Cooperating Intelligent Systems

Constraint satisfaction

Chapter 5, AIMA

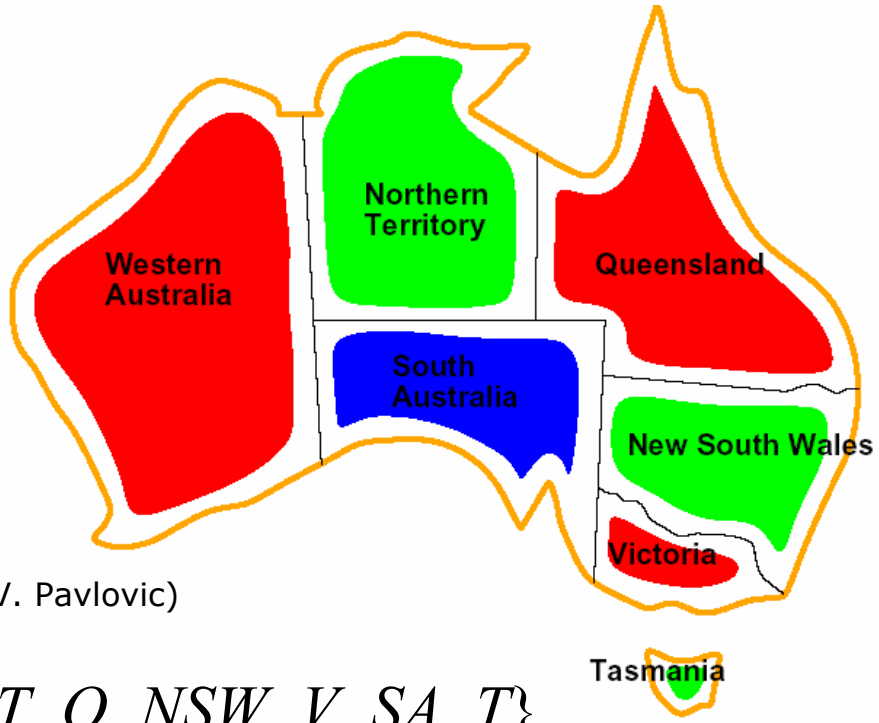
Constraint Satisfaction Problem (CSP)

- A set $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ of variables
- Each variable X_i has a nonempty domain $\mathbf{D}_i = \{v_1, v_2, \ldots, v_{k(i)}\}$ of possible values
- A set $\mathbf{C} = \{C_1, C_2, \ldots, C_m\}$ of constraints

- A solution is a complete assignment of values for the variables, which satisfies all the constraints.

- Examples: Map coloring, scheduling, logistics (transportation), N-queens, crossword puzzles,...

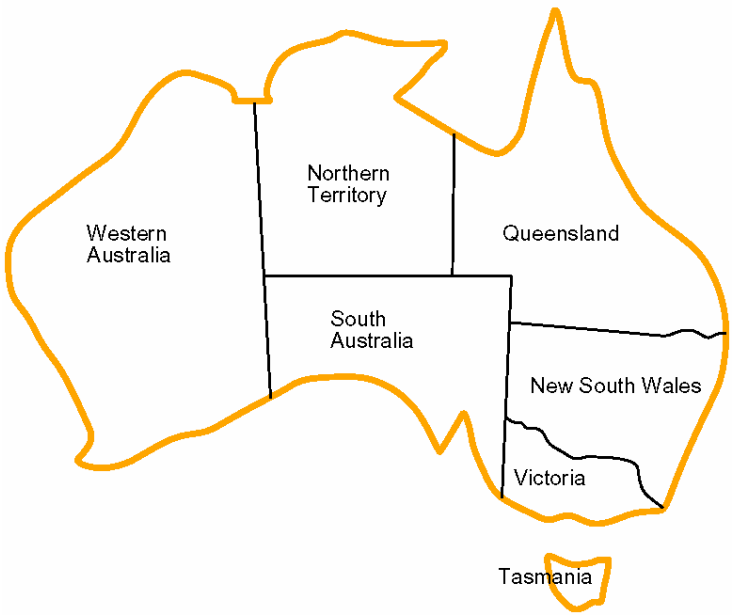
Map coloring

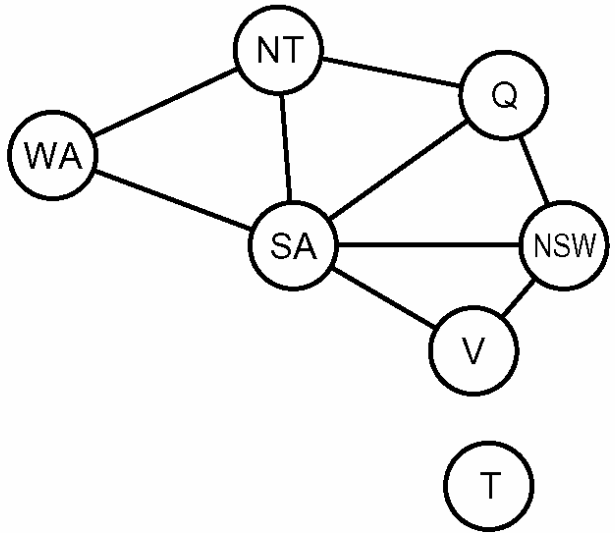


(Image borrowed from V. Pavlovic)

- $\mathbf{X} = \{WA, NT, Q, NSW, V, SA, T\}$
- $\mathbf{D} = \{R,G,B\}$ (for all variables)
- $\mathbf{C} = \{WA \neq NT, WA \neq SA, NT \neq SA, NT \neq Q, SA \neq Q, SA \neq NSW, SA \neq V, Q \neq NSW, NSW \neq V\}$
  (Neighboring regions must have different colors)
- Goal: All regions must have a color

Map coloring





- CSP can be visualized with constraint graph
  - Nodes = variables
  - Arcs = constraints

Types of constraints

- Unary: $X_i \neq v_j$
- Binary: $X_i \neq X_j$
- Higher order

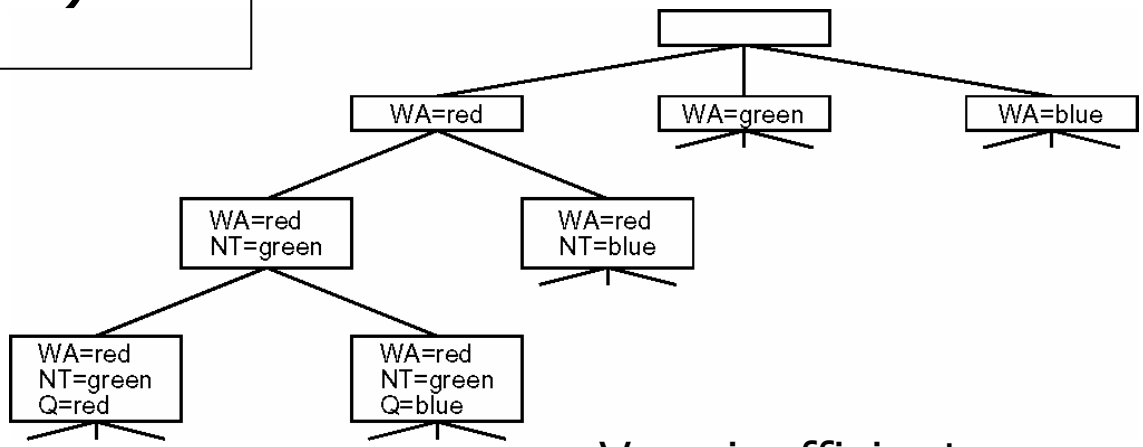
- Preferences

CSP cast as a search problem

- Initial state
- Successor function
- Goal test
- Path cost (not relevant)

- Depth-first search (depth limit = n)

  $\Rightarrow$ Backtracking search

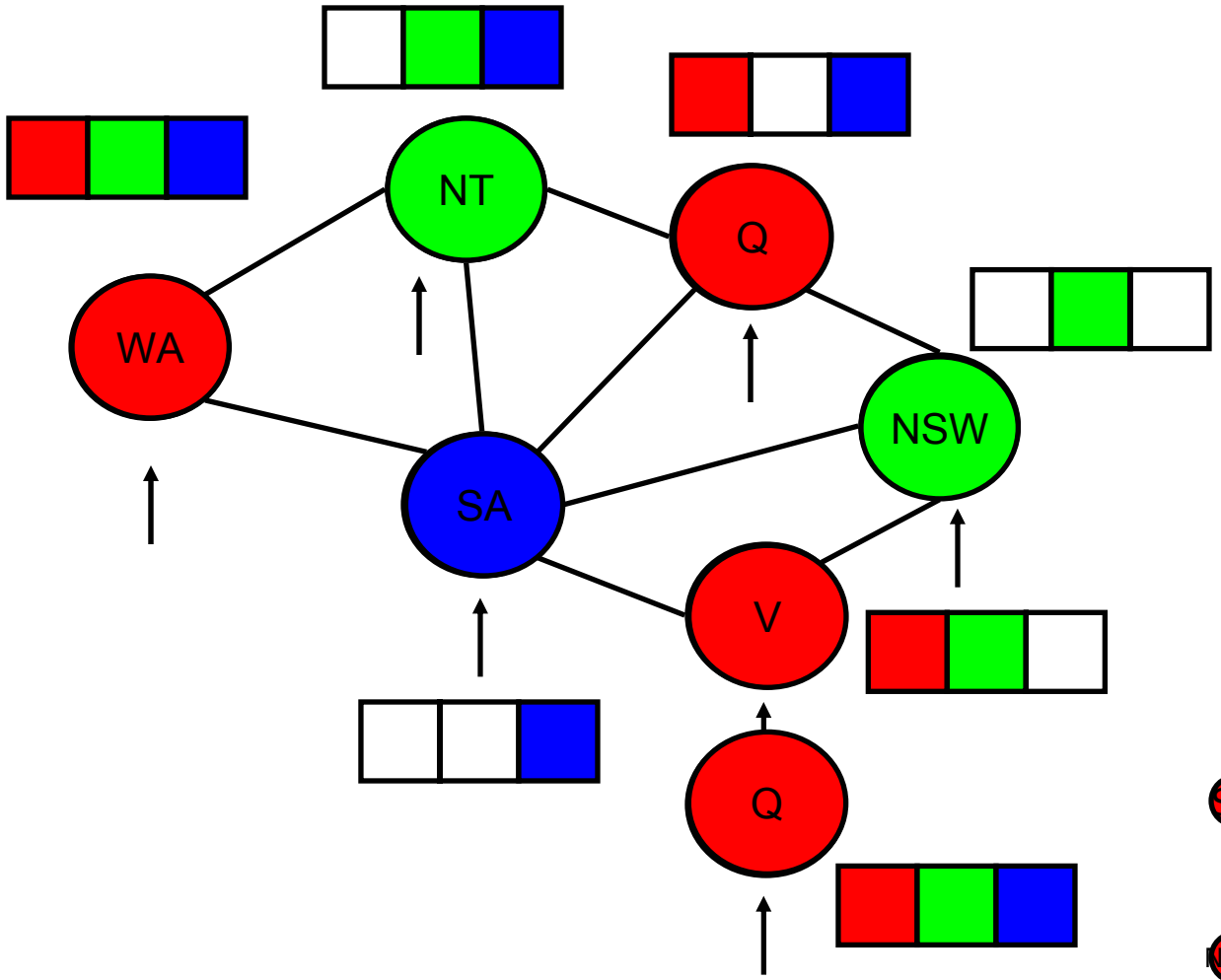


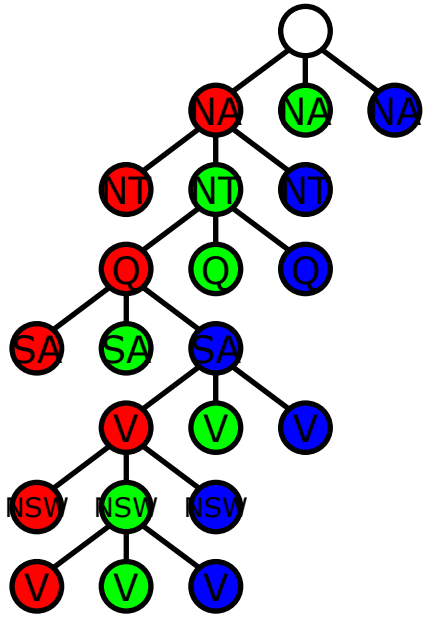
Very inefficient

Improve ”vanilla” Backtracking

1. Smart choice of next variable
   - Min. remaining values (+ degree heuristic)
   - Least constraining value
2. Check consequences of choice
   - Forward checking
   - Arc consistency
3. Intelligent backtracking
   - Backjumping

”Vanilla” backtracking

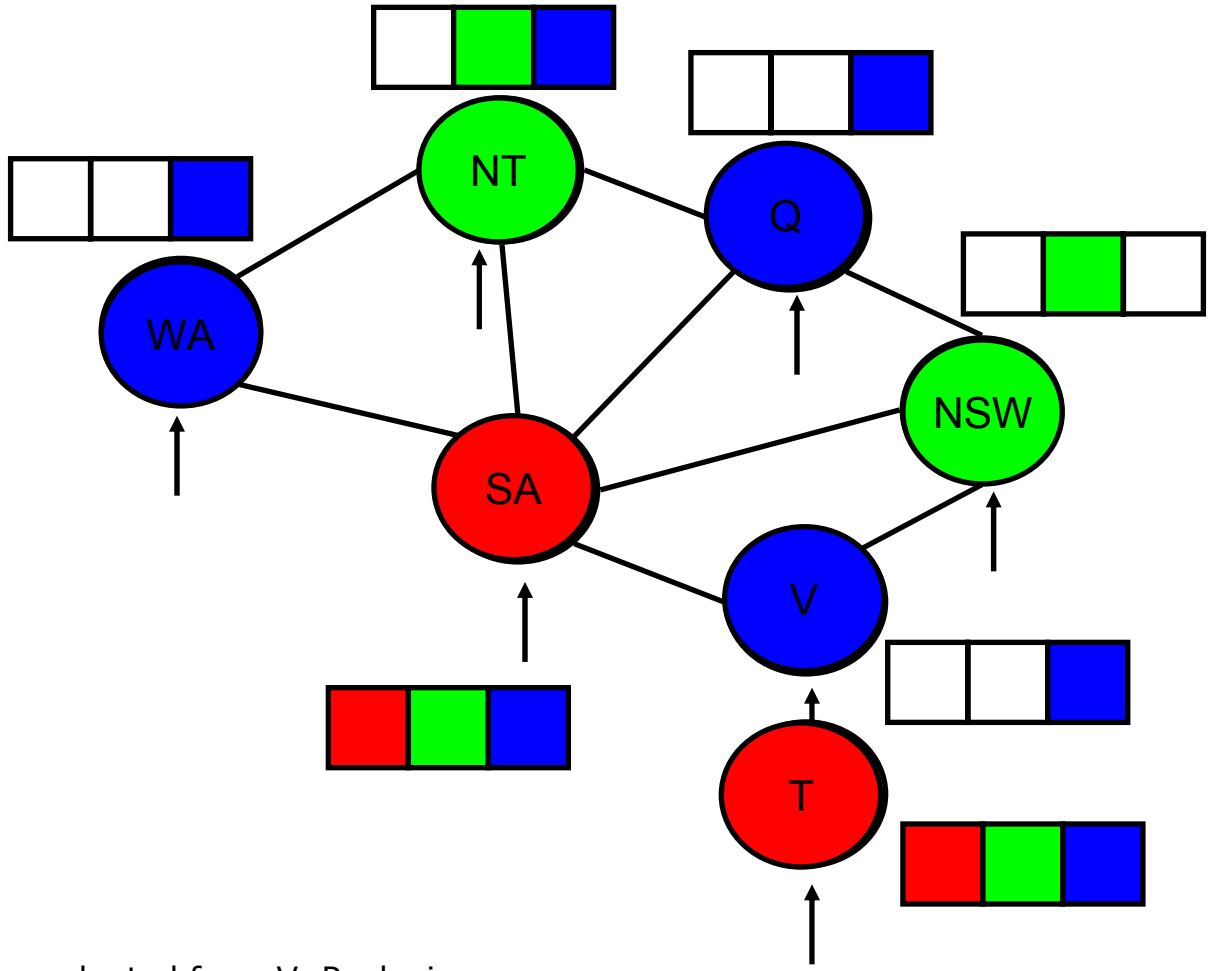




Animation adapted from V. Pavlovic

Minimum remaining values

Choose next the variable that is most constrained based on current assignment (& choose the one with the highest degree first in ties)



Animation adapted from V. Pavlovic

Least constraining value

Select the value that least constrains the other variables
(rules our fewest other variables)

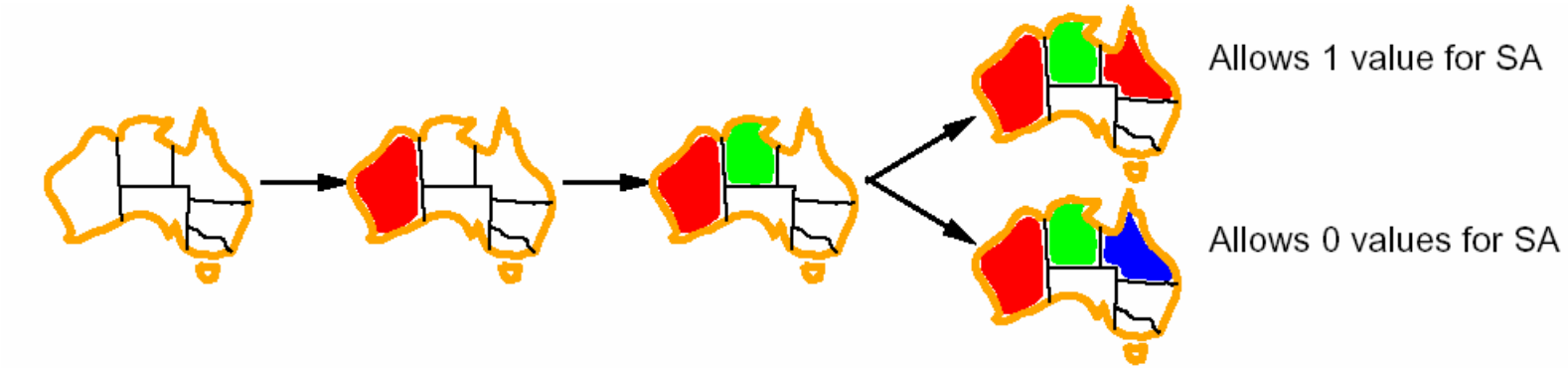
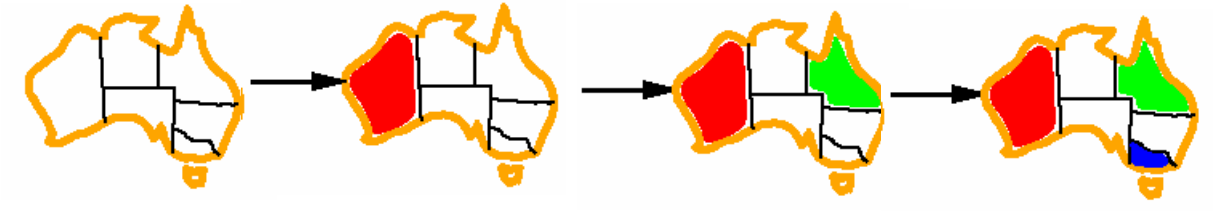
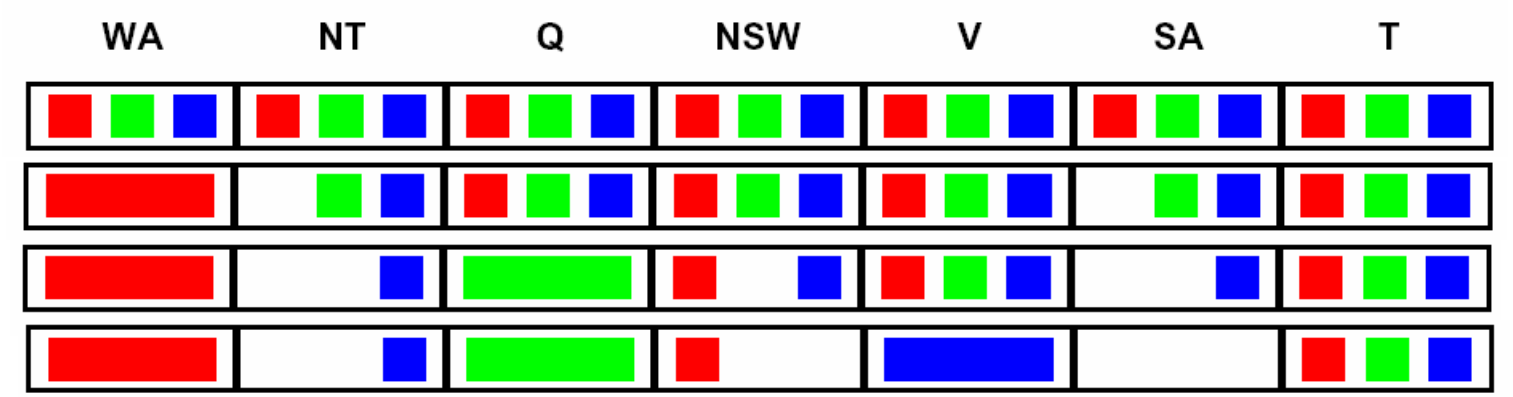


Image borrowed from V. Pavlovic

Forward checking

Delete values that are inconsistent with the last selection



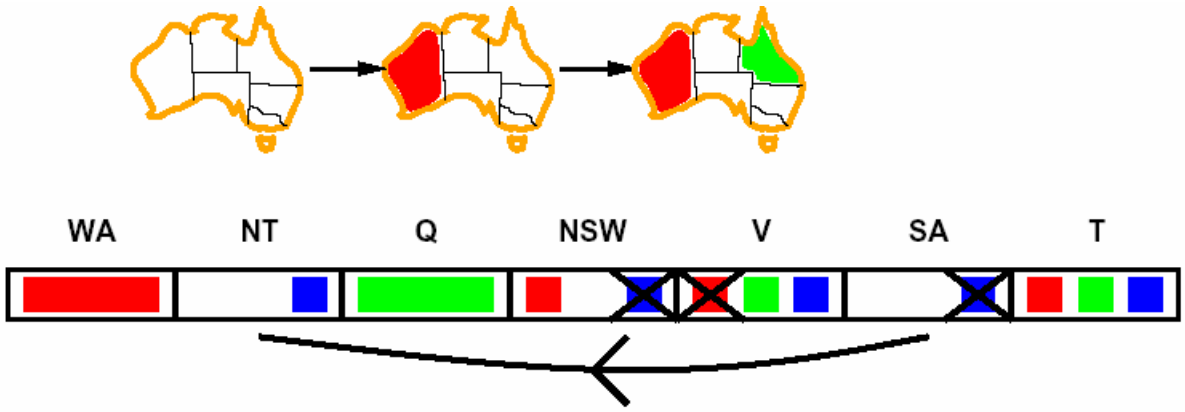


Stop search early, if necessary

Animation adapted from V. Pavlovic

Arc consistency

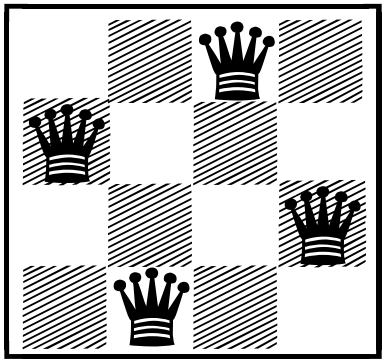
- The arc $X_i \rightarrow X_j$ is consistent if for every value of X_j there exists some valid value of X_i.

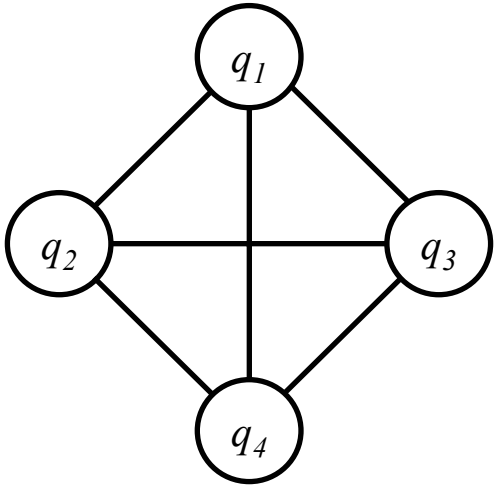


- Detects failures sooner than forward checking.
- Use as preprocessor or as check after each move (MAC)

Animation adapted from V. Pavlovic

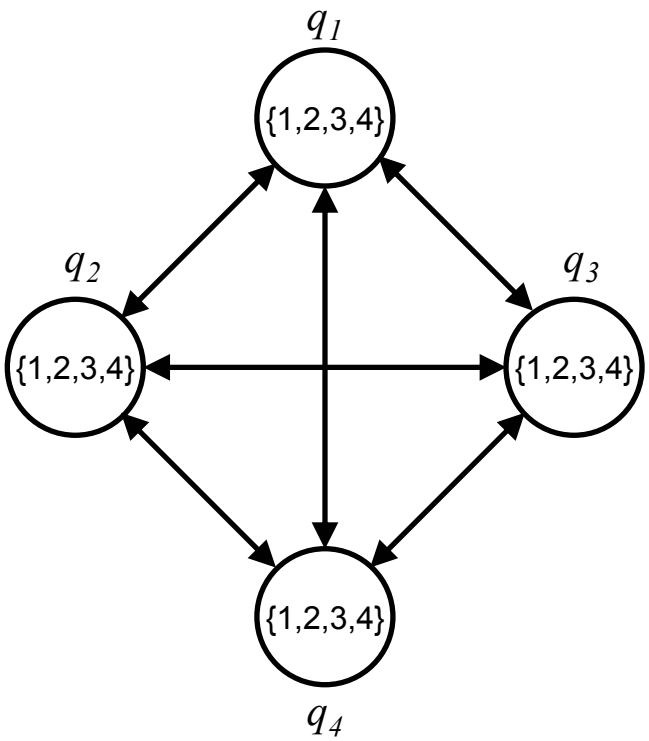
4-queens





- $\mathbf{X} = \{q_1, q_2, q_3, q_4\}$ (q_i = queen in column i)
- $\mathbf{D} = \{1,2,3,4\}$ (row for queen)
- **C**: No queen may attack another queen
- Goal: Place all queens on the board

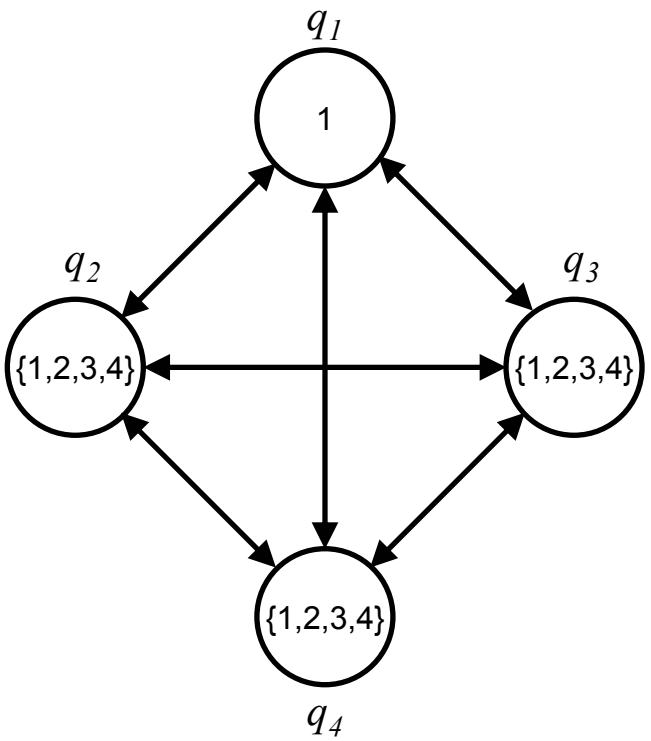
Arc consistency: 4-queens



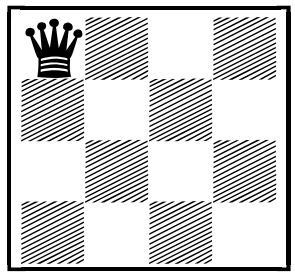
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens

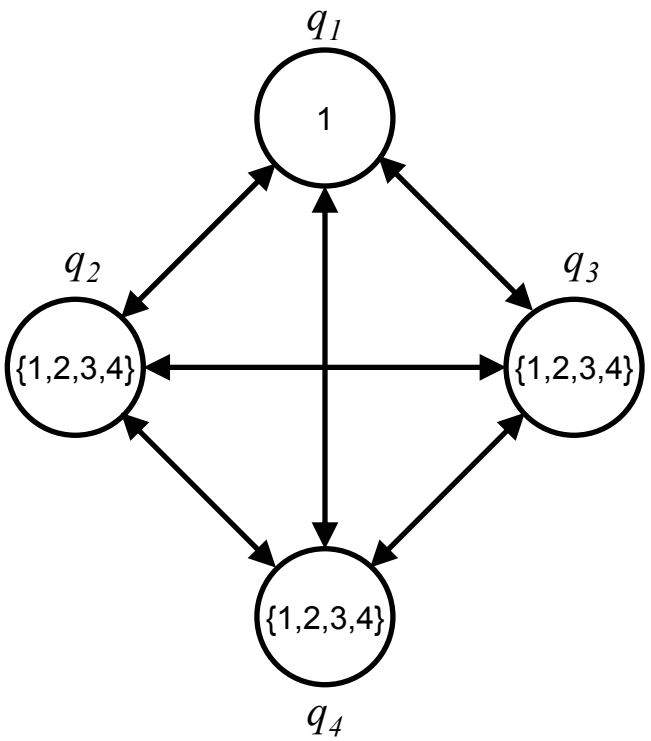


- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$

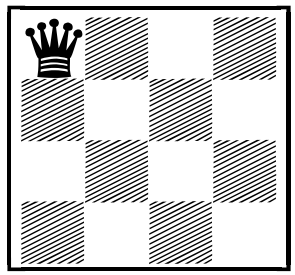


Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



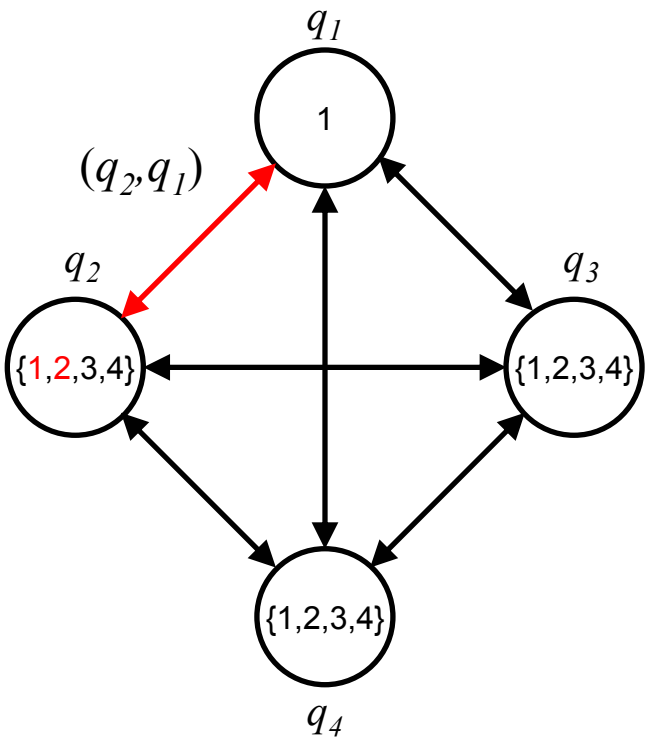
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



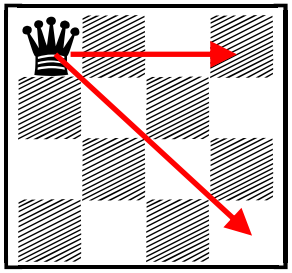
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



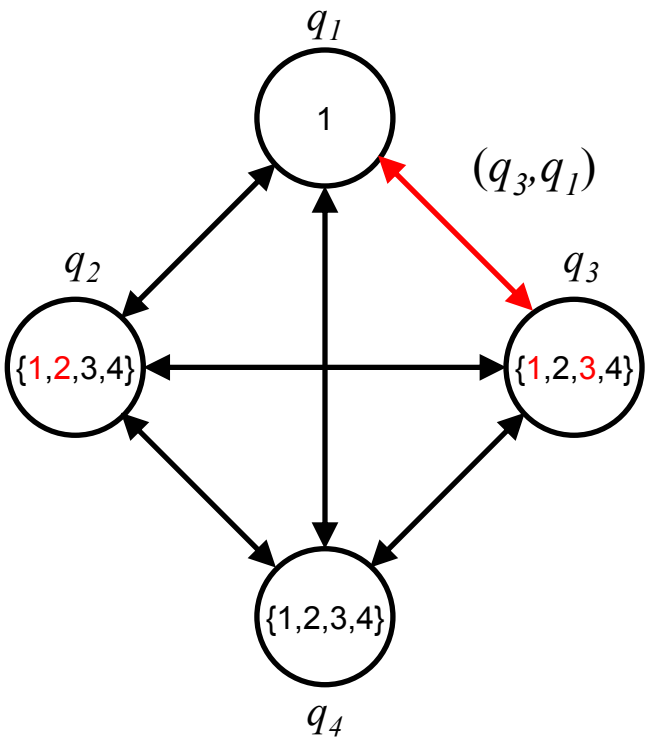
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



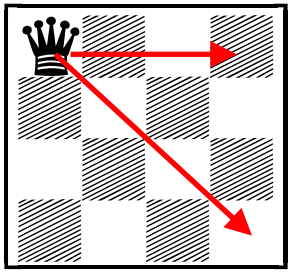
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has at least one value at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



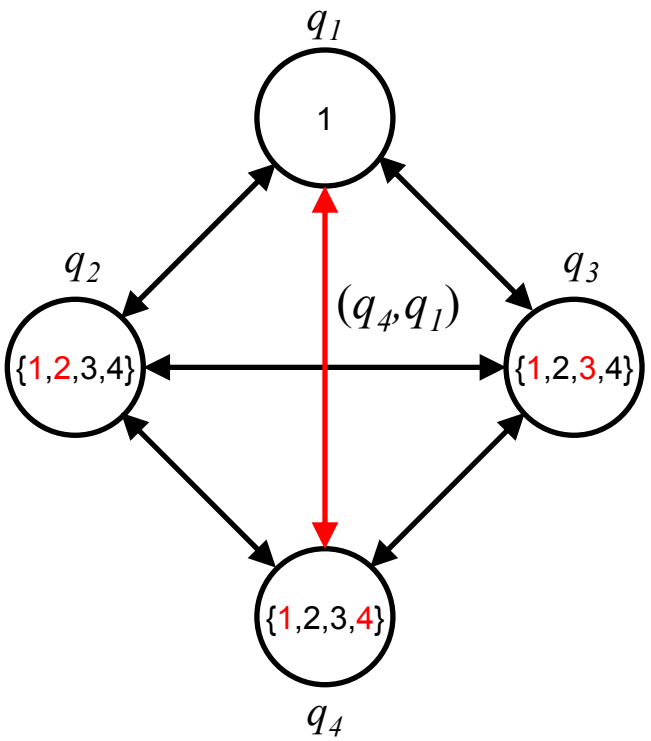
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



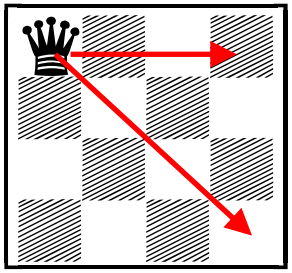
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has at least one value at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



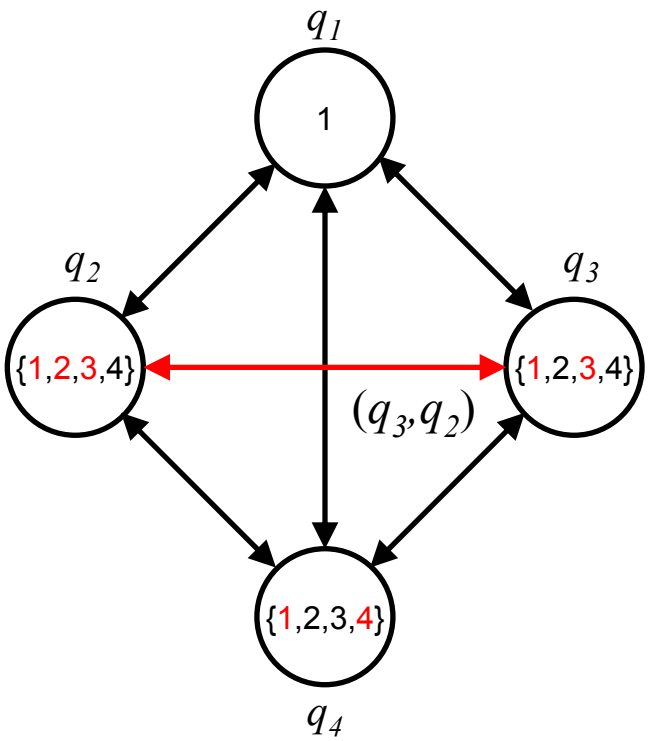
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



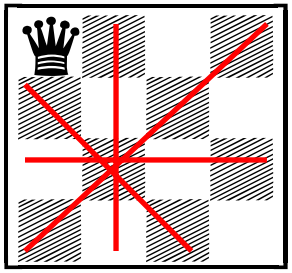
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



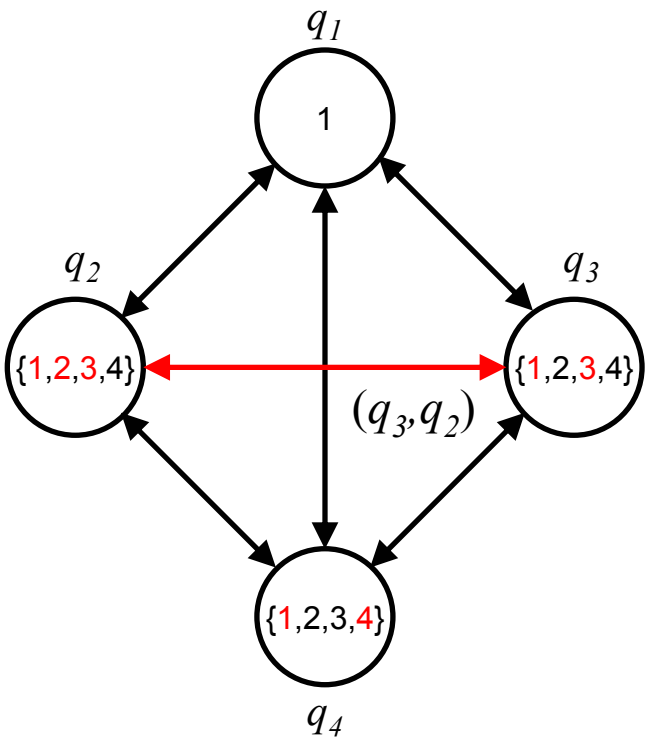
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



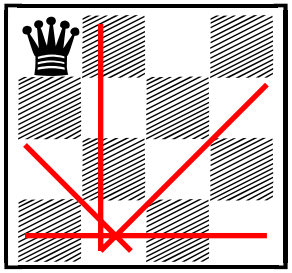
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



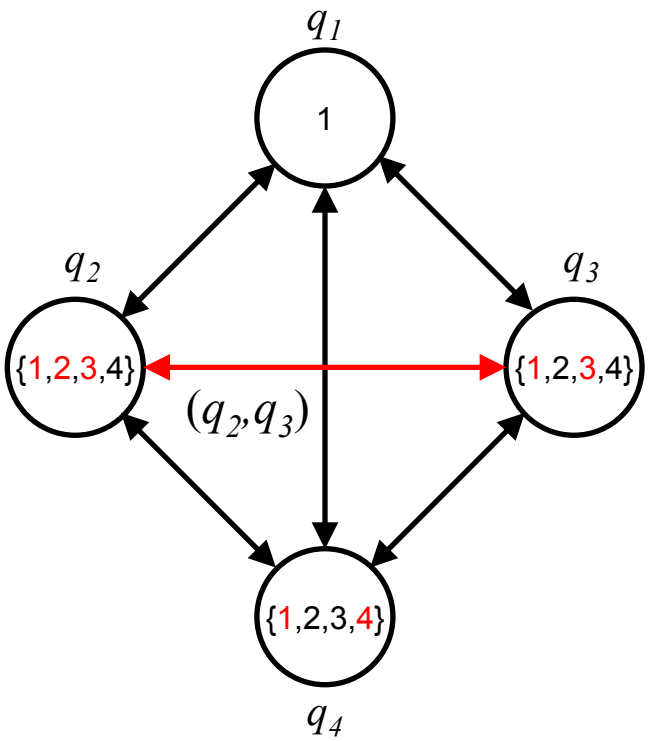
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



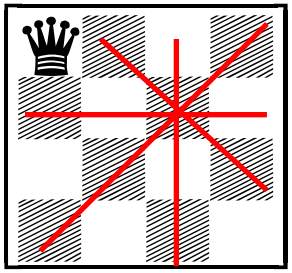
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



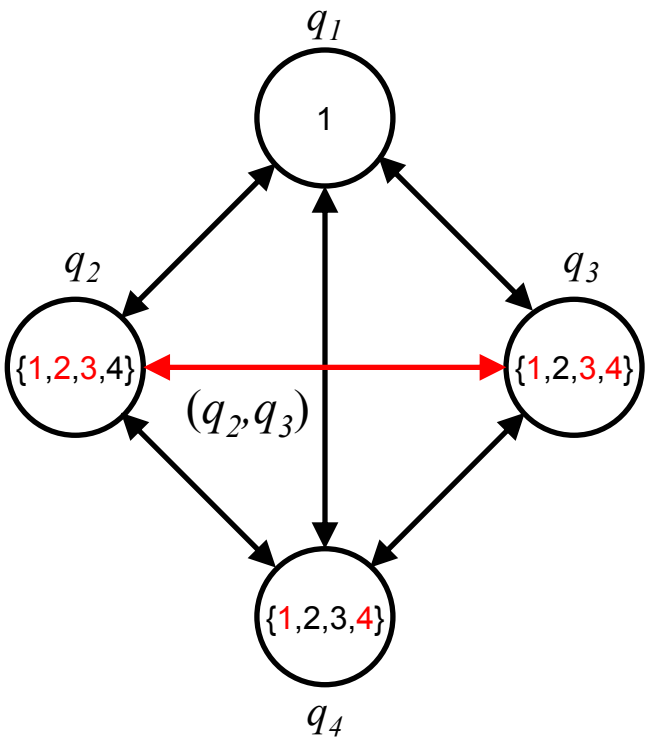
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



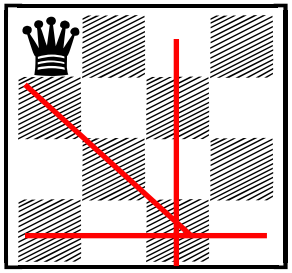
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



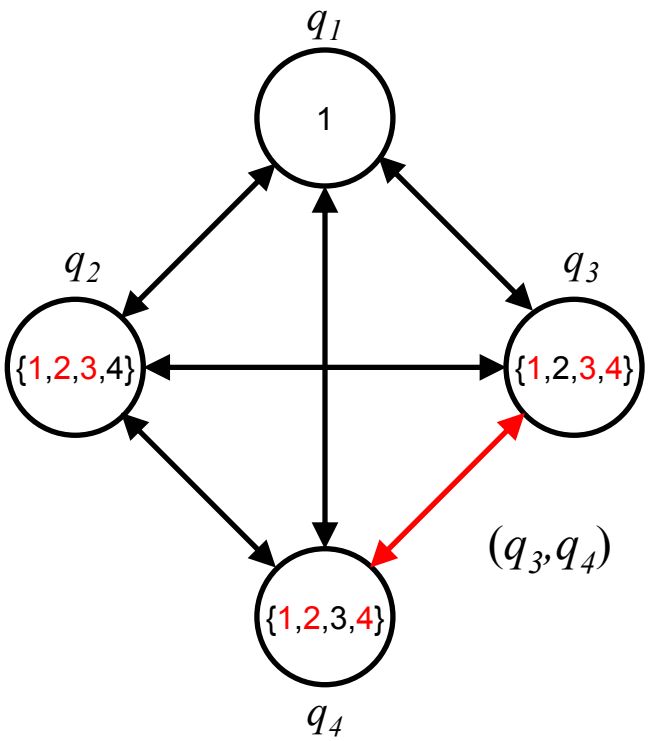
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



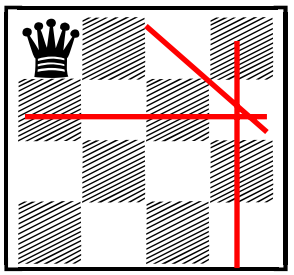
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



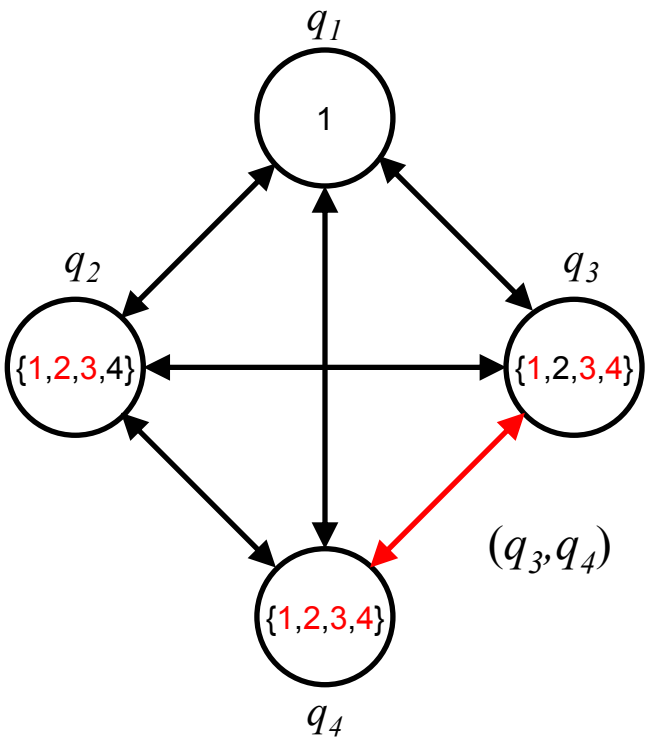
- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$



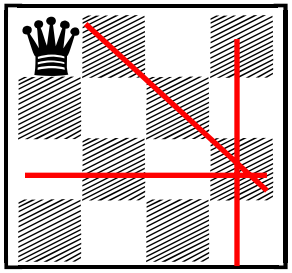
The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 1$

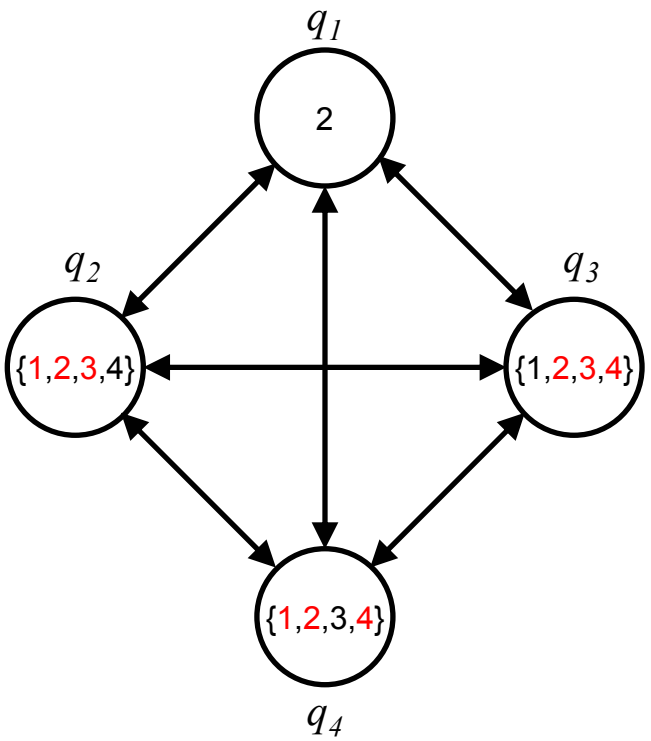


The arc *(i,j)* is *consistent* if *each* value at the tail j of the arc has <u>at least one value</u> at the head i that does not violate the constraints.

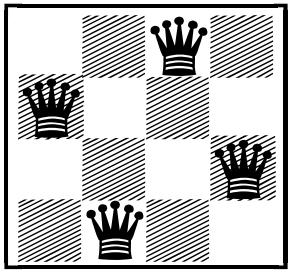
$\Rightarrow$ No solution with $q_1 = 1$

Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Arc consistency: 4-queens



- q_i = queen placed in column i
- {1,2,3,4} = The allowed values (row number) for the queen.
- First move: $q_1 = 2$

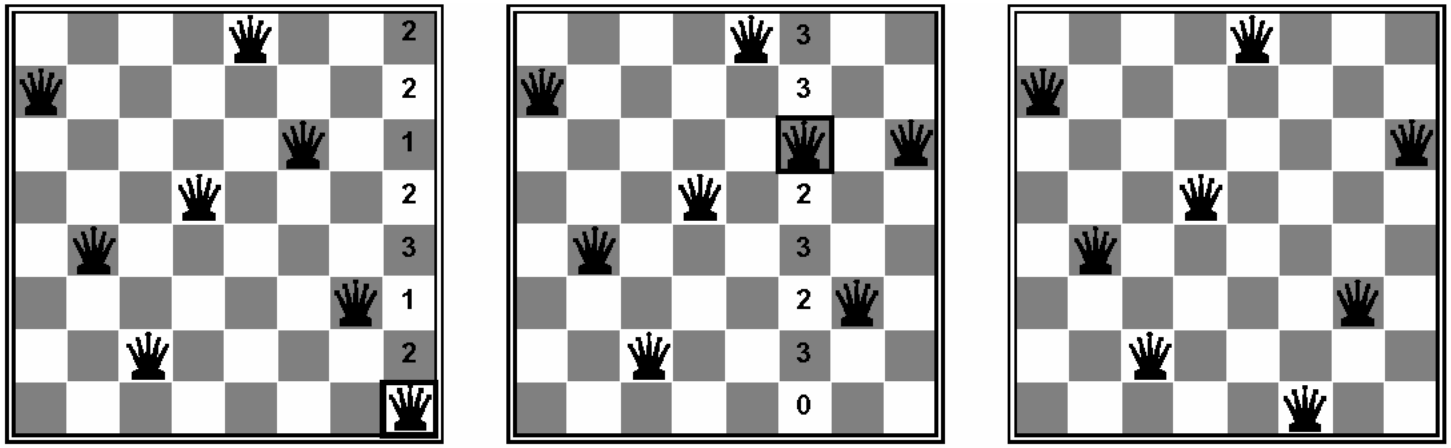


Arc consistency (constraint propagation) finds the solution

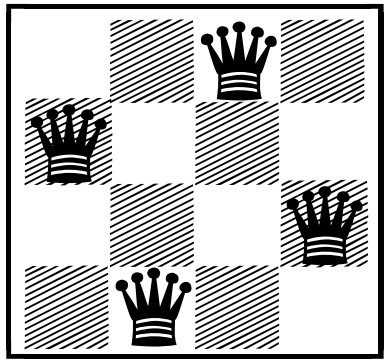
Example from N.J. Nilsson, ”Artificial Intelligence: A New Synthesis”, 1998

Local search: Simple and efficient

- Start with initial (invalid) state
- Modify this state, using the min-conflicts heuristic: Select the value that minimizes the number of conflicts
- Continue until a solution is found



Local search: 4-queens

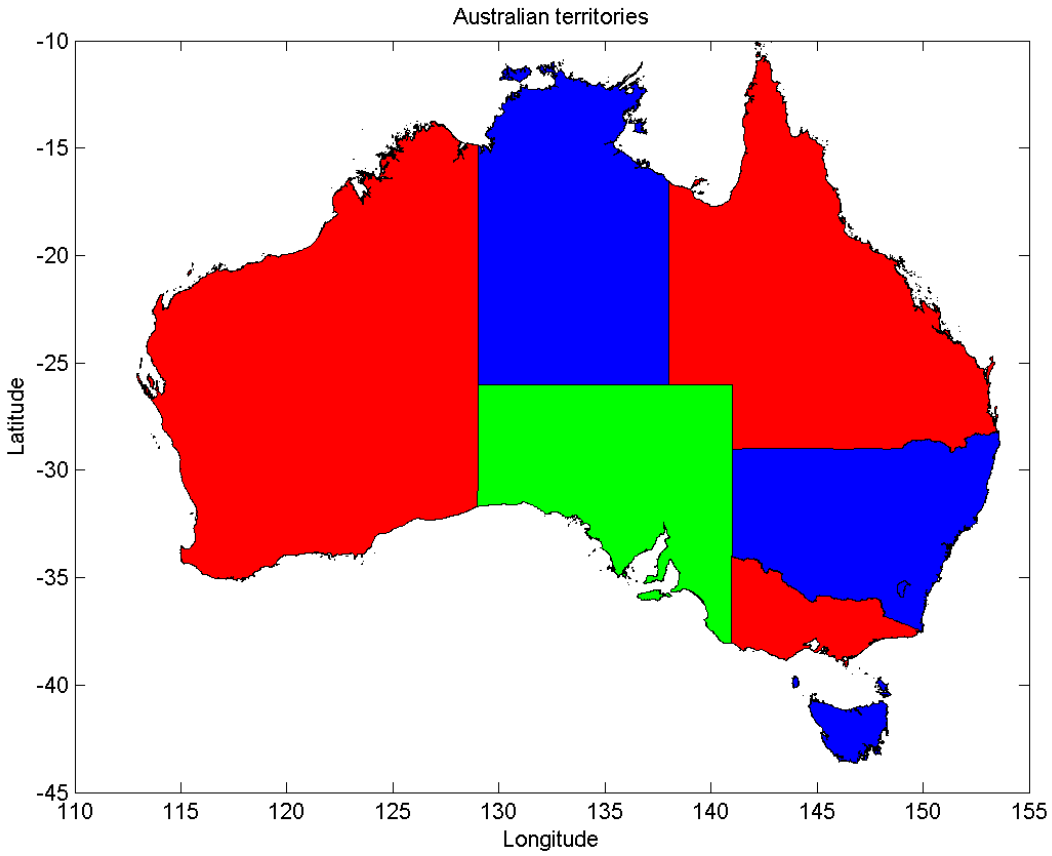


Stuck!

Choose random move that does not increase conflicts.

Solution in three moves!

Map coloring: Australia

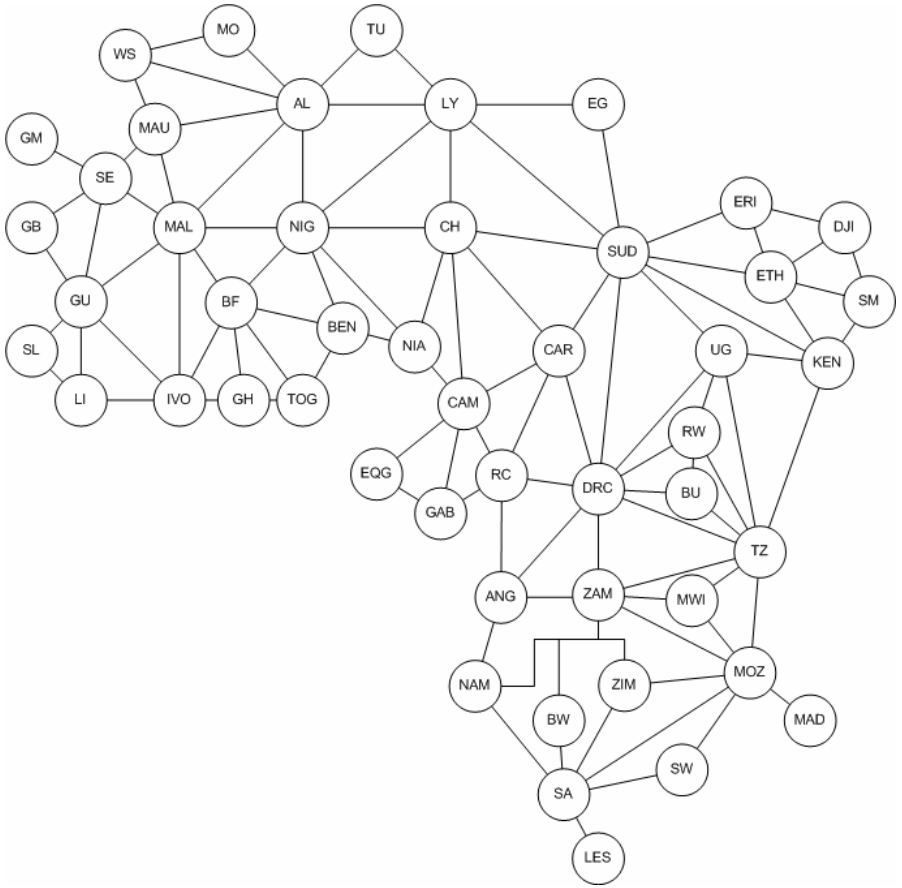


Using Java code @ AIMA site,
calling from within MATLAB.

- Backtracking ~ 1.5 sec

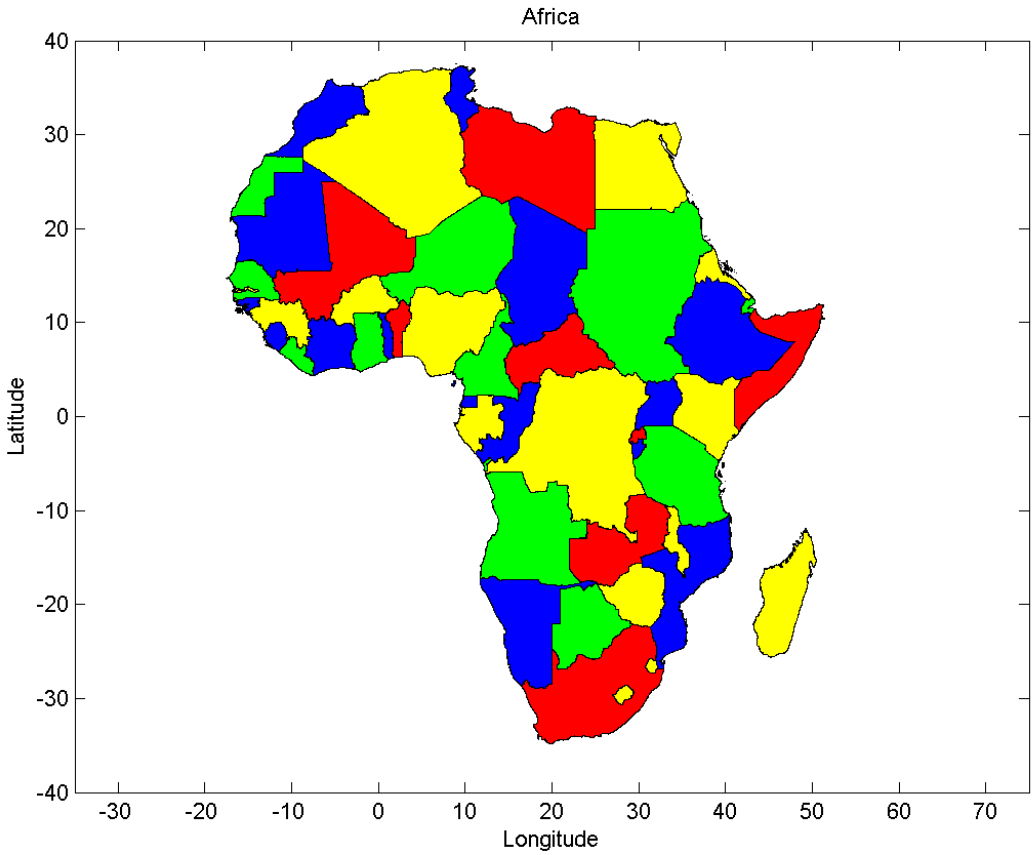
7 variables (countries)
9 constraints
3 values (R,G,B)

Map coloring: Africa



49 variables (countries)
107 constraints
4 values (R,G,B,Y)

Map coloring: Africa



Modified the AIMA Java code, calling from within MATLAB.

- Backtracking ~ 5.5 min