AI Search

Chao Lan
Outline

Introduction

Uninformed Search vs. Informed Search
Introduction

Uninformed Search vs. Informed Search
Introduction

What is search?

What are the basic concepts in search?

What are the applications of search?

Find solution by tree search.

What is the data structure of search algorithm?

How to evaluate search algorithm?
What is search?

Search is about an agent
“finding a sequence of actions to achieve its goal when no single action can do”.

We will assume a solution exists and is a fixed sequence of known actions.
Example: 8-Puzzle

Start State

Goal State
Example: Vacuum Cleaner
Example: Route-Finding
Introduction

What is search?

What are the basic concepts in search?

What are the applications of search?

Find solution by tree search.

What is the data structure of search algorithm?

How to evaluate search algorithm?
state space: set of all states reachable from the initial state by any sequence of actions
Concepts

state space: set of all states reachable from the initial state by

initial state: state where agent starts
Concepts

state space: set of all states reachable from the initial state by any sequence of actions

initial state: state where agent starts

applicable action: an action that can be taken from the current state
Concepts

state space: set of all states reachable from the initial state by any sequence of actions

initial state: state where agent starts

applicable action: an action that can be taken from the current state

transition model: a function mapping (current state, action) to a new state.
Concepts

*state space*: set of all states reachable from the initial state by any sequence of actions.

*initial state*: state where agent starts

*applicable action*: an action that can be taken from the current state

*transition model*: a function mapping (current state, action) to a new state.

*successor*: a state reachable from a given state by a single action.
Concepts (Cont.)

graph representation of state space (node, link, cost)
Concepts (Cont.)

graph representation of state space (node, link, cost)

path: a sequence of states connected by a sequence of actions
Concepts (Cont.)

graph representation of state space (node, link, cost)

path: a sequence of states connected by a sequence of actions

step cost: cost of taking an action in one state to reach another state
Concepts (Cont.)

graph representation of state space \text{(node, link, cost)}

path: a sequence of states connected by a sequence of actions

step cost: cost of taking an action in one state to reach another state

path cost: sum of step costs in the path
Concepts (Cont.)

graph representation of state space (node, link, cost)

path: a sequence of states connected by a sequence of actions

step cost: cost of taking an action in one state to reach another state

path cost: sum of step costs in the path

goal test: determine whether a given state is a goal state
Concepts (Cont.)

graph representation of state space (node, link, cost)

path: a sequence of states connected by a sequence of actions

step cost: cost of taking an action in one state to reach another state

path cost: sum of step costs in the path

goal test: determine whether a given state is a goal state

solution: a sequence of actions that leads from the initial state to a goal state
Concepts (Cont.)

graph representation of state space (node, link, cost)

path: a sequence of states connected by a sequence of actions

step cost: cost of taking an action in one state to reach another state

path cost: sum of step costs in the path

goal test: determine whether a given state is a goal state

solution: a sequence of actions that leads from the initial state to a goal state

optimal solution: a solution with the lowest path cost
**Abstraction** is the process of removing detail from a representation in problem formulation.
Introduction

What is search?

What are the basic concepts in search?

What are the applications of search?

Find solution by tree search.

What is the data structure of search algorithm?

How to evaluate search algorithm?
Route-Finding Problem
Touring Problem

Visit each city at least once, and travel back to Arad.
Touring Problem

Visit each city at least once, and travel back to Arad.

Traveling Salesperson Problem

Visit each city exactly once, and travel back to Arad, with the *shortest* path.
VLSI Layout Problem

Position millions of components and connections on a chip to minimize area, circuit delays, stray capacitances, and maximize manufacturing yield.
Robot Navigation

Agent can move in a continuous space with an infinite set of possible actions and states.

(Route-finding problem focuses on discrete space with a finite set of possible actions and states.)
Automatic Assembly Sequencing

Find an order to assemble different parts of some object.
Introduction

What is search?

What are the basic concepts in search?

What are the applications of search?

Find solution by tree search.

What is the data structure of search algorithm?

How to evaluate search algorithm?
Find Solution by Search Tree

Searching for a solution is equivalent to searching in a tree (or, generating a tree).
Find Solution by Search Tree

(a) The initial state
Find Solution by Search Tree

(b) After expanding Arad
Find Solution by Search Tree

(c) After expanding Sibiu
Concept: Frontier

A frontier is a list of nodes to be expanded, e.g. \{Sibiu, Timisoara, Zerind\}.
Avoid repeated state and loopy path by maintaining a closed set.
Introduction

What is search?

What are the basic concepts in search?

What are the applications of search?

Find solution by tree search.

What is the data structure of search algorithm?

How to evaluate search algorithm?
Several concepts in this section are from data structure (COSC3020)...
Concept Review

Node
Edge
Parent Node
Child Node
Root Node
Leaf Node
Data Structure of a Node

A node contains four components

1. **state**: state of current node
2. **parent node**: state of parent node
3. **action**: action applied to the parent to generate the node
4. **path-cost**: cost of path from initial state to the node
Example

The circled node has

1. State =
2. parent node =
3. action =
4. path-cost =
Example

The circled node has

1. State = Sibiu
2. parent node = Arad
3. action = turn east
4. path-cost = 140
Example 2

The circled node has

1. State =

2. parent node =

3. action =

4. path-cost =
Example 2

The circled node has

1. State = Fagaras
2. parent node = Sibiu
3. action = turn east
4. path-cost = 239
Data Structure to Store Nodes

Nodes are stored in a queue, e.g., $Q = (\text{Arad}, \text{Sibiu}, \text{Fagaras})$. 
Nodes are stored in a queue, e.g., Q = (Arad, Sibiu, Fagaras).

Three queue operations

- **empty**: return true if there is no node in the queue
- **pop**: return the first node in the queue, then remove it from the queue
- **insert**: insert a node to the queue, then return the resulting queue

Q: what is empty(Q)?
Data Structure to Store Nodes

Nodes are stored in a queue, e.g., Q = (Arad, Sibiu, Fagaras).

Three queue operations

- **empty**: return true if there is no node in the queue
- **pop**: return the first node in the queue, then remove it from the queue
- **insert**: insert a node to the queue, then return the resulting queue

Q: what is empty(Q)? False.

Q: what is pop(Q)?
Data Structure to Store Nodes

Nodes are stored in a queue, e.g., Q = (Arad, Sibiu, Fagaras).

Three queue operations

- **empty**: return true if there is no node in the queue
- **pop**: return the first node in the queue, then remove it from the queue
- **insert**: insert a node to the queue, then return the resulting queue

Q: what is empty(Q)? False.
Q: what is pop(Q)? Arad.
Q: what is insert(Zerind)?
Nodes are stored in a queue, e.g., \( Q = (\text{Arad}, \text{Sibiu}, \text{Fagaras}) \).

**Three queue operations**

- **empty**: return true if there is no node in the queue
- **pop**: return the first node in the queue, then remove it from the queue
- **insert**: insert a node to the queue, then return the resulting queue

Q: what is empty(Q)? False.
Q: what is pop(Q)? Arad.
Q: what is insert(Zerind)? \( Q = (\text{Arad}, \text{Sibiu}, \text{Fagaras}, \text{Zerind}) \)
Nodes are stored in a queue, e.g., (Arad, Sibiu, Fagaras).

Three queue operations

- empty: return true if there is no node in the queue
- pop: return the first node in the queue, then remove it from the queue
- insert: insert a node to the queue, then return the resulting queue

Three types of queue

- **FIFO** (First-In-First-Out): pop the oldest node in the queue
- **LIFO** (Last-In-First-Out): pop the newest node in the queue
- **Priority**: pop node with the highest priority in the queue
Data Structure to Store Nodes

Nodes are stored in a queue, e.g., (Arad, Sibiu, Fagaras).

Q: what is pop(Q) if FIFO?
Q: what is pop(Q) if LIFO?
Q: what about pop(Q) if priority?

Three types of queue

- **FIFO** (First-In-First-Out): pop the oldest node in the queue
- **LIFO** (Last-In-First-Out): pop the newest node in the queue
- **Priority**: pop node with the highest priority in the queue
Introduction

What is search?

What are the basic concepts in search?

What are the applications of search?

Find solution by tree search.

What is the data structure of search algorithm?

How to evaluate search algorithm?
Evaluate Search Algorithm

Four metrics to evaluate a search algorithm

1. **Completeness**: is the algorithm guaranteed to find a solution (if it exists)?

2. **Optimality**: does the algorithm find the optimal solution?

3. **Time Complexity**: how long does it take to find a solution?

4. **Space Complexity**: how much memory is needed to perform search?
Example

Is the following search algorithm complete?

Search Algorithm 1
- always turn left
Example

Is the following search algorithm complete?

Search Algorithm 1
- always turn left
(not complete)
Complexities of Search Algorithm

Complexities of a search algorithm are often measure indirectly, e.g.,

- branching factor (# successors), depth, maximum length
- \# nodes |V| + \# edges |E|
- time = \# generated nodes
- space = \# stored nodes