# Search

Monday, August 29, 2016 8:55 AM

Planning

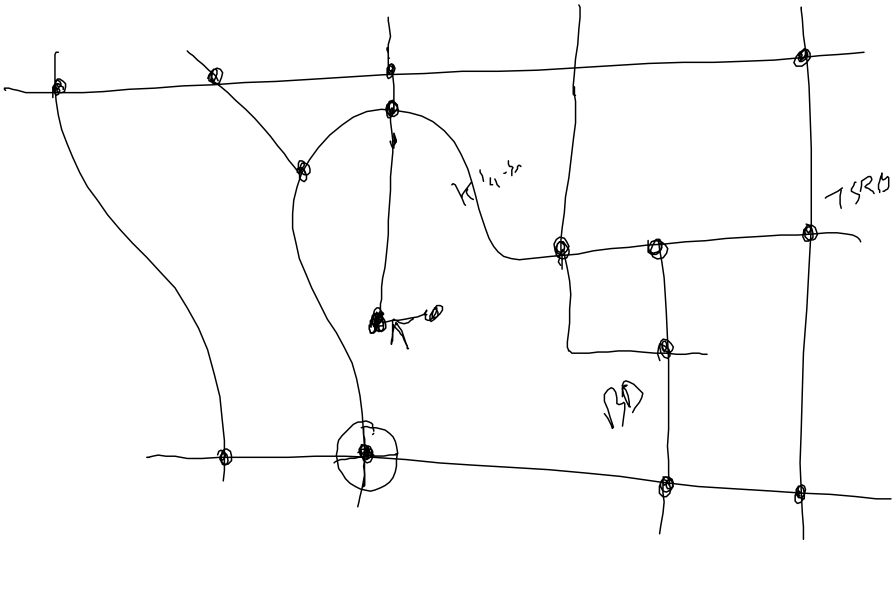
Fully Observable, discrete, deterministic, Sequential

**State**: unique configuration of relevant facts about the world/environment

**Goal**: State we want to be in ⟸ objective

**Actions**: Things to do to transition states

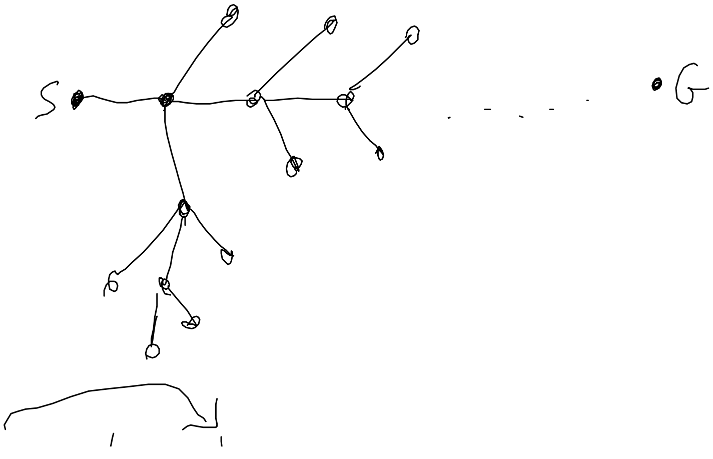
**Problem**: It's not in a goal state & doesn't know how to get to a goal state

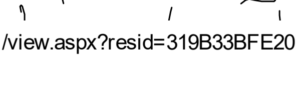


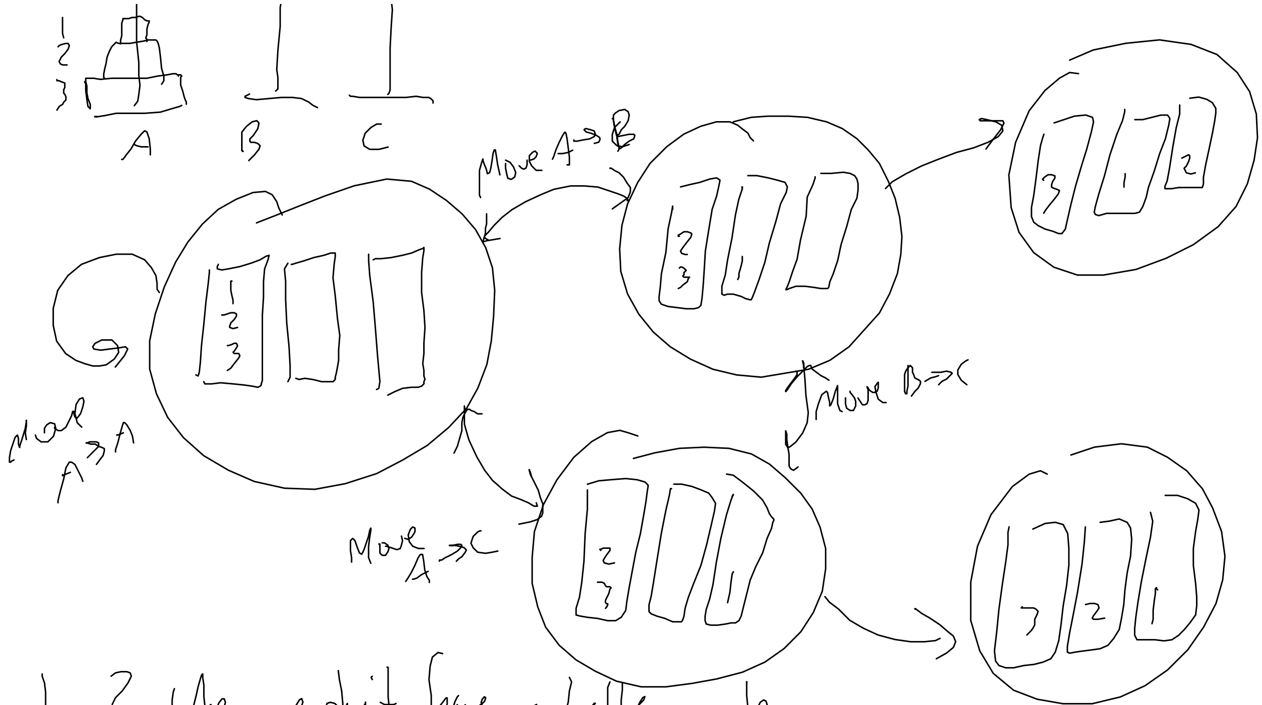
States: Make decisions

Actions: Roads

Transitions btwn states

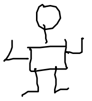






When? When we dont have a better idea

Designing Search Agents: ~~Per~~ Precise Definition of problem

Init state: only one of these  No coffee Location

Goal situation: Desription of the world I want to be in

A goal state: hasCoffee
Location: B&N

Multiple states: { hasCoffee Clough , hasCoffee B&N , hasCoffee Octane .... }

→ A function: f(state) → T/F

Ex: chess
Tells us when to stop searching

Actions: State → state transitions

walk(x,y)

List
Rules of the game

- Walk (x,y) where X & Y are adjacent
- Buy Coffee

## Action Costs: (optional)

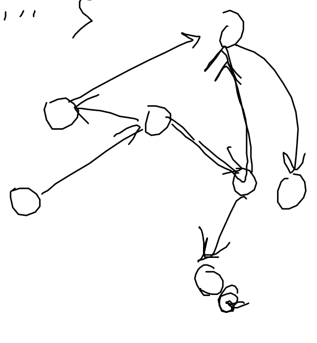
## Pick Algorithm

- uninformed search
- informed search
- Adversarial search
- Constraint satisfaction
- Conformant
- Markov Decision Processes

**Search Problem:** Find a sequence of actions that transforms the initial state into a state recognized as the goal situation

initial ↓ goal state ↓

**Solution:** sequence of actions {a₁, a₂, a₃, a₄ ... }



Brute force: (uninformed)

**Informed search:** give algorithm some intuition

**Pros:** Broadly applicable
Many flavors
Generality - find solutions unanticipated
Necessity - sometimes we don't know a better way

Necessity [illegible]

Cons:

- Computationally expensive
- Not great for stochastic environments