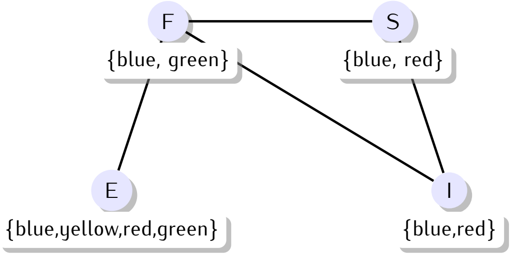
Constraint Satisfaction

Consider the problem of coloring (such that adjacent vertices have different colors) the following graph :



1. Use the Backtrack algorithm below to solve the problem. Take the variables in the order F, E, S, I; take the values in the order blue, yellow, red, green.

```
BackTrack (CSP net, Assignment a)
  if the assignment a is complete then return a
  var ← next non-assigned variable
  for each value val ∈ D(var)
      if {var=val} does not violate any constraint
          a = a ∪ {var=val}
          result ← Backtrack(net, a)
          if result ≠ failure
              return result
  return failure
```

2. Once we have assigned a value to a variable, we can look forward and limit the number of possible values.

```
ForwardChecking (CSP net, Assignment a)
  if the assignment a is complete then return a
  var ← next non-assigned variable
  for each value val ∈ D(var)
      if {var=val} does not violate any constraint
          a = a ∪ {var=val}
          for each variable v connected to var
              apply arc-consistency from v to var
          result ← Backtrack(net, a)
          if result ≠ failure
              return result
  return failure
```

   Use ForwardChecking to find a solution, using the same orders as above.

3. Repeat ForwardChecking using the following heuristics for the order on values and variables :
   — choose from the variables with the minimum remaining possible values (fail-first heuristic)
   — from the above variables, choose a variable involved in the most constraints with non-assigned variables (degree heuristic)
   — choose a value which constrains the fewest neighbours (least-constraining-value heuristic)