Forward Checking Idea.

During Backtracking:

When a variable $x$ is assigned value $v$:
- For each variable $y$ adjacent to $x$:
  - Remove from $y$'s domain any value that is inconsistent with $v$.
- If any variable has no legal values, go back up to previously assigned variable.
CSP with Backtracking and Forward Checking

Colors: rgb

<table>
<thead>
<tr>
<th>N</th>
<th>Q</th>
<th>S</th>
<th>V</th>
<th>W</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>rgb</td>
<td>rgb</td>
<td>rgb</td>
<td>rgb</td>
<td>rgb</td>
<td>rgb</td>
</tr>
<tr>
<td>2</td>
<td>b</td>
<td>b</td>
<td>gb</td>
<td>b</td>
<td>gb</td>
<td>rb</td>
</tr>
<tr>
<td>3</td>
<td>b</td>
<td>b</td>
<td>gb</td>
<td>b</td>
<td>gb</td>
<td>rb</td>
</tr>
</tbody>
</table>

Fewest Remaining Values / Highest Degree (Not always optimal)

1- N, Q, V, W, X have 2 values, Q has highest degree.

2- N: green (can choose either)

3- Q, W have 1 value, degree = 3 ⇒ tie. Arbitrarily choose Q: blue

W has empty domain so backtrack to Q
- Setting Q to green doesn't help.

- Does choosing W instead of Q help? No. Q: φ
Notice: After assigning just 2 variables, there was an inconsistency.

\[
\begin{array}{ccccccc}
N & Q & S & V & W & X & Y \\
| q | b & r | gb & b & gb & rb |
\end{array}
\]

Q & W are adjacent and have only blue in their domains!

What went wrong?

Forward checking checks only consistencies between pairs of assigned and unassigned variables, but not between pairs of unassigned variables.

So checked (N with Q), (N with W) but not: (Q with W)!

Idea: Also forward check constraints b/w unassigned variables.

Can do this easily with arcs.

After assigning N green, detect that W & Q's domains are inconsistent.