Announcements

Project 0: Python Tutorial.
Due on Monday 1/27. 11:59 pm.
Announcements

Project 0: Python Tutorial.
   Due on Monday 1/27. 11:59 pm.

Homework 0: Math Diagnostic.
   Due on Wednesday 1/29. 11:59 pm.
Announcements

Project 0: Python Tutorial.
   Due on Monday 1/27. 11:59 pm.

Homework 0: Math Diagnostic.
   Due on Wednesday 1/29. 11:59 pm.

Project 1: Search.
   Out: longer than average. Get started.
   Due Friday. 2/7. 11:59 pm.
Announcements

Project 0: Python Tutorial.
   Due on Monday 1/27. 11:59 pm.

Homework 0: Math Diagnostic.
   Due on Wednesday 1/29. 11:59 pm.

Project 1: Search.
   Out: longer than average. Get started.
   Due Friday. 2/7. 11:59 pm.

Sections.
   Start next week!
Announcements

Project 0: Python Tutorial.
   Due on Monday 1/27. 11:59 pm.

Homework 0: Math Diagnostic.
   Due on Wednesday 1/29. 11:59 pm.

Project 1: Search.
   Out: longer than average. Get started.
   Due Friday. 2/7. 11:59 pm.

Sections.
   Start next week!

Office Hours: still working on rooms.
   Department/university issues still being dealt with.
Announcements

Project 0: Python Tutorial.
   Due on Monday 1/27. 11:59 pm.

Homework 0: Math Diagnostic.
   Due on Wednesday 1/29. 11:59 pm.

Project 1: Search.
   Out: longer than average. Get started.
   Due Friday. 2/7. 11:59 pm.

Sections.
   Start next week!

Office Hours: still working on rooms.
   Department/university issues still being dealt with.

Check: signed up for Piazza and Gradescope?
Announcements

Project 0: Python Tutorial.
   Due on Monday 1/27. 11:59 pm.

Homework 0: Math Diagnostic.
   Due on Wednesday 1/29. 11:59 pm.

Project 1: Search.
   Out: longer than average. Get started.
   Due Friday. 2/7. 11:59 pm.

Sections.
   Start next week!

Office Hours: still working on rooms.
   Department/university issues still being dealt with.

Check: signed up for Piazza and Gradescope?
Pinned Post: AI in the news!
Lecture Attendance Link

Lecture Attendance Link: http://bit.ly/2GEMokS
Search.
Today.

Agents that Plan Ahead
Agents that Plan Ahead
Search Problems.
Agents that Plan Ahead

Search Problems.
Model world with state space.
Agents that Plan Ahead

Search Problems.
Model world with state space.
Setting up state spaces.
Agents that Plan Ahead
Search Problems.
Model world with state space.
Setting up state spaces.

Maybe today.
Agents that Plan Ahead

Search Problems.
  Model world with state space.
  Setting up state spaces.

Maybe today.

Uninformed Search Methods:
Agents that Plan Ahead

Search Problems.
Model world with state space.
Setting up state spaces.

Maybe today.

Uninformed Search Methods:
Depth-First Search
Agents that Plan Ahead
Search Problems.
Model world with state space.
Setting up state spaces.

Maybe today.

Uninformed Search Methods:
Depth-First Search
Breadth-First Search
Agents that Plan Ahead
Search Problems.
  Model world with state space.
  Setting up state spaces.
Maybe today.
Uninformed Search Methods:
  Depth-First Search
  Breadth-First Search
  Uniform-Cost Search
Frame of AI

An agent perceives its environment with sensors and acts on environment using actuators.
An agent percieves its environment with sensors and acts on environment using actuators.

Car:
  Sensors:
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
  Sensors: camera,
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
Sensors: camera, lidar,
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
  Sensors: camera, lidar, speed gauge,
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
- Sensors: camera, lidar, speed gauge, ...
- Actuators:
An agent **perceives** its environment with **sensors** and **acts** on environment using **actuators**.

**Car:**
- Sensors: camera, lidar, speed gauge,..
- Actuators: gas pedal,
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
  Sensors: camera, lidar, speed gauge, ..
  Actuators: gas petal, , steering wheel, brake petal,
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
- Sensors: camera, lidar, speed gauge, ..
- Actuators: gas petal, , steering wheel, brake petal, ...
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
- Sensors: camera, lidar, speed gauge, ...
- Actuators: gas petal, , steering wheel, brake petal, ...
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
  - Sensors: camera, lidar, speed gauge, ..
  - Actuators: gas petal, , steering wheel, brake petal, ...

Website, Program,
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
- Sensors: camera, lidar, speed gauge, ..
- Actuators: gas petal, , steering wheel, brake petal, ...

Website, Program, ...
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
- Sensors: camera, lidar, speed gauge, ...
- Actuators: gas pedal, , steering wheel, brake petal, ...

Website, Program, ...

Input/Output:
An agent **perceives** its environment with **sensors** and **acts** on environment using **actuators**.

**Car:**
- Sensors: camera, lidar, speed gauge, ..
- Actuators: gas petal, , steering wheel, brake petal, ...

**Website, Program, ...**

Input/Output: Outputs affect environment, which affects input
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
- Sensors: camera, lidar, speed gauge, ..
- Actuators: gas petal, , steering wheel, brake petal, ...

Website, Program, ...

Input/Output: Outputs affect environment, which affects input, which outputs,
An agent perceives its environment with sensors and acts on environment using actuators.

Car:
- Sensors: camera, lidar, speed gauge, ...
- Actuators: gas petal, , steering wheel, brake petal, ...

Website, Program, ...

Input/Output: Outputs affect environment, which affects input, which outputs, ...
Rationality

A rational agent chooses actions that maximize expected utility.
A *rational agent* chooses actions that maximize *expected* utility.

Today: agents that have a goal, and a cost.
E.g., reach goal with lowest cost.
A rational agent chooses actions that maximize expected utility.

Today: agents that have a goal, and a cost.
E.g., reach goal with lowest cost.

Later: agents have numerical utilities, rewards, etc.
E.g., takes action that maximizes total reward over time.
(Reward: largest total profit. or expected total profit.)
Agent Design

The environment largely determines the agent design.
Agent Design

The environment largely determines the agent design.

Fully/partially observable $\rightarrow$ agent request memory (internal state)
Agent Design

The environment largely determines the agent design.

- Fully/partially observable → agent requests memory (internal state)
- Discrete/continuous → agent can/can’t enumerate all states
Agent Design

The environment largely determines the agent design.

- Fully/partially observable → agent request memory (internal state)
- Discrete/continuous → agent can/can’t enumerate all states
- Stochastic/deterministic → agent deals with contingencies
Agent Design

The environment largely determines the agent design.

- Fully/partially observable → agent request memory (internal state)
- Discrete/continuous → agent can/can’t enumerate all states
- Stochastic/deterministic → agent deals with contingencies
- Single-agent/multi-agent → agent may need to behave randomly.
Agents that Plan
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)

Examples:
- Stove: hot, ouch!
- Car: deer, brake!
- Go to Tahoe: skid in snow, ???

[Demo: reflex (L2D1 and L2D2)]
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
Reflex Agents

Reflex agents:

▶ Choose action based on current percept (and maybe memory)

▶ May have memory or a model of the world’s current state

▶ Do not consider the future consequences of their actions.

▶ Consider how the world **is**.

Examples:

- **Stove:** hot, ouch!
- **Car:** deer, brake!
- **Go to Tahoe:** skid in snow, ???

[Demo: reflex (L2D1 and L2D2)]
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.

Can a reflex agent be rational?
Reflex Agents

Reflex agents:

▶ Choose action based on current percept (and maybe memory)

▶ May have memory or a model of the world’s current state

▶ Do not consider the future consequences of their actions.

▶ Consider how the world IS.

Can a reflex agent be rational?

Examples:
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.

Can a reflex agent be rational?
Examples:
  Stove:
Reflex Agents

Reflex agents:

▶ Choose action based on current percept (and maybe memory)
▶ May have memory or a model of the world’s current state
▶ Do not consider the future consequences of their actions.
▶ Consider how the world IS.

Can a reflex agent be rational?
Examples:
   Stove: hot,
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.

Can a reflex agent be rational?
Examples:
Stove: hot, ouch!
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.

Can a reflex agent be rational?
Examples:
  - Stove: hot, ouch!
  - Car:
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.

Can a reflex agent be rational?
Examples:
- Stove: hot, ouch!
- Car: deer,
Reflex Agents

Reflex agents:

▶ Choose action based on current percept (and maybe memory)

▶ May have memory or a model of the world’s current state

▶ Do not consider the future consequences of their actions.

▶ Consider how the world IS.

Can a reflex agent be rational?
Examples:
  Stove: hot, ouch!
  Car: deer, brake!
Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.

Can a reflex agent be rational?

Examples:
Stove: hot, ouch!
Car: deer, brake!
Go to Tahoe:
Reflex Agents

Reflex agents:

▶ Choose action based on current percept (and maybe memory)

▶ May have memory or a model of the world’s current state

▶ Do not consider the future consequences of their actions.

▶ Consider how the world IS.

Can a reflex agent be rational?

Examples:

Stove: hot, ouch!
Car: deer, brake!
Go to Tahoe: skid in snow,
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world is.

Can a reflex agent be rational?
Examples:
- Stove: hot, ouch!
- Car: deer, brake!
- Go to Tahoe: skid in snow, ???
Reflex Agents

Reflex agents:

▶ Choose action based on current percept (and maybe memory)

▶ May have memory or a model of the world’s current state

▶ Do not consider the future consequences of their actions.

▶ Consider how the world IS.

Can a reflex agent be rational?

Examples:

Stove: hot, ouch!
Car: deer, brake!
Go to Tahoe: skid in snow, ???
Reflex Agents

Reflex agents:

- Choose action based on current percept (and maybe memory)
- May have memory or a model of the world’s current state
- Do not consider the future consequences of their actions.
- Consider how the world IS.

Can a reflex agent be rational?

Examples:
- Stove: hot, ouch!
- Car: deer, brake!
- Go to Tahoe: skid in snow, ???

[Demo: reflex (L2D1 and L2D2)]
Video of Demo Reflex Optimal
Video of Demo Reflex Odd
Planning agents:

Planning agents:

- Ask "what if?"
- Decisions based on (hypothesized) consequences of actions.
- Must have a model of how the world evolves in response to actions.
- Must formulate a goal (test).
- Consider how the world WOULD BE.
- Optimal or not optimal.
- Complete or not.

Planning vs. replanning

[Demo: nearest dot re-planning (L2D3), mastermind (L2D4)]
Planning agents:

Ask “what if?”
Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Must have a model of how the world evolves in response to actions.
Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Must have a model of how the world evolves in response to actions.
Must formulate a goal (test).
Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Must have a model of how the world evolves in response to actions.
Must formulate a goal (test).
Consider how the world WOULD BE.
Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Must have a model of how the world evolves in response to actions.
Must formulate a goal (test).
Consider how the world WOULD BE.
Planning Agents

Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Must have a model of how the world evolves in response to actions.
Must formulate a goal (test).
Consider how the world WOULD BE.

Optimal or not optimal.
Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Must have a model of how the world evolves in response to actions.
Must formulate a goal (test).
Consider how the world WOULD BE.

Optimal or not optimal.
Complete or not.
Planning agents:

- Ask “what if?”
- Decisions based on (hypothesized) consequences of actions.
- Must have a model of how the world evolves in response to actions.
- Must formulate a goal (test).
- Consider how the world WOULD BE.

Optimal or not optimal.
Complete or not.
Planning vs. replanning
Planning agents:

Ask “what if?”
Decisions based on (hypothesized) consequences of actions.
Must have a model of how the world evolves in response to actions.
Must formulate a goal (test).
Consider how the world WOULD BE.

Optimal or not optimal.
Complete or not.
Planning vs. replanning

[Demo: nearest dot re-planning (L2D3), mastermind (L2D4)]
Video of Demo

Replanning
Video of Demo Mastermind
Search Problems
Search Problems

A search problem consists of:
Search Problems

A search problem consists of:

A state space
Search Problems

A search problem consists of:

A state space

A state space
Search Problems

A search problem consists of:

A state space

A successor function
(with actions, costs)
Search Problems

A search problem consists of:

A state space

A successor function (with actions, costs)
Search Problems

A search problem consists of:

- A state space
- A successor function (with actions, costs)
- A start state and a goal test

A start state and a goal test
Search Problems

A **search problem** consists of:

- A state space
- A successor function (with actions, costs)
- A start state and a goal test

A solution is a sequence of actions (a plan) which transforms the start state to a goal state
Search Problems Are Models
Example: Traveling in Romania

State space:

Cities

Successor function:
Roads: Neighboring city with cost = distance

Start state:
Arad

Goal test:
Is state == Bucharest?

Solution?
Example: Traveling in Romania

State space:
Cities
Example: Traveling in Romania

State space:
Cities

Successor function:
Roads: Neighboring city with cost = distance

Start state:
Arad

Goal test:
Is state == Bucharest

Solution?
Example: Traveling in Romania

State space:
Cities

Successor function:
Roads: Neighboring city with cost = distance
Example: Traveling in Romania

State space:
Cities

Successor function:
Roads: Neighboring city with cost = distance

Start state:
Example: Traveling in Romania

State space:
Cities

Successor function:
Roads: Neighboring city with cost = distance

Start state:
Arad
Example: Traveling in Romania

State space:
Cities

Successor function:
Roads: Neighboring city with cost = distance

Start state:
Arad

Goal test:
Example: Traveling in Romania

State space:
- Cities

Successor function:
- Roads: Neighboring city with cost = distance

Start state:
- Arad

Goal test:
- Is state == Bucharest?
Example: Traveling in Romania

State space:
   Cities

Successor function:
   Roads: Neighboring city with cost = distance

Start state:
   Arad

Goal test:
   Is state == Bucharest?

Solution?
What’s in a State Space?

The world state includes every last detail of the environment.
What’s in a State Space?

The world state includes every last detail of the environment.
What’s in a State Space?

The world state includes every last detail of the environment.
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

Problem: Pathing
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing

States:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing

States: $(x, y)$ location
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing

States: \((x, y)\) location

Actions:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing

States: \((x, y)\) location
Actions: NSEW
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing

States: \((x, y)\) location
Actions: NSEW
Successor:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

Problem: Pathing
States: \((x, y)\) location
Actions: NSEW
Successor: update location only
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing

States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing

States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test: is \((x, y) = \text{END}\)?
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing
States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test: is \((x, y) = \text{END}\)?
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing
States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test: is \((x, y) = \text{END}\)?

**Problem:** Eat-All-Dots
States:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing  
States: \((x, y)\) location  
Actions: NSEW  
Successor: update location only  
Goal test: is \((x, y) = \text{END}\)?

**Problem:** Eat-All-Dots  
States: \{\((x, y), \text{dot booleans}\)\}
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing
States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test: is \((x, y) = \text{END}\)?

**Problem:** Eat-All-Dots
States: \{\((x, y)\), dot booleans\}
Actions:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem:** Pathing
States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test: is \((x, y) = \text{END}\)?

**Problem:** Eat-All-Dots
States: \\{(x, y), \text{dot booleans}\}
Actions: NSEW
Goal test: all dots false
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem: Pathing**  
States: \((x, y)\) location  
Actions: NSEW  
Successor: update location only  
Goal test: is \((x, y) = \text{END}\)?

**Problem: Eat-All-Dots**  
States: \(\{(x, y), \text{dot booleans}\}\)  
Actions: NSEW  
Successor:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem: Pathing**
- States: \((x, y)\) location
- Actions: NSEW
- Successor: update location only
- Goal test: is \((x, y) = \text{END}\)?

**Problem: Eat-All-Dots**
- States: \{\((x, y)\), dot booleans\}
- Actions: NSEW
- Successor: update location and possibly a dot boolean
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

Problem: Pathing
States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test: is \((x, y) = \text{END}\)?

Problem: Eat-All-Dots
States: \(\{(x, y), \text{dot booleans}\}\)
Actions: NSEW
Successor: update location and possibly a dot boolean
Goal test:
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem: Pathing**
States: \((x, y)\) location
Actions: NSEW
Successor: update location only
Goal test: is \((x, y) = \text{END}\)?
What’s in a State Space?

The world state includes every last detail of the environment.

Search state keeps only details needed for planning (abstraction).

**Problem: Pathing**
- **States:** \((x, y)\) location
- **Actions:** NSEW
- **Successor:** update location only
- **Goal test:** is \((x, y) = \text{END}\)?

**Problem: Eat-All-Dots**
- **States:** \{\((x, y)\), dot booleans\}
- **Actions:** NSEW
- **Successor:** update location and possibly a dot boolean
- **Goal test:** dots all false
State Space Sizes?

World state:
State Space Sizes?

World state:
Agent positions:
State Space Sizes?

World state:
Agent positions: 120
State Space Sizes?

- World state: 120
- Agent positions: 120
- Food count: 30
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions:
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12

States for pathing? 120
States for eat-all-dots? 120 x (2^30)
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing:
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW
How many World states?
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW
How many World states?
120
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW
How many World states?
\[120 \times (2^{30}) \times (12^2) \times 4\]
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW

How many World states?
120 \times (2^{30}) \times (12^2) \times 4

States for pathing?
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW

How many World states?
\[120 \times (2^{30}) \times (12^2) \times 4\]

States for pathing?
120
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW

How many World states?
$120 \times (2^{30}) \times (12^2) \times 4$

States for pathing?
120

States for eat-all-dots?
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW

How many World states?
\[ 120 \times (2^{30}) \times (12^2) \times 4 \]

States for pathing?
120

States for eat-all-dots?
120
State Space Sizes?

World state:
Agent positions: 120
Food count: 30
Ghost positions: 12
Agent facing: NSEW

How many World states?
$120 \times (2^{30}) \times (12^2) \times 4$

States for pathing?
120

States for eat-all-dots?
$120 \times (2^{30})$
Problem: Safe Passage

What does the state space have to specify?
- agent position
- dot booleans
- power pellet booleans
- remaining scared time
Problem: Safe Passage

Problem: eat all dots while keeping the ghosts perma-scared?
Problem: Safe Passage

Problem: eat all dots while keeping the ghosts perma-scared?

What does the state space have to specify?
Problem: eat all dots while keeping the ghosts perma-scared?

What does the state space have to specify?

agent position
Problem: Safe Passage

Problem: eat all dots while keeping the ghosts perma-scared?

What does the state space have to specify?

agent position
dot booleans
Problem: Safe Passage

Problem: eat all dots while keeping the ghosts perma-scared?

What does the state space have to specify?

agent position
dot booleans
power pellet booleans
Problem: Safe Passage

Problem: eat all dots while keeping the ghosts perma-scared?

What does the state space have to specify?

- agent position
- dot booleans
  - power pellet booleans
- remaining scared time
Agent Design

The environment largely determines the agent design.
Agent Design

The environment largely determines the agent design.

Fully/partially observable $\rightarrow$ agent request memory (internal state)
Agent Design

The environment largely determines the agent design.

Fully/partially observable $\rightarrow$ agent request memory (internal state)
Discrete/continuous $\rightarrow$ agent can/can’t enumerate all states
Agent Design

The environment largely determines the agent design.

- Fully/partially observable $\rightarrow$ agent request memory (internal state)
- Discrete/continuous $\rightarrow$ agent can/can’t enumerate all states
- Stochastic/deterministic $\rightarrow$ agent deals with contingencies
Agent Design

The environment largely determines the agent design.

- Fully/partially observable $\rightarrow$ agent request memory (internal state)
- Discrete/continuous $\rightarrow$ agent can/can’t enumerate all states
- Stochastic/deterministic $\rightarrow$ agent deals with contingencies
- Single-agent/multi-agent $\rightarrow$ agent may need to behave randomly.
State Space Graphs and Search Trees
State Space Graphs

State space graph: A mathematical representation of a search problem

In a state space graph, each state occurs only once! We can rarely build this full graph in memory (it's too big), but it's a useful idea. E.g., replanning.
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations.
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations.
- Arrows represent successors (action results).
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations.
- Arrows represent successors (action results).
- The goal test is a set of goal nodes (maybe only one).
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations.
- Arcs represent successors (action results).
- The goal test is a set of goal nodes (maybe only one).

In a state space graph, each state occurs only once!
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations.
- Arcs represent successors (action results).
- The goal test is a set of goal nodes (maybe only one).

In a state space graph, each state occurs only once!

We can rarely build this full graph in memory (it’s too big),
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations.
- Arcs represent successors (action results).
- The goal test is a set of goal nodes (maybe only one).

In a state space graph, each state occurs only once!

We can rarely build this full graph in memory (it’s too big), but it’s a useful idea.
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations.
- Arcs represent successors (action results).
- The goal test is a set of goal nodes (maybe only one).

In a state space graph, each state occurs only once!

We can rarely build this full graph in memory (it’s too big), but it’s a useful idea.

E.g., replanning.
State space graph: A mathematical representation of a search problem
State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations
State space graphs: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations
- Arcs represent successors (action results)

Tiny search graph for a tiny search problem
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations
- Arcs represent successors (action results)
- The goal test is a set of goal nodes (maybe only one)
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations
- Arcs represent successors (action results)
- The goal test is a set of goal nodes (maybe only one)

In a state space graph, each state occurs only once!
State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations
- Arcs represent successors (action results)
- The goal test is a set of goal nodes (maybe only one)

In a state space graph, each state occurs only once!

We can rarely build this full graph in memory (it’s too big),
State Space Graphs

State space graph: A mathematical representation of a search problem

- Nodes are (abstracted) world configurations
- Arcs represent successors (action results)
- The goal test is a set of goal nodes (maybe only one)

In a state space graph, each state occurs only once!

We can rarely build this full graph in memory (it’s too big), but it’s a useful idea.
Search Trees

This is now / start

A search tree:

Possible futures.
Search Trees

This is now / start

A search tree:
A “what if” tree of plans and their outcomes

Possible futures.
Search Trees

This is now / start

A search tree:
A “what if” tree of plans and their outcomes
The start state is the root node

Possible futures.
Search Trees

This is now / start

“N”  “E”

Possible futures.

A search tree:
A “what if” tree of plans and their outcomes
The start state is the root node
Children correspond to successors
Search Trees

This is now / start

A search tree:
A “what if” tree of plans and their outcomes
The start state is the root node
Children correspond to successors
Nodes show states, but correspond to PLANS that achieve those states

Possible futures.
Search Trees

This is now / start

A search tree:
A “what if” tree of plans and their outcomes

The start state is the root node

Children correspond to successors

Nodes show states, but correspond to PLANS that achieve those states

For most problems, we can never actually build the whole tree

Possible futures.
State Space Graphs vs. Search Trees

Each node in the search tree is an entire path in the state space graph. We construct both on demand and construct as little as possible.
State Space Graphs vs. Search Trees

Each NODE in search tree is an entire PATH in state space graph.
State Space Graphs vs. Search Trees

Each NODE in search tree is an entire PATH in state space graph.
We construct both on demand – and we construct as little as possible.
Quiz: State Space Graphs vs. Search Trees

Consider this 4-state graph:

How large could search tree be?
Consider this 4-state graph:

How large could search tree be?

$\infty$
Quiz: State Space Graphs vs. Search Trees

Consider this 4-state graph:

How large could search tree be?
Consider this 4-state graph:

How large could search tree be?

∞
Tree Search
Search Example: Romania
Searching with a Search Tree

Search:

1. Expand out potential plans (tree nodes)
2. Maintain a fringe of partial plans under consideration.
3. Try to expand as few tree nodes as possible
Searching with a Search Tree

Search:
(1) Expand out potential plans (tree nodes)
Searching with a Search Tree

Search:
(1) Expand out potential plans (tree nodes)
(2) Maintain a fringe of partial plans under consideration.
Searching with a Search Tree

Search:
1. Expand out potential plans (tree nodes)
2. Maintain a fringe of partial plans under consideration.
Searching with a Search Tree

Search:
(1) Expand out potential plans (tree nodes)
(2) Maintain a fringe of partial plans under consideration.
(3) Try to expand as few tree nodes as possible
General Tree Search

Important ideas:
General Tree Search

Important ideas:

Fringe
General Tree Search

Important ideas:

- Fringe
- Expansion
General Tree Search

Important ideas:

Fringe
Expansion
Exploration strategy
General Tree Search

Important ideas:

- Fringe
- Expansion
- Exploration strategy

Main question: which fringe nodes to explore?
General Tree Search

Important ideas:
- Fringe
- Expansion
- Exploration strategy

Main question: which fringe nodes to explore?

```
function TREE-SEARCH(problem, strategy) returns a solution, or failure
    initialize the search tree using the initial state of problem
    loop do
        if there are no candidates for expansion then return failure
        choose a leaf node for expansion according to strategy
        if the node contains a goal state then return the corresponding solution
        else expand the node and add the resulting nodes to the search tree
    end
```