Outline

1. Problem Solving Agents
2. Components of Well-Defined Problem
3. Problem Formulation
4. Example Problems
5. Basic search algorithms
Outline

1. Problem Solving Agents
2. Components of Well-Defined Problem
3. Problem Formulation
4. Example Problems
5. Basic search algorithms
In this chapter we consider a type of Goal-Based Agent, known as Problem Solving Agent that uses atomic representation.

Atomic Representation: Consist of indivisible states, i.e., that has no internal structure.

Example (Atomic Representation)
Finding the route from one part of the country to the other through some sequence of cities. For simplicity, only city names can represent the states of the world i.e, single atom or black box.

Obviously, we can have goal-based agents with more advanced state representations.
**Goal Formulation:** is the first step in problem solving and it is based on the current situation and the agent’s performance measure. It helps to simplify the decision problem. Goal is normally considered to be set of states in which the Goal is satisfied.

**Problem Formulation:** is the process of deciding what actions and states to consider, given a goal.

- It is not necessary that in the initial stage the agent action on the current state (transition) results in achieving the goal.
- Then generally an agent with several immediate options of unknown value can decide what to do by first examining future actions that eventually lead to states of known value.
Problem Solving

**Search-Solution-Execute**

**Search** is the process of looking for a sequence of actions that reaches the goal. A search algorithm takes a problem as input and returns a **solution** in the form of an action sequence. Once a solution is found, the actions it recommends can be carried out. This is called the **execution**.
function Simple-Problem-Solving-Agent(\textit{percept}) \textbf{returns} an action 

\textbf{static}: seq, an action sequence, initially empty
\textbf{state}, some description of the current world state
\textbf{goal}, a goal, initially null
\textbf{problem}, a problem formulation

\textit{state} $\leftarrow$ Update-State(\textit{state, percept})

\textbf{if} seq is empty \textbf{then}
\textbf{goal} $\leftarrow$ Formulate-Goal(\textit{state})
\textbf{problem} $\leftarrow$ Formulate-Problem(\textit{state, goal})
\textit{seq} $\leftarrow$ Search(\textit{problem})
\textbf{action} $\leftarrow$ Recommendation(\textit{seq, state})
\textit{seq} $\leftarrow$ Remainder(\textit{seq, state})

\textbf{return} action
On holiday in Romania; currently in Arad.
Non-Refundable ticket of flight that leaves tomorrow from Bucharest.
**Formulate goal**: be in Bucharest
**Formulate problem**: states: various cities, actions: drive between cities.
**Find solution**: sequence of cities, e.g., Arad, Sibiu, Fagaras, Bucharest
An Example (cont.)
Assumptions about the Environment

- Observable
- Discrete
- Deterministic

Open-Loop Systems

These assumptions result in a fixed sequence of actions. Therefore, when executing this solution sequence, the agent will ignore its percept while deciding the actions. Hence called **Open-Loop Systems** since the relationship between agent and environment through percept is no-more a loop.
Outline

1. Problem Solving Agents
2. Components of Well-Defined Problem
3. Problem Formulation
4. Example Problems
5. Basic search algorithms
Components of a Problem

- Initial State
- Actions
- Transition Model
- Goal Test
- Path Cost
Components of a Problem

**Definition (Initial State)**

As name suggest, the initial state agent starts in. For example, in tourist example, the initial state of our agent is in Arad, i.e., IN(Arad).

**Definition (Actions)**

Description of possible actions given a particular state \( s \), i.e., \( ACTIONS(s) \) will return all possible actions executable in \( s \) and these set of actions will be called applicable in \( s \). For example, \{Go(Sibiu), Go(Timisoara), Go(Zerind)\} are applicable in state Arad.
Components of a Problem

**Definition (Transition Model)**

Describes each action \(a\) with respect to state \(s\). i.e., \(\text{RESULT}(s,a)\) will return the state that result by doing action \(a\) in state \(s\). For example, \(\text{RESULT}(\text{IN}(\text{Arad}), \text{Go}(\text{Zerind})) = \text{IN}(\text{Zerind})\)

**Definition (State Space)**

The initial state, set of actions and the transition model forms a state space. It is the set of all possible states reachable from initial state by any sequence of action. State space forms a graph or directed network, where nodes are states and links represent the action. A *Path* in the state space describes the sequence of states connected by sequence of actions.
Componenets of a Problem

Definition (Goal Test)
That determines the given state is goal or not. It is not always as simple as in our example, i.e., to be in Bucharest, for example, in a chess game, the goal state is “Checkmate”.

Definition (Path Cost)
A function that assigns cost to each path. The cost function in problem solving agents is their performance measure. For example, in our tourist example, the cost is the distance between states that reflects time. The path cost, in simple case, can be described by the sum of costs of actions along the path. We denote it by $C(s,a,s')$, where $s$ is the current state, $a$ is the action and $s'$ is the resulting state.
Solution is set of actions that leads from initial state to goal state, there may be many solution for a problem solving agent. The quality of this solution is measured by the path cost and the solution with minimum path cost is called Optimal Solution.
Outline

1. Problem Solving Agents
2. Components of Well-Defined Problem
3. Problem Formulation
4. Example Problems
5. Basic search algorithms
Problem Formulation

- All five components, initial state, actions, transition model, goal test and path cost defined above forms a problem formulation.
- This formulation is abstract, i.e., details are hidden.
- Abstraction is useful since they simplify the problem by hiding many details but still covering the most important information about states and actions (retaining the state space in simple form), therefore abstraction needs to be valid.
- Abstraction is called valid when the abstract solution can be expanded to more detailed world.
- Abstraction is useful if the actions in the solution are easier than the original problem, i.e, no further planning and searching.
- Construction of useful and valid abstraction is challenging.
Outline

1. Problem Solving Agents
2. Components of Well-Defined Problem
3. Problem Formulation
4. Example Problems
5. Basic search algorithms
Types of Problems

Toy Problems: Intended to illustrate or exercise various problem-solving methods. It can be given a concise, exact description and hence is usable by different researchers to compare the performance of algorithms.

Real World Problems: has importance and whose solutions people actually care about. Such problems tend not to have a single agreed-upon description, but we can give the general flavor of their formulations.
Toy Problem: 8-Puzzle

- Consist of $3 \times 3$ board with eight numbered tiles and a blank space.
- A tile adjacent to the blank space can slide into the space.
- The objective is to reach a specified goal state, such as the one shown in the figure.

Start State

<table>
<thead>
<tr>
<th>7</th>
<th>2</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

Goal State

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>
8-Puzzle: Problem Formulation

**States:** A state description specifies the location of each of the eight tiles and the blank in one of the nine squares.

**Initial state:** Any state can be designated as the initial state. Note that any given goal can be reached from exactly half of the possible initial states.

**Actions:** The simplest formulation defines the actions as movements of the blank space Left, Right, Up, or Down.

**Transition model:** Given a state and action, this returns the resulting state; for example, if we apply Left to the start state in Figure above, the resulting state has the 5 and the blank switched.

**Goal test:** The goal configuration shown in Figure above (Other goal configurations are possible.)

**Path cost:** Each step costs 1, so the path cost is the number of steps in the path.
Description: 8 Puzzle

- Belongs to the family sliding-block puzzles.
- Often used as a test problem for new AI algorithms.
- Has 181440 reachable states.
- Best search algorithm may take few milliseconds to solve from any random initial state.
- Other similar example includes: 15 puzzle (4 × 4 board), and 24 puzzle (5 × 5 board) with trillions of states.

Readings

Read from AI book or other source about example problems such as 8 queen, traveling salesperson problem (TSP) and robot navigation.
Outline

1. Problem Solving Agents
2. Components of Well-Defined Problem
3. Problem Formulation
4. Example Problems
5. Basic search algorithms
Search Algorithms

- Search plays important role in solving AI problems.
- Normally, the result of a search algorithm is a solution in the form of a set of actions that lead to the goal. (But not always)
- For any problem, once the problem is properly formulated and initial state is identified, search algorithm is applied to identify best possible sequence of state-action to achieve the goal.