

At the end of the class you should be able to:

- define a directed graph
- represent a problem as a state-space graph

- Often we are not given an algorithm to solve a problem, but only a specification of what is a solution — we have to search for a solution.

- Often we are not given an algorithm to solve a problem, but only a specification of what is a solution — we have to search for a solution.
- A typical problem is when the agent is in one state, it has a set of deterministic actions it can carry out, and wants to get to a goal state.

- Often we are not given an algorithm to solve a problem, but only a specification of what is a solution — we have to search for a solution.
- A typical problem is when the agent is in one state, it has a set of deterministic actions it can carry out, and wants to get to a goal state.
- Many AI problems can be abstracted into the problem of finding a path in a directed graph.

- Often we are not given an algorithm to solve a problem, but only a specification of what is a solution — we have to search for a solution.
- A typical problem is when the agent is in one state, it has a set of deterministic actions it can carry out, and wants to get to a goal state.
- Many AI problems can be abstracted into the problem of finding a path in a directed graph.
- Often there is more than one way to represent a problem as a graph.

- **flat** or modular or hierarchical
- **explicit states** or features or individuals and relations
- static or finite stage or **indefinite stage** or infinite stage
- **fully observable** or partially observable
- **deterministic** or stochastic dynamics
- **goals** or complex preferences
- **single agent** or multiple agents
- **knowledge is given** or knowledge is learned
- **perfect rationality** or bounded rationality

State-space Problem

A **state-space problem** consists of

- a set of states
- a subset of states called the **start states**

State-space Problem

A **state-space problem** consists of

- a set of states
- a subset of states called the **start states**
- a set of actions

State-space Problem

A **state-space problem** consists of

- a set of states
- a subset of states called the **start states**
- a set of actions
- an **action function**: given a state and an action, returns a new state

State-space Problem

A **state-space problem** consists of

- a set of states
- a subset of states called the **start states**
- a set of actions
- an **action function**: given a state and an action, returns a new state
- a set of goal states, specified as function, $goal(s)$

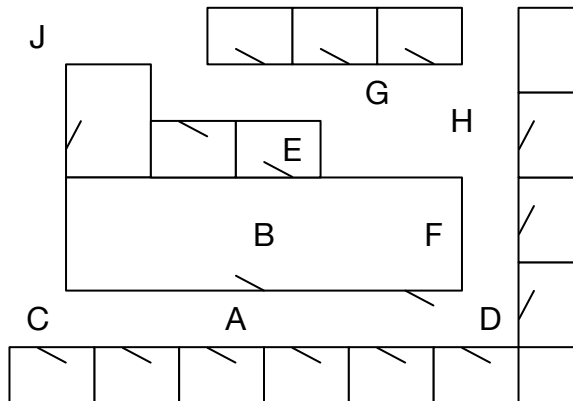
State-space Problem

A **state-space problem** consists of

- a set of states
- a subset of states called the **start states**
- a set of actions
- an **action function**: given a state and an action, returns a new state
- a set of goal states, specified as function, $goal(s)$
- a criterion that specifies the quality of an acceptable solution.

Example Problem for Delivery Robot

The robot is at A and the goal is to get to G:



Directed Graphs

- A (directed) **graph** consists of a set N of **nodes** and a set A of ordered pairs of nodes, called **arcs**.

Directed Graphs

- A (directed) **graph** consists of a set N of **nodes** and a set A of ordered pairs of nodes, called **arcs**.
- Node n_2 is a **neighbor** of n_1 if there is an arc from n_1 to n_2 . That is, if $\langle n_1, n_2 \rangle \in A$.

Directed Graphs

- A (directed) **graph** consists of a set N of **nodes** and a set A of ordered pairs of nodes, called **arcs**.
- Node n_2 is a **neighbor** of n_1 if there is an arc from n_1 to n_2 . That is, if $\langle n_1, n_2 \rangle \in A$.
- A **path** is a sequence of nodes $\langle n_0, n_1, \dots, n_k \rangle$ such that $\langle n_{i-1}, n_i \rangle \in A$.

Directed Graphs

- A (directed) **graph** consists of a set N of **nodes** and a set A of ordered pairs of nodes, called **arcs**.
- Node n_2 is a **neighbor** of n_1 if there is an arc from n_1 to n_2 . That is, if $\langle n_1, n_2 \rangle \in A$.
- A **path** is a sequence of nodes $\langle n_0, n_1, \dots, n_k \rangle$ such that $\langle n_{i-1}, n_i \rangle \in A$.
- Given **start nodes** and **goal nodes**, a **solution** is a path from a start node to a goal node.

Directed Graphs

- A (directed) **graph** consists of a set N of **nodes** and a set A of ordered pairs of nodes, called **arcs**.
- Node n_2 is a **neighbor** of n_1 if there is an arc from n_1 to n_2 . That is, if $\langle n_1, n_2 \rangle \in A$.
- A **path** is a sequence of nodes $\langle n_0, n_1, \dots, n_k \rangle$ such that $\langle n_{i-1}, n_i \rangle \in A$.
- Given **start nodes** and **goal nodes**, a **solution** is a path from a start node to a goal node.
- When there is a **cost** associated with arcs, the cost of a path is the sum of the costs of the arcs in the path:

$$\text{cost}(\langle n_0, n_1, \dots, n_k \rangle) = \sum_{i=1}^k \text{cost}(\langle n_{i-1}, n_i \rangle)$$

An **optimal solution** is one with minimum cost.

What is a state?

What is a state?

- A **state** needs to include enough information to
- determine what is the next state

What is a state?

A **state** needs to include enough information to

- determine what is the next state
- determine whether the goal is achieved

What is a state?

A **state** needs to include enough information to

- determine what is the next state
- determine whether the goal is achieved
- determine the cost.

What is a state?

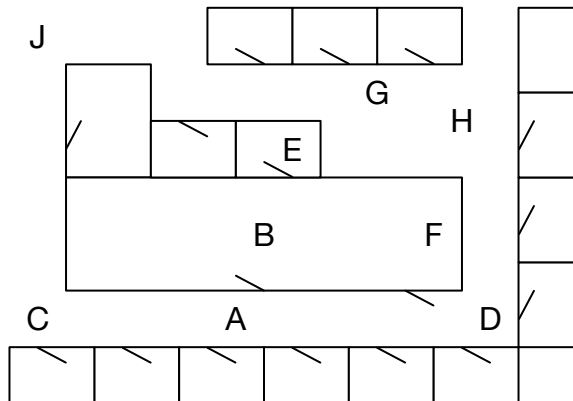
A **state** needs to include enough information to

- determine what is the next state
- determine whether the goal is achieved
- determine the cost.

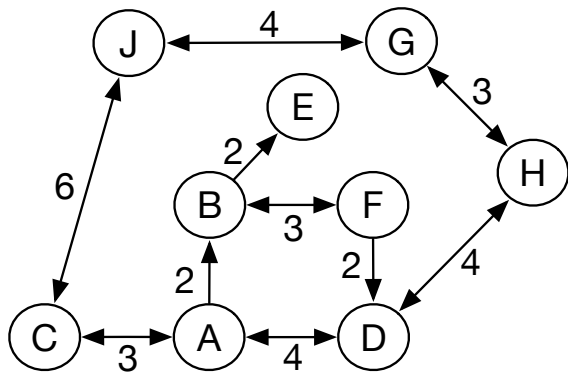
Often there are many options for what to include in the state.
Keep the states as simple as possible but no simpler.

Example Problem for Delivery Robot

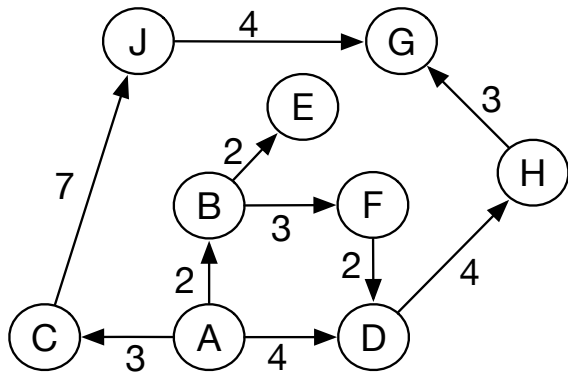
The robot is at A and the goal is to get to G:



State-Space Graph for the Delivery Robot



State-Space Graph for the Delivery Robot (Acyclic)



Example: Google Maps

Example: Google Maps

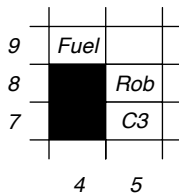
- single start location and goal location
- cost is estimated time
- state

Example: Google Maps

- single start location and goal location
- cost is estimated time
- state needs to include direction because the cost depends on directions (e.g., turning left).

Partial Search Space for a Video Game

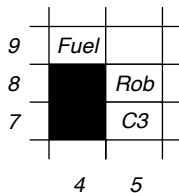
Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1 , C_2 , C_3 , C_4 , without running out of fuel, and end up at location (1, 1):



Partial Search Space for a Video Game

Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1 , C_2 , C_3 , C_4 , without running out of fuel, and end up at location (1, 1):

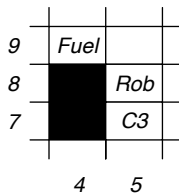
State:



Partial Search Space for a Video Game

Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1 , C_2 , C_3 , C_4 , without running out of fuel, and end up at location (1, 1):

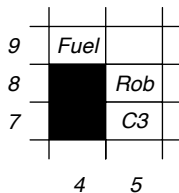
State: $\langle X\text{-pos}, Y\text{-pos},$



Partial Search Space for a Video Game

Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1 , C_2 , C_3 , C_4 , without running out of fuel, and end up at location (1, 1):

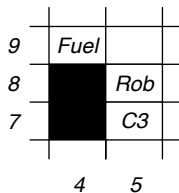
State: $\langle X\text{-pos}, Y\text{-pos}, \text{Fuel} \rangle$,



Partial Search Space for a Video Game

Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1, C_2, C_3, C_4 , without running out of fuel, and end up at location $(1, 1)$:

State: $\langle X\text{-pos}, Y\text{-pos}, \text{Fuel}, C_1, C_2, C_3, C_4 \rangle$

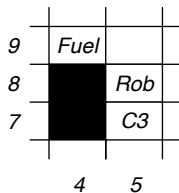


Partial Search Space for a Video Game

Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1, C_2, C_3, C_4 , without running out of fuel, and end up at location (1, 1):

State: $\langle X\text{-pos}, Y\text{-pos}, \text{Fuel}, C_1, C_2, C_3, C_4 \rangle$

Goal:

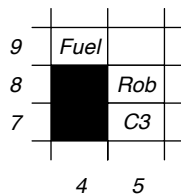


Partial Search Space for a Video Game

Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1, C_2, C_3, C_4 , without running out of fuel, and end up at location (1, 1):

State: $\langle X\text{-pos}, Y\text{-pos}, \text{Fuel}, C_1, C_2, C_3, C_4 \rangle$

Goal: $\langle 1, 1, ?, t, t, t, t \rangle$

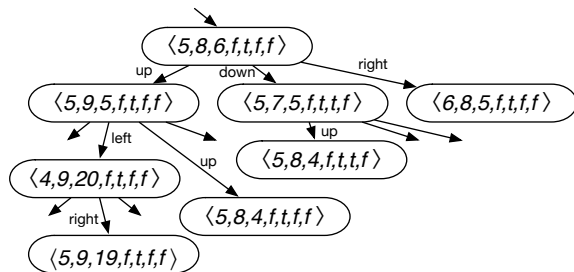
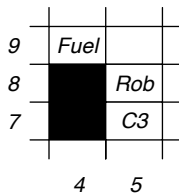


Partial Search Space for a Video Game

Grid game: Rob is on a grid and can move up, down, left or right and needs to collect coins C_1, C_2, C_3, C_4 , without running out of fuel, and end up at location $(1, 1)$:

State: $\langle X\text{-pos}, Y\text{-pos}, \text{Fuel}, C_1, C_2, C_3, C_4 \rangle$

Goal: $\langle 1, 1, ?, t, t, t, t \rangle$



- 2 rooms, one cleaning robot
- rooms can be clean or dirty
- robot actions:
 - suck: makes the room that the robot is in clean
 - move: move to other room
- Goal: have both rooms clean

- 2 rooms, one cleaning robot
- rooms can be clean or dirty
- robot actions:
 - suck: makes the room that the robot is in clean
 - move: move to other room
- Goal: have both rooms clean
- How many states are there? What are they?