

Algorithms and Complexity

Backtracking

Lecturer: Dr. Alaa Ahmed Abboud

Lecture 8.

Class 2nd.

Time: 8:30-10:30

Department: Businesses Information Technology (BIT)

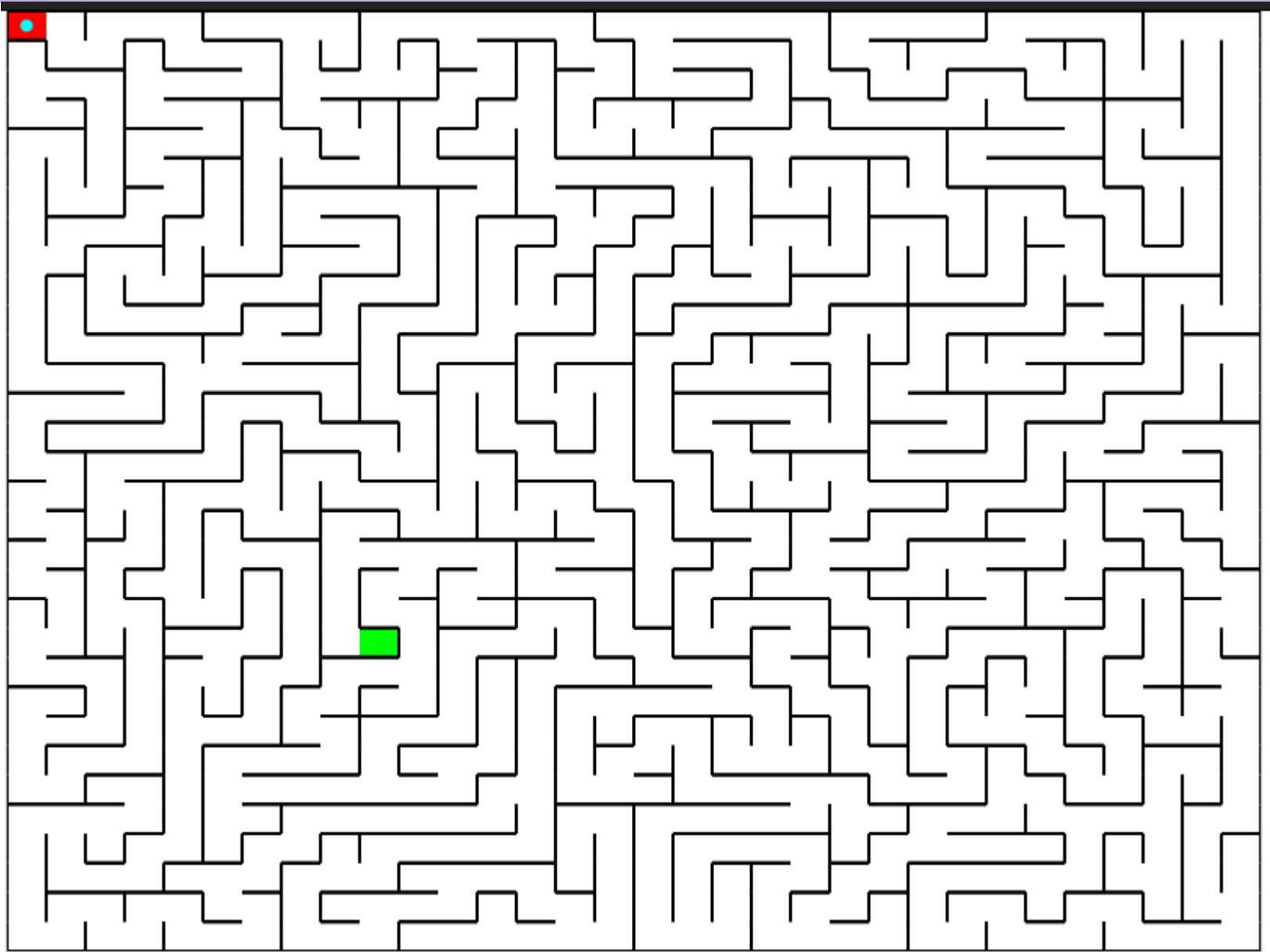
Backtracking

- Suppose you have to make a series of *decisions*, among various *choices*, where you don't have enough information to know what to choose
 - Each decision leads to a new set of choices
 - Some sequence of choices (possibly more than one) may be a solution to your problem
- **Backtracking** *is a methodical way of trying out various sequences of decisions, until you find one that “works”*

Solving a maze

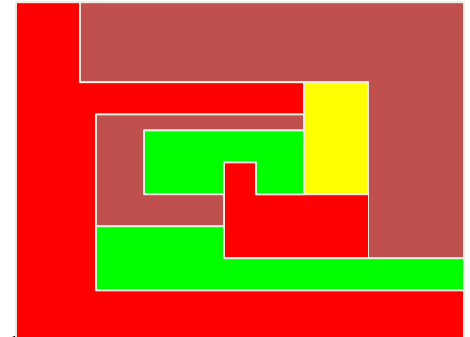
- Given a maze, find a path from start to finish. At each intersection, you have to decide between three or fewer choices:
 - ✓ Go straight
 - ✓ Go left
 - ✓ Go right
- You don't have enough information to choose correctly. Each choice leads to another set of choices. One or more sequences of choices may (or may not) lead to a solution. Many types of maze problem can be solved with backtracking

Solving a maze



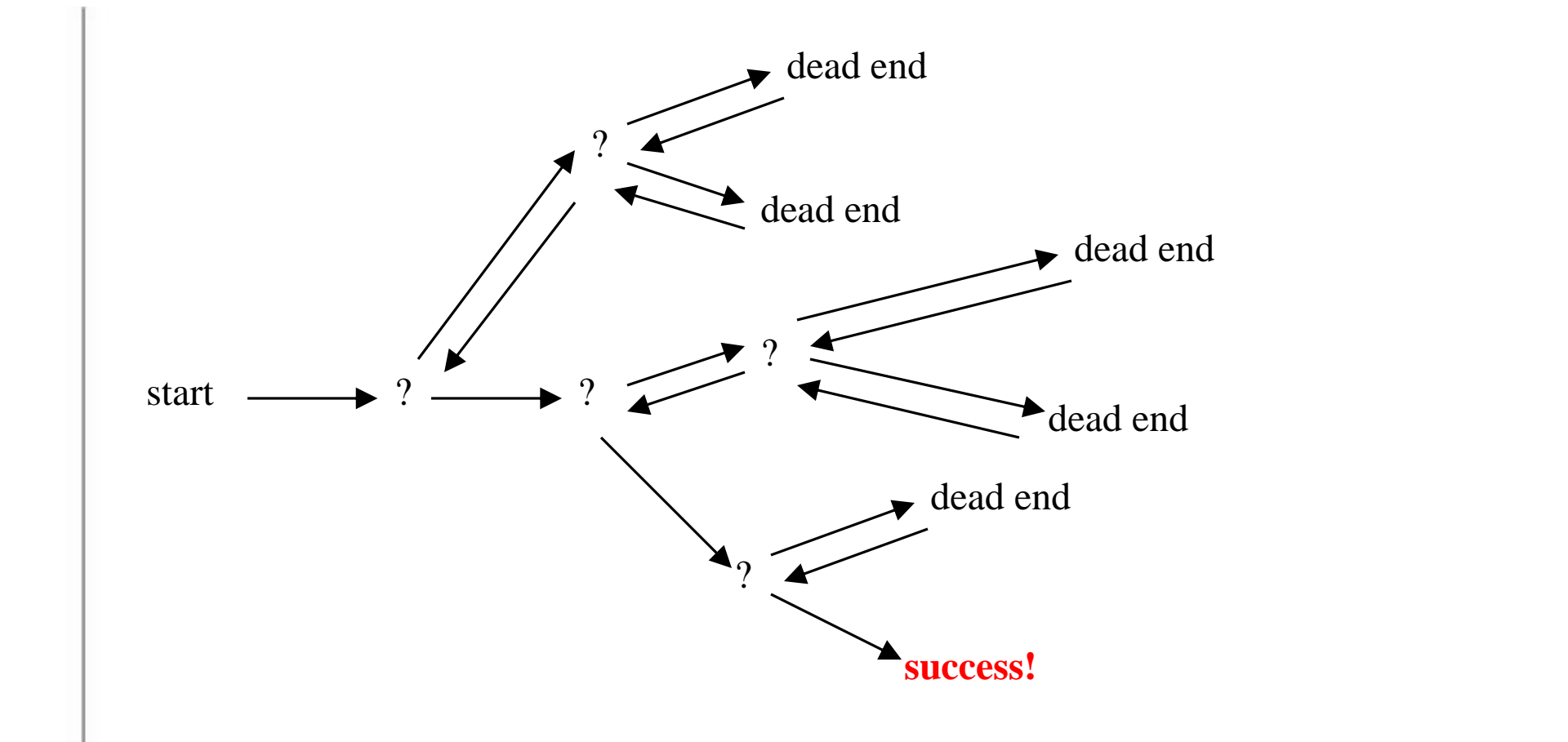
Coloring a map

- You wish to color a map with not more than four colors red, yellow, green, blue
- Adjacent countries must be in different colors.
- You don't have enough information to choose colors.



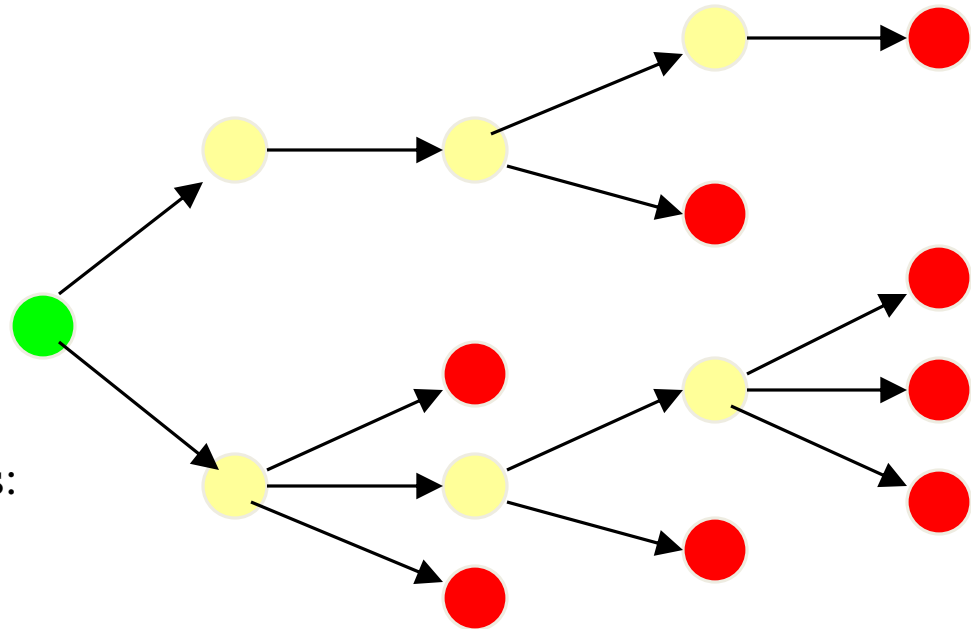
Each choice leads to another set of choices. One or more sequences of choices may (or may not) lead to a solution. Many coloring problems can be solved with backtracking

Backtracking (animation)






Terminology I

A tree is composed of **nodes**



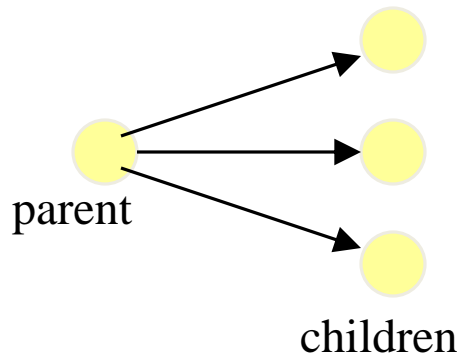
There are three kinds of nodes:

-  The (one) **root** node
-  **Internal** nodes
-  **Leaf** nodes

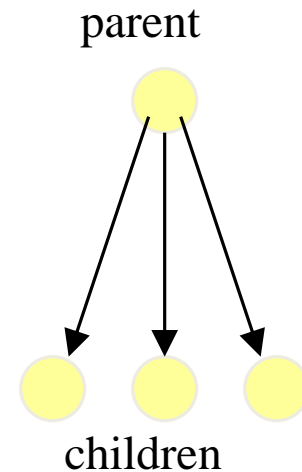
Backtracking can be thought of as searching a tree for a particular “goal” leaf node

Terminology II

- Each non-leaf node in a tree is a **parent** of one or more other nodes (its **children**)
- Each node in the tree, other than the root, has exactly one **parent**



Usually, however, we draw our trees *downward*, with the root at the top



Real and virtual trees

- There is a type of data structure called a tree. But we are **not** using it here.
- If we diagram the sequence of choices we make, the diagram looks like a tree.

The backtracking algorithm

Backtracking is really quite simple--we “explore” each node, as follows:

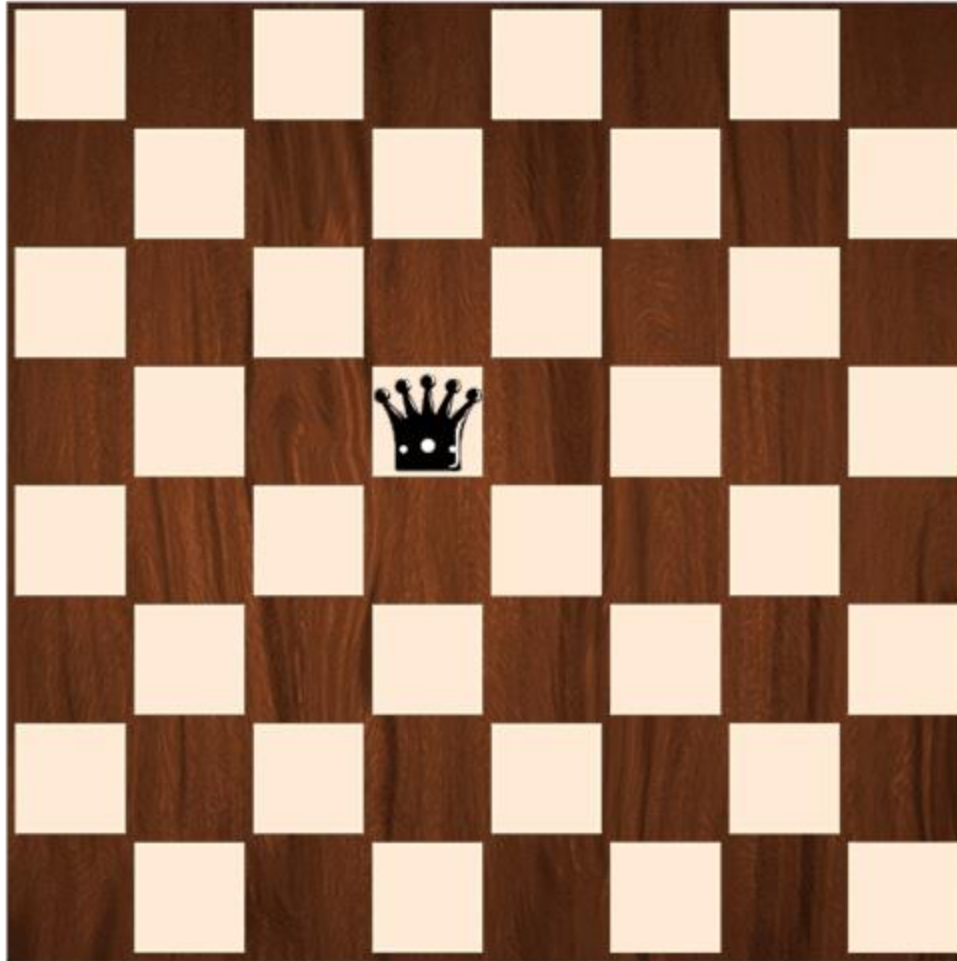
To “explore” node N:

1. If N is a goal node, return “success”
2. If N is a leaf node, return “failure”
3. For each child C of N,
 - 3.1. Explore C
 - 3.1.1. If C was successful, return “success”
4. Return “failure”

N-Queens Problem

- The problem is to place n queens on an $n \times n$ chessboard so that no two queens attack each other by being in the same row or in the same column or on the same diagonal.
- For $n = 1$, the problem has a trivial solution, and it is easy to see that there is no solution for $n = 2$ and $n = 3$.
- So let us consider the four-queens problem and solve it by the backtracking technique.

N -Queens Problem



Cont..

- Since each of the four queens has to be placed in its own row, all we need to do is to assign a column for each queen on the board presented in Figure .
- We start with the empty board and then place queen 1 in the first possible position of its row, which is in column 1 of row 1.
- Then we place queen 2, after trying unsuccessfully columns 1 and 2, in the first acceptable position for it, which is square (2, 3), the square in row 2 and column 3.

Cont..

- This proves to be a dead end because there is no acceptable position for queen 3. So, the algorithm backtracks and puts queen 2 in the next possible position at (2, 4). Then queen 3 is placed at (3, 2), which proves to be another dead end.
- The algorithm then backtracks all the way to queen 1 and moves it to (1, 2). Queen 2 then goes to (2,4), queen 3 to (3, 1), and queen 4 to (4, 3), which is a solution to the problem. The state-space tree of this search is shown in Figure.

Board for the four-queens problem

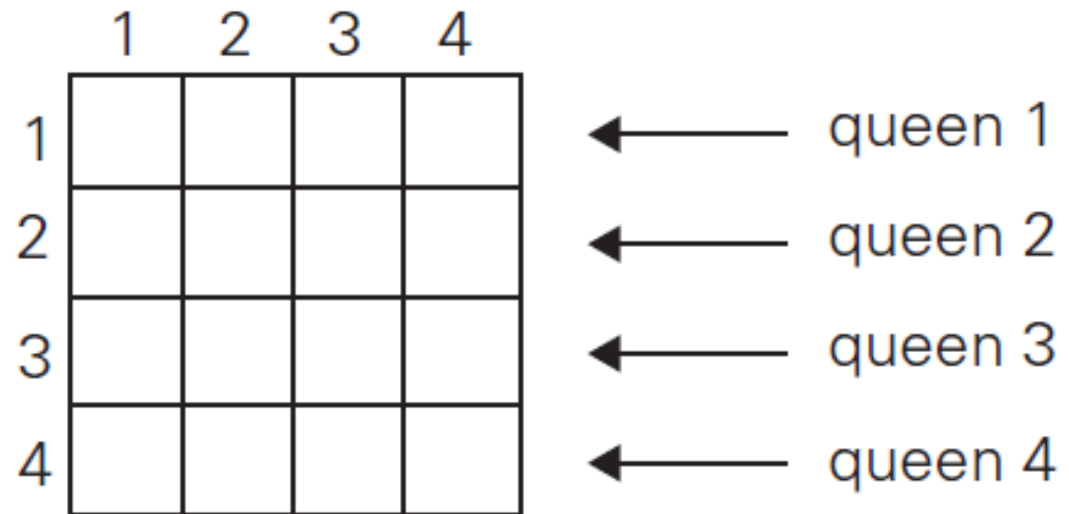


FIGURE 12.1 Board for the four-queens problem.

Cont..

- State-space tree of solving the four-queens problem by backtracking. \times denotes an unsuccessful attempt to place a queen in the indicated column.
- The numbers above the nodes indicate the order in which the nodes are generated.

Cont..



Coloring Problem

Coloring Problem

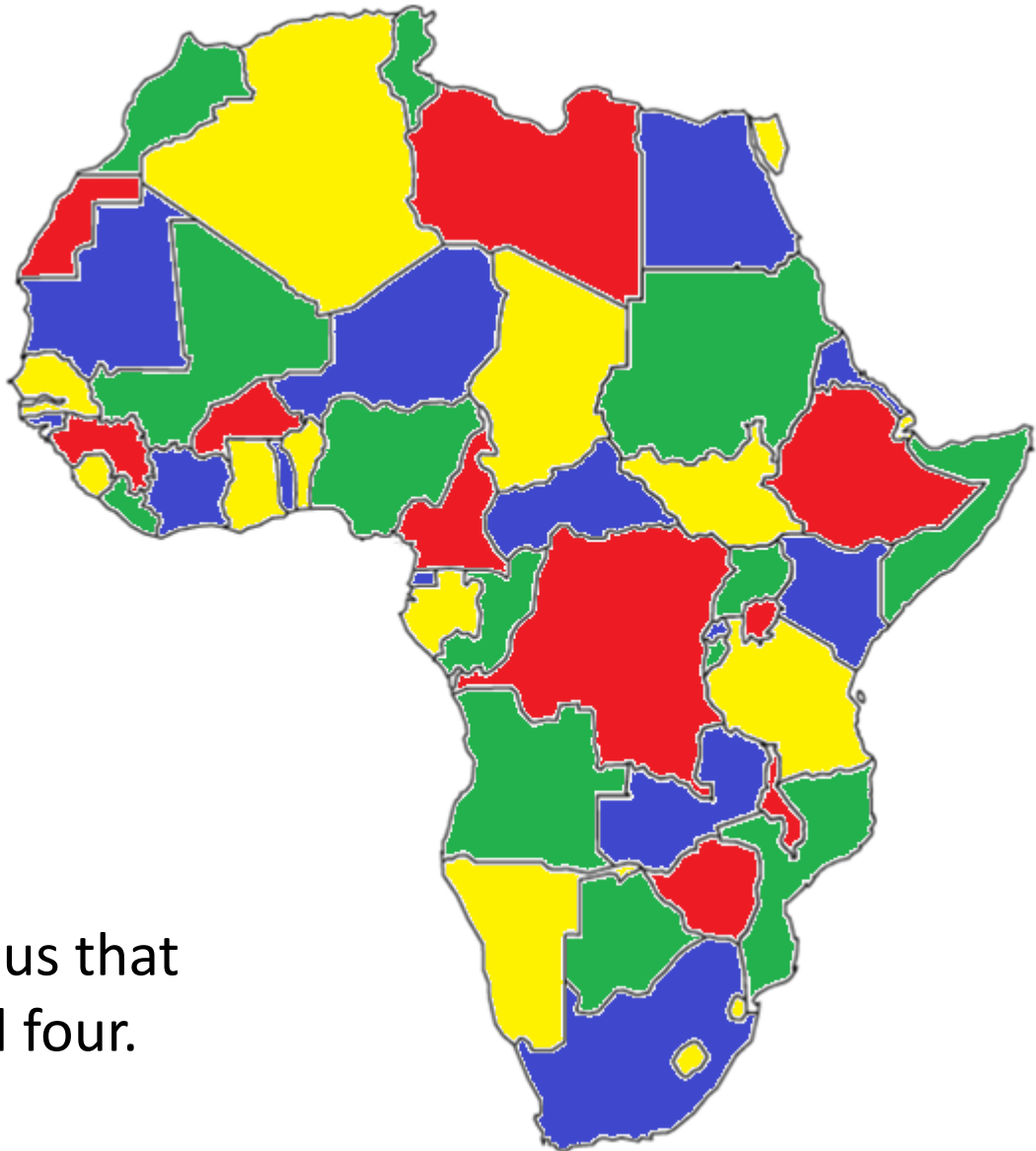
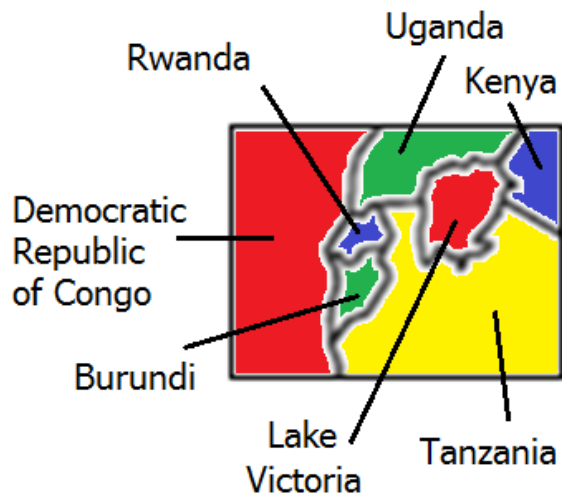
You want to shade the map of countries on the right figure so that **no two countries sharing a border are shaded with the same colour.**



How many colours do you need?

(We won't worry about the sea...)

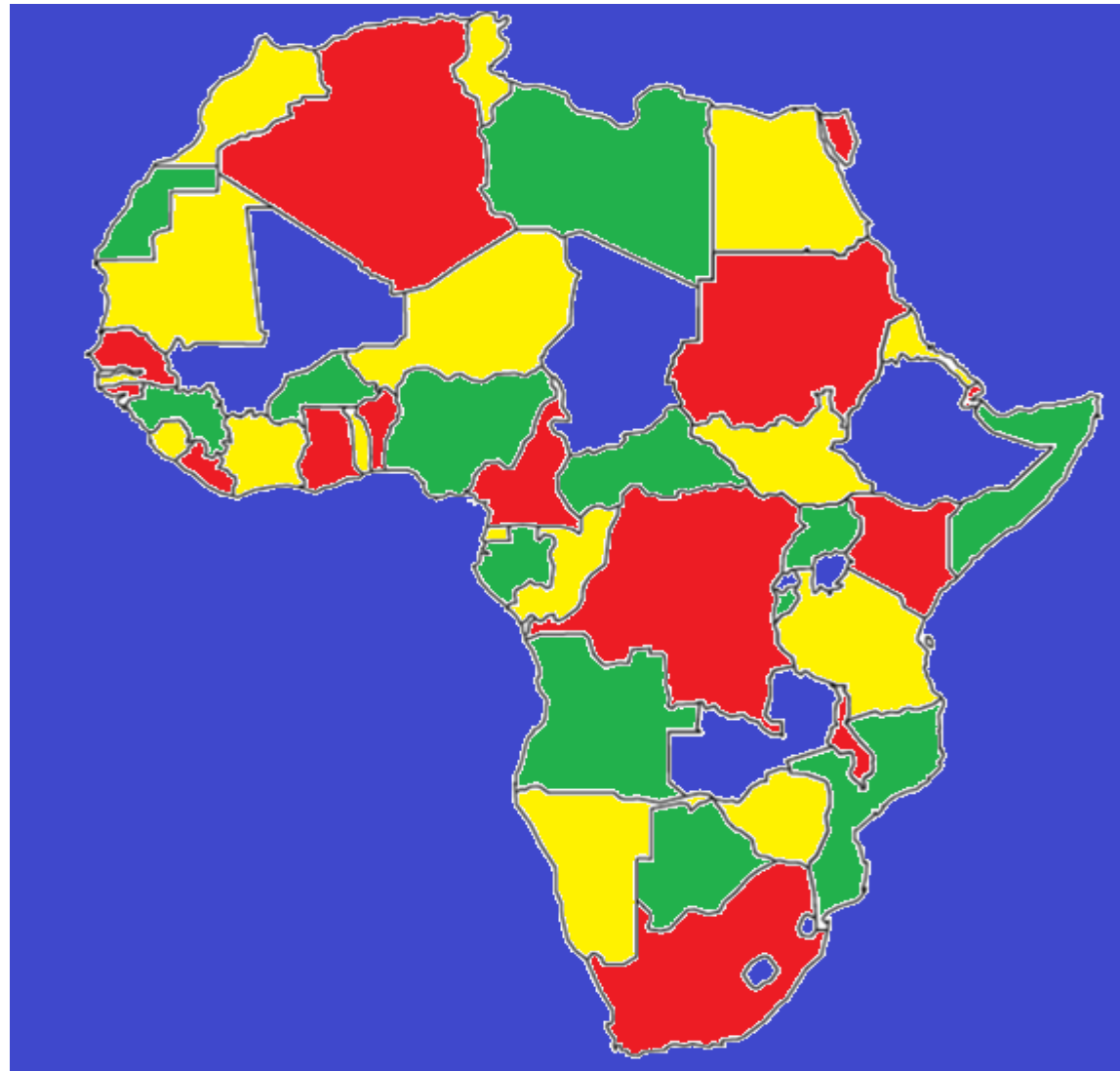
It turns out that
FOUR will do.



This section shows us that
we certainly need four.

Coloring Problem

Actually,
including the
sea is no
problem.



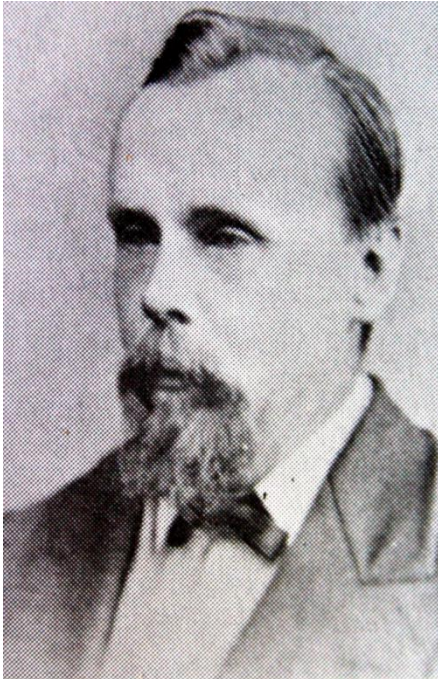
Coloring Problem

But will **four colours** do for **ANY** map?

Might there be some configuration of countries
so that **five colours** are essential? **Six**?

Coloring Problem

This problem has a fascinating history...



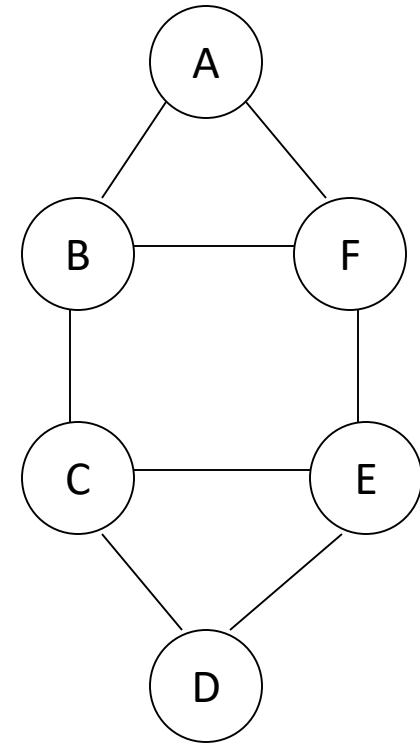
In 1852, **Frances Guthrie** (left) discovered that he seemed to be able to **four-colour** any map of countries.

In 1879, a mathematician called **Kempe** claimed to have proved this, but in 1890, **Heawood** found a fatal error in his argument.

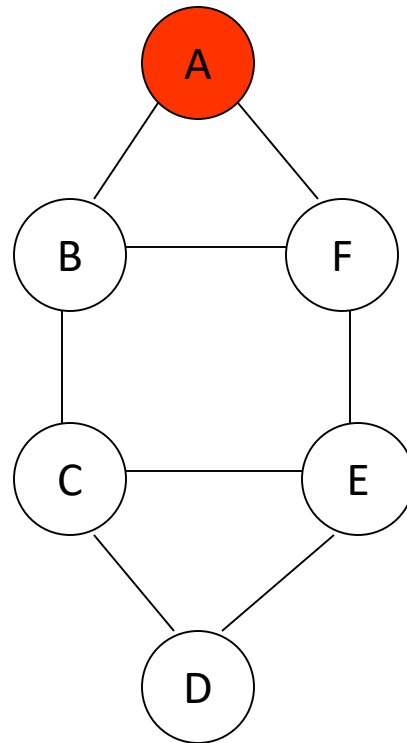
In 1922, **Franklin** proved that **four colours** would suffice for every map with 25 or fewer countries. This number was gradually increased over the years, to 95 in 1976.

Graph Coloring

- As an example:
 - The vertices are enumerated in order A-F
 - The colors are given in order: R, G, B

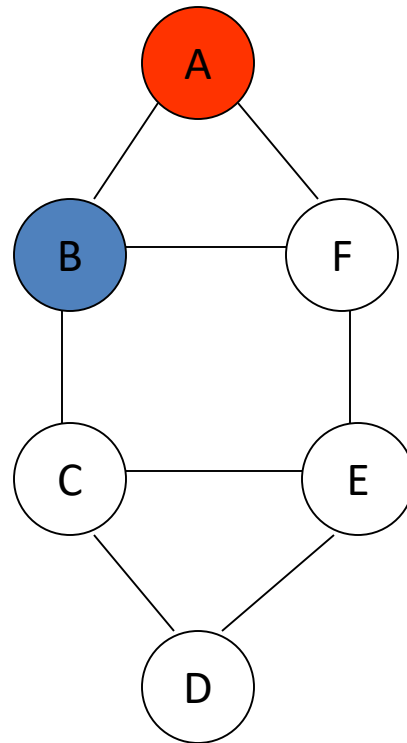


Graph Coloring



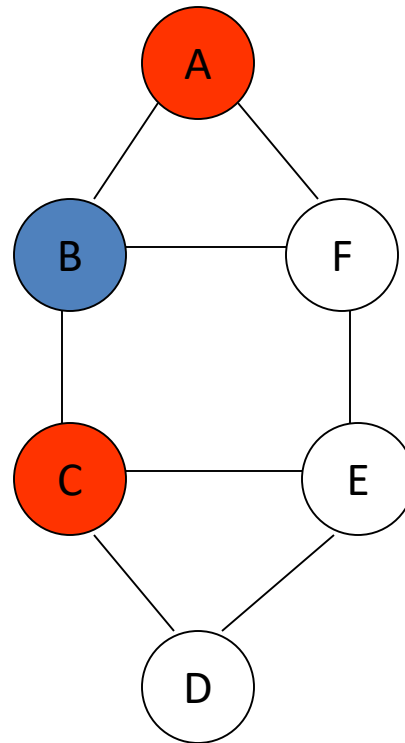
Coloring Problem

Graph Coloring



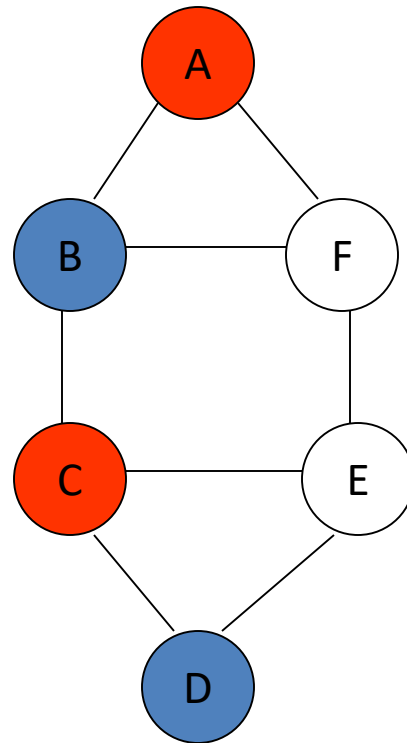
Coloring Problem

Graph Coloring



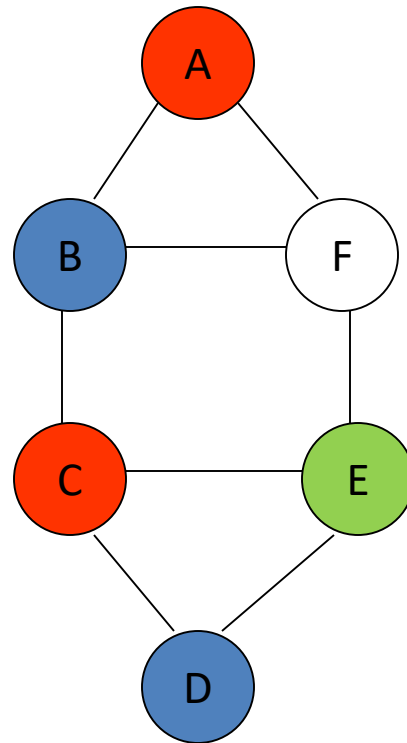
Coloring Problem

Graph Coloring



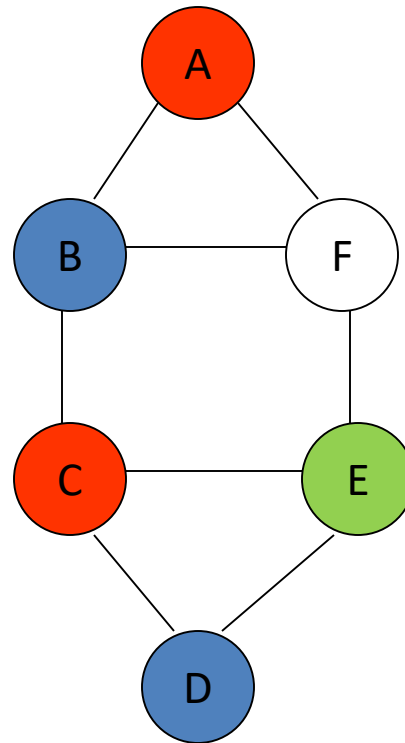
Coloring Problem

Graph Coloring



Coloring Problem

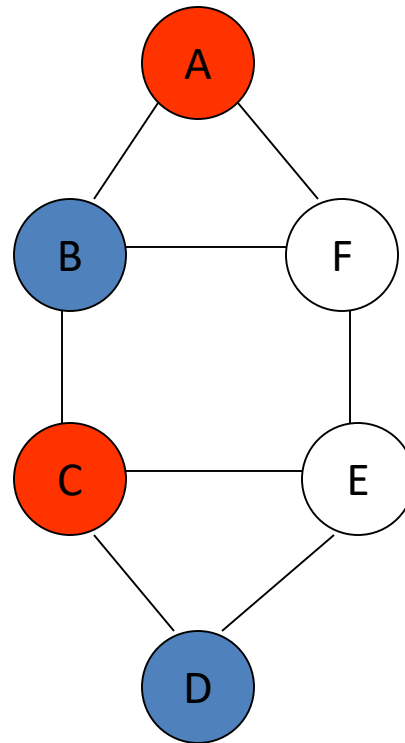
Graph Coloring



Stuck!

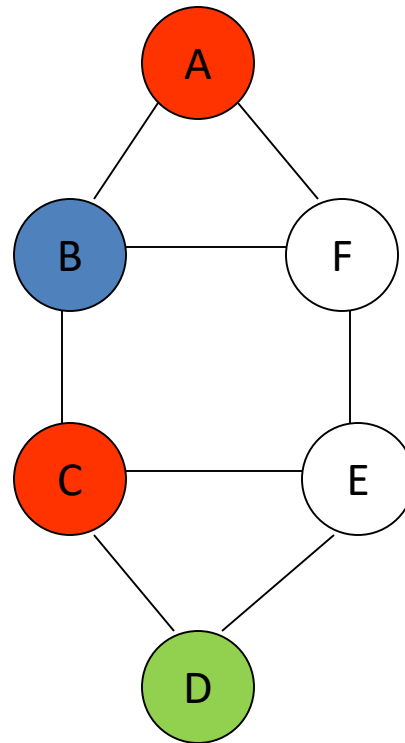
Coloring Problem

Graph Coloring



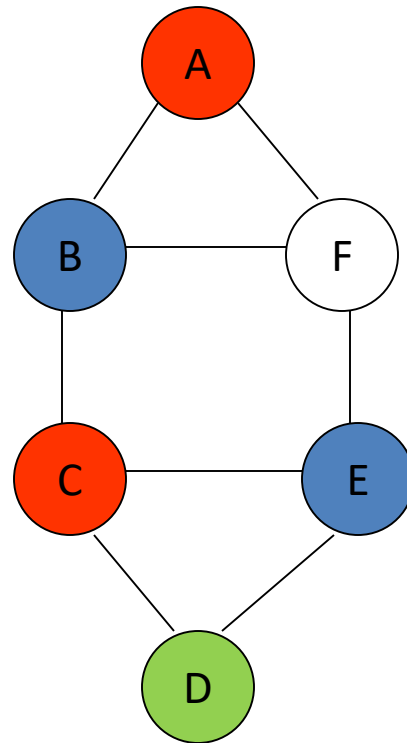
Coloring Problem

Graph Coloring



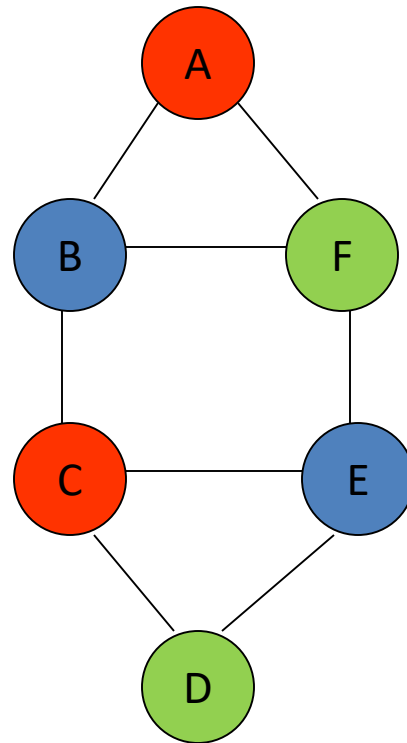
Coloring Problem

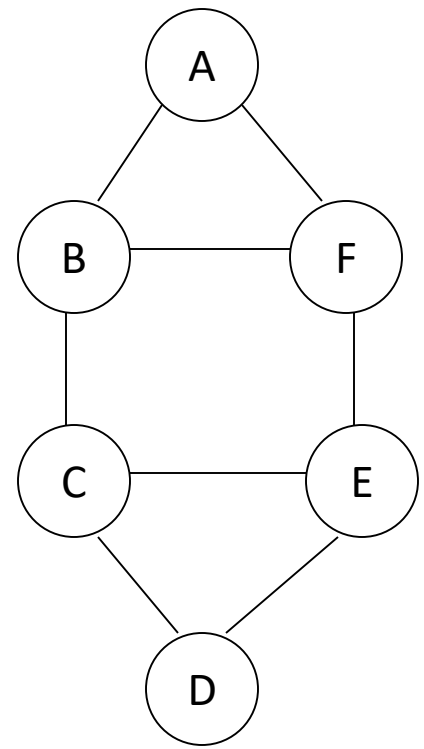
Graph Coloring



Coloring Problem

Graph Coloring



[illegible]

A
B
C
D
E
F

Assignment

Q1// Continue the backtracking search for a solution to the four-queens problem, which was started in this section, to find the second solution to the problem.

Q2// Which is the *last* solution to the five-queens problem found by the backtracking algorithm?



THANK YOU