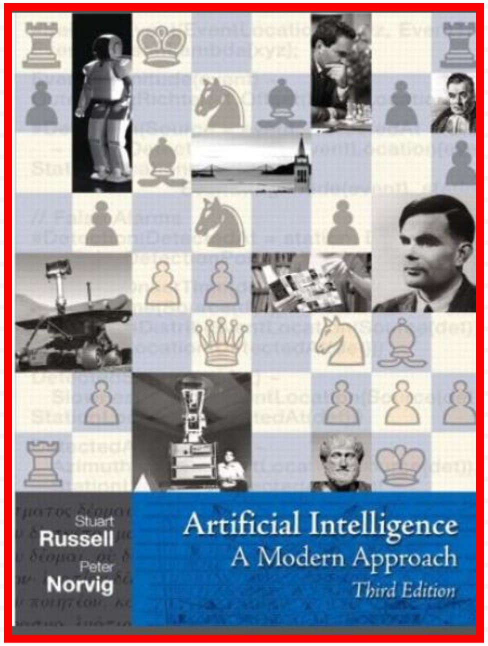
Princess Nora University
Faculty of Computer & Information Systems

ARTIFICIAL INTELLIGENCE
(CS 370D)

Computer Science Department

جامعة الأميرة نورة بنت عبدالرحمن
Princess Nora Bint Abdul Rahman University



(CHAPTER-3-PART1)

PROBLEM SOLVING AND SEARCH

Dr. Abeer Mahmoud
(Course coordinator)

WHY SEARCH?

- Search : Finding a good/best solution to a problem amongst many possible solutions.

- Many AI problems can be posed as search
- If goal found=>success; else, failure

- Not just city route search
  - Many AI problems can be posed as search
- Game-playing:
  Sequence of moves to win a game.
- Speech Recognition
  Sequence of moves to recognize the speech

O Shortest path on a map.

CHAPTER OUTLINE

- Problem-solving agents
- Problem types, formulation & Examples
- Basic search algorithms

1. **Uninformed search** algorithms (**blind search**)
   - (these algorithms are given no information about the problem other than its definition)
2. **Informed search** algorithms (**heuristic search**)
   - (these algorithms have some idea of where to look for solutions and whether one non goal state is more promising than another in reaching goal)

Problem-solving agents

Dr. Abeer Mahmoud
(course coordinator)

- The simplest agent (reflex agent) which base their actions on direct mapping from states to actions

- Disadv: such agent cannot operate well in environments for which this **mapping would be too large**

- But Goal based agents can achieve successes by considering **future** actions desirability of their outcomes

  - One kind of goal based agent **called** **problem solving agent**

- **Problem solving agents**: decide what to do by finding sequences of actions that lead to desirable states

Problem types, formulation & Examples

How problem is solved?

Step 1	Goal formulation
Step 2.	Problem formulation – a process of deciding what actions and states to consider
Step 3	Search – systematic exploration of the sequence of alternative states that appear in a problem solving process
Step 4	Solution – reach the right action
Step 5	Execution – recommended actions can be accomplished

1. Define the problem and its solution

Formulate	**Search**	**Executes**

1. Define the problem and its solution

Formulate

- Agent task is to find out which sequence of actions will get to a goal state
- Hence, before it can do this , it needs to decide what sorts of **actions** & **states** to consider

1. Define the problem and its solution

Formulate

- Ex , **if** agent will consider details **"move left foot forward an inch "** or **" turn the steering wheel one degree left"**, **then** the agent will probably never find a way out……why?

- Because at this level of details there are too many steps to find solution

Formulate =The process of deciding actions and states to consider

Note: The type of problem formulation can have a serious influence on the difficulty of finding a solution.

1. Define the problem and its solution

Search

- Ex , **if** agent at a specific city **"Riyad"** and **" want to go Madenah",** **and there are three paths to achieve the goal** **then** which to select ? May be random?

- If agent has a map (additional knowledge) , finding the best choice= Search

Search Algorithm =takes problem as input and returns a solution in the form of an action sequence

1. Define the problem and its solution

Search

Requirements of a good search strategy:

1. It causes motion

   Otherwise, it will never lead to a solution.

2. It is systematic

   Otherwise, it may use more steps than necessary.

3. It is efficient

   Find a good, but not necessarily the best, answer.

1. Define the problem and its solution

Executes

- Once a solution is found the action it recommends can be carried out

14

1. Define the problem and its solution

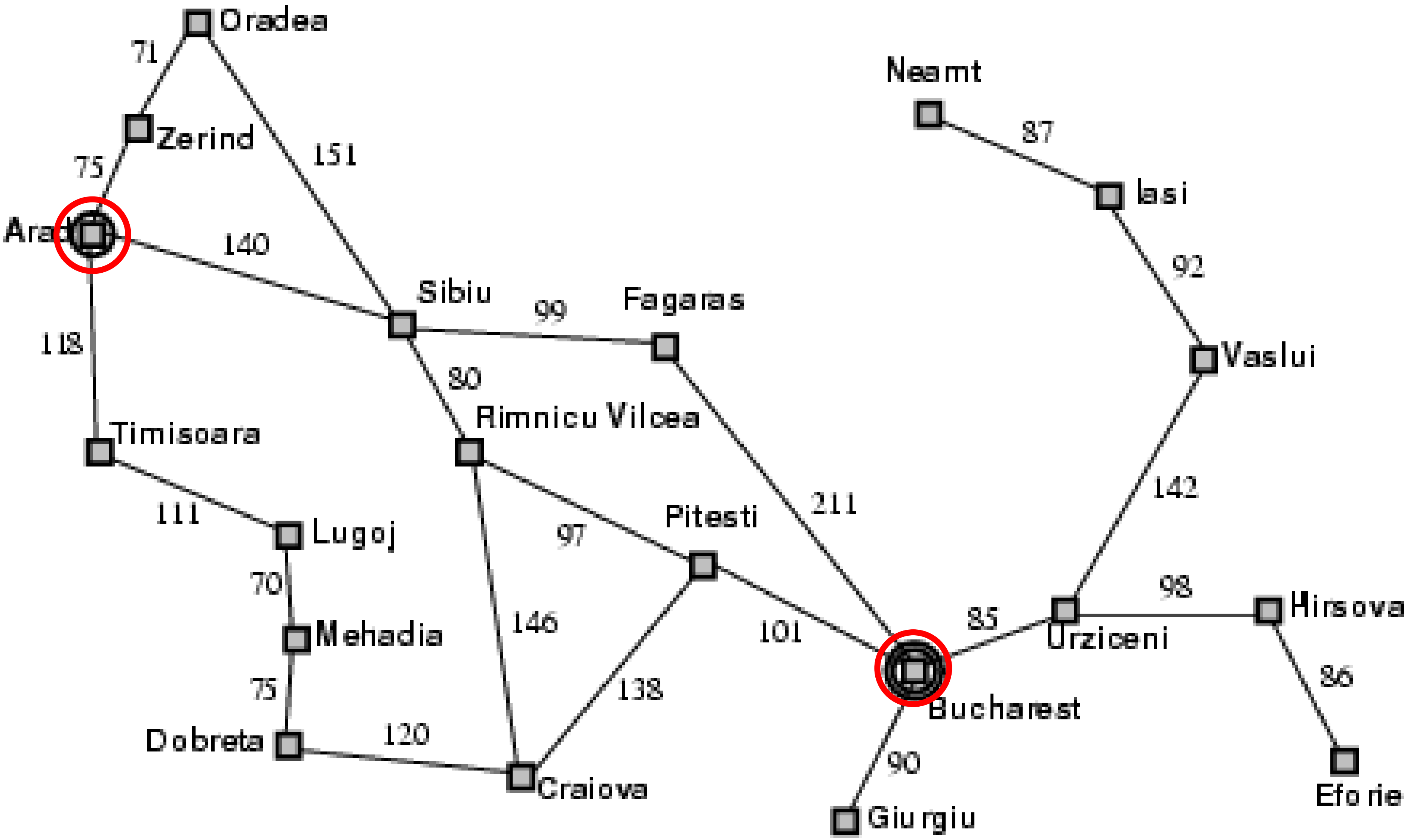
Initial state	Operator	Neighbourhood (Successor Function)	State Space	Goal test	Path cost
The **initial state** of the problem, defined in some suitable manner	**A set of actions** that moves the problem from one state to another	The set of all possible states reachable from a given state	The set of all states reachable from the initial state	A test applied to a state which returns if we have reached a state that solves the problem	How much it costs to take a particular path

Dr. Abeer Mahmoud
(course coordinator)

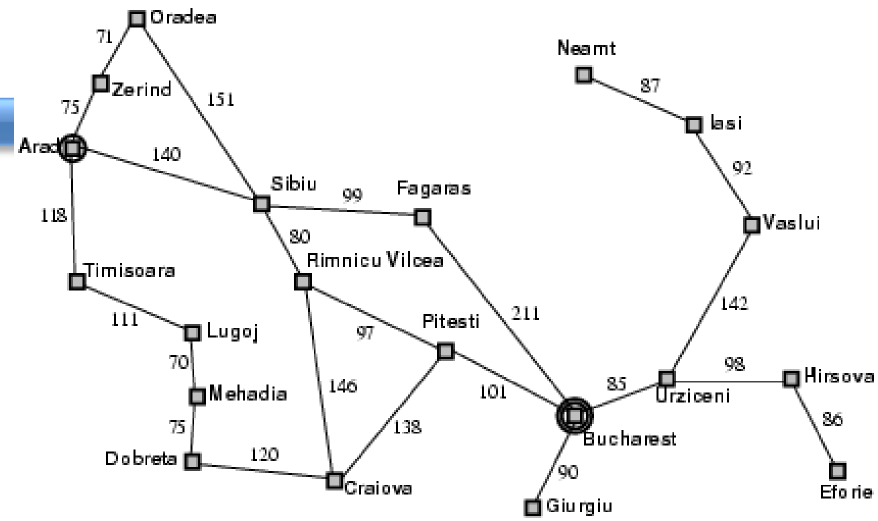
Examples

Dr. Abeer Mahmoud
(course coordinator)

Example: Traveling in Romania



State-space Problem Formulation



A **problem** is defined by four items:

1. **initial state** e.g., "at Arad"
2. **actions** or **successor function**

   $S(x)$ = set of action–state pairs
   e.g., $S(Arad) = \{<Arad \rightarrow Zerind, Zerind>, \ldots\}$

3. **goal test** (or set of goal states)

   e.g., x = "at Bucharest", $Checkmate(x)$

4. **path cost** (additive)

   e.g., 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

Dr. Abeer Mahmoud
(course coordinator)

Problem Ex: The 8-puzzle

7	2	4
5		6
8	3	1

Initial state

	1	2
3	4	5
6	7	8

Goal state

- states?
- operators?
- goal test?
- path cost?

Dr. Abeer Mahmoud
(course coordinator)

Problem Ex: The 8-puzzle

7	2	4
5		6
8	3	1

Initial state

	1	2
3	4	5
6	7	8

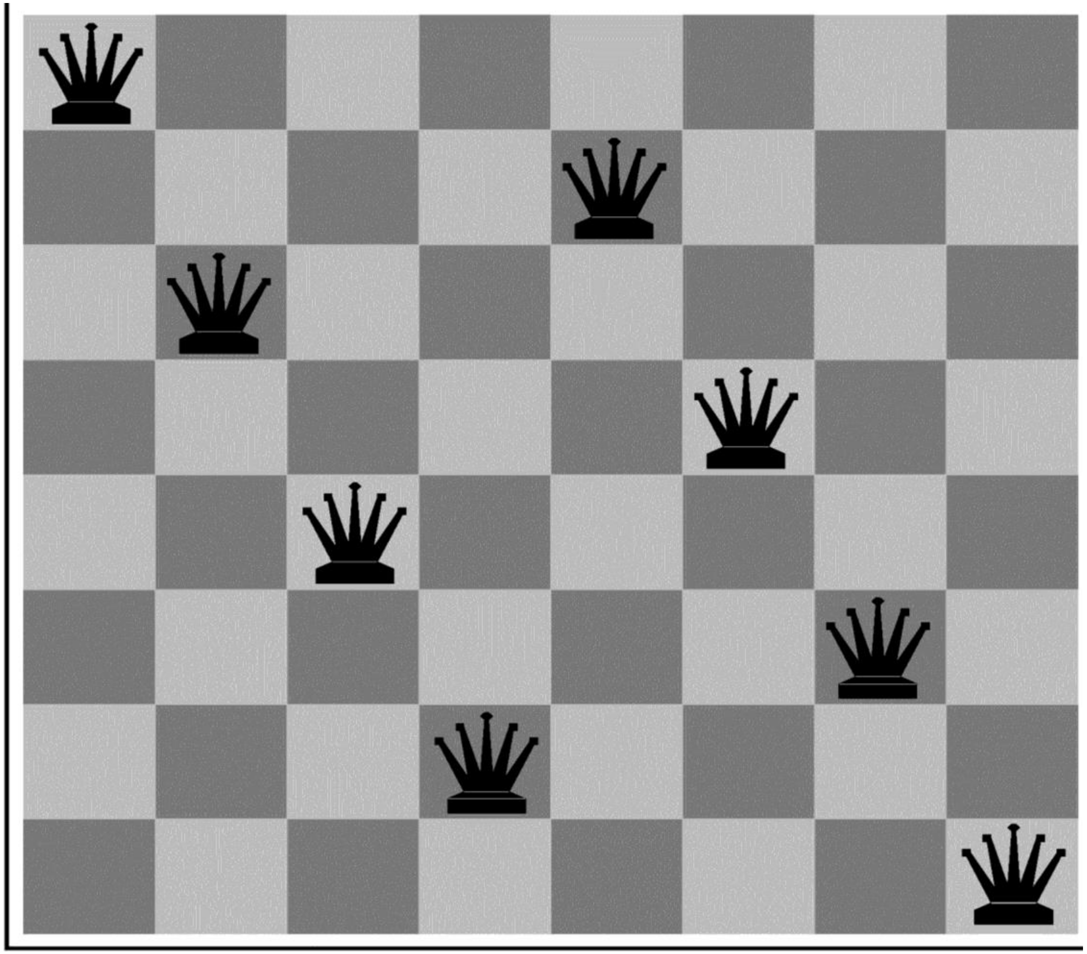
Goal state

- **states?** locations of tiles
- **operators?** move blank left, right, up, down
- **goal test?** = goal state (given)
- **path cost?** 1 per move

Dr. Abeer Mahmoud
(course coordinator)

20

Problem Ex: The 8-queens problem



Dr. Abeer Mahmoud
(course coordinator)

Problem Ex: The 8-queens problem

- **states?** -any arrangement of n<=8 queens

  -such that no queen attacks any other.[not on same row or same column or diagonal]

- **initial state?** no queens on the board

- **actions?** -add queen to any empty square

  -*or* add queen to leftmost empty square such that it is not attacked by other queens.

- **goal test?** 8 queens on the board, none attacked.

- **path cost?** 1 per move

Dr. Abeer Mahmoud
(course coordinator)

State Space Search: Water Jug Problem

- “You are given two jugs, a 4-litre one and a 3-litre one.
- Neither has any measuring markers on it.
- There is a pump that can be used to fill the jugs with water.
- How can you get exactly 2 litres of water into 4-litre jug.”

Dr. Abeer Mahmoud
(course coordinator)

A

B

24

4-litre

A
empty

3-litre

B
empty

4-litre

A
4
3
2
1

3-litre

B
empty

Dr. Abeer Mahmoud
(course coordinator)

4-litre 3-litre

A
4
3
2
1

Fill B from A

B
empty

Dr. Abeer Mahmoud
(course coordinator)

4-litre | 3-litre

A
1

Fill B from A

B
3
2
1

Dr. Abeer Mahmoud
(course coordinator)

4-litre

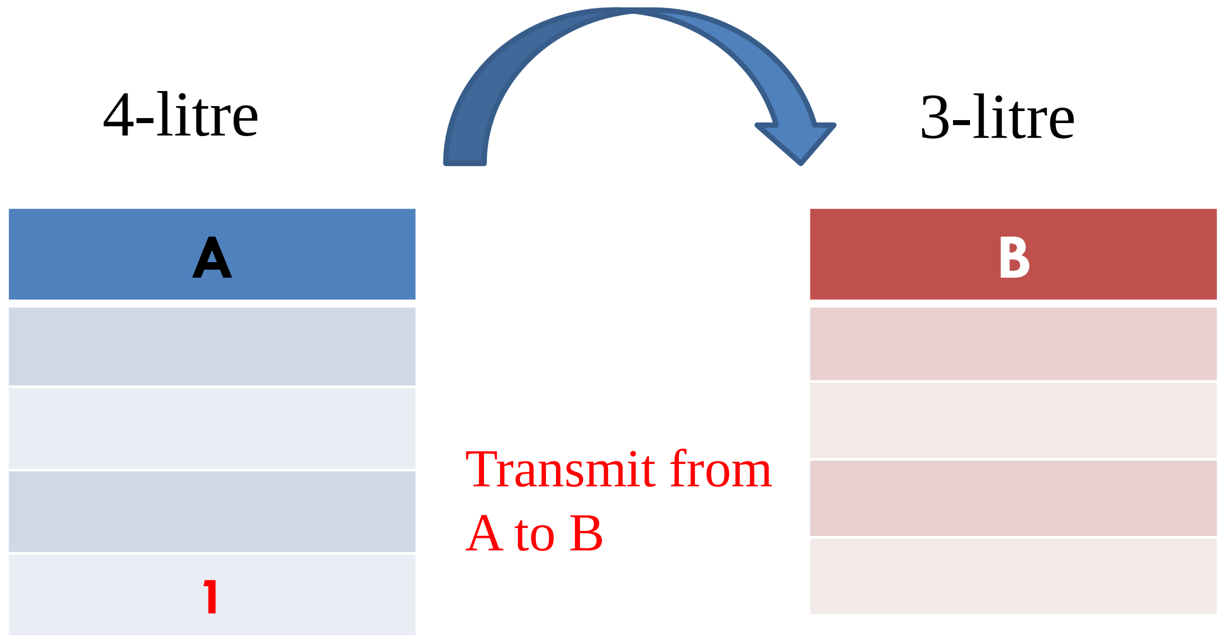
A
1

empty B

3-litre

B

Dr. Abeer Mahmoud
(course coordinator)



Dr. Abeer Mahmoud
(course coordinator)

4-litre

A

Fill A again

3-litre

B
1

31

4-litre

A
4
3
2
1

Fill A again

3-litre

B
1

32

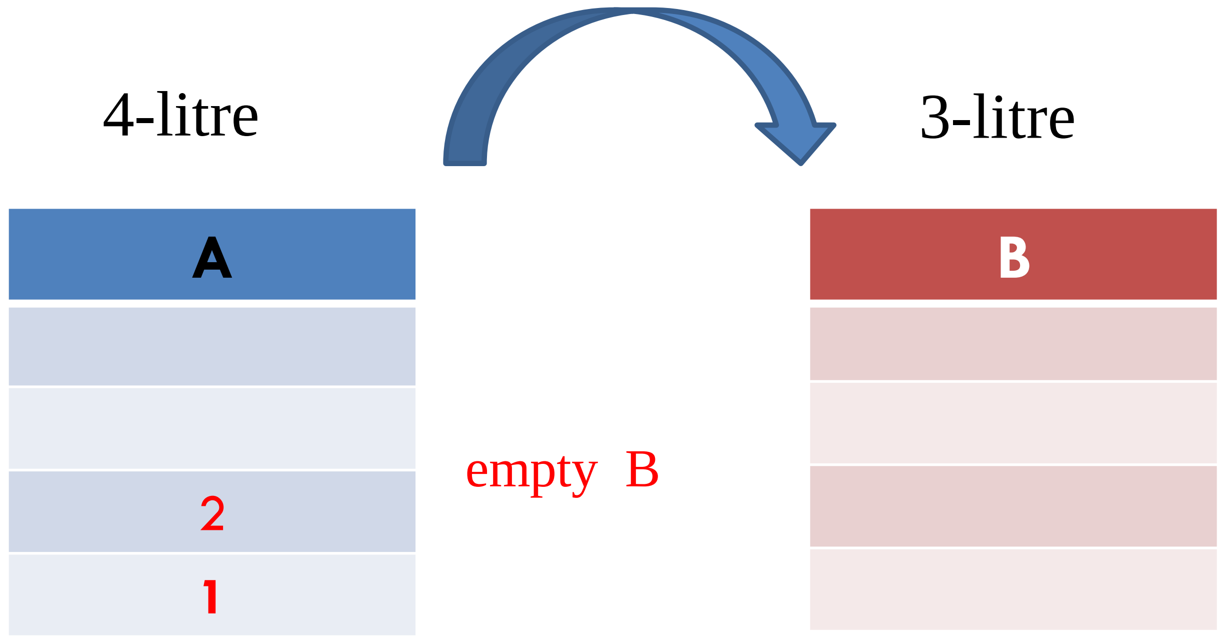
4-litre

A
2
1

Transmit from A to B

3-litre

B
3
2
1



goal

Water Jug Problem: A State Space Search

- **State space:**
  - set of ordered pairs of integers (x, y) such as
  - x = 0,1,2,3, or 4 for amount of water in 4-gallon jug,
  - y = 0, 1, 2, or 3 for amount of water in the 3-gallon jug.
- **The start state :** (0,0).
- **The goal state :** is (2,n) for any value of n.

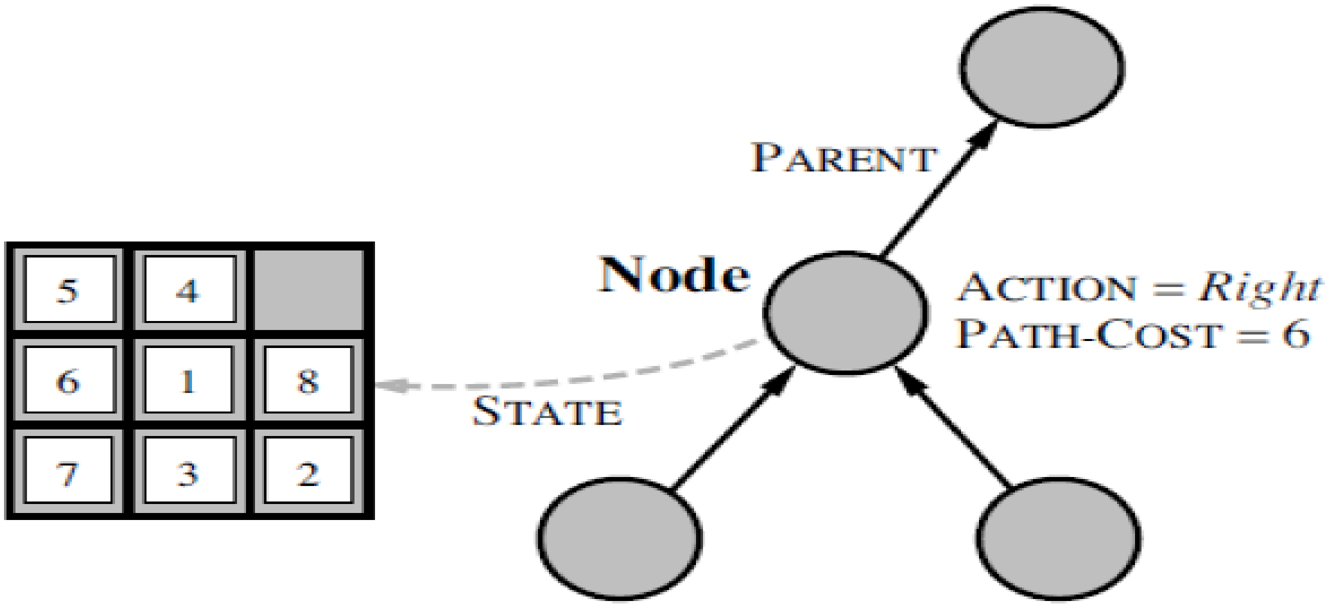
After formulating the problem ,

a **<u>search</u>** through the states is needed to find a solution

Dr. Abeer Mahmoud
(course coordinator)

- One of searching techniques is search **tree or graph**

A state is a (representation of) a physical configuration.
A node is a data structure constituting part of a search tree includes : state, parent node, action, path cost g(x).



- Many ways to represent **node** ,
- **ex** : data structure with 5 components

Implementing a Search-What we need to store

state	Parent node	action	Path cost	depth
The state in state space which the node corresponds	The node in search tree that generated this node	The action that was applied to parent to generate the node	Cost from initial state to the node	The number of steps along the path from the initial states

Dr. Abeer Mahmoud
(course coordinator)

State-Space Search Algorithm

- Search process constructs a "Search tree"
- Root is the start node (initial state).
- Leaf nodes are: unexpanded nodes (in the nodes list).
- "dead ends" (nodes that aren't goals and have no successors).
- Solution desired may be:
  - just the goal state.
  - a path from start to goal state .
- The search tree is the explicit *tree* generated during the search by the search strategy.

- The search space is the implicit *tree* (OR *graph*) defined by initial state and the operators.

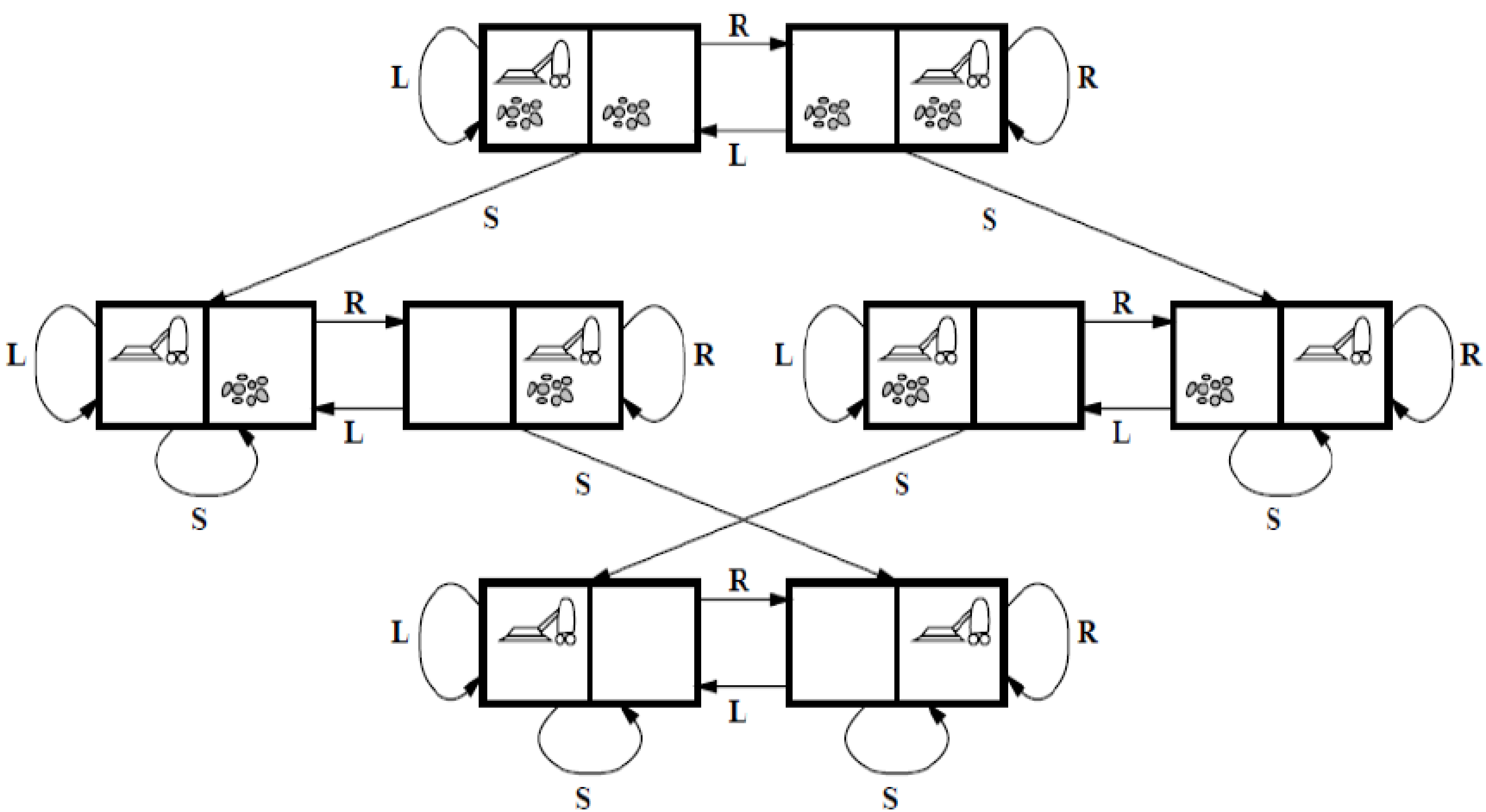
Tree Search Algorithms

Basic idea:

- offline, simulated exploration of state space by generating successors of already-explored states (expanding states).

```
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
```

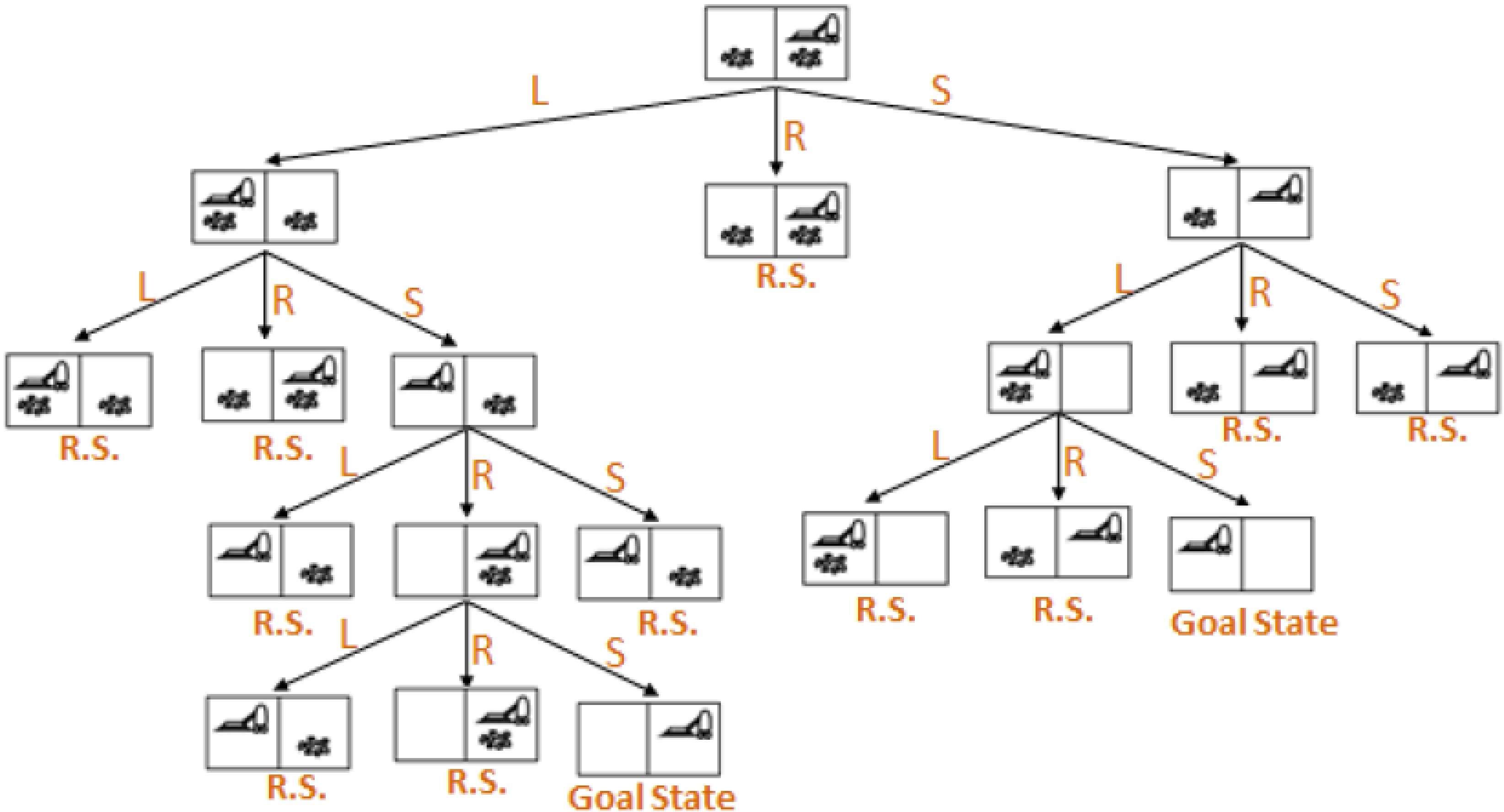
Search Space of Vacuum World Problem: **Graph**



Dr. Abeer Mahmoud
(course coordinator)

Search Space of Vacuum World Problem: **Tree**

- trace every path from the root until you reach a leaf node (goal) or a node already in that path (Repeated State or R.S.)



How Good is the found Solution?

43

Dr. Abeer Mahmoud
(course coordinator)

- **Completeness**
  - Is the strategy guaranteed to find a solution if there is a one

- **Time Complexity**
  - How long does it take to find a solution?

- **Space Complexity**
  - How much memory does it take to perform the search?

- **Optimality**
  - Does the strategy find the optimal solution where there are several solutions?

Actions in Searching a Tree

Fundamental actions (operators) that you can take:

1. "**Expand**": Ask a node for its children
2. "**Test**": Test a node for whether it is a goal

Undiscovered Nodes

- The set of nodes that have not yet been discovered as being reachable from the root

Actions in Searching a Tree (cont..)

Fringe Nodes

This is the set of nodes that (open nodes)

- have been **discovered**
- have not yet been "**processed**":
  1. have not yet expanded for the children
  2. (have not yet tested if they are a goal)

Dr. Abeer Mahmoud
(course coordinator)

Actions in Searching a Tree (cont..)

Visited Nodes

- This is the set of nodes that
  - have been discovered
  - have been processed:
    1. have discovered all their children
    2. (have tested whether are a goal)
- Also called
  - closed nodes

Dr. Abeer Mahmoud
(course coordinator)

Action on finding a Goal

- "**First match**": Usually we just want one goal, or just to know whether or not one exists
  - **on discovering a goal, then "return true"**
- "**All Matches**": Sometimes want all goals
  - **on discovering a goal, then record the fact that have found it, but continue with the search**

Dr. Abeer Mahmoud
(course coordinator)

Thank you

End of Chapter 3-part1

Dr. Abeer Mahmoud
(course coordinator)