V.
   → Color v with a "feasible" used color, if possible, else use unused color.

2. If all nodes are colored return coloring, else go to 1.

# The Graph Coloring Problem

## DSatur Heuristic

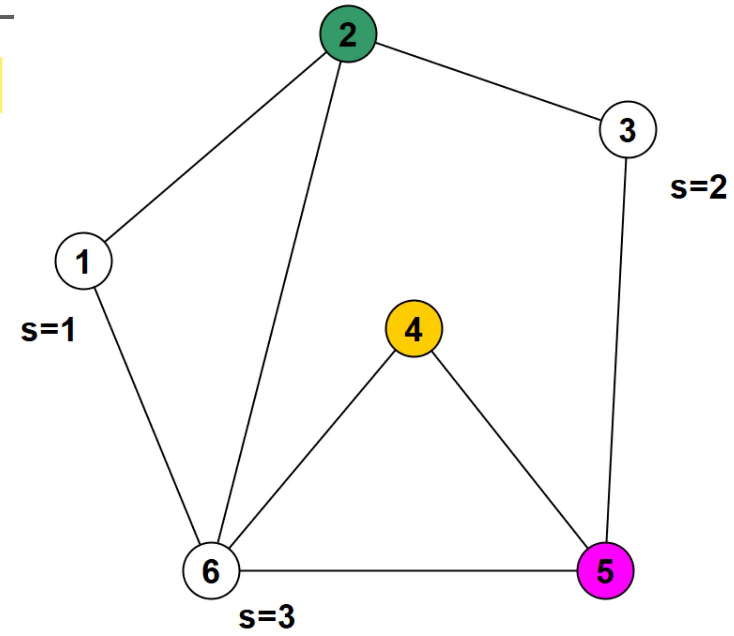
**Definition:** For a partial coloring of graph **G**, the saturation of an (uncolored) node is the number of distinct colors of its neighbors.

Saturazione di un nodo non colorato = numero di colori distinti dei nodi adiacenti al nodo non colorato

Cerco il/i nodo/i con alta saturazione e se ne ho più di uno, controllo il degre e se ne ho più di uno, scelgo quello con indice più basso

Se ho nodi con stessa saturazione e degre, scelgo il nodo con indice più basso

Scelgo nodo non colorato con saturazione più alta e degree più alto (in caso di nodi non colorati con simil saturazione). Coloro con un colore diverso da adiacenti (uso colore già usato, ma se non posso, uso un nuovo colore)



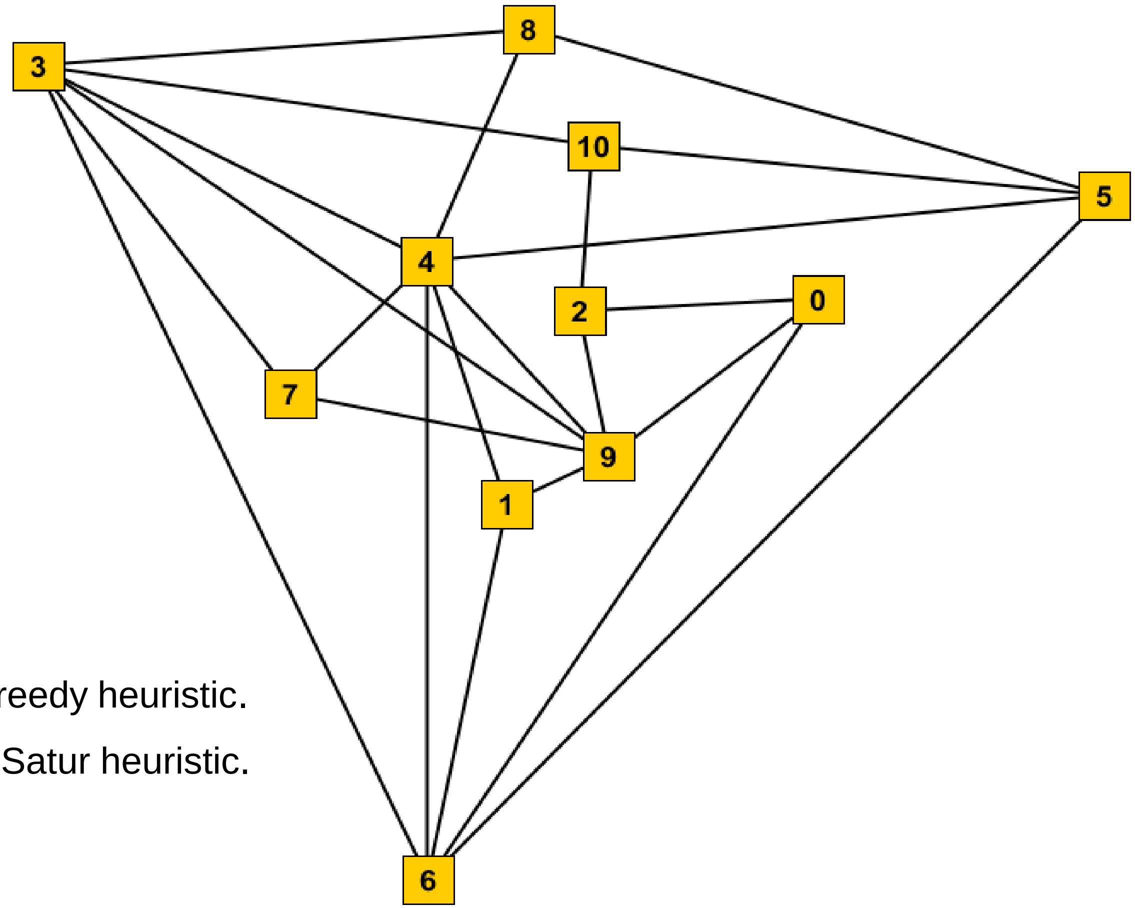

**Input:** Undirected graph G=(V,E); Colors C={1,…,|V|}.

1. Pick an uncolored node with (1) highest saturation and (2) highest degree.
   → Color node with a "feasible" used color, if possible, else use unused color.

2. If all nodes are colored return coloring, else go to 1.

# The Graph Coloring Problem

**Example:**



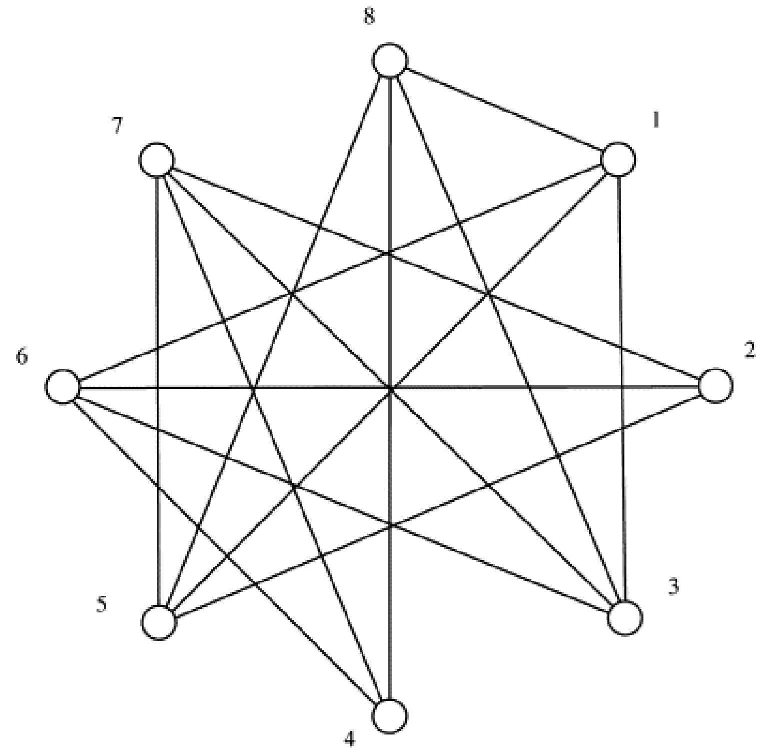

a) Find a feasible coloring using the greedy heuristic.

b) Find a feasible coloring using the DSatur heuristic.

c) Find an optimal coloring using IP.

# The Graph Coloring Problem

## 1) Exercises:




**1. Consider the graph with 8 nodes.**

**2. Find an optimal coloring using IP.**
Verify your results visually.

**3. Find a coloring using the DSatur heuristic.**
Verify your results visually.

**4. Add a minimal number of edges to increase the number of needed colors by one.**
Verify your results using the IP model.

# The Graph Coloring Problem

## 2) Exercises:

**1. Create a random graph with 100 nodes and an arbitrary number of edges in yEd.**

Use a convenient layout for your visualization.
Format as necessary.
You can export the edge list using the TGF format.

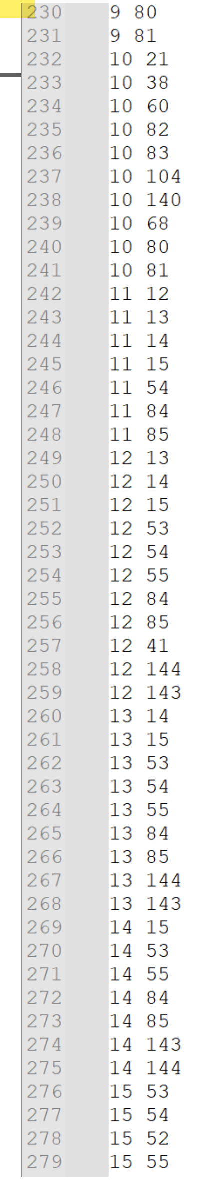
**2. Find an optimal coloring using IP.**

Verify your results visually.

**3. Find an optimal coloring using MiniZinc.**

Verify your results visually.

**4. For what larger/denser graphs can you still answer the questions above?**



```
9 80
9 81
10 21
10 38
10 60
10 82
10 83
10 104
10 140
10 68
10 80
10 81
11 12
11 13
11 14
11 15
11 54
11 84
11 85
12 13
12 14
12 15
12 53
12 54
12 55
12 84
12 85
12 41
12 144
12 143
13 14
13 15
13 53
13 54
13 55
13 84
13 85
13 144
13 143
14 15
14 53
14 55
14 84
14 85
14 143
14 144
15 53
15 54
15 52
15 55
```

# The Graph Coloring Problem

**3) Exercises:**

**1. Use the class social network data (Virtuale).**

**2. Can you find a partition of minimum size such that people in the partition set do not know each other?**

Solve the graph coloring problem.

Then nodes of the same color correspond to individuals that do not know each other.

**3. Can you visualize your results from 2?**