Graph colouring II

1 Graph Colouring

A proper colouring of a graph with k colours:

- A colouring of the vertices, such that any two adjacent vertices have different colours

The graph colouring problem:

- Find the smallest number k such that the graph has a proper k colouring

A brute force algorithm for 3 colouring:

- exhaustively enumerate all possible 3-colourings
- For each colouring, check whether all vertex pairs have different colours
- correct answer is always guaranteed
- Exponential time needed (3^n iterations)

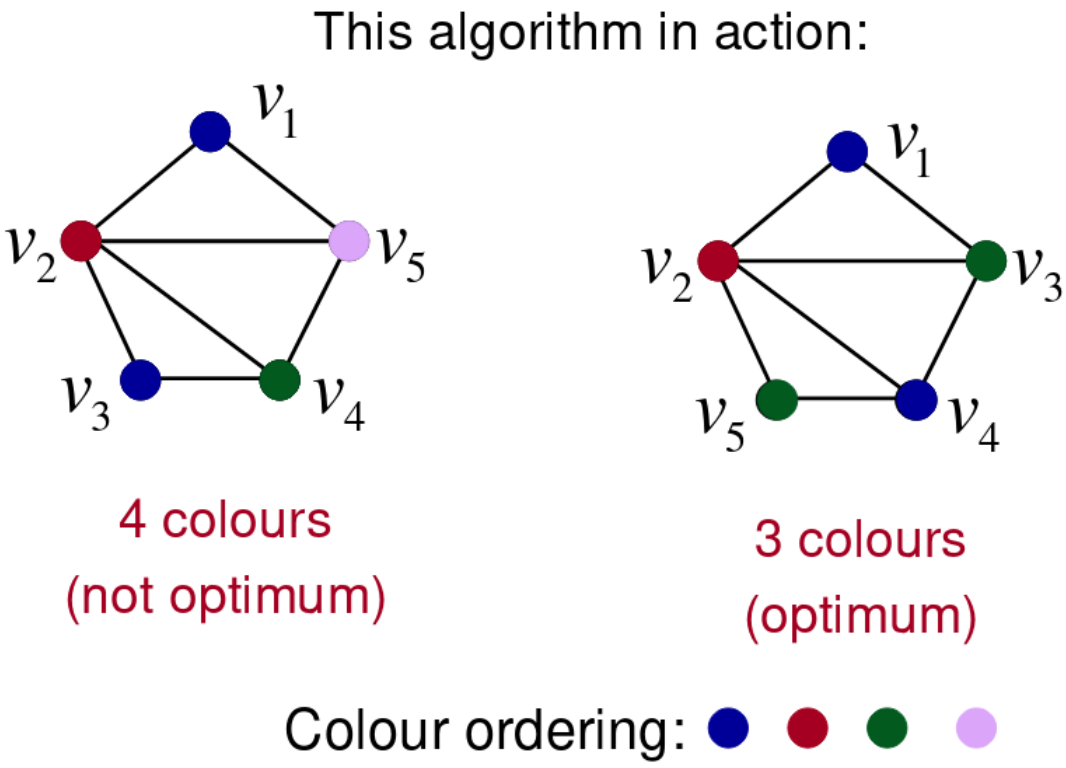
Sometimes we just want ”some” proper colouring:

- not necessarily with the smallest number of colours
- but we want it quickly

Then: a greedy algorithm

2 A ”greedy” algorithm

- Assign labels to the vertices $v_1, v_2, v_3, v_4, ..., v_n$ such that
  v_2 is adjacent to v_1
  v_3 is adjacent to one of $\{v_1, v_2\}$
  v_4 is adjacent to one of $\{v_1, v_2, v_3\}$
  ...
  v_n is adjacent to one of $\{v_1, v_2, v_3, ..., v_{n-1}\}$
- Visit the vertices sequentially
- Assign to each vertex the ”smallest” colour that is not being used by its neighbours so far



- This greedy algorithm:
  - finishes quickly (polynomial number of iterations)
  - Makes at every step a decision that ”looks the best for now”
  - but the final output may not be the best possible
- The choice of vertex ordering:
  - determines the quality of the solution (i.e. number of colours)
- It is hard (NP-complete) to compute the best possible vertex ordering

3 Which algorithm to choose

	brute-force algorithm	greedy algorithm
+	computes always the optimum	finishes quickly (polynomial time)
-	finishes very slowly (exponential time)	not always the optimum solution

Two measures of algorithmic performance:

- quality of the solution (how far from the optimum)
- time needed

4 A ”greedy” algorithm

- Ok, it is hard to find the optimum in general
- What is I just look for specific small values of k?
- For k=1
  - only the trivial graph (one vertex) can be coloured with only one colour
- For k=2 bipartite graphs

5 Bipartite graph

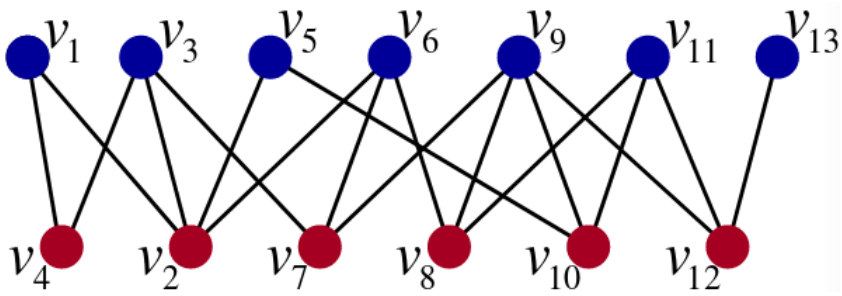
- Our greedy algorithm
  - always decides correctly whether 2 colours are enough for the graph
  - for any choice of the vertex ordering
- In other words, this algorithm:
  - constructs a proper 2 colouring, if one exist
  - otherwise reports that we need more colours

6 Greedy with 2 colours

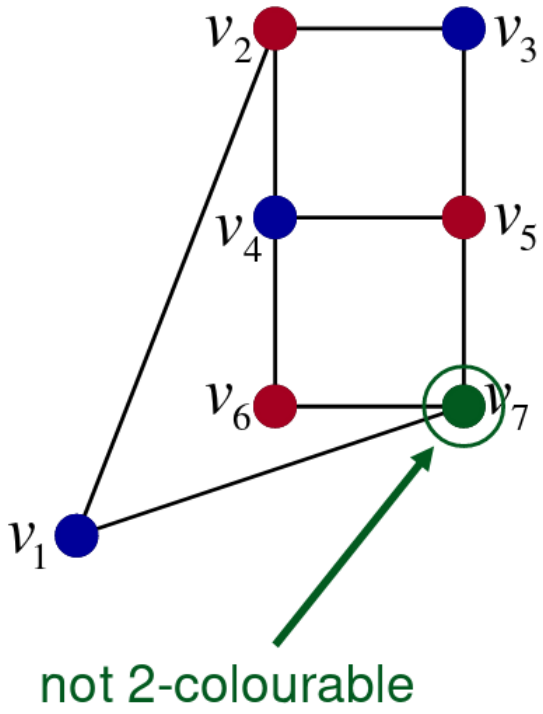
This algorithm in action:

- Colour the first vertex blue
- Thus the next vertex (which is neighbour) is necessarily red
- and the next one is necessarily blue
- etc

In other words



In some cases the last vertex needs a third colour



7 Bipartite graphs

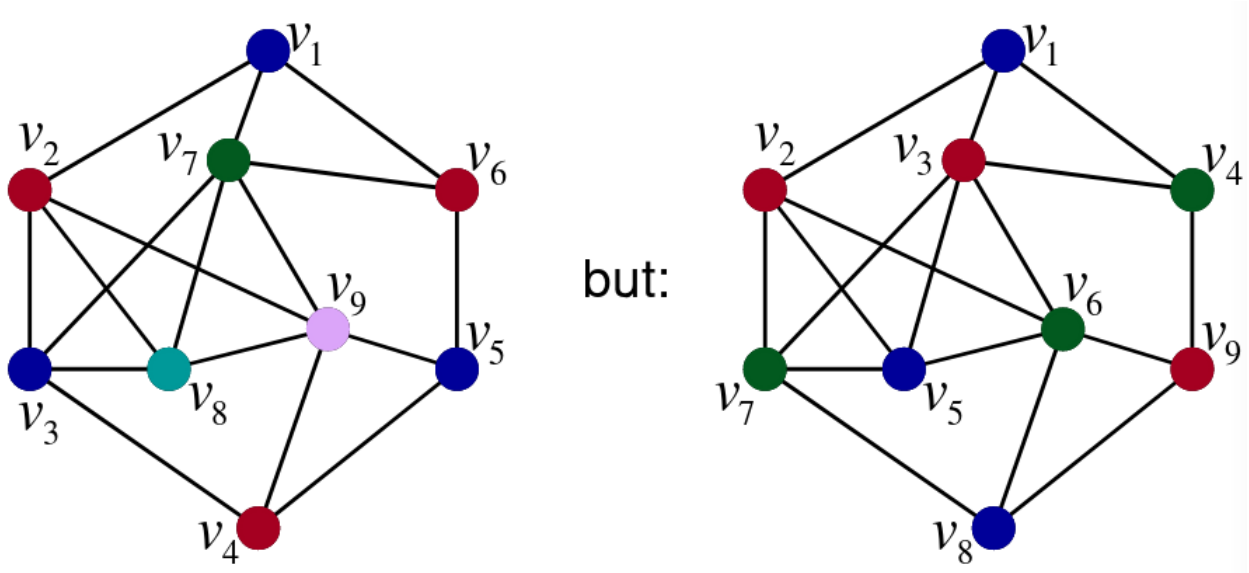
- Therefore:
  - Using the greedy 2 colouring algorithm we can decide whether a given graph is bipartite
- Is there any other way to do this?

Theorem: A graph is bipartite if, and only if it has no cycles with odd length

- Can we use this theorem algorithmically? There can be too many cycles in a graph (exponential)
- we can not check all of them efficiently

8 Greedy colouring

- Can the greedy algorithm solve 3 colouring?



greedy can not be used for 3 colouring

9 Colouring of planar graphs

Consider an arbitrary map:

- We want to distinguish different countries using different colours
- How many colours do we need

This is the famous 4 colour problem.
How can we model a map by a graph

- region of the map ↔ vertex of the graph
- an edge when two regions have common borders

Planar graphs:

- the graphs that are obtained by maps equivalently
- the graphs that can be drawn on the plane such that no two edges cross each other