# INTRODUCTION TO AI

## LECTURE 3

# Problem Formulation

**Dr. Tamal Ghosh**
**Computer Science & Engineering**
**Adamas University**

# PROBLEM-SOLVING AGENT

- A problem-solving agent is a goal-based agent
- Intelligent agents are supposed to maximize their performance measures. Achieving this is sometimes simplified if the agent can adopt a goal and aim at satisfying it.
- A problem-solving agent has three phases:
    - problem formulation, searching for solutions, and executing actions in the solution.
- **Problem formulation** is the process of deciding what actions and states to consider, given a goal.
- The process of looking for a sequence of actions that reaches the goal is called **searching for solutions**. A search algorithm takes a problem as input and returns a solution in the form of an action sequence.
- The **execution of this action** sequence produces the solution.

# PROBLEM FORMULATION STEPS

- A problem can be defined by five components:
  - initial state, actions, transition model, goal test, path cost.
- INITIAL STATE: The initial state that the agent starts in.
- ACTIONS: A description of the possible actions available to the agent.

  Given a particular state s, ACTIONS(s) returns the set of actions that can be executed in s. Each of these actions is applicable in s.
- TRANSITION MODEL: A description of what each action does is known as the transition model
  - A function RESULT(s, a) that returns the state that results from doing action a in state s.
  - The term successor refers to any state reachable from a given state by a single link
  - The state space of the problem is the set of all states reachable from the initial state by any sequence of actions.
  - The state space forms a graph in which the nodes are states and the links between nodes are actions.
  - A path in the state space is a sequence of states connected by a sequence of actions.

# CONTD..

- GOAL TEST: The goal test determines whether a given state is a goal state.
- PATH COST: A path cost is a function that assigns a numeric cost to each path.
  - The problem-solving agent chooses a cost function that reflects its own performance measure.
  - The cost of a path can be described as the sum of the costs of the individual actions along the path.
  - The step cost of taking action a in state s to reach state s′ is denoted by c(s, a, s′).
- A SOLUTION to a problem is an action sequence that leads from the initial state to a goal state.
  - Solution quality is measured by the path cost function, and an OPTIMAL SOLUTION has the lowest path cost among all solutions.

# PROBLEM EXAMPLE: TRAVEL IN ROMANIA

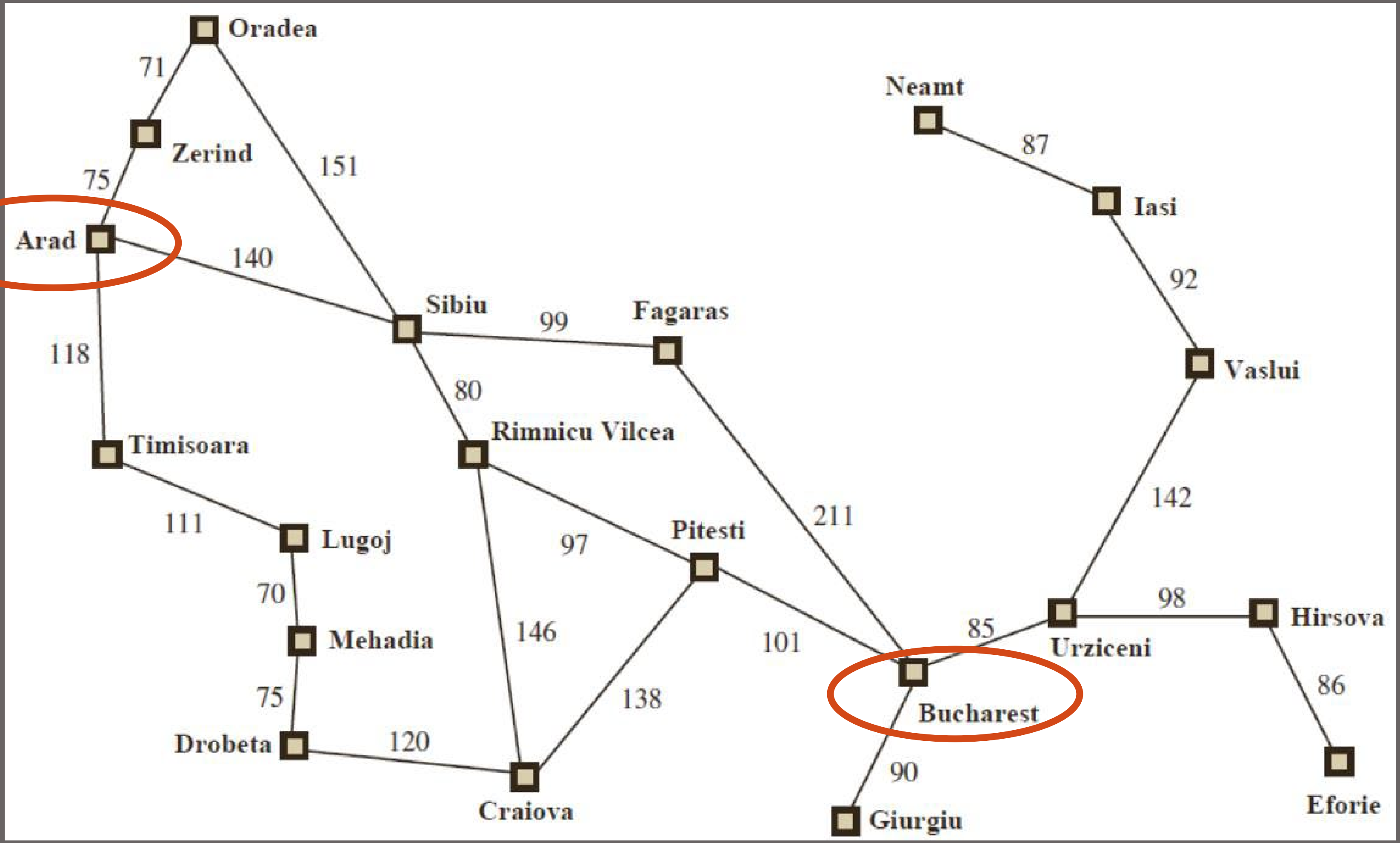
- On holiday in Romania; currently in Arad.
- Flight leaves tomorrow from Bucharest

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

# CONTD…



Oradea
71
Zerind
151
75
Arad
140
Neamt
87
Iasi
92
Sibiu
99
Fagaras
118
Vaslui
80
Rimnicu Vilcea
Timisoara
142
211
111
Lugoj
97
Pitesti
70
98
Hirsova
Mehadia
146
101
85
Urziceni
86
75
138
Bucharest
Drobeta
120
90
Craiova
Giurgiu
Eforie

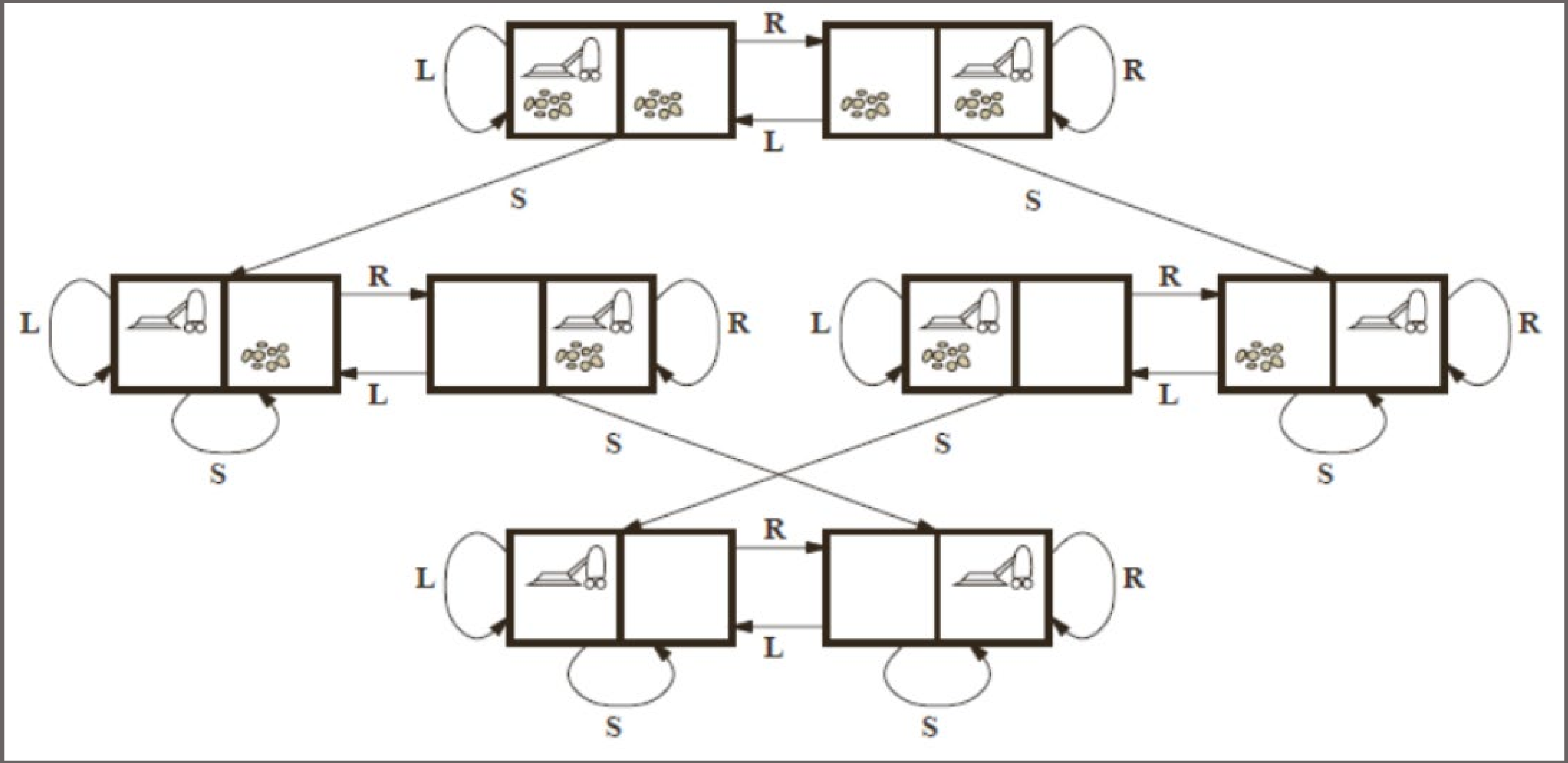
# CONTD…

- Travelling in Romania problem can be defined by:
- **initial state:** in Arad
- **actions: transition model:** is the graph
- **successor function** S(x) = set of action, state pairs
- e.g., S(Arad) = { <Arad →Zerind, Zerind>, <Arad → Sibiu, Sibiu>, < Arad → Timisoara, Timisoara> }
- **goal test:** in Bucharest
- **path cost:** sum of distances, number of actions executed, etc.
- c(x; a; y) is the **step cost**, assumed to be ≥0
- **A solution is a sequence of actions leading from the initial state to a goal state**

# VACUUM CLEANING PROBLEM

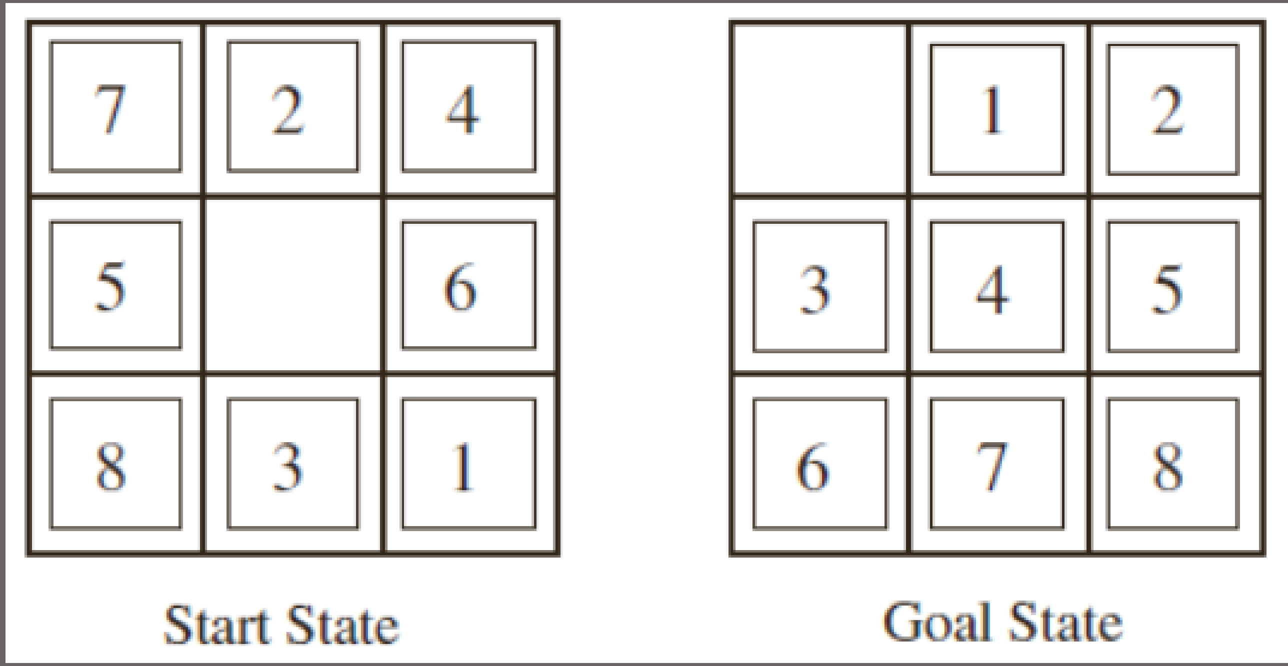
- **States:**
  - The state is determined by both the agent location and the dirt locations.
  - The agent is in one of two locations, each of which might or might not contain dirt.
  - Thus, there are $2 \times 2^2 = 8$ possible world states.
- **Initial state:** Any state can be designated as the initial state.
- **Actions:** Each state has just three actions: Left, Right, and Suck.
- **Transition model:**
  - The actions have their expected effects, except that moving Left in the leftmost square, moving Right in the rightmost square, and Sucking in a clean square has no effect.
  - The transition model defines a state space.
- **Goal test:** This checks whether all the squares are clean.
- **Path cost:** Each step costs 1, so the path cost is the number of steps in the path.

# CONTD…



# 8 PUZZLE PROBLEM

- The 8-puzzle consists of a 3×3 board with eight numbered tiles and a blank space.
- A tile adjacent to the blank space can slide into the space.
- The object is to reach a specified goal state.



# CONTD…

- **States:** A state 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 the goal can be reached from exactly half of the possible initial states.
- **Actions:** Movements of the blank space *Left*, *Right*, *Up*, or *Down.*
- •Different subsets of these are possible depending on where the blank is.
- **Transition model:** Given a state and action, this returns the resulting state;
- **Goal test:** This checks whether the state matches the goal configuration
- **Path Cost:** Each step costs 1, so the path cost is the number of steps in the path.

# CONTD…

- The 8-puzzle belongs to the family of sliding-block puzzles,
  - This family is known to be NP-complete.
  - The optimal solution of the n-Puzzle family is NP-hard. i.e. NO polynomial solution for the problem.
- •The 8-puzzle has 9!/2=181440 reachable states.
- •The 15-puzzle (on a 4×4 board) has around 1.3 trillion states,
- •The 24-puzzle (on a 5 ×5 board) has around $10^{25}$ states

# REAL-WORLD PROBLEM

- Route-finding problem is defined in terms of specified locations and transitions along links between them.
- Route-finding algorithms are used in a variety of applications such as Web sites and in-car systems that provide driving directions.
- VLSI layout problem requires positioning millions of components and connections on a chip to minimize area, minimize circuit delays, minimize stray capacitances, and maximize manufacturing yield.