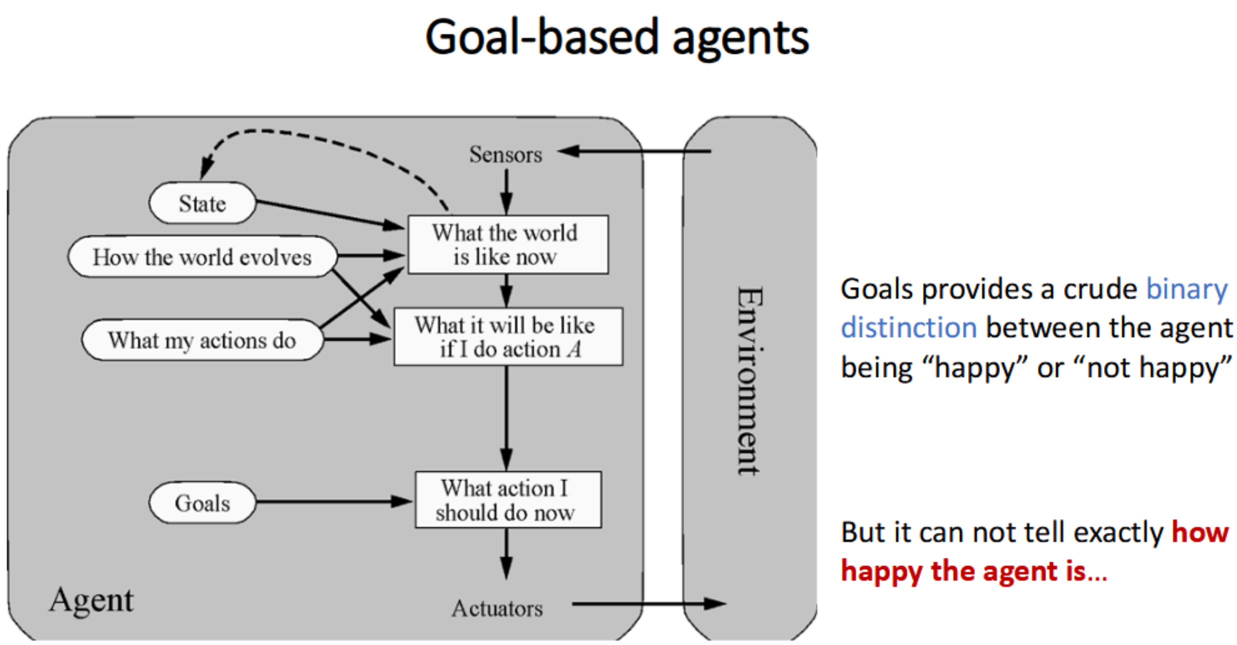
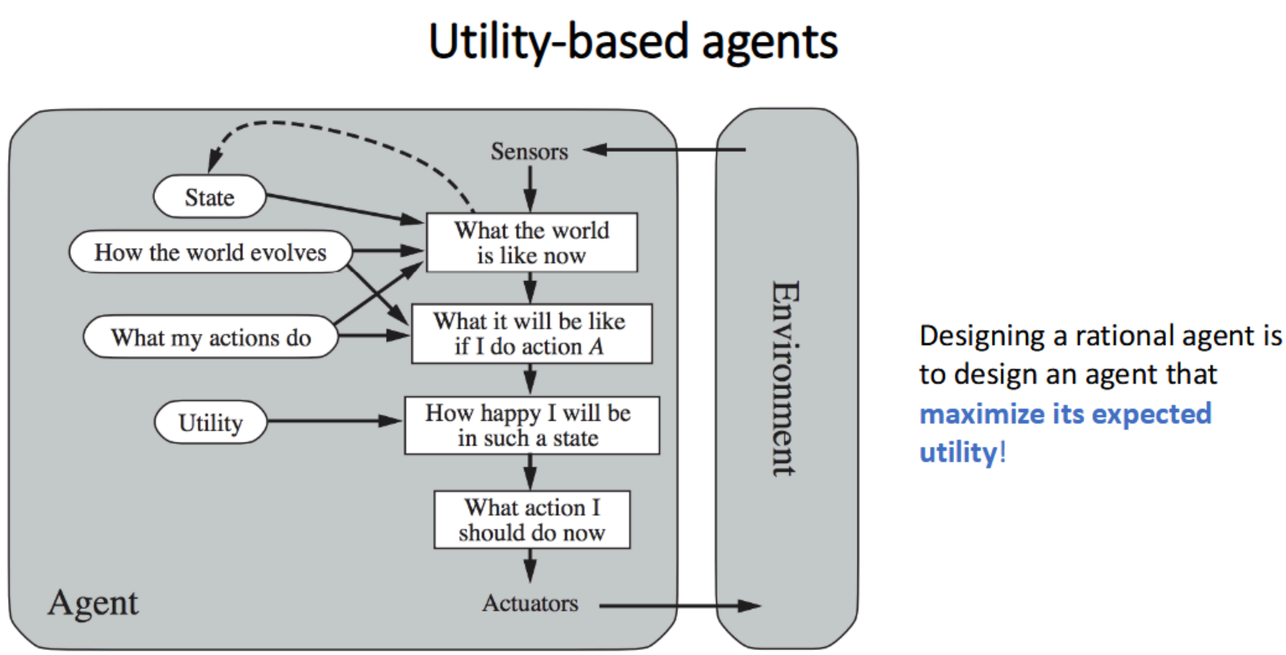
The key challenge for AI is to find out how to write programs that, to the extent possible, produce rational behavior from a smallish program rather than from a vast table.

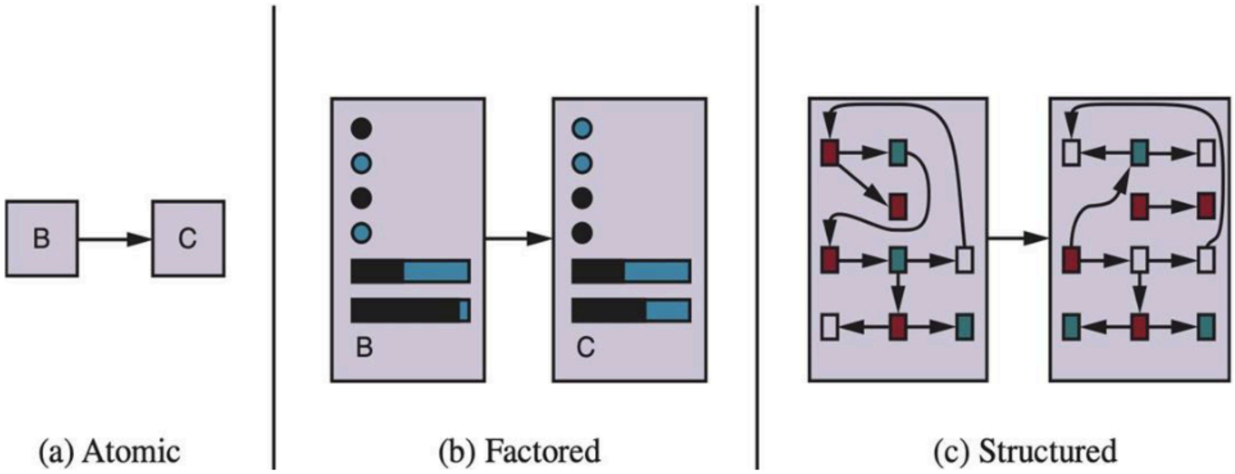
GOAL BASED AGENTS



UTILITY_BASED AGENTS



Spectrum of state representations



Examples:

- Atomic: West Lafayette, Gary, Chicago
- Factored Representation: GPS coordinates, current speed, current direction, current gas in the tank
- Structured representation: Objects -> Car A, Car B, my car, traffic signs/ Relations: Car A is in front of Car B, etc...

SEARCH PROBLEMS AND SOLUTIONS

Search: Assumptions on task environment:

- Fully observable: states can be fully determined by observations
- single agent: only one agent is involved
- Deterministic: outcomes of actions are certain
- Static: the world does not change on its own
- Discrete: a finite number of individual states exist rather than a continuous space of options
- Known: the result functions of actions are known

BASICS

Yucky terminology!!

State: s -> describe the configuration of the environment.
Initial state: s_0 -> the state that an agent starts in.
Actions (Operators):a, activities which move the agent from one state to another.

- Actions(s) -> returns a finite set of actions that can be executed in state s
  Transition model (Successor function): describes what each action does.
- Result(s, a) -> returns the state that results from doing action a at state s
  State space: the set of states reachable from the initial state by applying any sequence of

actions

- usually represented as a graph

Goal state: a set of possible states to reach

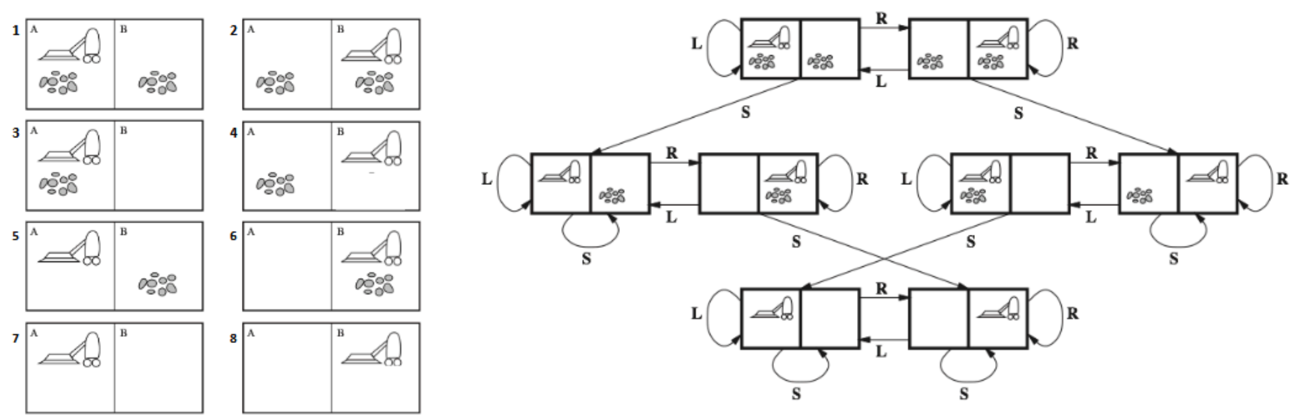
- IsGoal(s) -> indicates whether a state s belongs to the goal states.
- **Step/Action cost**: $ActionCost(s', a, s)$ -> the cost of taking an action a which moves the agent from state s to s'
- **Path**: a sequence of action
- **Path Cost**: the cost of a path
- **Solution**: a path from the initial state to a goal state.

Note the quality of a solution is measure by its path cost.

Optimal solution: Any solution with the lowest path cost.

Search algorithm: the algorithm that determines which actions should be tried in which order to find a path from the initial state to a goal state.

Example



- **States** — Robot locations, A dirty? B dirty?
- **Actions** — Left, Right, Suck
- **Goal test** — no dirt
- **Path cost** — 1 per operation

Note: continuous state space does NOT satisfy search's assumption on task environment.

Also, the **world state** includes every last detail of the environment.
A **search state** keeps only the details needed for problem solving(abstraction)

Be smart in how you formulate your problem because it can get ugly fast!

Goal: Place 8 queens on the chessboard such that no queen attacks any other,

- A queen attacks any piece in the same row, column, or diagonal

- **Idea 1:** **(Complete-state formulation)**
  - **State:** Position of all 8 queens on the board
  - **Actions**: Select one queen and move it up/down/to the left/to the right

Size of state space:

$64 \times 63 \times \cdots \times 57 \approx 1.8 \times 10^{14}$!

- **Idea 2:** **(Incremental formulation)**
  - State: The row number of the n queens ($1 \leq \mathrm{n} \leq 8$) in the leftmost n columns, one per column
  - Actions: Add a queen to any row in the leftmost empty column such that it is not attacked by any existing queens

Size of state space: 2057

The state does not need to be of the same "length"!