Graph Coloring

CSC 1300 – Discrete Structures
Villanova University

Major Themes

- Vertex coloring
- Chromatic number $\chi(G)$
- Map coloring
- Greedy coloring algorithm
- Applications

Vertex Colorings

Adjacent vertices cannot have the same color

$H : \begin{array}{c}
1 & 2 & 3 \\
2 & 3 & 4 \\
3 & 4 & 1 \\
4 & 1 & 2 \\
\end{array}$

What is the least number of colors needed for the vertices of this graph so that no two adjacent vertices have the same color?

$\chi(G) =$

Chromatic number $\chi(G) =$ least number of colors needed to color the vertices of a graph so that no two adjacent vertices are assigned the same color.


What is the least number of colors needed for the vertices of this graph so that no two adjacent vertices have the same color?

More Examples


Map Coloring

What is the least number of colors needed to color a map?

Region ➔ vertex
Common border ➔ edge

Map Coloring
What is the least number of colors needed to color a map?

Coloring the USA

Four color theorem
Every planar graph is 4-colorable

The proof of this theorem is one of the most famous and controversial proofs in mathematics, because it relies on a computer program. It was first presented in 1976. A more recent formulation can be found in this article:


Do you always need four colors?

C₆  C₉  Cₙ
What about non-planar graphs?

- $K_5$
- $K_{3,3}$
- $K_6, K_7, K_8, ..., K_n$?
- $K_{3,4}, K_{4,7}, K_{5,18}, ..., K_{n,m}$?

**Four color theorem**

Every planar graph is 4-colorable

**Chromatic Numbers of Some Graphs**

- For $K_n$, the complete graph with $n$ vertices, $\chi(K_n) =$
- For any bipartite graph $G$, $\chi(G) =$
- For $C_n$, the cycle with $n$ vertices, $\chi(C_n) =$
- $\chi(G) = 1$ iff ...
- For any planar graph $G$, $\chi(G) \leq 4$ (Four Color Theorem)

**Applications of Graph Coloring**

- map coloring
- scheduling
  - eg: Final exam scheduling
- Frequency assignments for radio stations
- Index register assignments in compiler optimization
- Phases for traffic lights

**Example: Schedule these exams, avoiding conflicts**

<table>
<thead>
<tr>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSC 2053</td>
<td>CSC 1052</td>
<td>CSC 1300</td>
</tr>
<tr>
<td>CSC 1700</td>
<td>CSC 2400</td>
<td>CSC 4480</td>
</tr>
<tr>
<td>CSC 2014</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CSC 2053</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CSC 1052</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CSC 1300</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CSC 4480</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CSC 2014</td>
</tr>
</tbody>
</table>

Dr Papalaskari
Scheduling constraints – edges represent conflicts

Computing the Chromatic Number

There is no efficient algorithm for finding $\chi(G)$ for arbitrary graphs. Most computer scientists believe that no such algorithm exists.

Greedy algorithm: **sequential coloring**:
1. Order the vertices in nonincreasing order of their degrees.
2. Scan the list to color each vertex in the first available color, i.e., the first color not used for coloring any vertex adjacent to it.

Graph coloring sequential algorithm: Assign colors in order

Example: Index Registers

<table>
<thead>
<tr>
<th>Code Sequence</th>
<th>Interference Graph</th>
<th>Colored Graph</th>
<th>Final Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A = ...$</td>
<td>$A$</td>
<td>$A$</td>
<td>$R1 = ...$</td>
</tr>
<tr>
<td>$B = ...$</td>
<td>$B$</td>
<td>$B$</td>
<td>$R2 = ...$</td>
</tr>
<tr>
<td>$C = ...$</td>
<td>$C$</td>
<td>$C$</td>
<td>$R3 = ...$</td>
</tr>
<tr>
<td>... $A = ...$</td>
<td>... $A = ...$</td>
<td>... $A = ...$</td>
<td>... $R1 = ...$</td>
</tr>
<tr>
<td>... $D = ...$</td>
<td>... $D = ...$</td>
<td>... $D = ...$</td>
<td>... $R2 = ...$</td>
</tr>
<tr>
<td>... $C = ...$</td>
<td>... $C = ...$</td>
<td>... $C = ...$</td>
<td>... $R3 = ...$</td>
</tr>
</tbody>
</table>


Source: http://www.computer.org/portal/web/csdl/doi/10.1109/TSE.2010.111

http://upload.wikimedia.org/wikipedia/commons/0/00/Greedy_colourings.svg

http://www.cis.ksu.edu/~hastings/comp/graphcoloring.png

Not always optimal! (order matters)