Knowledge Representation and Search

Stephen G. Ware

CSCI 4525 / 5525

Classical AI

AI = Representation + Search

1. Consider the problem to be solved.
2. Decide on a data structure for representing a (partial) solution to the problem which provides as much **abstraction** as possible.
3. Consider all the operations which change the data structure.
4. Intelligently make changes to the data structure until it is a solution to the problem.

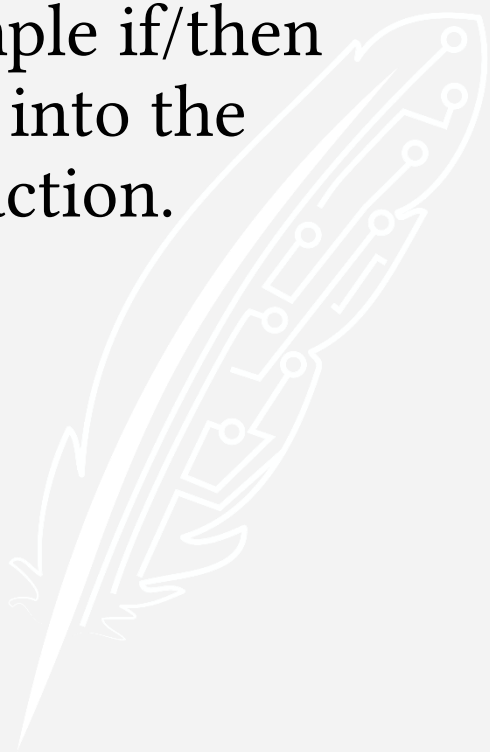
Classical AI

AI = Representation + Search

1. Consider the problem to be solved.
2. Decide on a data structure for representing a (partial) solution to the problem which provides as much **abstraction** as possible.
3. Consider all the operations which change the data structure.
4. *Intelligently make changes to the data structure until it is a solution to the problem.*

Search

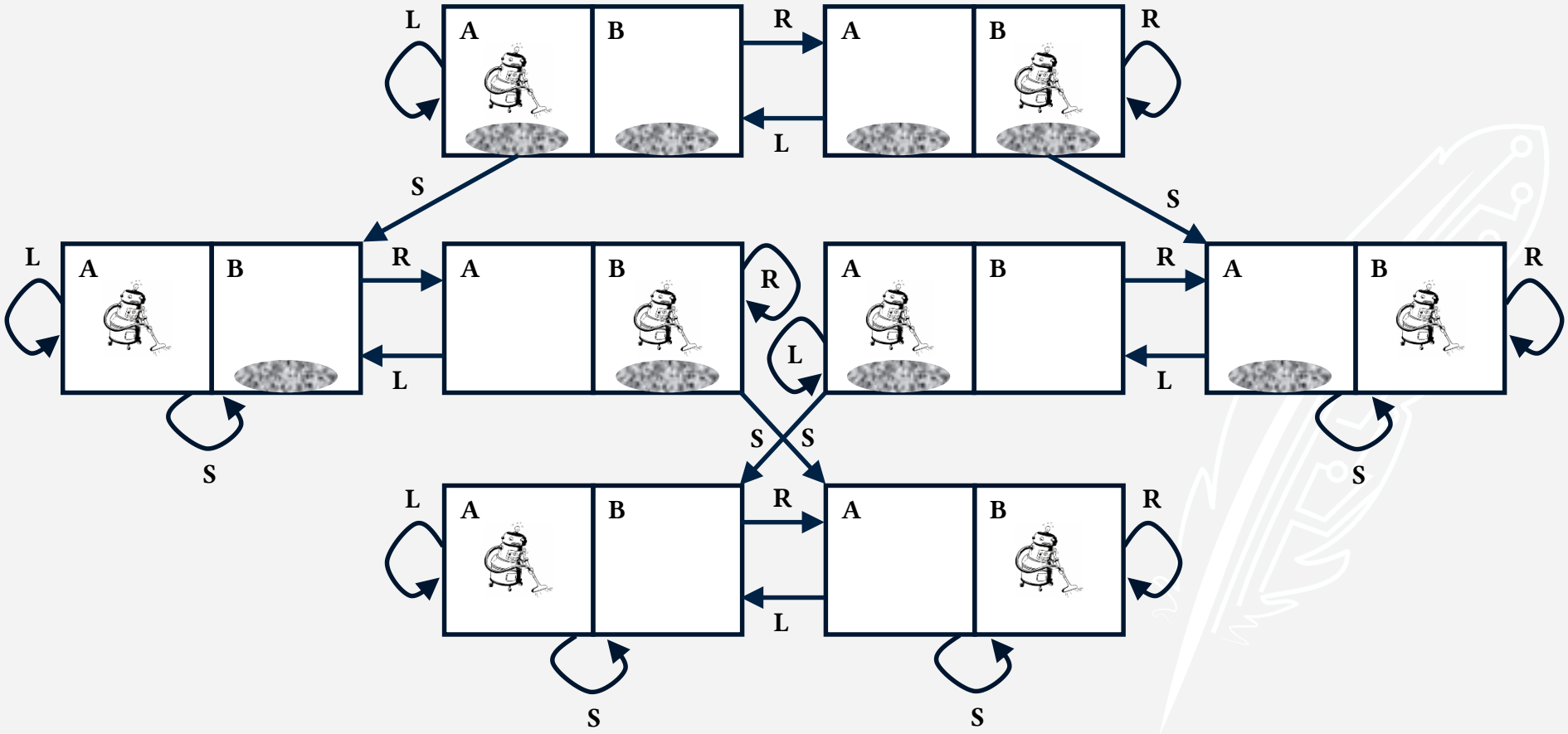
- **Search** is thinking before acting.
- When a problem is too complex for simple if/then logic, an agent may need to look ahead into the future to determine the best course of action.



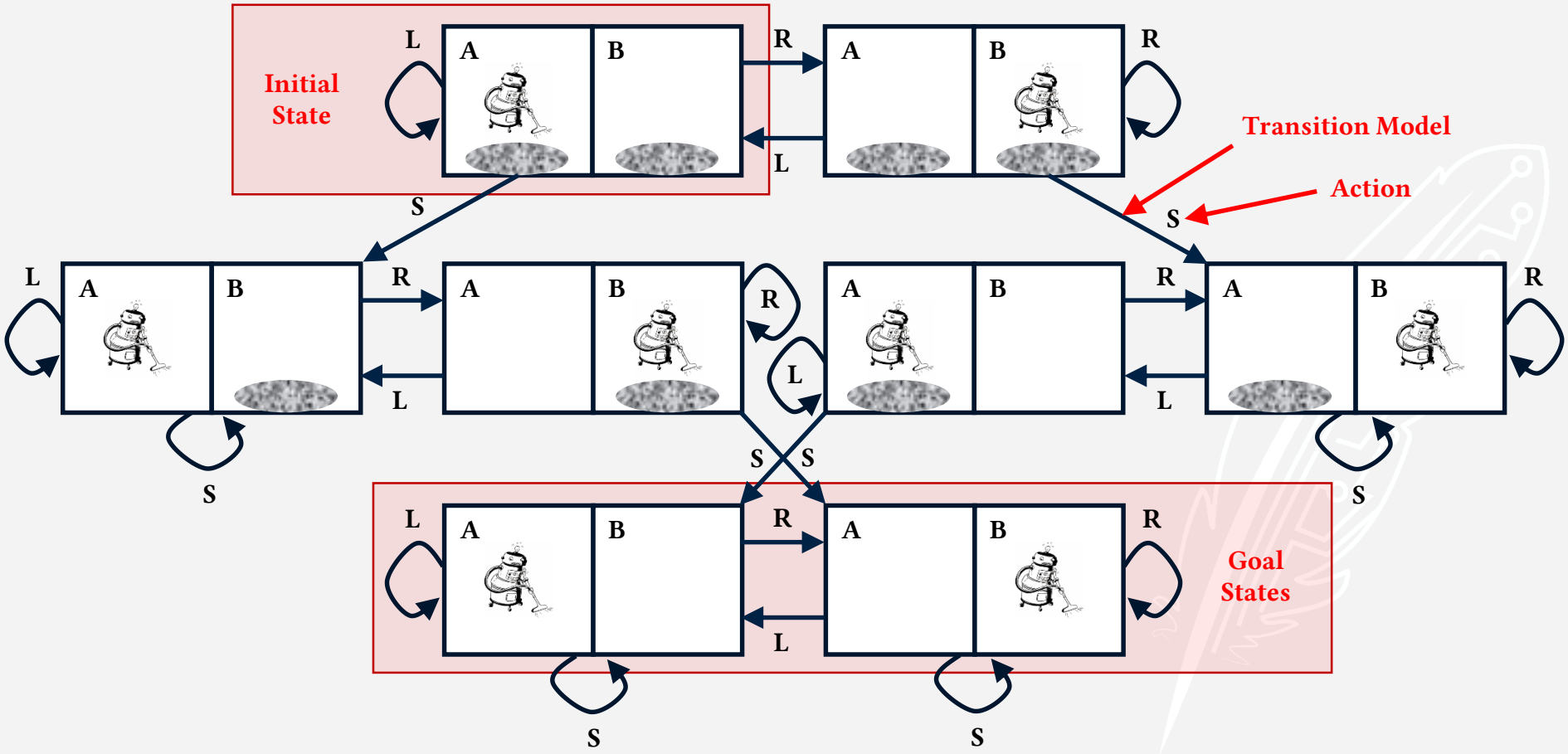
Search Problems

- **Initial State:** The starting point.
- **Actions:** Given the current state, which actions are available?
- **Transition Model:** Given the current state and the next action, what is the next state?
- **Goal Test:** When is a state a solution?
- **Path Cost:** What rewards or penalties are incurred by taking an action?

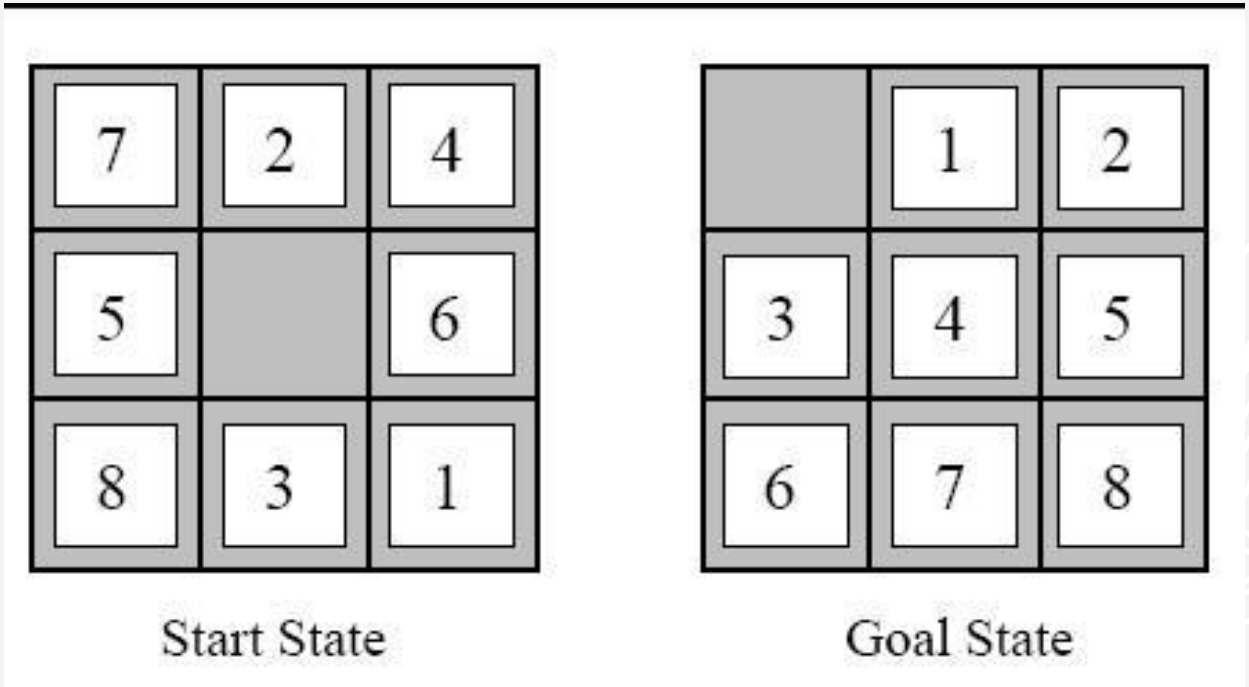
Vacuum World State Space



Vacuum World State Space



8 Puzzle



n Puzzle

Given an *n* x *n* puzzle, how large is the state space?

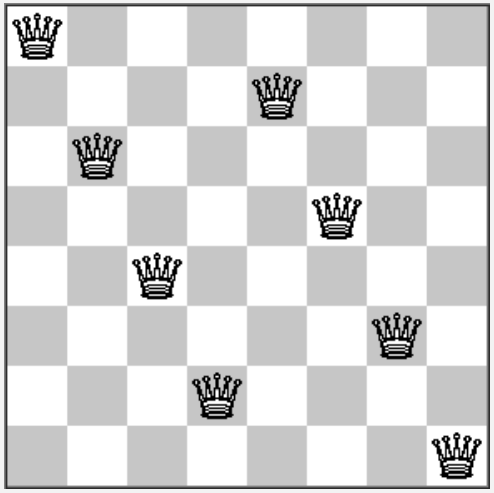
- *n*=3: ~200,000 states

  solved in milliseconds
- *n*=4: ~1,300,000,000,000 states

  solved in milliseconds
- *n*=5: ~10^{25} states

  solved in hours

8 Queen Problem (naïve)



States: Any board with 0 to 8 queens

Initial: Empty board

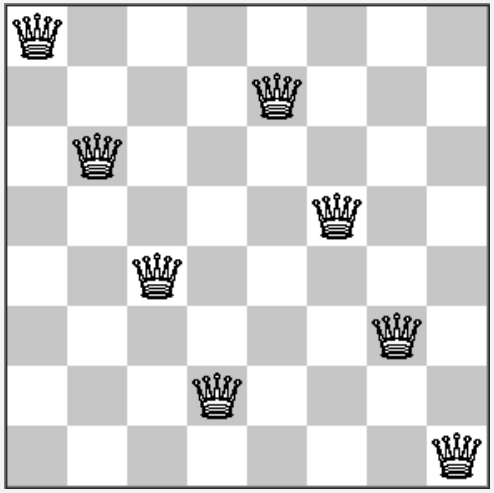
Actions: Place queen on empty square

Transition: New board with +1 queen

Goal: 8 queens, none attacking

Space: ~10^{15} states

8 Queen Problem (better)



States: 1 queen per column, leftmost filled first, none attacking

Initial: Empty board

Actions: Place queen in leftmost empty column, none attacking

Transition: New board with +1 queen

Goal: 8 queens, none attacking

Space: ~2000 states

Navigation

States: Current city

Initial: Starting city

Actions: Drive to neighboring city

Transition: Neighboring city

Goal: Destination city

Search

```
1. Let V be the set of visited nodes, empty.
2. Let F be the frontier, initially containing
   only the initial state.
3. Loop:
4.   If F is empty, return failure.
5.   Choose a node n to remove from F.
6.   If n is a solution, return n.
7.   Add n to V.
8.   For every successor s of n not in V or F:
9.     Add s to F.
```

Tradeoffs

- **Soundness:** Return value is always a solution.
- **Completeness:** If a solution exists, it will be found.
- **Optimality:** The solution is the best possible one.
- **Time Complexity:** Time to find a solution.
- **Space Complexity:** Memory to find a solution.

"**Satisficing**" means solutions may not be optimal but are usually good enough.