MEASI
Institute *of*
Information Technology

Problem Solving Agents

.

Intelligent Agents and Performance Measures: Intelligent agents aim to maximize their performance by achieving certain goals.

Goal Formulation: Goals help organize behavior by limiting the objectives an agent tries to achieve.

Problem Formulation: Deciding which actions and states to consider based on the goal.

Knowledge and Search: The search algorithm takes the problem as input and returns a solution in the form of an action sequence.

Execution: Once a solution is found, the agent executes the recommended actions. This process is summarized as "formulate, search, execute."

MEASI Institute of Information Technology

Problem Solving Agents

.

Example:

Let's consider the example of the agent in Arad, Romania:

Initial Situation: The agent starts in Arad with multiple performance measures, such as enjoying the holiday and improving its suntan.

Goal Formulation: Given that the agent has a nonrefundable ticket to fly out of Bucharest the next day, the agent formulates the goal of reaching Bucharest on time.

Goal State: The goal state here is being in Bucharest before the flight departs.

Problem Solving Agents

Current Situation and Performance Measure:

The agent assesses its current situation (in Arad) and its performance measures (enjoying holiday activities, catching the flight) .

Adopting a Goal:

To ensure it catches the flight, the agent adopts the goal of reaching Bucharest. This goal overrides other performance measures as it becomes the primary objective.

Problem Formulation:

The agent decides what actions and states to consider. Instead of considering detailed actions like every turn or step, the agent simplifies the problem by considering actions at the level of driving from one town to another.

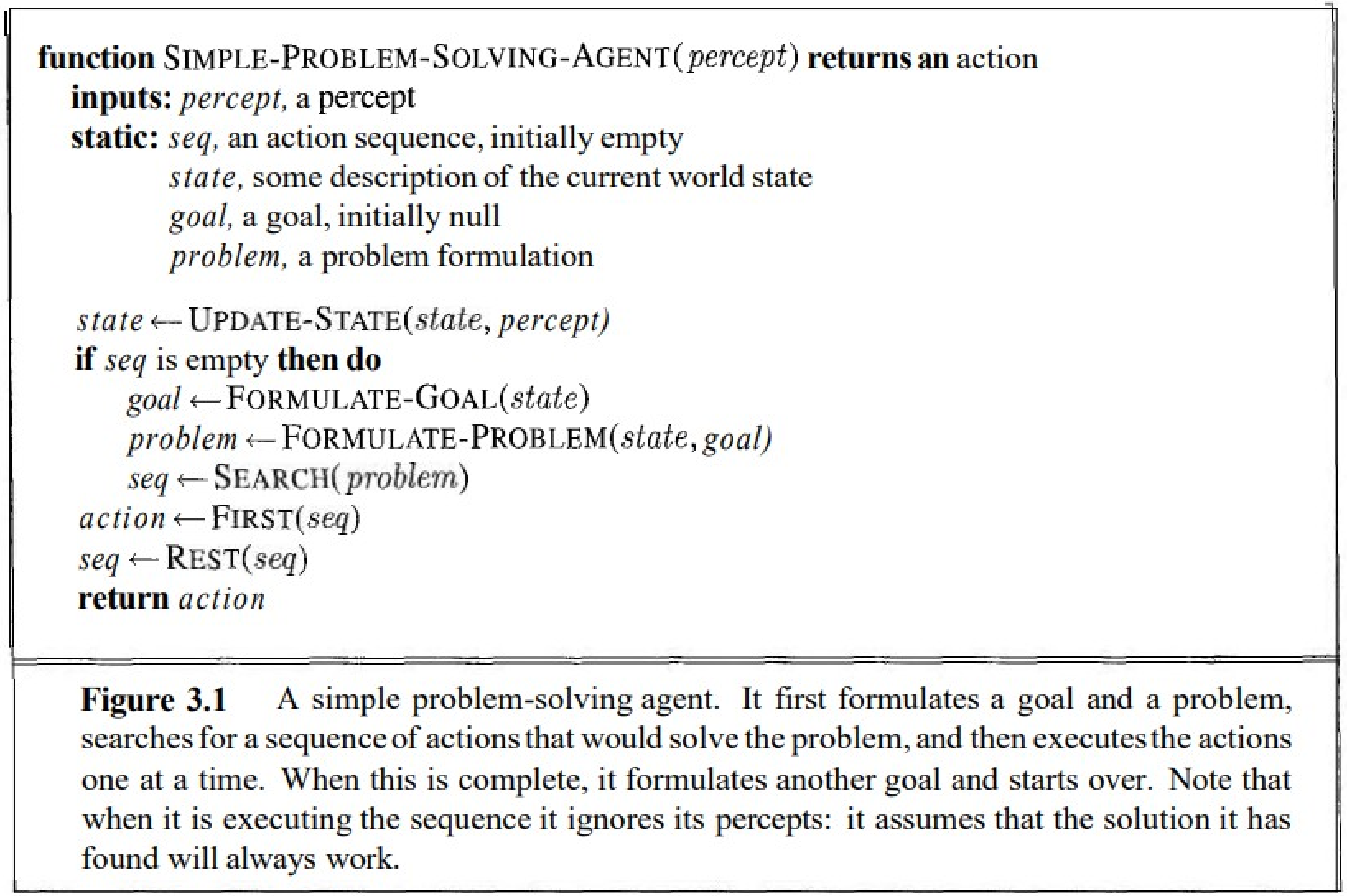
Search for Solution:

Using available information (e.g., a map), the agent searches for a sequence of towns it needs to travel through to reach Bucharest.

Execution:

Once the agent identifies the best path, it executes the actions (driving to each town in the sequence) to achieve the

Problem Solving Agents



```
function SIMPLE-PROBLEM-SOLVING-AGENT(percept) returns an action
   inputs: percept, a percept
   static: seq, an action sequence, initially empty
           state, some description of the current world state
           goal, a goal, initially null
           problem, a problem formulation

   state ← UPDATE-STATE(state, percept)
   if seq is empty then do
       goal ← FORMULATE-GOAL(state)
       problem ← FORMULATE-PROBLEM(state, goal)
       seq ← SEARCH(problem)
   action ← FIRST(seq)
   seq ← REST(seq)
   return action
```

Figure 3.1 A simple problem-solving agent. It first formulates a goal and a problem, searches for a sequence of actions that would solve the problem, and then executes the actions one at a time. When this is complete, it formulates another goal and starts over. Note that when it is executing the sequence it ignores its percepts: it assumes that the solution it has found will always work.

MEASI Institute of Information Technology

Problem Formulation

Ms.Rafath Zahra

Problem Definition: A problem is defined by four components:

Initial State: The starting point of the agent.
Successor Function: Defines the possible actions and resulting states.
Given a particular state x, SUCCESSOR-FN(x) returns a set of (action, successor) ordered pairs

$$\{\langle Go(Sibiu), In(Sibiu)\rangle, \langle Go(Timisoara), In(Tzmisoara)\rangle, \langle Go(Zerind), In(Zerind)\rangle\}$$

State Space: The combination of the initial state and successor function defines the state space, which can be visualized as a graph with nodes (states) and arcs (actions).

A **path** in the state space is a sequence of states connected by a sequence of actions.

Goal Test: Determines if a state is a goal state.
Path Cost: Measures the cost of a path from the initial state to the goal state.

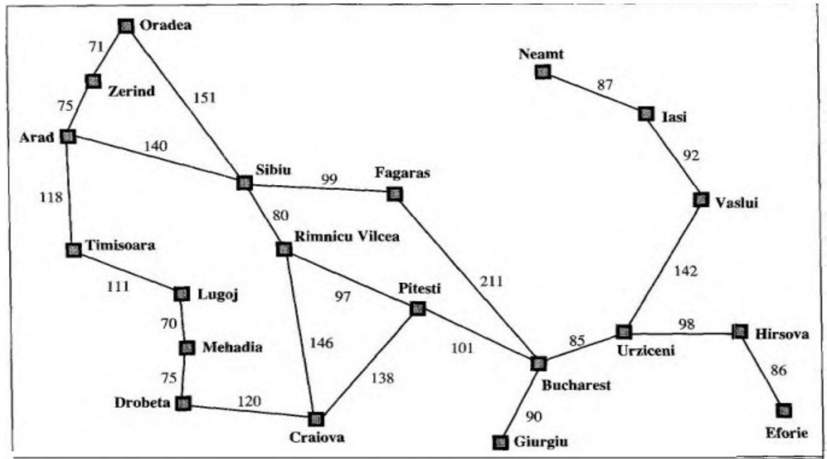


Figure 3.2 A simplified road map of part of Romania.

Problem Formulation – Vaccum World

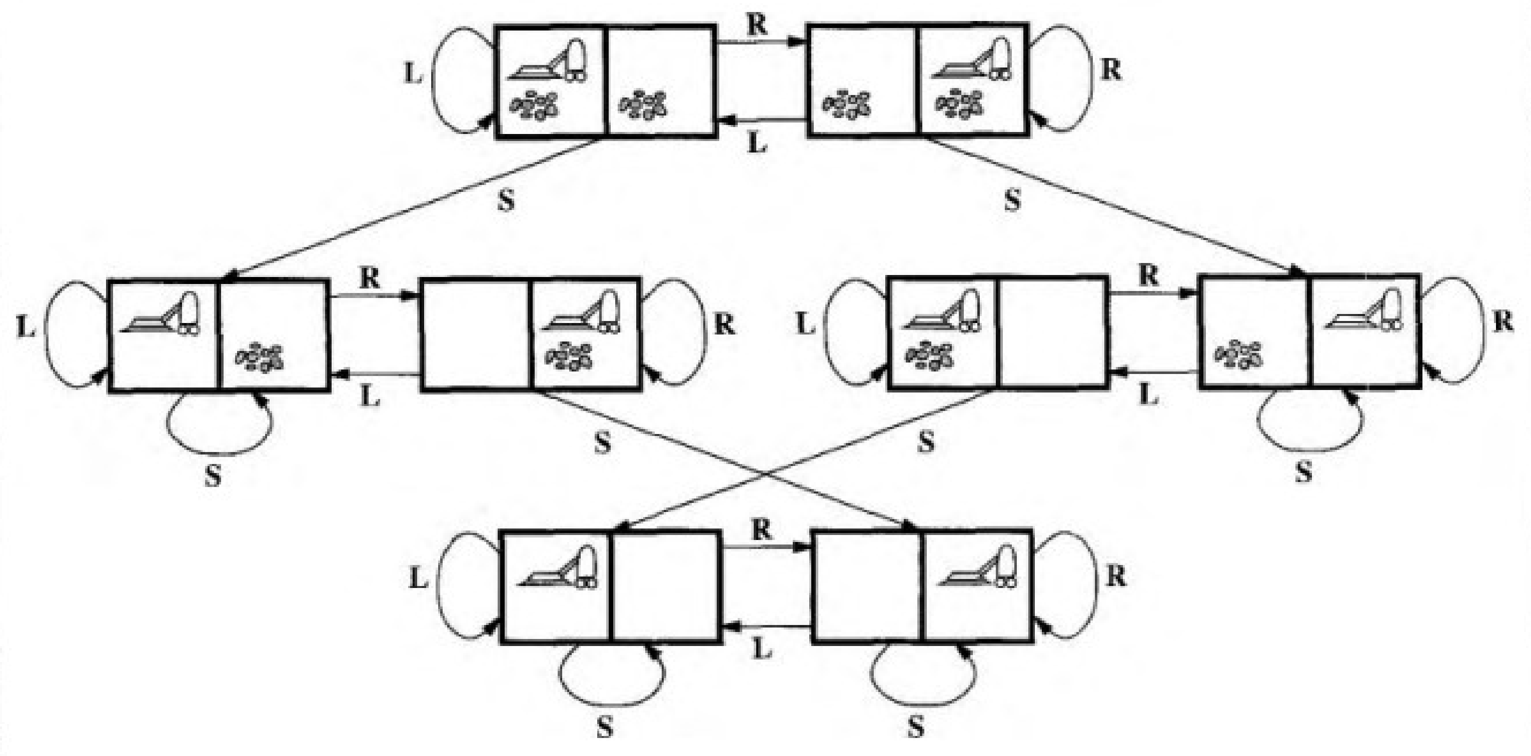


Figure 3.3 The state space for the vacuum world. Arcs denote actions: L = *Left*, R = *Right*, S = Suck.

States: The agent (vacuum) can be in one of two locations, each of which may or may not contain dirt, resulting in 8 possible states (2 locations x 2 dirt conditions per location).

Initial State: Any of these 8 states can be the starting state.

Successor Function: Generates the legal states resulting from the three actions: Left, Right, and Suck. The state space, showing all possible states and transitions, is depicted in Figure 3.3.

Goal Test: Checks if all squares (locations) are clean.

Path Cost: Each action (step) has a cost of 1, so the path cost is the number of steps taken.

A larger environment with n locations has n 2n states.

MEASI Institute *of* Information Technology

Problem Formulation

8 Puzzle Problem

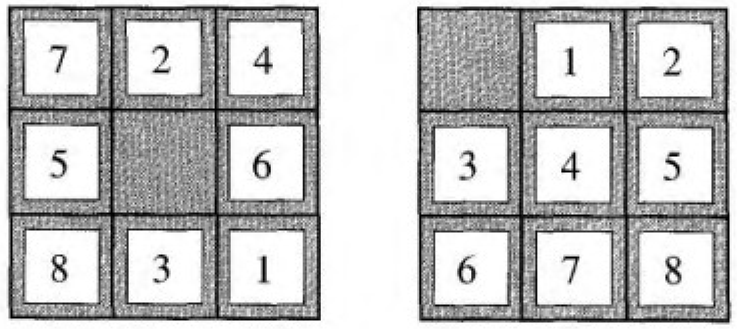
In a 3x3 grid, each cell can contain one of the following elements:
Numbered tiles, typically from 1 to 8. A blank tile, represented as an empty cell.
The arrangement of these elements in the grid defines the state of the puzzle.

1. Initial State: The initial state can be any arrangement of the tiles, which can be specified manually or generated randomly.

2.Goal State: The goal state represents the desired configuration that the puzzle should reach.

3.The state space of the 8-puzzle encompasses all possible states that the puzzle can transition through, from the initial state to the goal state.

4.There are 9 positions (8 numbered tiles and 1 blank space), so there are 9! (9 factorial) possible arrangements.
9!=362,880.



Start State Goal State

Not all these arrangements are reachable due to the constraints of tile movement (you can't reach all 362,880 states from any starting configuration).
The puzzle can be divided into two sets of states, each containing

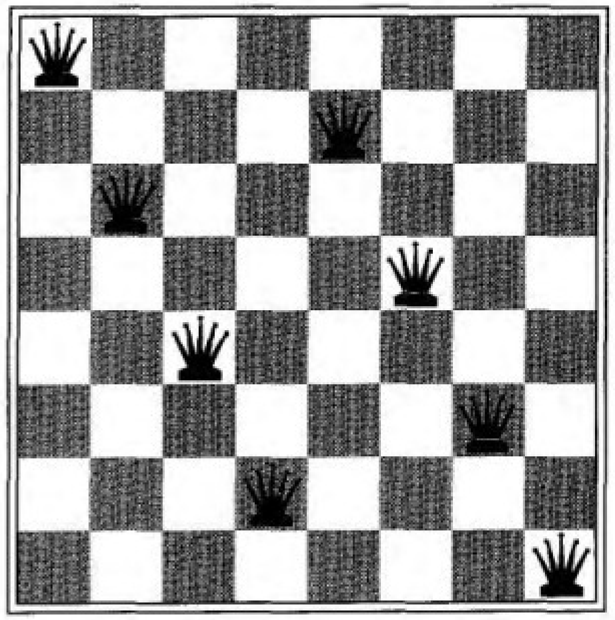
9!/2 states, that are mutually unreachable from one another

Problem Formulation

Ms.Rafath Zahra

8 Queens Puzzle

The goal of the 8-queens problem is to place eight queens on a chessboard such That no queen attacks any other. (A queen attacks any piece in the same row, column or diagonal.) Figure 3.5 shows an attempted solution that fails: the queen in the rightmost column is attacked by the queen at the top left



An incremental formulation involves operators that augment the state description, starting with an empty state; for the 8-queens problem, this means that each action adds a queen to the state. A complete-state formulation starts with all 8 queens on the board and moves them around. In either case, the path cost is of no interest because only the final state counts. The first incremental formulation one might try is the following:

States: Any arrangement of 0 to 8 queens on the board is a state.
Initial state: No queens on the board.
Successor function: Add a queen to any empty square.
Goal test: 8 queens are on the board, none attacked.

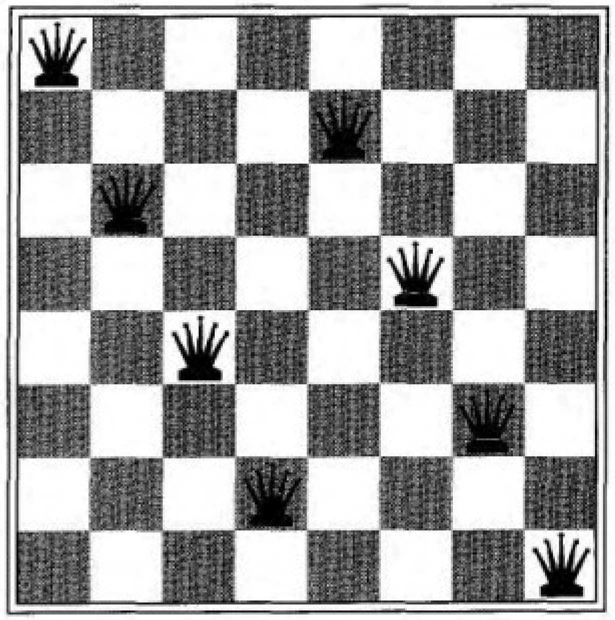
States: Arrangements of n queens (0 < n < 8), one per column in the leftmost n columns, with no queen attacking another are states.
Successor function: Add a queen to any square in the leftmost empty column such that it is not attacked b:y any other queen.

Problem Formulation

Ms.Rafath Zahra

8 Queens Puzzle

The goal of the 8-queens problem is to place eight queens on a chessboard such That no queen attacks any other. (A queen attacks any piece in the same row, column or diagonal.) Figure 3.5 shows an attempted solution that fails: the queen in the rightmost column is attacked by the queen at the top left



An incremental formulation involves operators that augment the state description, starting with an empty state; for the 8-queens problem, this means that each action adds a queen to the state. A complete-state formulation starts with all 8 queens on the board and moves them around. In either case, the path cost is of no interest because only the final state counts. The first incremental formulation one might try is the following:

States: Any arrangement of 0 to 8 queens on the board is a state.
Initial state: No queens on the board.
Successor function: Add a queen to any empty square.
Goal test: 8 queens are on the board, none attacked.

States: Arrangements of n queens ($0 < n < 8$), one per column in the leftmost n columns, with no queen attacking another are states.
Successor function: Add a queen to any square in the leftmost empty column such that it is not attacked b:y any other queen.

A* Algorithm

Ms.Rafath Zahra

The A* algorithm or A star algorithm in AI is a powerful pathfinding algorithm that efficiently finds the shortest path in a graph while considering both the actual cost incurred so far and an estimate of the remaining cost.

It combines the features of Dijkstra's algorithm and Greedy Best-First-Search to efficiently compute the shortest path.

The core of the A* algorithm is based on cost functions and heuristics. It uses two main parameters:

g(n): The actual cost from the starting node to any node n.
h(n): The heuristic estimated cost from node n to the goal

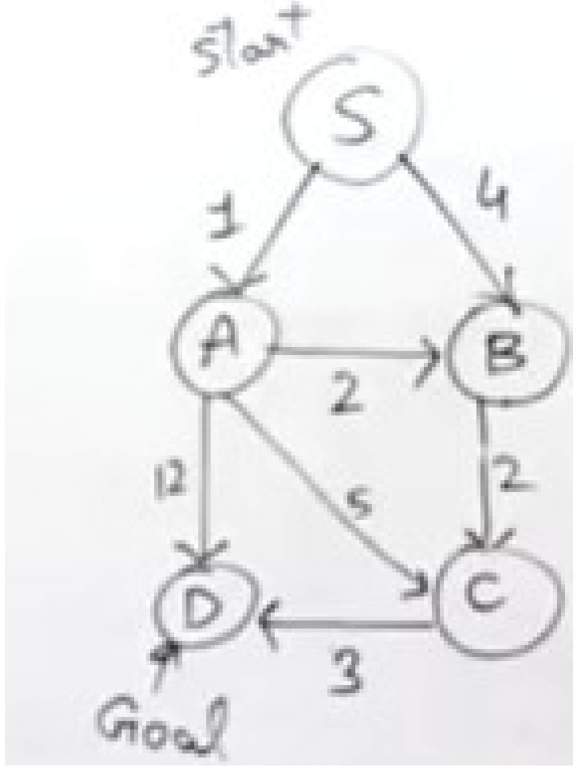
The sum, f(n)=g(n)+h(n), represents the total estimated cost of the cheapest solution

The A* algorithm's ability to find the most efficient path with a given heuristic makes it suitable for various practical applications:

Pathfinding in Games and Robotics: A* is extensively used in the gaming industry to control characters in dynamic environments, as well as in robotics for navigating between points.
Network Routing: In telecommunications, A* helps in determining the shortest routing path that data packets should take to reach the destination.
AI and Machine Learning: A* can be used in planning and decision-making algorithms, where multiple stages of decisions and movements need to be evaluated.

A* algorithm

Ms.Rafath Zahra



f(n) = g(n) + h(n)

Heuristic Value

S = 7	
A = 6	
B = 2	
C = 1	
D = 0	

S –> A
g(n) + h(n)
1 + 6 =7
f(n)=7

S –> B
g(n) + h(n)
4+2 = 6

S – >A –> B
g(n) + h(n)
3 + 2 = 5

S->A ->C
g(n) + h(n)
6+ 1 = 7

S->A->D
g(n) + h(n)
13+0 =13

S->A->B->C
g(n) + h(n)
5+1 = 6

S->A->B->C->D
g(n) + h(n)
8+0 = 8

S->A->C->D : f(n) = 9
S->B->C->D : f(n) = 9

THANK YOU

MEASI
Institute *of*
Information Technology