Graph Algorithms

Vertex Coloring
The Input Graph

\[ G = (V, E) \text{ a simple and undirected graph:} \]

- \( V \): a set of \( n \) vertices.
- \( E \): a set of \( m \) edges.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Vertex Coloring

Definition I:
★ A disjoint collection of independent sets that cover all the vertices in the graph.
★ A partition $V = I_1 \cup I_2 \cup \cdots \cup I_\chi$ such that $I_j$ is an independent set for all $1 \leq j \leq \chi$.

Definition II:
★ An assignment of colors to the vertices such that two adjacent vertices are assigned different colors.
★ A function $c : V \to \{1, \ldots, \chi\}$ such that if $(u, v) \in E$ then $c(u) \neq c(v)$.

Observation: Both definitions are equivalent.
Example: Coloring
Example: Coloring with Minimum Number of Colors
The Vertex Coloring Problem

The optimization problem: Find a vertex coloring with minimum number of colors.

Notation: The chromatic number of $G$, denoted by $\chi(G)$, is the minimum number of colors required to color all the vertices of $G$.

Hardness: A very hard problem (an NP-Complete problem).
It is NP-Hard to color a 3-colorable graph with 3 colors.

It is NP-Hard to construct an algorithms that colors a graph with at most $n^\varepsilon \chi(G)$ colors for any constant $0 < \varepsilon < 1$. 
Known Algorithms for Vertex Coloring

★ There exists an optimal algorithm for coloring whose running time is $O\left( mn \left( 1 + 3^{1/3} \right)^n \right) \approx mn1.442^n$.

★ There exists a polynomial time algorithm that colors any graph with at most $O(n/ \log n)\chi(G)$ colors.

★ There exists an algorithm that colors a 3-colorable graph with $O(n^{1/3})$ colors.
Properties of Vertex Coloring

**Observation:** \( K(G) \leq \chi(G) \).

- Because in any vertex coloring, each member of a clique must be colored by a different color.

**Observation:** \( \chi(G) \geq \left\lceil \frac{n}{I(G)} \right\rceil \).

- A pigeon hole argument: the size of each color-set is at most \( I(G) \).
Example: $\chi(G) = K(G)$

- $K(G) = 4$ and $\chi(G) = 4$.
- Every member of the only clique of size 4 must be colored with a different color.
Example: $\chi(G) > K(G)$

$K(G) = 2$ and $\chi(G) = 3$
Example: \( \chi(G) > K(G) \)

\( K(G) = 2 \) and \( \chi(G) = 4 \)
Theorem: For any $k \geq 3$, there exists a triangle-free graph $G_k$ ($K(G_k) = 2$) for which $\chi(G_k) = k$.

A construction: $G_3$ and $G_4$ are the examples above. Construct $G_{k+1}$ from $G_k$.

- Let $V = \{v_1, \ldots, v_n\}$ be the vertices of $G_k$.
- The vertices of $G_{k+1}$ include $V$, a new vertex $w$, and a new set of vertices $U = \{u_1, \ldots, u_n\}$ for a total of $2n+1$ vertices.
- The edges of $G_{k+1}$ include all the edges of $G_k$, $w$ is connected to all the vertices in $U$, and $u_i \in U$ is connected to all the neighbors of $v_i$ in $G_k$. 

$\chi(G') >> K(G')$
Constructing $G_4$ from $G_3$. 

Graph Algorithms
Constructing $G_4$ from $G_3$. 
Constructing $G_4$ from $G_3$. 

Graph Algorithms

15
Constructing $G_4$ from $G_3$. 
\( G_{k+1} \) is a Triangle-Free graph

\begin{itemize}
  \item \( U \) is an independent set in \( G_{k+1} \) and therefore there is no triangle with at least 2 vertices from \( U \).
  \item \( w \) is not adjacent to \( V \) and is adjacent to the independent set \( U \). Therefore \( w \) cannot be a member in a triangle.
  \item \( V \) contains no triangles because \( G_k \) is a triangle-free graph.
  \item The remaining case is a triangle with 1 vertex \( u_i \in U \) and 2 vertices \( v, v' \in V \).
  \item This is impossible since \( u_i \) is connected to the neighbors of \( v_i \) and therefore the triangle \( u_i v v' \) would imply the triangle \( v_i v v' \) in the triangle-free graph \( G_k \).
\end{itemize}
\[ \chi(G_{k+1}) \leq k + 1 \]

- Color the vertices in \( V \) with \( k \) colors as in \( G_k \).
- Color \( u_i \) with the color of \( v_i \). This is a legal coloring since \( u_i \) is connected to the neighbors of \( v_i \).
- Color \( w \) with a new color.
Coloring $G_4$
Coloring $G_4$
Coloring $G_4$
Coloring $G_4$
\[ \chi(G_{k+1}) > k \]

- Assume that \( G_{k+1} \) is colored with the colors 1, \ldots, \( k \).
- Let the color of \( w \) be \( k \).
- Since \( w \) is adjacent to all the vertices in \( U \) it follows that the vertices in \( U \) are colored with the colors 1, \ldots, \( k - 1 \).
- Color each \( v_i \) that is colored by \( k \) with the color of \( u_i \).
- This produces a legal coloring of the \( G_k \) subgraph of the \( G_{k+1} \) graph because \( u_i \) is adjacent to all the neighbors of \( v_i \) and the set of all the \( k \)-colored \( v_i \) is an independent set.
- A contradiction since \( \chi(G_k) = k \).
Perfect Graphs

• In a perfect graph $\chi(G) = K(G)$ for any “induced” subgraph of $G$.

• Coloring is not Hard for perfect graphs.

• The complement of a perfect graph is a perfect graph.

• Interval graphs are perfect graphs.
Observation: A graph with $n \geq 1$ vertices needs at least 1 color and at most $n$ colors.

$\star 1 \leq \chi(G) \leq n$.

Null Graphs: No edges $\Rightarrow$ 1 color is enough.

$\star \chi(N_n) = 1$.

Complete Graphs: All edges $\Rightarrow n$ colors are required.

$\star \chi(K_n) = n$. 
Theorem: The following three statements are equivalent for a simple undirected graph $G$:

1. $G$ is a bipartite graph.

2. There are no odd length cycles in $G$.

3. $G$ can be colored with 2 colors.
Proof: 1 $\Rightarrow$ 2

- The vertices of $G$ can be partitioned into 2 sets $A$ and $B$ such that each edge connects a vertex from $A$ with a vertex from $B$.

- The vertices of any cycle alternate between $A$ and $B$.

- Therefore, any cycle must have an even length.
Proof: \( 2 \Rightarrow 3 \)

★ Run BFS on \( G \) starting with an arbitrary vertex.
★ Color odd-levels vertices 1 and even-level vertices 2.
★ Tree edges connect vertices with different colors.
★ In a BFS there are no forward and backward edges and a cross edge connects level \( \ell \) with level \( \ell' \) only if \( |\ell - \ell'| \leq 1 \).
★ If \( \ell = \ell' + 1 \) then the cross edge connects vertices with different colors.
★ If \( \ell = \ell' \) then the cross edge closes an odd-length cycle contradicting the assumption.
★ Thus, all the edges connect vertices with different colors.
Proof: $3 \Rightarrow 1$

- Let $A$ be all the vertices with color 1 and let $B$ be all the vertices with color 2.

- By the definition of coloring, any edge connects a vertex from $A$ with a vertex from $B$.

- Therefore, the graph is bipartite.
Coloring 2-colorable graphs

- Apply the BFS algorithm from the $2 \Rightarrow 3$ proof.
- $O(n + m)$-time complexity using adjacency lists.
- Can be used to recognize bipartite graphs: If there exists an edge connecting vertices with the same color then the graph is not bipartite.
A tree is a bipartite graph and therefore can be colored with 2 colors.
A cycle graph with an even number of vertices is a bipartite graph and therefore can be colored with 2 colors.
Odd Length Cycles

* A cycle graph with an odd number of vertices is not a bipartite graph \( \Rightarrow \) it cannot be colored with 2 colors.

* 3-Coloring: Color one vertex 3. The rest of the vertices induce a bipartite graph and therefore can be colored with colors 1 and 2.
Greedy Vertex Coloring

**Theorem:** Let $\Delta$ be the maximum degree in $G$. Then $G$ can be colored with $\Delta + 1$ colors.

**Proof:**

- Color the vertices in a sequence.
- A vertex is colored with a free color, one that is not the color of one of its neighbors.
- Since the maximum degree is $\Delta$, there is always a free color among $1, 2, \ldots, \Delta + 1$ (a pigeon hole argument).
A Proof by Induction

★ Assume $G$ has $n$ vertices.
★ The theorem is true for a graph with 1 vertex since $\Delta = 0$.
★ Let $n \geq 2$ and assume that the theorem is correct for any graph with $n - 1$ vertices.
★ Omit an arbitrary vertex and all of its edges from the graph.
★ By the induction hypothesis the remaining graph can be colored with at most $\Delta + 1$ colors.
★ Since the degree of the omitted vertex is at most $\Delta$ it follows that one of the colors $1, \ldots, \Delta + 1$ will be available to color the omitted vertex (a pigeon hole argument).
The star graph is a bipartite graph and therefore can be colored with 2 colors.

$\Delta = n - 1$ in a star graph. The above theorem guarantees a performance that is very far from the optimal performance.
First-Fit Implementation

★ Consider the vertices in any sequence.
★ Color a vertex with the smallest available color.

Greedy Coloring \((G)\)

\[
\text{for } i = 1 \text{ to } n \\
\quad c = 1 \\
\quad \text{while } (\exists j \{(i, j) \in E\}) \text{ AND } (c(j) = c) \\
\quad \quad c = c + 1 \\
\quad c(i) = c
\]

**Complexity:** Possible in \(O(m + n)\) time.
Sometimes Greedy is optimal

**Complete graphs:** $\Delta = n - 1$ and $n = \Delta + 1$ colors are required.

**Odd-length cycles:** $\Delta = 2$ and 3 colors are required.
The order of the vertices is crucial

A bipartite graph $G$.

- $2k$ vertices $v_1, v_2, \ldots, v_k$ and $u_1, u_2, \ldots, u_k$.
- All $(v_i, u_j)$ edges for $1 \leq i \neq j \leq k$. 
Suppose the order is \( v_1, v_2, \ldots, v_k, u_1, u_2, \ldots, u_k \).

- The algorithm colors \( G \) with 2 colors.
A Bad Order

★ Suppose the order is \( v_1, u_1, v_2, u_2, \ldots, v_k, u_k \).

- The algorithm colors \( G \) with \( k \) colors.
**Greedy with a Decreasing Order of Degrees**

**Notation:** Let the vertices be $v_1, v_2, \ldots, v_n$ and let their degrees be $\Delta = d_1 \geq d_2 \geq \cdots \geq d_n$.

**Theorem:** $\chi(G) \leq \max_{1 \leq i \leq n} \min \{d_i + 1, i\}$.

**Proof:**

- The input order for greedy is $v_1, v_2, \ldots, v_n$.
- When coloring $v_i$ at most $i - 1$ colors are used by its neighbors since greedy has colored only $i - 1$ vertices.
- When coloring $v_i$ at most $d_i$ colors are used by its neighbors because the degree of $v_i$ is $d_i$. 
Back Degrees

Notation:
★ Let the vertices be $v_1, v_2, \ldots, v_n$ and let their degrees be $d_1, d_2, \ldots, d_n \leq \Delta$.
★ Let $d'_i \leq d_i$ be the number of neighbors of $v_i$ among $v_1, \ldots, v_{i-1}$ (in particular: $d'_i \leq i - 1$).

Theorem: $\chi(G') \leq \max_{1 \leq i \leq n} \{d'_i + 1\}$.

Proof:
★ The input order for greedy is $v_1, v_2, \ldots, v_n$.
★ When coloring $v_i$ at most $d'_i$ colors are used by its neighbors.
A Marginal Improvement to the Greedy Algorithm

**Theorem:** A connected non-clique $G$ can be colored with $\Delta$ colors where $\Delta \geq 3$ is the maximum degree in $G$.

**Cliques:** $K_n$ requires $n = \Delta + 1$ colors.

**Cycles:** $C_n$, for an odd $n$, requires $3 = \Delta + 1$ colors.
By induction implying an algorithm.

Let $v$ be an arbitrary vertex with degree $d(v)$.

Let $G' = G \setminus \{v\}$:
- If $G'$ is not a clique or a cycle, then color it recursively with $\Delta$ colors.
- If $G'$ is a clique, then it is a $K_\Delta$ graph that can be colored with $\Delta$ colors. $G'$ cannot be a $K_{\Delta+1}$ graph since then the neighbors of $v$ would have degree $\Delta + 1$.
- If $G'$ is a cycle, then it can be colored with $3 \leq \Delta$ colors.
★ If \( d(v) \leq \Delta - 1 \), then color \( v \) with a free color (pigeon hole argument).

★ If \( v \) has 2 neighbors colored with the same color, then color \( v \) with a free color (pigeon hole argument).

★ From now on assume that \( d(v) = \Delta \) and that each neighbor of \( v \) is colored with a different color.
Let the neighbors of $v$ be $v_1, v_2, \ldots, v_\Delta$ and let their colors be $c_1, c_2, \ldots, c_\Delta$ respectively.
Definitions and an Observation

★ For colors $x$ and $y$, let $G(x, y)$ be the subgraph of $G$ containing only the vertices whose colors are $x$ or $y$.

★ For a vertex $w$ whose color is $x$, let $G_w(x, y)$ be the connected component of $G(x, y)$ that contains $w$.

★ Interchanging the colors $x$ and $y$ in the connected component $G_w(x, y)$ of $G(x, y)$ results with another legal coloring in which the color of $w$ is $y$. 
The Observation
Let $v_i$ and $v_j$ be any 2 neighbors of $v$.

If $G_{v_i}(c_i, c_j)$ does not contain $v_j$, then interchange the colors $c_i$ and $c_j$ in $G_{v_i}(c_i, c_j)$.

The color of both $v_i$ and $v_j$ is now $c_j$ and no neighbor of $v$ is colored with $c_i$.

Color $v$ with $c_i$. 
Graph Algorithms 51

Proof Continue

\[
V_i \quad V_d \quad V \quad V_1 \quad V_2
\]

\[
V_i \quad V_j \quad V \quad V_1 \quad V_2
\]
If \( v_i \) has 2 neighbors colored with \( c_j \), then color \( v_i \) with a different color and color \( v \) with \( c_i \).
From now on assume that $v_i$ and $v_j$ belong to the same connected component in $G(c_i, c_j)$ and that $v_i$ has only 1 neighbor colored with $c_j$. 
If $G_{v_i}(c_i, c_j)$ is not a path, then let $w \in G_{v_i}(c_i, c_j)$ be the closest to $v_i$ whose color is $c_i$ (or $c_j$) and who has more than 2 neighbors whose colors are $c_j$ (or $c_i$).
⋆ Color \( w \) with a different color.

⋆ \( v_i \) and \( v_j \) are not anymore in the same connected component of \( G(c_i, c_j) \).
★ Interchange the colors $c_i$ and $c_j$ in $G_{v_i}(c_i, c_j)$.
★ The color of both $v_j$ and $v_i$ is now $c_j$ and no neighbor of $v$ is colored with $c_i$: Color $v$ with $c_i$. 
From now on assume that for any 2 neighbors $v_i$ and $v_j$ of $v$, the subgraph $G_{v_i}(c_i, c_j)$ is a path (could be an edge) starting with $v_i$ and ending with $v_j$. 
If for some $v_k$ the path $G_{v_i}(c_i, c_k)$ intersects the path $G_{v_i}(c_i, c_j)$ in a vertex $w \neq v_i$ whose color is $c_i$, then $w$ has 2 neighbors colored $c_k$ and 2 neighbors colored $c_j$.

Color $w$ with a different color than $c_i, c_j, c_k$. 

Graph Algorithms
\* $v_i$ and $v_j$ are not anymore in the same connected component of $G(c_i, c_j)$.

\* Interchange the colors $c_i$ and $c_j$ in $G_{v_i}(c_i, c_j)$.

\* The color of both $v_j$ and $v_i$ is now $c_j$ and no neighbor of $v$ is colored with $c_i$: Color $v$ with $c_i$. 
⋆ From now on assume that the path $G_{v_i}(c_i, c_k)$ intersects the path $G_{v_i}(c_i, c_j)$ only at $v_i$. 
By assumption the graph is not a clique. Therefore, there exist $2$ neighbors of $v$, $v_i$ and $v_j$, that are not adjacent. Let $w$ be the $c_j$ neighbor of $v_i$.

By assumption $\Delta \geq 3$. Therefore, there exists another neighbor of $v$, $v_k$ that is different than $v_i$ and $v_j$. 

Proof Continue
★ Interchange the colors $c_i$ and $c_k$ in $G_{v_i}(c_i, c_k)$.
★ The color of $v_i$ is $c_k$ and the color of $v_k$ is $c_i$. 
★ Repeat the arguments as before and assume that
  - $G_{v_j}(c_j, c_i)$ is a path from $v_j$ to $v_k$.
  - $G_{v_j}(c_j, c_k)$ is a path from $v_j$ to $v_i$.
★ These paths must intersect with $w$ because $w$ is the only $c_j$ neighbor of $v_i$. 
• Color $w$ with a different color than $c_i, c_j, c_k$.

• $v_i$ has no $c_j$ neighbor.
* Color $v_i$ with $c_j$.
* Color $v$ with $c_k$. 

Proof End
Complexity

★ Possible in $O(nm)$. 

★ Each correction can be done in $O(m)$.
Definition: A cubic graph is a regular graph in which the degree of every vertex is 3.

Corollary: The chromatic number of a non-bipartite cubic graph that is not $K_4$ is 3.
**Coloring 3-Colorable Graphs with $O(\sqrt{n})$ colors**

**Observation:** In 3-colorable graphs, the subgraph containing only the neighbors of a particular vertex is a 2-colorable graph (a bipartite graph).
Let $G$ be a 3-colorable graph.

Allocate 3 colors to a vertex and all of its neighbors if the degree of this vertex is larger than $\sqrt{n}$.

There are at most $\sqrt{n}$ such vertices and therefore so far at most $3\sqrt{n}$ colors were used.
Now, all the degrees in the graph are less than $\sqrt{n}$.

The greedy algorithm needs at most $\sqrt{n}$ colors to color the rest of the graph.

All together, the algorithm uses $O(\sqrt{n})$ colors.

- If all omitted vertices are colored with the same color, then at most $2\sqrt{n} + 1$ colors are used before applying the greedy algorithm.
- Therefore, the algorithm uses about $3\sqrt{n}$ colors.