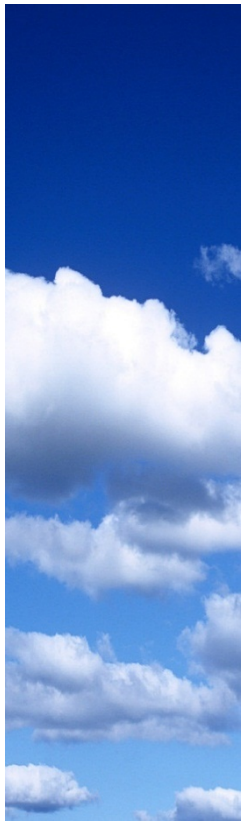
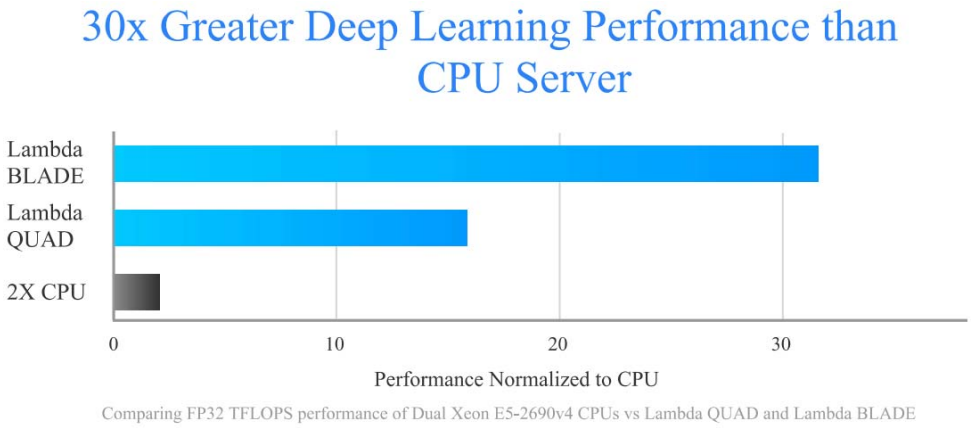


CS 4810 Artificial Intelligence
CS 6810 Topics in Artificial Intelligence

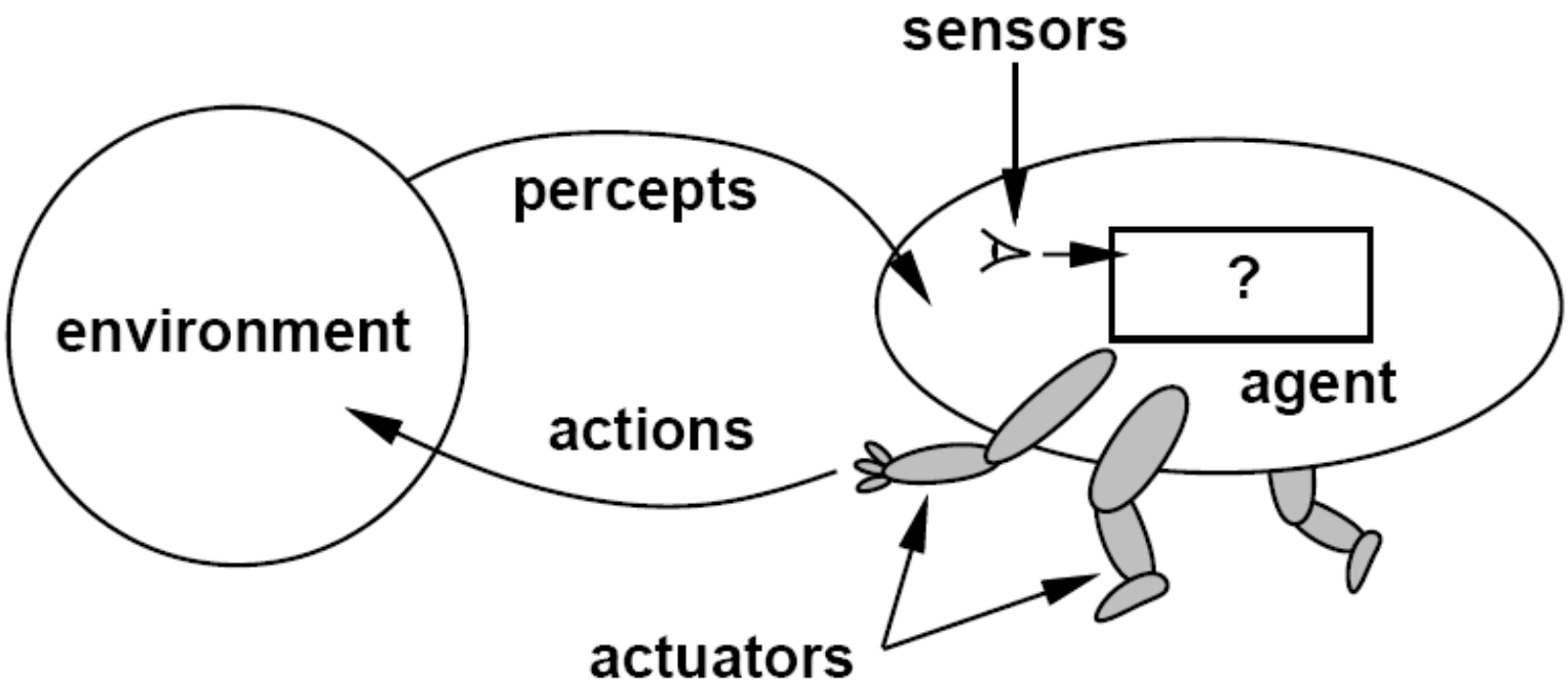
LAMBDA BLADE



Rational Agents Chapter 2

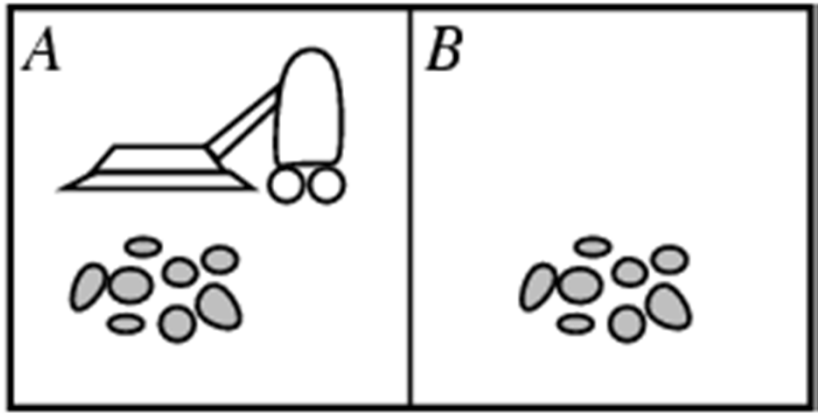
Agents

- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators



Example: Vacuum-Agent

- **Percepts:**
  Location and status,
  e.g., [A,Dirty]
- **Actions:**
  Left, Right, Suck, NoOp



function Vacuum-Agent([location,status]) returns an action

- *if* status = Dirty *then* return Suck
- *else if* location = A *then* return Right
- *else if* location = B *then* return Left

Rational agents

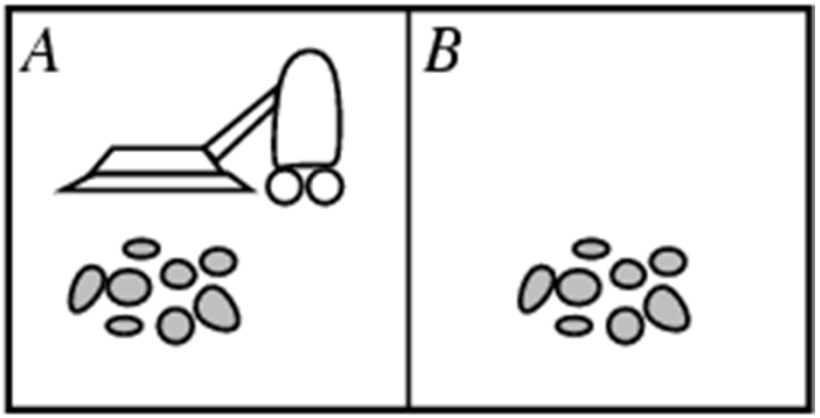
- For each possible percept sequence, a **rational agent** should select an action that is expected to maximize its **performance measure**, *given the evidence provided by the percept sequence and the agent's built-in knowledge*
- **Performance measure (utility function):** An *objective* criterion for success of an agent's behavior
- **Expected utility:**

$$EU(action) = \sum_{outcomes} P(outcome \mid action) U(outcome)$$

- Can a rational agent make mistakes?

Back to Vacuum-Agent

- **Percepts:**
  Location and status,
  e.g., [A,Dirty]
- **Actions:**
  Left, Right, Suck, NoOp



function Vacuum-Agent([location,status]) returns an action

- *if* status = Dirty *then* return Suck
- *else if* location = A *then* return Right
- *else if* location = B *then* return Left

- Is this agent rational?
  - Depends on performance measure, environment properties

Specifying the task environment

- **PEAS: Performance measure, Environment, Actuators, Sensors**
- **P:** a function the agent is maximizing (or minimizing)
  - Assumed given
  - In practice, needs to be computed somewhere
- **E:** a formal representation for *world states*
  - For concreteness, a tuple $(var_1=val_1, var_2=val_2, \ldots, var_n=val_n)$
- **A:** actions that change the state according to a *transition model*
  - Given a state and action, what is the successor state (or distribution over successor states)?
- **S:** observations that allow the agent to infer the world state
  - Often come in very different form than the state itself
  - E.g., in tracking, observations may be pixels and state variables 3D coordinates

PEAS Example: Autonomous taxi

- Performance measure
  - Safe, fast, legal, comfortable trip, maximize profits
- Environment
  - Roads, other traffic, pedestrians, customers
- Actuators
  - Steering wheel, accelerator, brake, signal, horn
- Sensors
  - Cameras, LIDAR (Light Detection and Ranging), speedometer, GPS, odometer, engine sensors, keyboard

Another PEAS example: Spam filter

- Performance measure
  - Minimizing false positives, false negatives
- Environment
  - A user's email account, email server
- Actuators
  - Mark as spam, delete, etc.
- Sensors
  - Incoming messages, other information about user's account

Environment types

- Fully observable vs. partially observable
- Deterministic vs. stochastic
- Episodic vs. sequential
- Static vs. dynamic
- Discrete vs. continuous
- Single agent vs. multi-agent
- Known vs. unknown

Fully observable vs. partially observable

- Do the agent's sensors give it access to the complete state of the environment?
  - For any given world state, are the values of all the variables known to the agent?
- A vacuum agent only knows if current location is dirt

vs.

Source: L. Zettlemoyer

Deterministic vs. stochastic

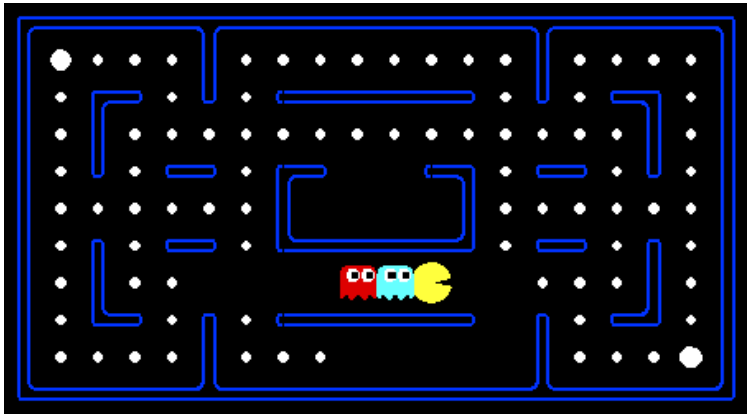
- Is the next state of the environment completely determined by the current state and the agent's action?
  - Is the transition model deterministic (unique successor state given current state and action) or stochastic (distribution over successor states given current state and action)?
  - **Strategic:** the environment is deterministic except for the actions of other agents

vs.

Episodic vs. sequential

- Is the agent's experience divided into unconnected single decisions/actions, or is it a coherent sequence of observations and actions in which the world evolves according to the transition model?

vs.



Static vs. dynamic

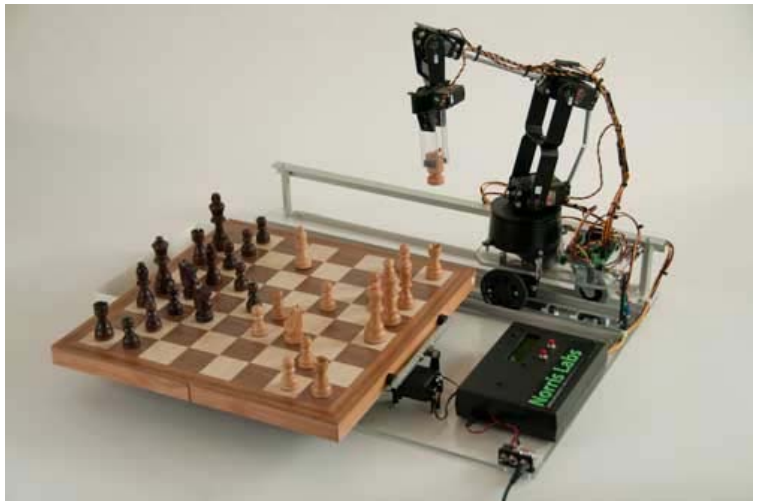
- Is the world changing while the agent is thinking?
  - **Semidynamic:** the environment does not change with the passage of time, but the agent's performance score does

vs.

Discrete vs. continuous

- Does the environment provide a fixed number of distinct percepts, actions, and environment states?
  - Are the values of the state variables discrete or continuous?
  - Time can also evolve in a discrete or continuous fashion

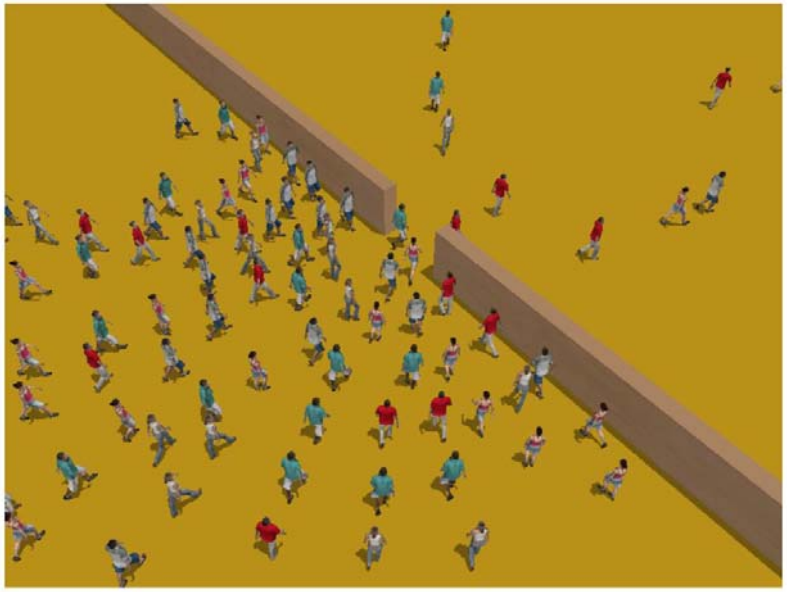
vs.



Single-agent vs. multiagent

- Is an agent operating by itself in the environment?

vs.

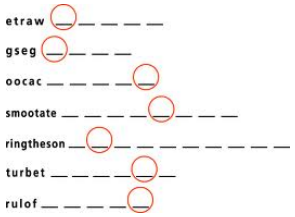


Known vs. unknown

- Are the rules of the environment (transition model and rewards associated with states) known to the agent?
  - Strictly speaking, not a property of the environment, but of the agent's state of knowledge

vs.

Examples of different environments



Word jumble solver

Chess with a clock

Scrabble

Autonomous driving

	Word jumble solver	Chess with a clock	Scrabble	Autonomous driving
Observable	Fully	Fully	Partially	Partially
Deterministic	Deterministic	Strategic	Stochastic	Stochastic
Episodic	Episodic	Sequential	Sequential	Sequential
Static	Static	Semidynamic	Static	Dynamic
Discrete	Discrete	Discrete	Discrete	Continuous
Single agent	Single	Multi	Multi	Multi

Preview of the course

- Deterministic environments: search, constraint satisfaction, classical planning
  - Can be sequential or episodic
- Multi-agent, strategic environments: minimax search, games
  - Can also be stochastic, partially observable
- Stochastic environments
  - Episodic: Bayesian networks, pattern classifiers
  - Sequential, known: Markov decision processes
  - Sequential, unknown: reinforcement learning

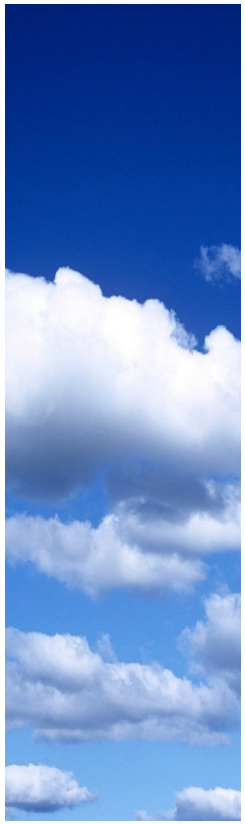
Review: PEAS

Review: PEAS

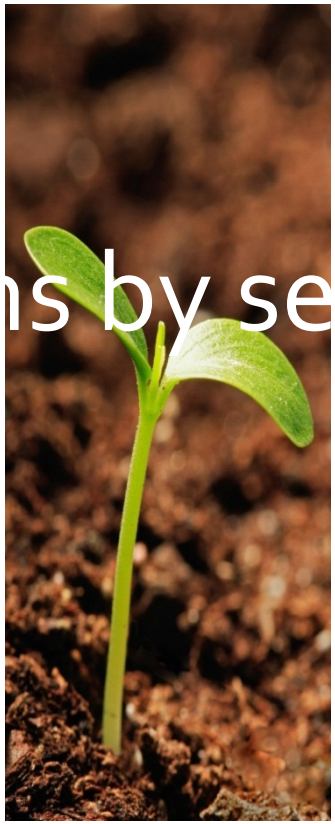
- **P: Performance measure**
  - Function the agent is maximizing (or minimizing)
- **E: Environment**
  - A formal representation for *world states*
  - For concreteness, a tuple (var_1=val_1, var_2=val_2, … ,var_n=val_n)
- **A: Actions**
  - *Transition model:* Given a state and action, what is the successor state (or distribution over successor states)?
- **S: Sensors**
  - Observations that allow the agent to infer the world state
  - Often come in very different form than the state itself

Review: Environment types

- Fully observable vs. partially observable
- Deterministic vs. stochastic (vs. strategic)
- Episodic vs. sequential
- Static vs. dynamic (vs. semidynamic)
- Discrete vs. continuous
- Single agent vs. multi-agent
- Known vs. unknown

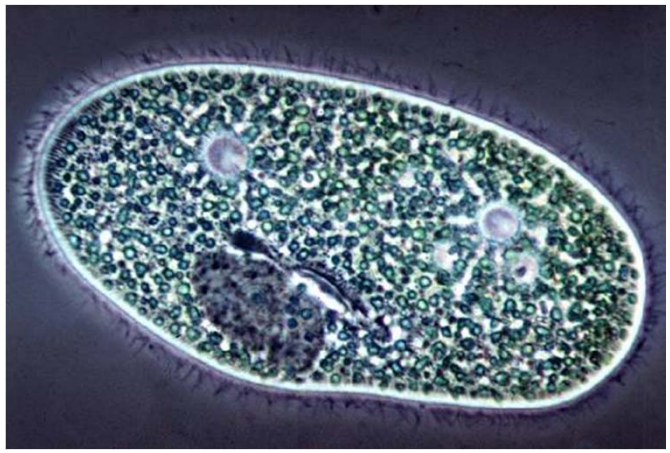


Chapter 3
Solving problems by searching



Types of agents

Reflex agent



- Consider how the world IS
- Choose action based on current percept
- Do not consider the future consequences of actions

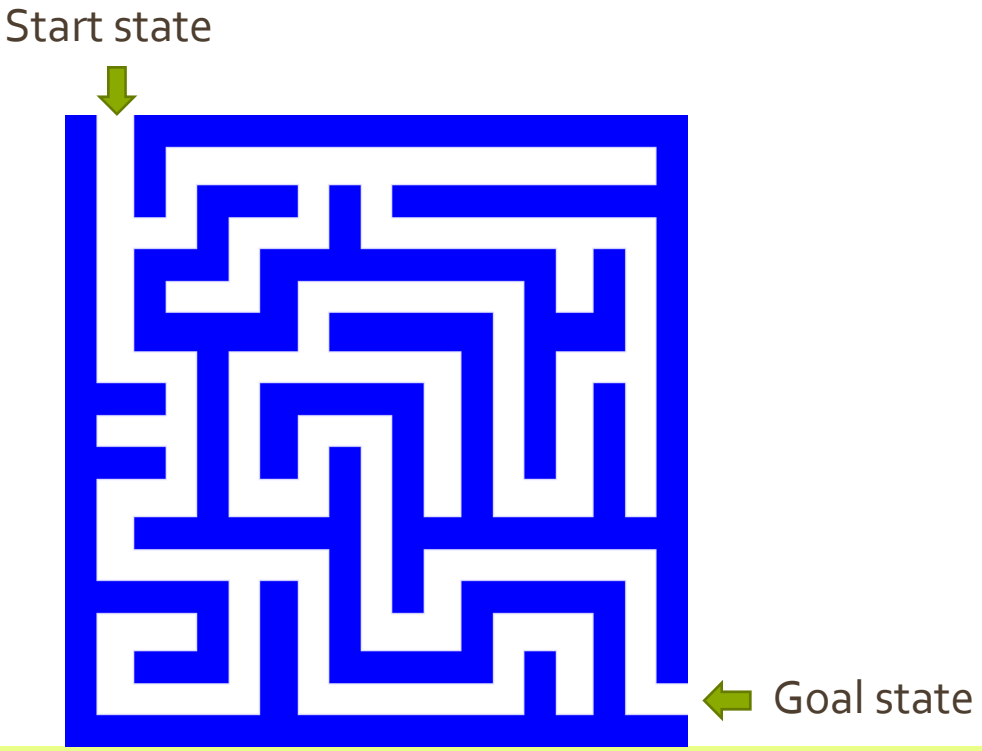
Planning agent

- Consider how the world WOULD BE
- Decisions based on (hypothesized) consequences of actions
- Must have a model of how the world evolves in response to actions
- Must formulate a goal

Source: D. Klein, P. Abbeel

Search

- We will consider the problem of designing **goal-based agents** in **fully observable**, **deterministic**, **discrete**, **known** environments

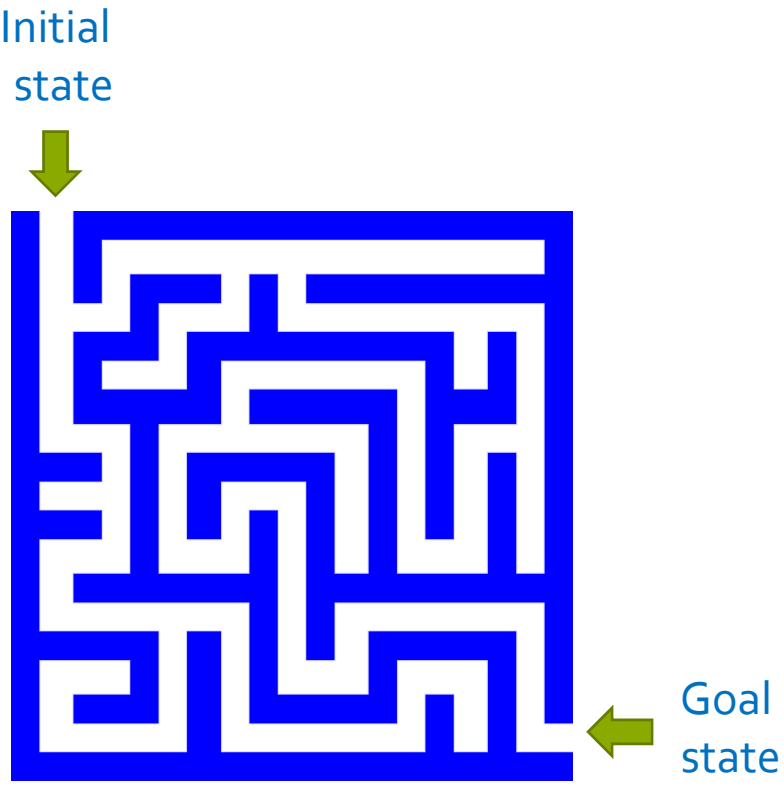


Search

- We will consider the problem of designing **goal-based agents** in **fully observable**, **deterministic**, **discrete**, **known** environments
  - The agent must find a *sequence of actions* that reaches the goal
  - The **performance measure** is defined by (a) reaching the goal and (b) how "expensive" the path to the goal is
  - We are focused on the process of finding the solution; while executing the solution, we assume that the agent can safely ignore its percepts (**open-loop system**)

Search problem components

- **Initial state**
- **Actions**
- **Transition model**
  - What state results from performing a given action in a given state?
- **Goal state**
- **Path cost**
  - Assume that it is a sum of nonnegative *step costs*

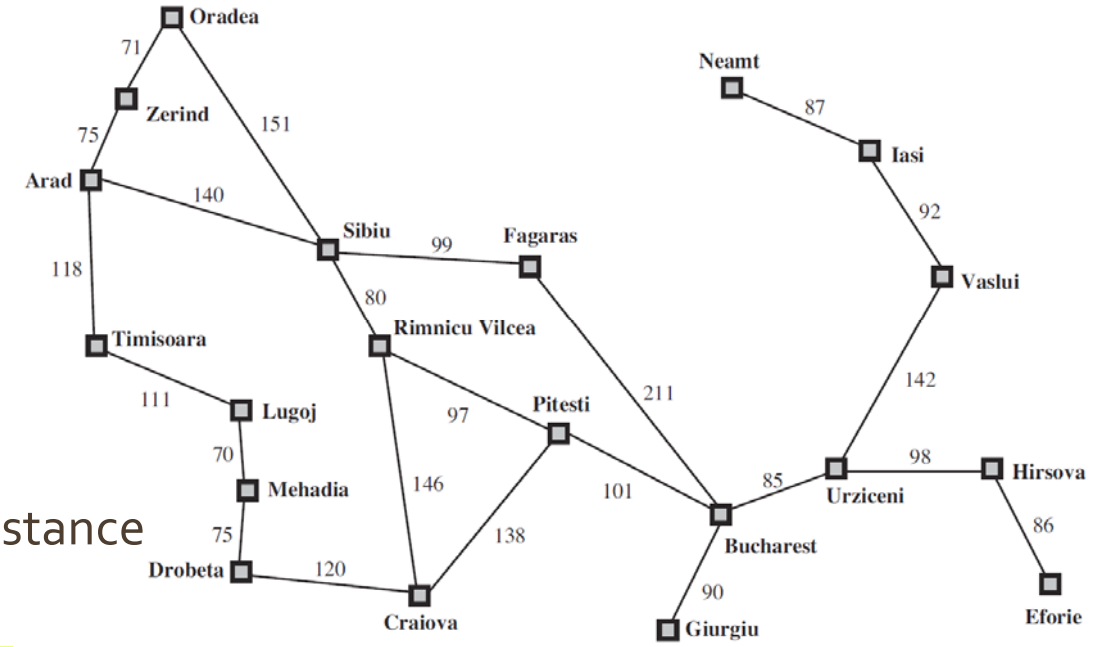


- The **optimal solution** is the sequence of actions that gives the *lowest* path cost for reaching the goal

Example: Romania

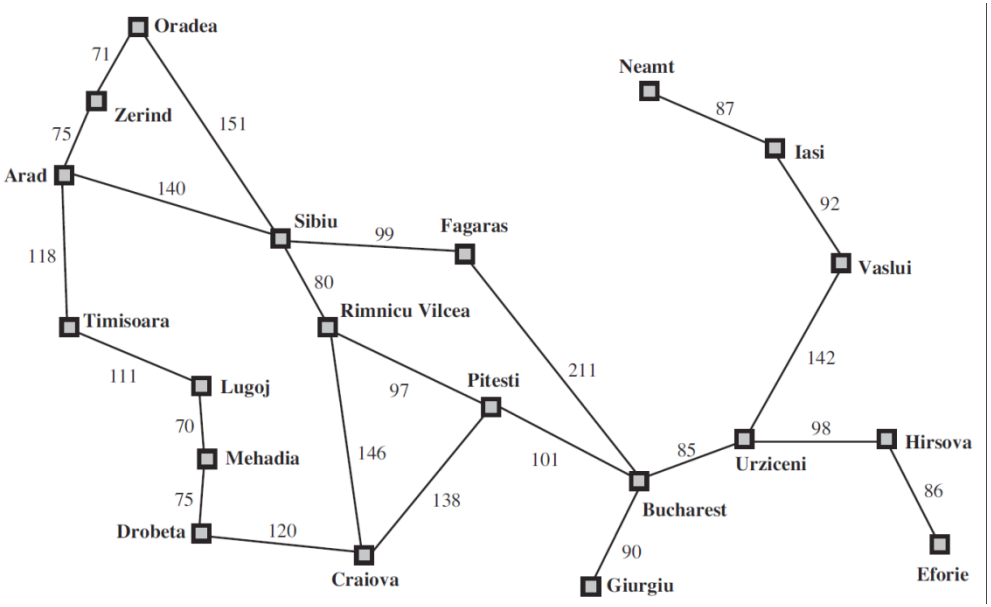
- On vacation in Romania; currently in Arad
- Flight leaves tomorrow from Bucharest

- **Initial state**
  - Arad
- **Actions**
  - Go from one city to another
- **Transition model**
  - If you go from city A to city B, you end up in city B
- **Goal state**
  - Bucharest
- **Path cost**
  - Sum of edge costs (total distance traveled)

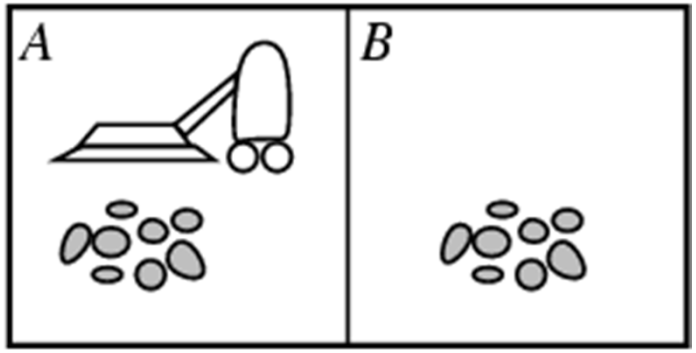


State space

- The initial state, actions, and transition model define the **state space** of the problem
  - The set of all states reachable from initial state by any sequence of actions
  - Can be represented as a **directed graph** where the nodes are states and links between nodes are actions
- What is the state space for the Romania problem?

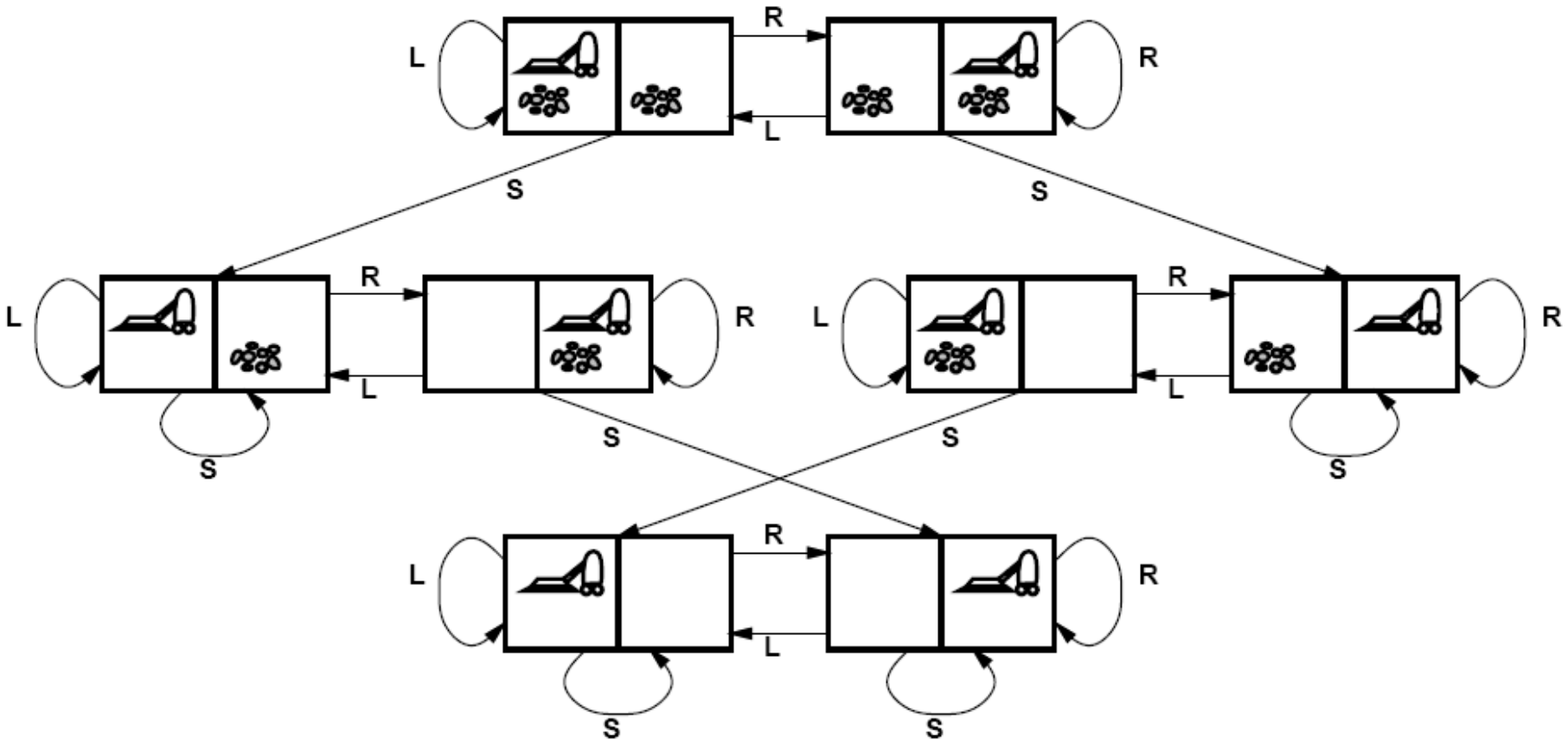


Example: Vacuum world



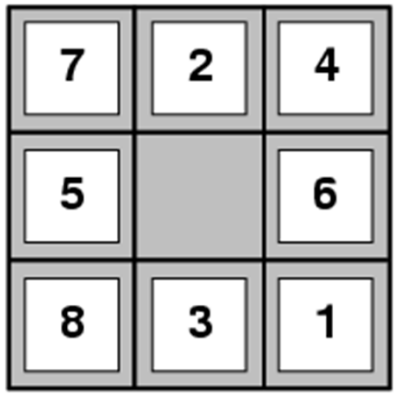
- **States**
  - Agent location and dirt location
  - How many possible states?
  - What if there are *n* possible locations?
    - The size of the state space grows exponentially with the "size" of the world!
- **Actions**
  - Left, right, suck
- **Transition model**

Vacuum world state space graph

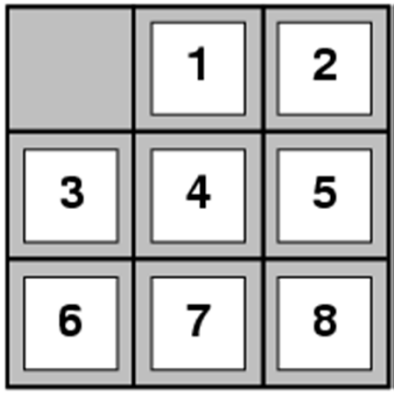


Example: The 8-puzzle

- **States**
  - Locations of tiles
    - 8-puzzle: 181,440 states (9!/2)
    - 15-puzzle: ~10 trillion states
    - 24-puzzle: ~10^{25} states
- **Actions**
  - Move blank left, right, up, down
- **Path cost**
  - 1 per move



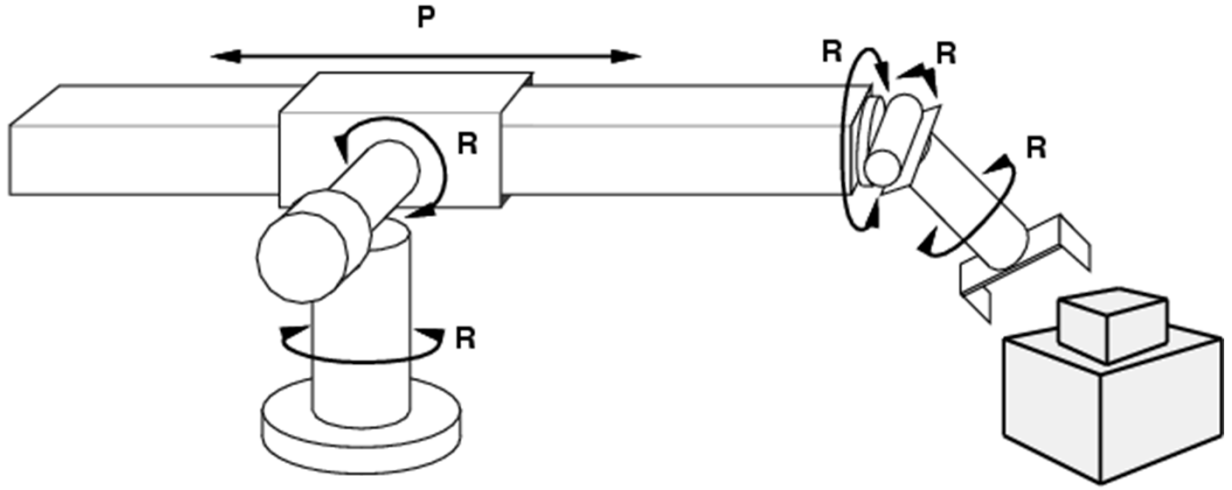
Start State



Goal State

- Finding the optimal solution of n-Puzzle is NP-hard

Example: Robot motion planning



- **States**
  - Real-valued joint parameters (angles, displacements)
- **Actions**
  - Continuous motions of robot joints
- **Goal state**
  - Configuration in which object is grasped
- **Path cost**
  - Time to execute, smoothness of path, etc.

Search

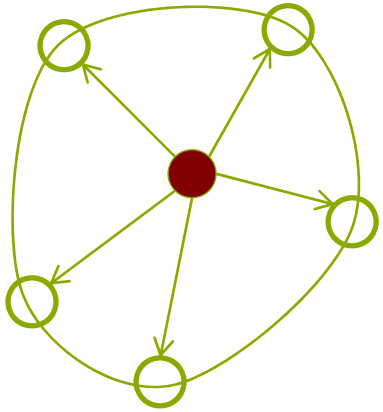
- Given:
  - **Initial state**
  - **Actions**
  - **Transition model**
  - **Goal state**
  - **Path cost**
- How do we find the optimal solution?
  - How about building the state space and then using Dijkstra's shortest path algorithm?
    - Complexity of Dijkstra's is $O(E + V \log V)$, where V is the size of the state space
    - The state space may be huge!

Search: Basic idea

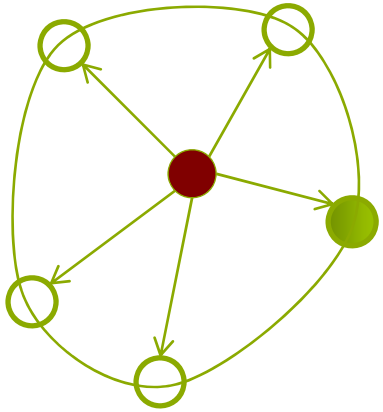
- Let's begin at the start state and **expand** it by making a list of all possible successor states
- Maintain a **frontier** or a list of unexpanded states
- At each step, pick a state from the frontier to expand
- Keep going until you reach a goal state
- Try to expand as few states as possible

Search: Basic idea

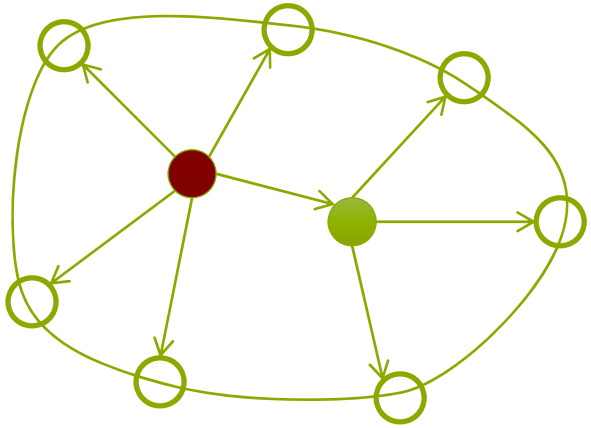
Search: Basic idea



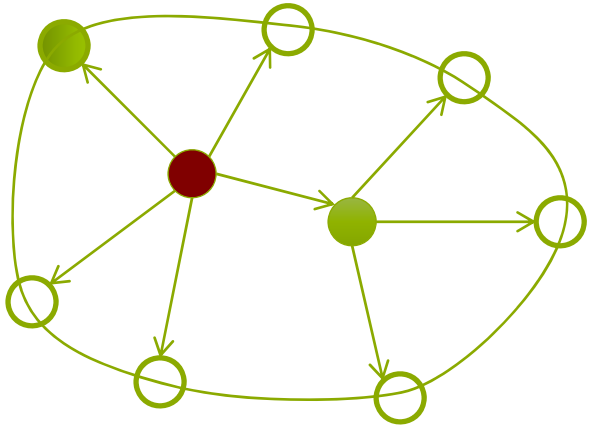
Search: Basic idea



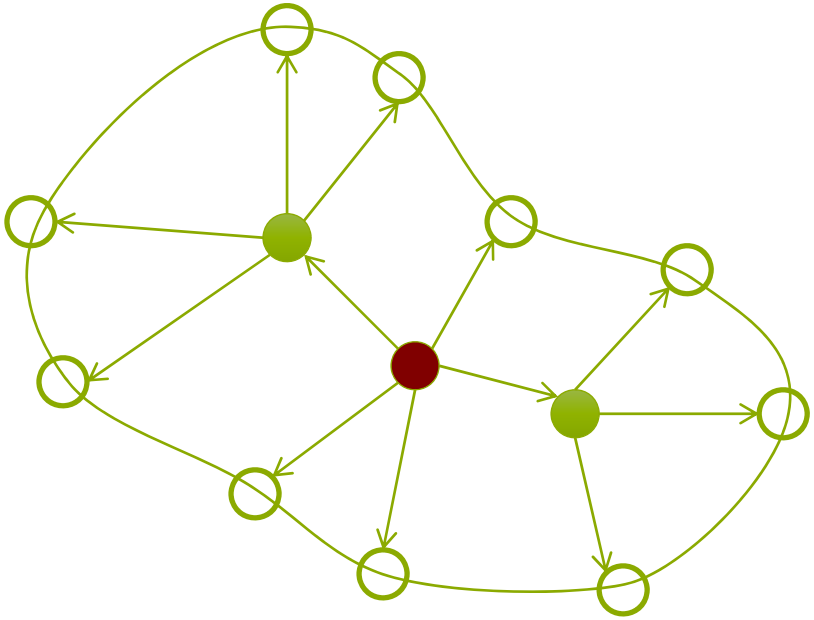
Search: Basic idea



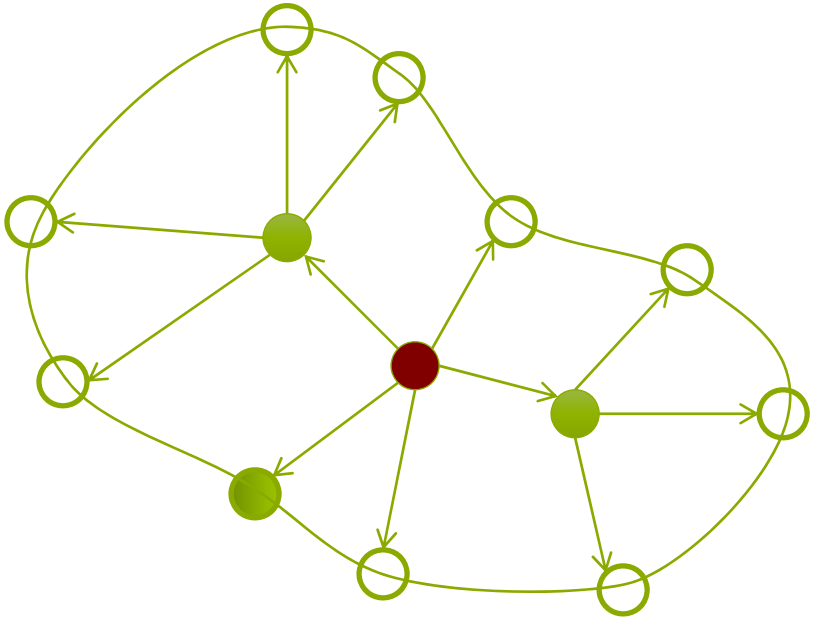
Search: Basic idea



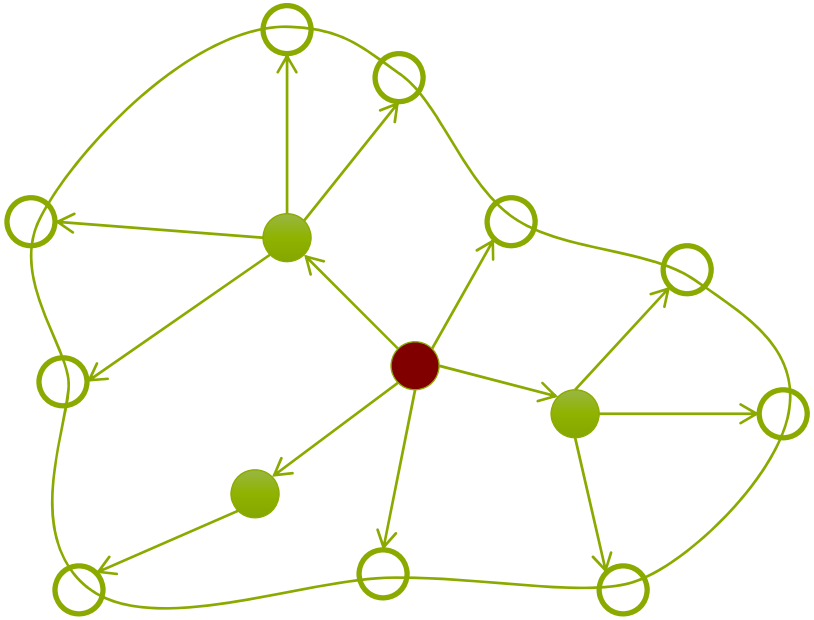
Search: Basic idea



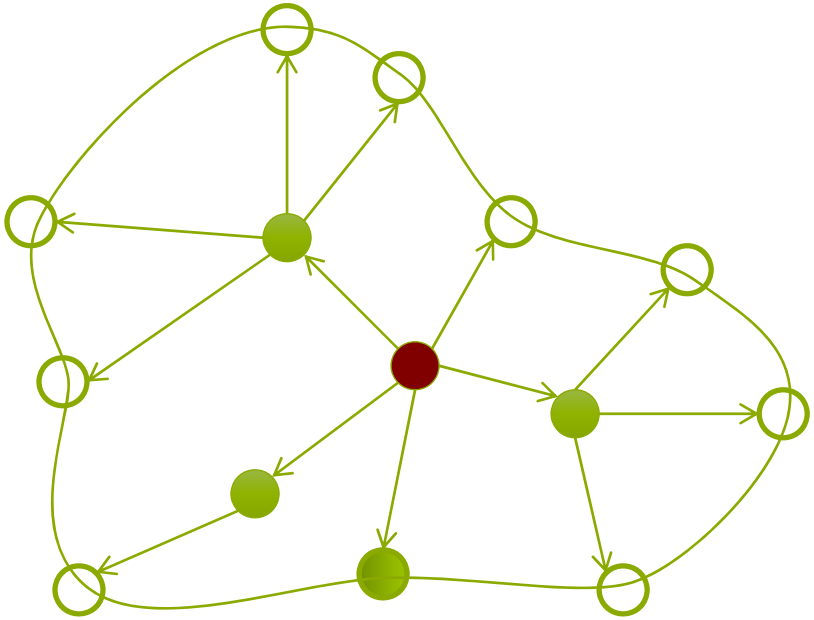
Search: Basic idea



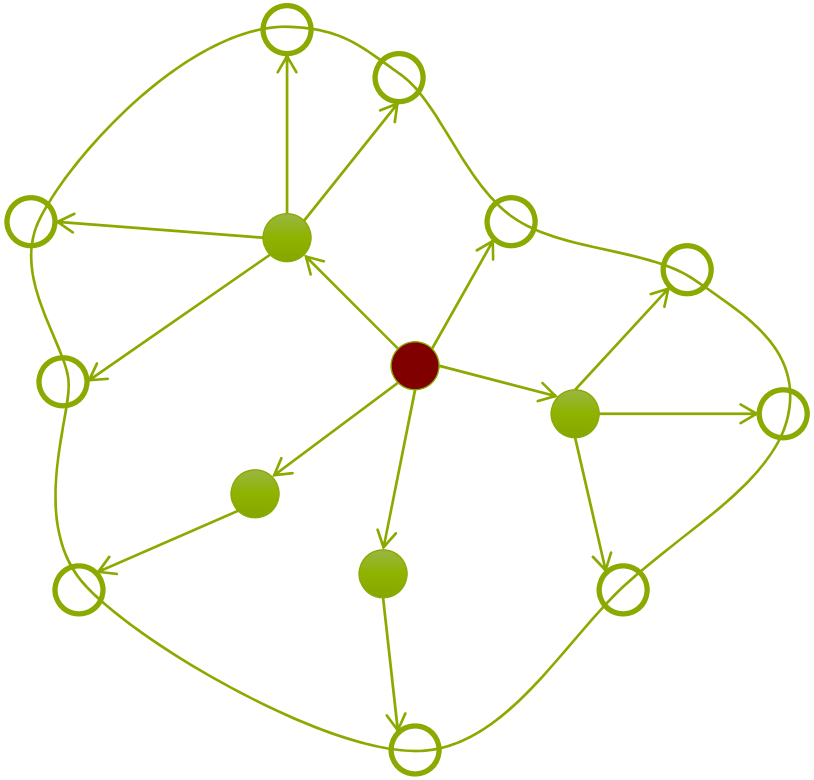
Search: Basic idea



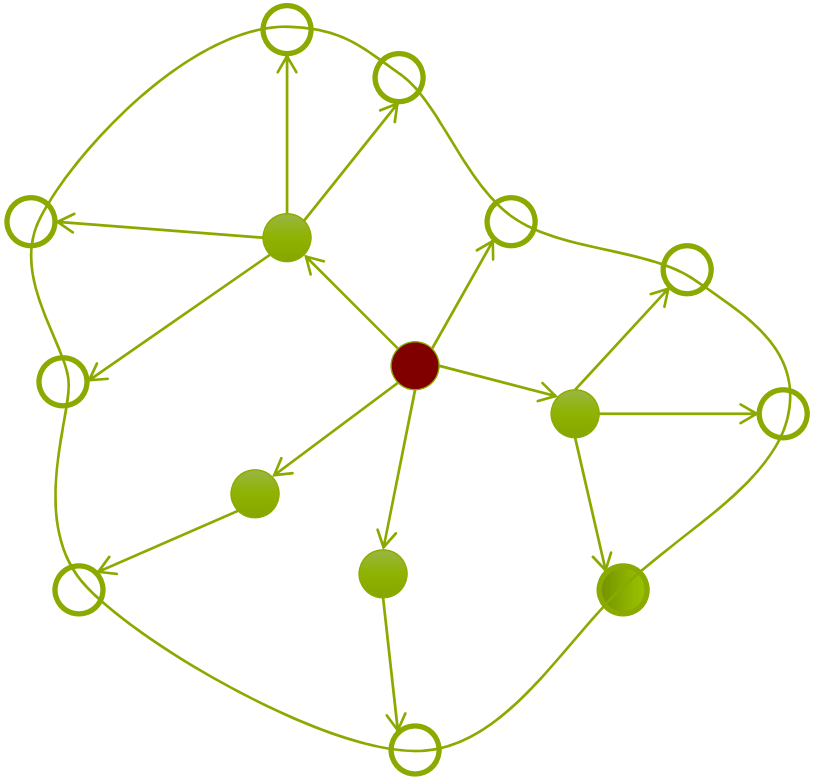
Search: Basic idea



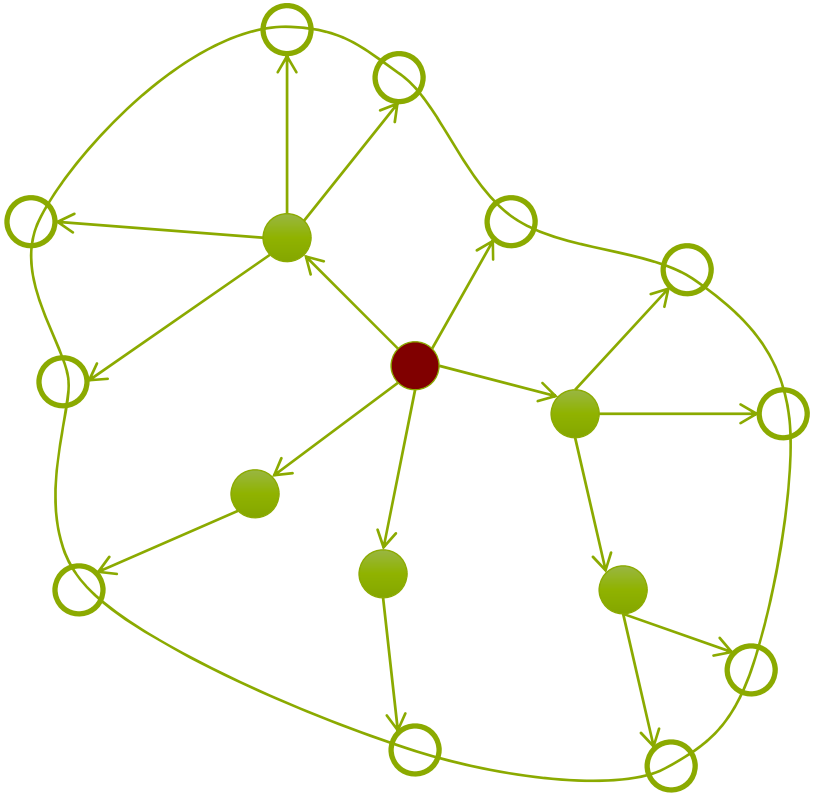
Search: Basic idea



Search: Basic idea

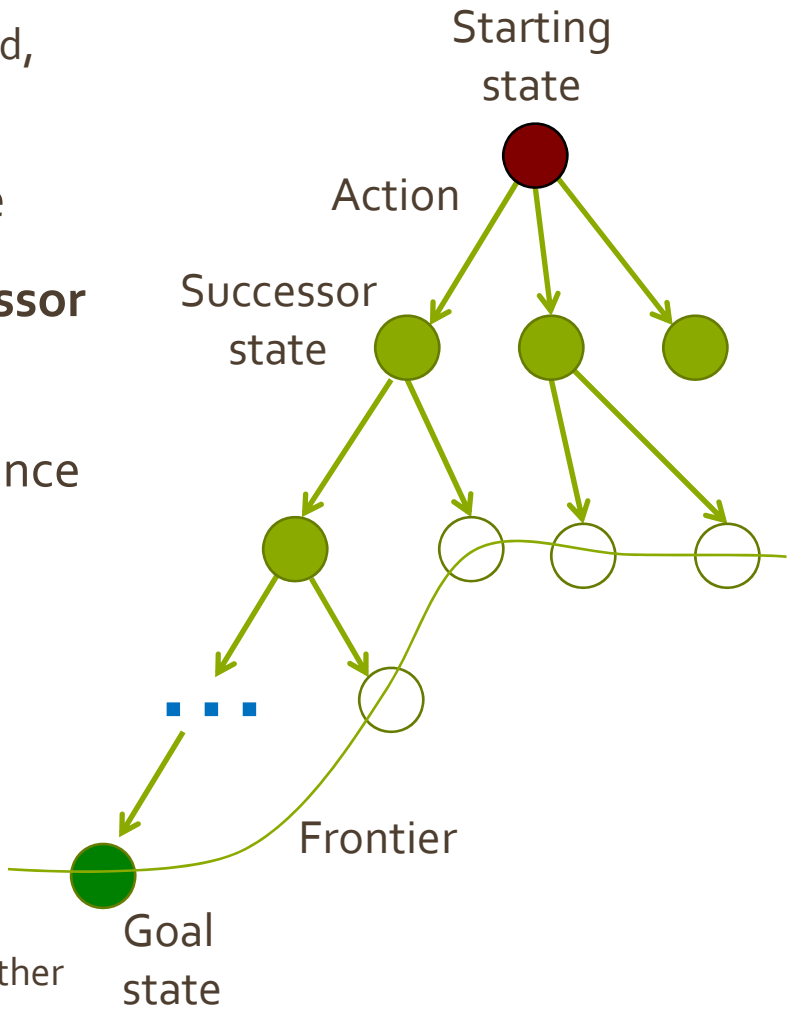


Search: Basic idea



Search tree

- "What if" tree of sequences of actions and outcomes
  - When we are searching, we are not acting in the world, merely "thinking" about the possibilities
- The root node corresponds to the starting state
- The children of a node correspond to the **successor states** of that node's state
- A path through the tree corresponds to a sequence of actions
  - A solution is a path ending in the goal state
- Nodes vs. states
  - A state is a representation of the world, while a node is a data structure that is part of the search tree
    - Node has to keep pointer to parent, path cost, possibly other info



Tree Search Algorithm Outline

- Initialize the **frontier** using the **starting state**
- While the frontier is not empty
  - Choose a frontier node according to **search strategy** and take it off the frontier
  - If the node contains the **goal state**, return solution
  - Else **expand** the node and add its children to the frontier