*Artificial Intelligence Programming*

# Constraint Satisfaction

Chris Brooks

Department of Computer Science
University of San Francisco



## 7-2: Constraint Satisfaction

- So far, we've focused on using search to find the **best** solution.
- In many cases, you just need to find a solution that satisfies some criteria.
- These criteria are called constraints.
- A problem in which we want to find any solution that satisfies our constraints is a **constraint satisfaction problem**
- Constraints provide us with additional knowledge about the problem that we can exploit.
  - We can also consider optimizing constrained problems.



## 7-3: Examples

- Toy Problems
  - Map coloring, N-queens, cryptarithmetic
- Real life problems
  - Scheduling, register allocation, resource allocation



## 7-4: Formalizing a CSP

- We can think of a CSP as a set of variables $\{x_1, x_2, ..., x_n\}$
- Each variable as a domain of possible values $D_1, D_2, ..., D_n$
- We also have a set of constraints $C_1, C_2, ...$
  - Unary constraints: $x < 10$, $grade != 0$, etc
  - Binary constraints: $x < y, x + y < 50$, no two adjacent squares can be the same color, etc
  - N-ary constraints: $x1 + x2 + ... + x_n = 75$, weight of chassis plus engine plus body must be less than 3000 lbs, etc.
- An assignment of values to variables that satisfies all constraints is called a **consistent** solution.
- We might might also have an objective function $y = f(x_1, ..., x_n)$ that lets us compare solutions.



## 7-5: Approaches

- If the domain of all variables is continuous (i.e. real numbers) and constraints are all linear functions, we can use **linear programming** to solve the problem.
  - Express the problem as a system of equations
- In other cases, we can use **dynamic programming**.
- Dynamic programming is a form of search.
- In the most general case, we can express a CSP as a search problem.



## 7-6: Solving CSPs with search

- We'll begin with an initial state: no values assigned to $x_1, ..., x_n$
- An action assigns values to variables.
- A goal is an assignment to each variable such that all constraints are met.
- The successor function returns all possible single assignments such that constraints are still met.

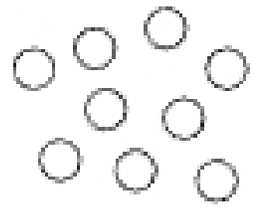
## 7-7: Constraint graph

- Often, the most difficult part of solving a CSP is formulating the problem.
- It is often useful to visualize the CSP as a **constraint graph**
- Nodes represent variables, edges represent binary constraints.
  - For n-ary constraints, we must add nodes that represent combinations of values.

## 7-8: Example - Coloring a Bay Area Map


- Can we color this map using only four colors (R,Y,B,G), so that no adjacent counties have the same color?

## 7-9: Example - Coloring a Bay Area Map



## 7-13: Example - Australia

- Neighbors of Q have a domain of Green, Blue - smallest.
- Choose a neighbor and select a color that satisfies all constraints.
- NSW = Green.
- Now SA has a domain of size 1 (Blue)
- Coloring SA then fixes the choices for V and NT.

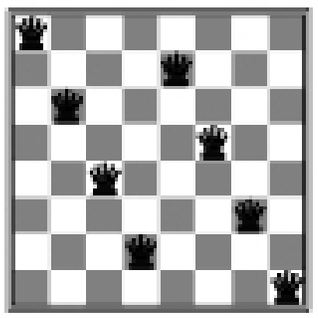
## 7-14: Heuristics

- How do we pick which country to color next?
- How do we choose what color to give it?
- Intuition: always try to make decisions that leave as much flexibility as possible.
- Most constrained variable: pick the variable (country) that has the fewest possible choices.
- Least constraining value: pick the value (color) that has the least effect on possible values for other variables.

## 7-15: Backtracking

- In the previous example, we were fortunate.
  - We never made a bad choice.
  - What if we had colored Q = red, NSW = green, V = blue?
- Usually, when solving a CSP, there are times when you have to 'undo' a bad choice.
- This is called **backtracking**.

## 7-16: Example - 8-queens



- Problem: place each queen on the board so that no queen is attacking another.
- We can reduce this to: What row should each queen be in?

## 7-17: Chronological Backtracking

- The easiest approach is to use depth-first search.
- If we reach a point where we can't place a queen, back up and undo the most recent placement.
- If that placement can't be changed, undo its predecessor.
- Always undo assignments in reverse order of when they were done.
- This is called **chronological backtracking**.

## 7-18: Chronological Backtracking