Advanced Topics in AI

Solving CSPs

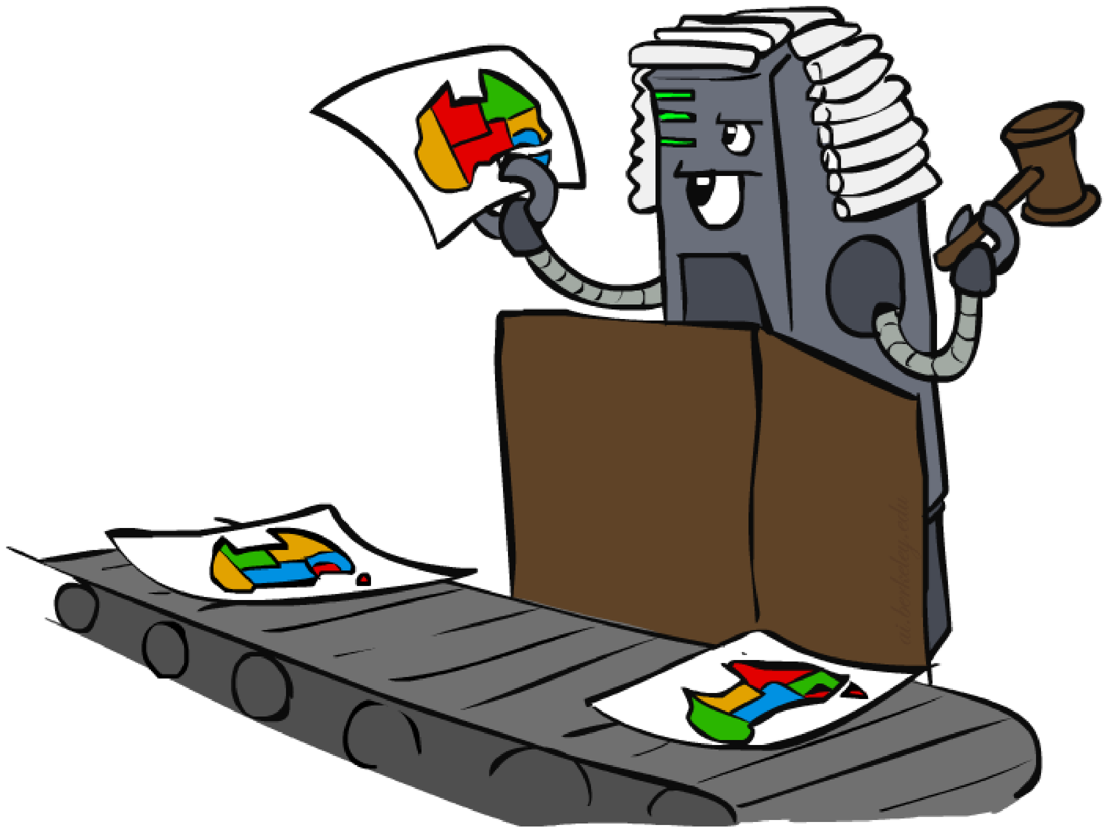
Instructor: Prof. Dr. techn. Wolfgang Nejdl

Leibniz University Hannover

Co-financed by the Connecting Europe Facility of the European Union

Standard Search Formulation

- Standard search formulation of CSPs
- States defined by the values assigned so far (partial assignments)
  - **Initial state**: the empty assignment, {}
  - **Successor function**: assign a value to an unassigned variable
  - **Goal test**: the current assignment is complete and satisfies all constraints
- We'll start with the straightforward, naïve approach, then improve it



Co-financed by the Connecting Europe Facility of the European Union

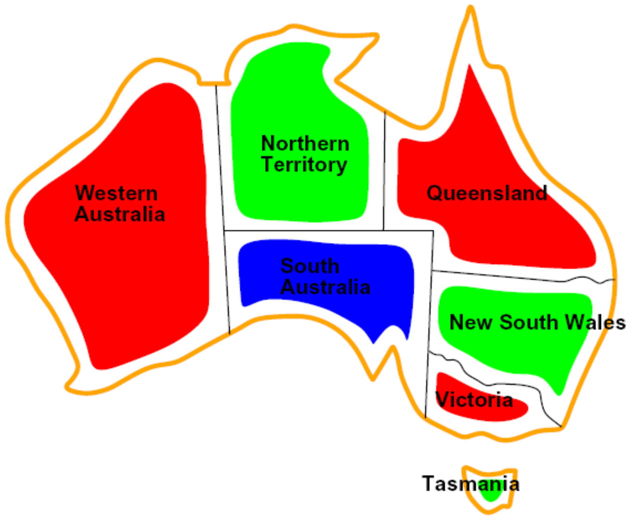
Search Methods

- What would BFS do?

$\{\}$

$\{WA = g\}$ $\{WA = r\}$ … $\{NT = g\}$ …

…



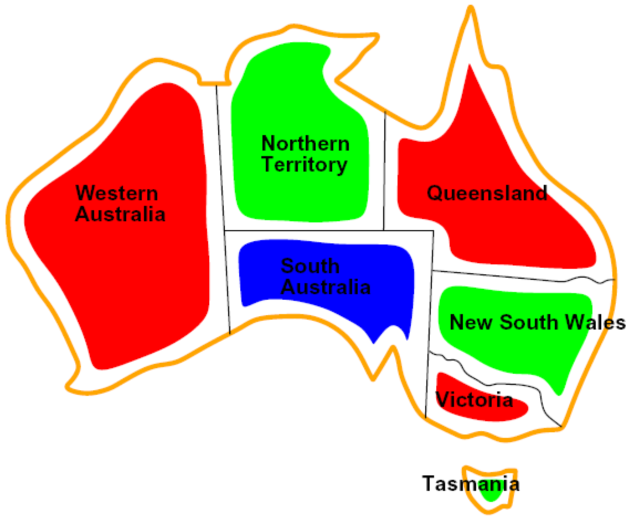
Co-financed by the Connecting Europe Facility of the European Union

Search Methods

- What would BFS do?

- What would DFS do?
  - let's see!

- What problems does naïve search have?



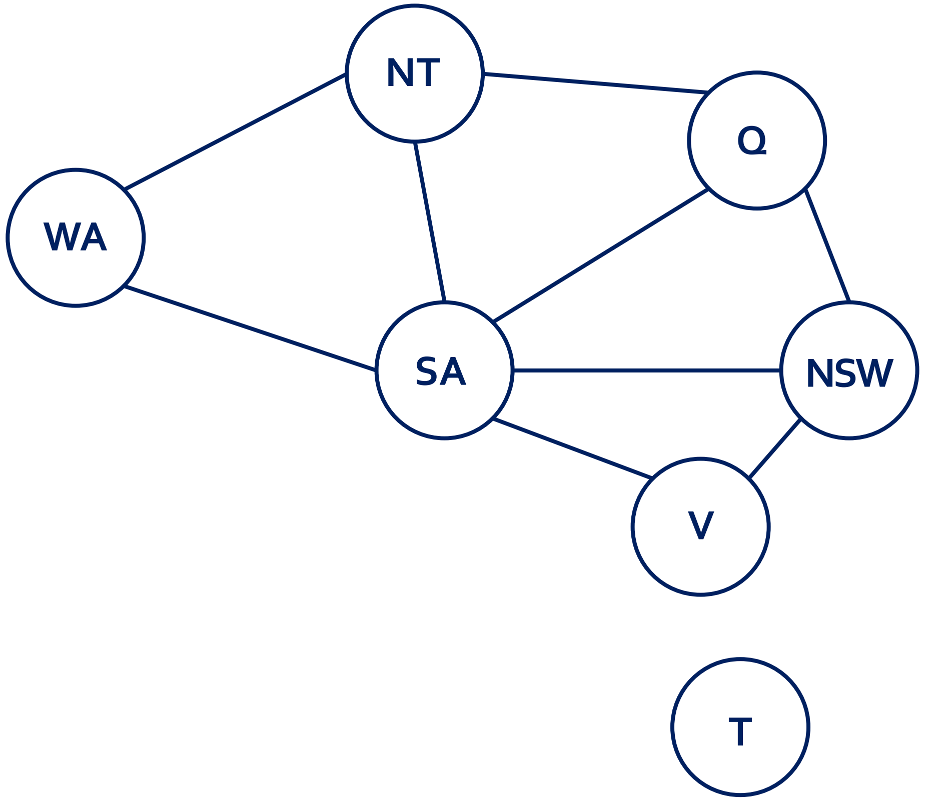
Co-financed by the Connecting Europe Facility of the European Union

Search Methods

- What would BFS do?

- What would DFS do?

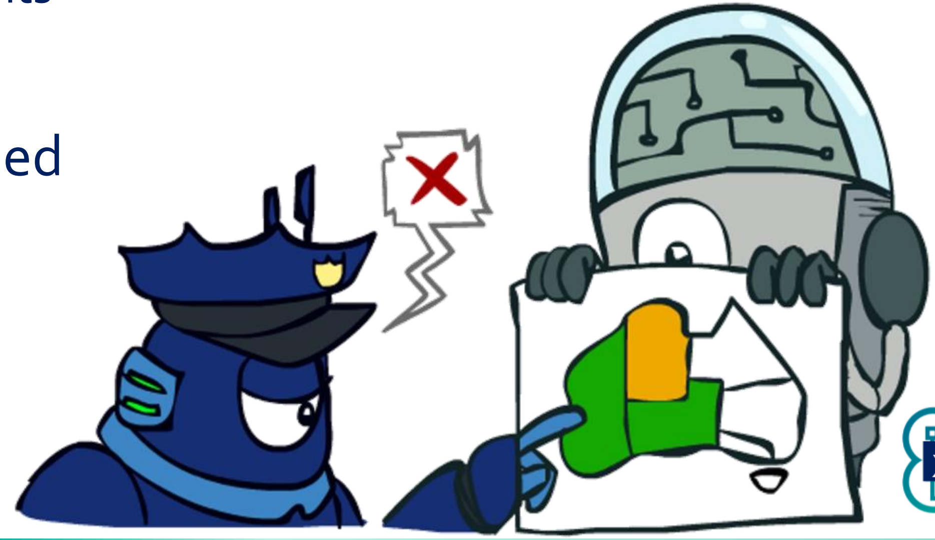
- What problems does naïve search have?



Co-financed by the Connecting Europe Facility of the European Union

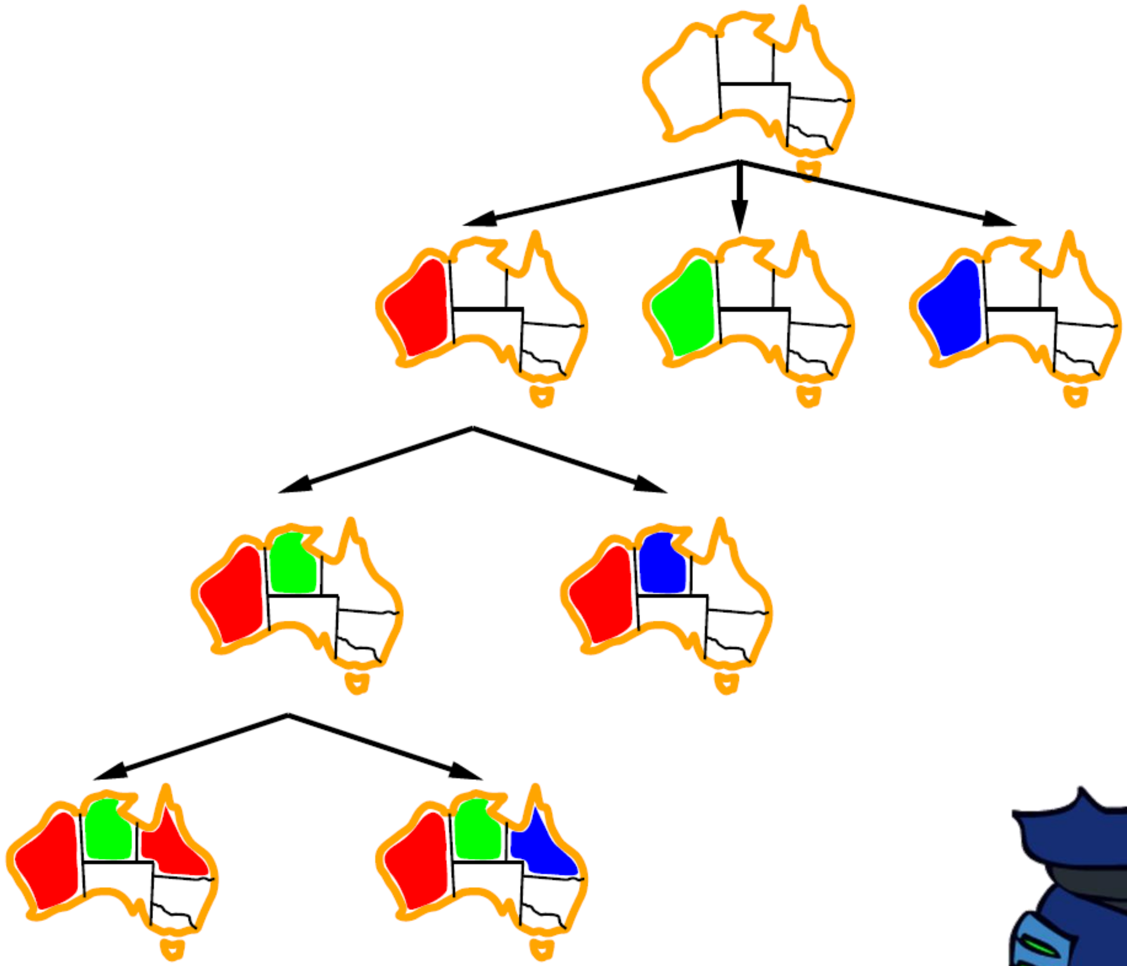
Backtracking Search

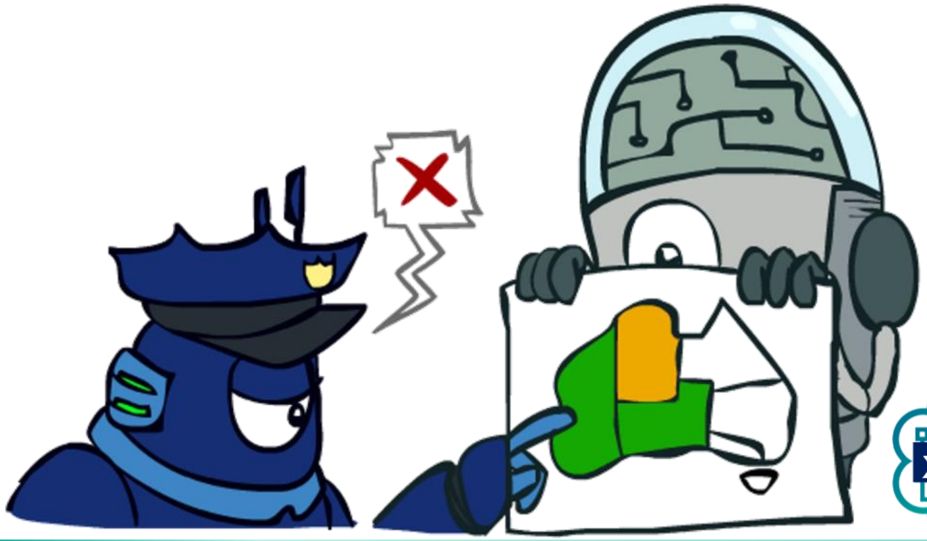
- Backtracking search is the basic uninformed algorithm for solving CSPs
- Idea 1: One variable at a time
  - Variable assignments are commutative, so fix ordering
  - I.e., [WA = red then NT = green] same as [NT = green then WA = red]
  - Only need to consider assignments to a single variable at each step
- Idea 2: Check constraints as you go
  - I.e. consider only values which do not conflict with previous assignments
  - Might have to do some computation to check the constraints
  - "Incremental goal test"
- Depth-first search with these two improvements is called *backtracking search* (not the best name)
- Can solve n-queens for n ≈ 25



Co-financed by the Connecting Europe Facility of the European Union

Backtracking Example





Co-financed by the Connecting Europe Facility of the European Union

Backtracking Search

```
function BACKTRACKING-SEARCH(csp) returns a solution, or a failure
    return RECURSIVE-BACKTRACKING({ }, csp)

function RECURSIVE-BACKTRACKING(assignment, csp) returns soln or failure
    if assignment is complete then return assignment
    var ← SELECT-UNASSIGNED-VARIABLE(VARIABLES[csp], assignment, csp)
    for each value in ORDER-DOMAIN-VALUES(var, assignment, csp), do
        if value is consistent with assignment given CONSTRAINTS[csp] then
            add {var = value} to assignment
            result ← RECURSIVE-BACKTRACKING(assignment, csp)
            if result ≠ failure then return result
            remove {var = value}  from assignment
    return failure
```

- Backtracking = DFS + variable-ordering + fail-on-violation
- What are the choice points?

Co-financed by the Connecting Europe Facility of the European Union

Improving Backtracking

- General-purpose ideas give huge gains in speed
- Ordering:
  - Which variable should be assigned next?
  - In what order should its values be tried?
- Filtering: Can we detect inevitable failure early?
- Structure: Can we exploit the problem structure?

Co-financed by the Connecting Europe Facility of the European Union

Advanced Topics in AI

Next: Filtering

Instructor: Prof. Dr. techn. Wolfgang Nejdl

Leibniz University Hannover

Co-financed by the Connecting Europe Facility of the European Union