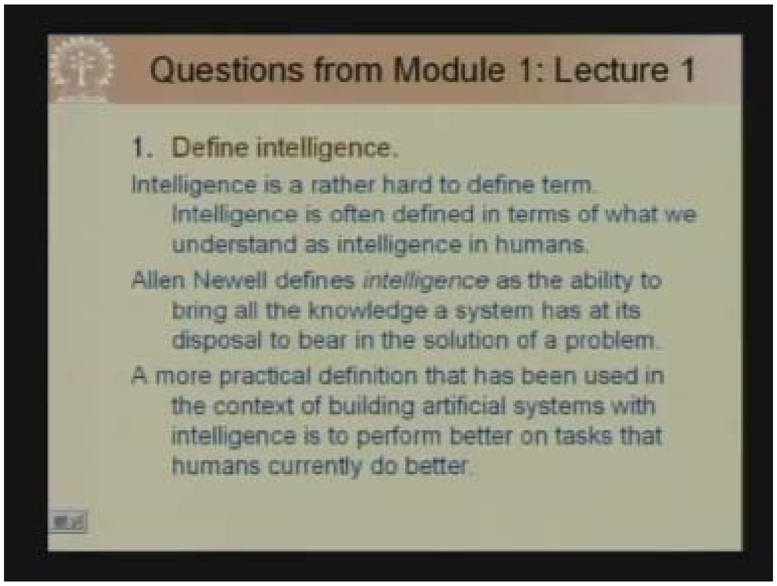
Artificial Intelligence

Prof. Sudeshna Sarkar
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 3
State Space Search

Today we will start the third lecture of our course on artificial intelligence. In the last two days we discussed about the first module which was an introduction. Today we will start on the second module and the first lecture is on State Space Search. Before I continue on to this module I would like to discuss the questions from the previous module. So let us go back to the questions of lecture 1.

(Refer Slide Time: 01:04)



Questions from Module 1: Lecture 1

1. Define intelligence.

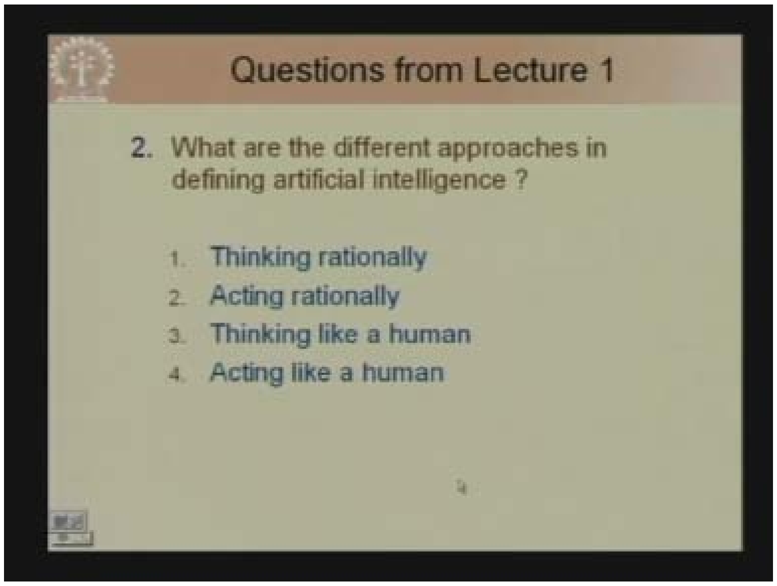
Intelligence is a rather hard to define term. Intelligence is often defined in terms of what we understand as intelligence in humans.

Allen Newell defines *intelligence* as the ability to bring all the knowledge a system has at its disposal to bear in the solution of a problem.

A more practical definition that has been used in the context of building artificial systems with intelligence is to perform better on tasks that humans currently do better.

The first question was; define intelligence. Intelligence is actually a rather hard to define term. It is often defined in terms of what we understand as intelligence in humans. Allen Newell has defined intelligence as the ability to bring all the knowledge of a system to bear on the solution of a problem. A more practical definition has been given in the context of building artificial systems. Intelligence is the ability to perform better on tasks that humans currently do better. So, that was question number 1.

(Refer Slide Time: 01:55)



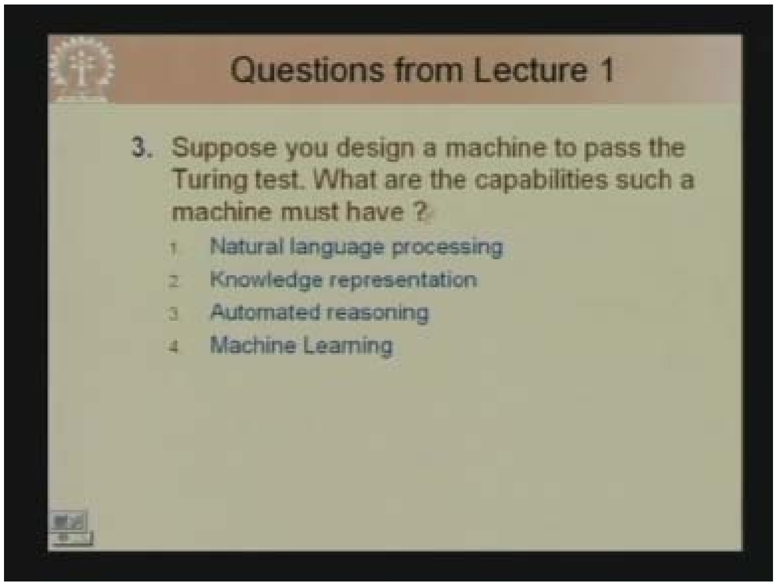
Questions from Lecture 1

2. What are the different approaches in defining artificial intelligence ?
   1. Thinking rationally
   2. Acting rationally
   3. Thinking like a human
   4. Acting like a human

Question number 2 of the first lecture was, what are the different approaches in defining AI?

- There are four approaches:
- Thinking rationally
- Acting rationally
- Thinking like a human and
- Acting like a human

(Refer Slide Time: 02:27)



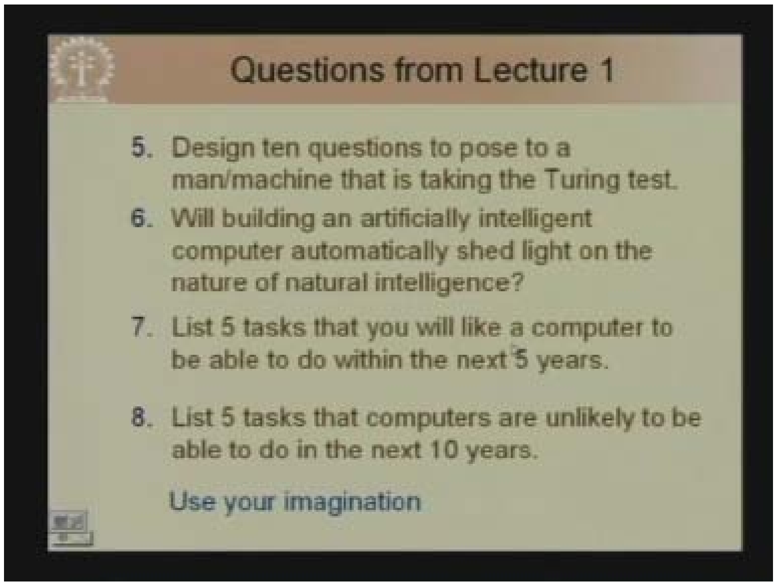
Questions from Lecture 1

3. Suppose you design a machine to pass the Turing test. What are the capabilities such a machine must have ?
   1. Natural language processing
   2. Knowledge representation
   3. Automated reasoning
   4. Machine Learning

The third question was; suppose you design a machine to pass the Turing test what are the capabilities such a machine must have?
The answer is the machine must be capable of natural language processing in order to understand the questions that the interrogator is posing to it. Secondly, the machine must have capability for knowledge representation to represent not only the prior knowledge about the world but also the facts given to the machine by the interrogator. Thirdly, the machine should be should have the capability for automated reasoning so that it can do inferences based on the facts that it knows.

Finally the machine must have capabilities of machine learning so that it can learn and adapt to new situations. Also, in order to pass the so called complete Turing test or to really act like a human the machine would also have to have expertise in computer vision as well as robotics.

(Refer Slide Time: 03:36)



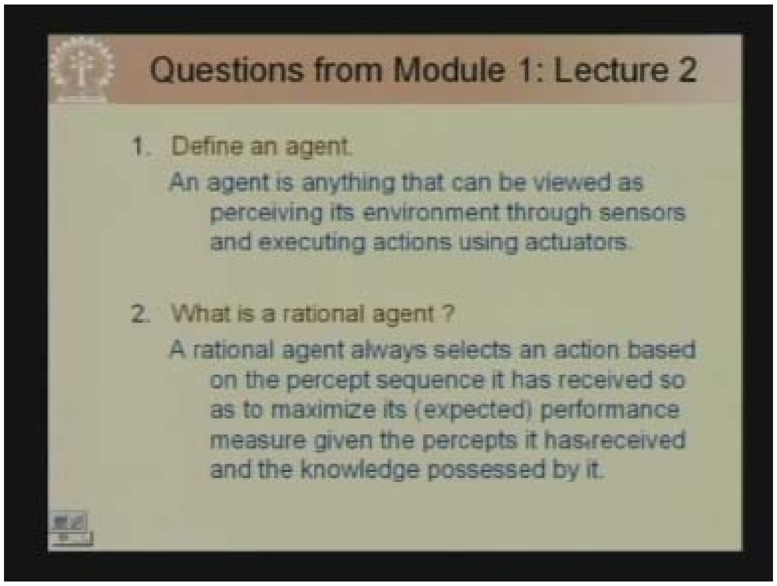
Questions from Lecture 1

5. Design ten questions to pose to a man/machine that is taking the Turing test.
6. Will building an artificially intelligent computer automatically shed light on the nature of natural intelligence?
7. List 5 tasks that you will like a computer to be able to do within the next 5 years.
8. List 5 tasks that computers are unlikely to be able to do in the next 10 years.

Use your imagination

The other four questions from the first lecture are:
Design ten questions to pose to a machine or a man taking the Turing test. It is the same for the next three questions. Will building an artificially intelligent computer automatically shed light on the nature of natural intelligence?
In a sense yes, because people who propose the thinking humanly approach to intelligence believe that if we can have a deep understanding of how humans' reason we can build machines like that and the vice versa we can gain some insights. It is the same for the next two questions.

(Refer Slide Time: 04:27)



Questions from Module 1: Lecture 2

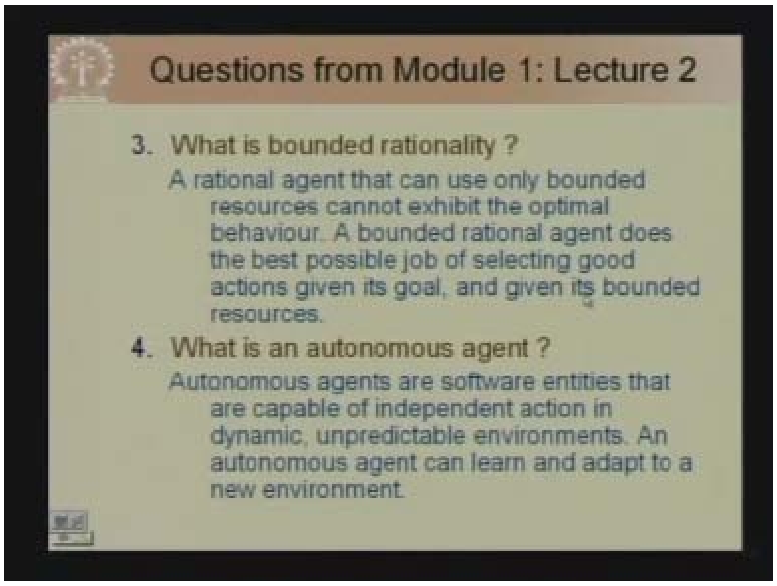
1. Define an agent.
   An agent is anything that can be viewed as perceiving its environment through sensors and executing actions using actuators.

2. What is a rational agent ?
   A rational agent always selects an action based on the percept sequence it has received so as to maximize its (expected) performance measure given the percepts it has received and the knowledge possessed by it.

Let us go to lecture 2. The first question was; define an agent. The answer is straight forward. An agent is anything that can be viewed as perceiving its environment through sensors and executing actions using actuator.

The second question was; what is a rational agent?
A rational agent is one who selects an action based on the percept sequence that it has received so far so as to maximize the expected performance measure given the percepts it has received and the knowledge possessed by it. So, within the bounds of what the agent knows and what are the percepts that it can receive the agent should behave so as to maximize its expected performance measure.

(Refer Slide Time: 05:36)



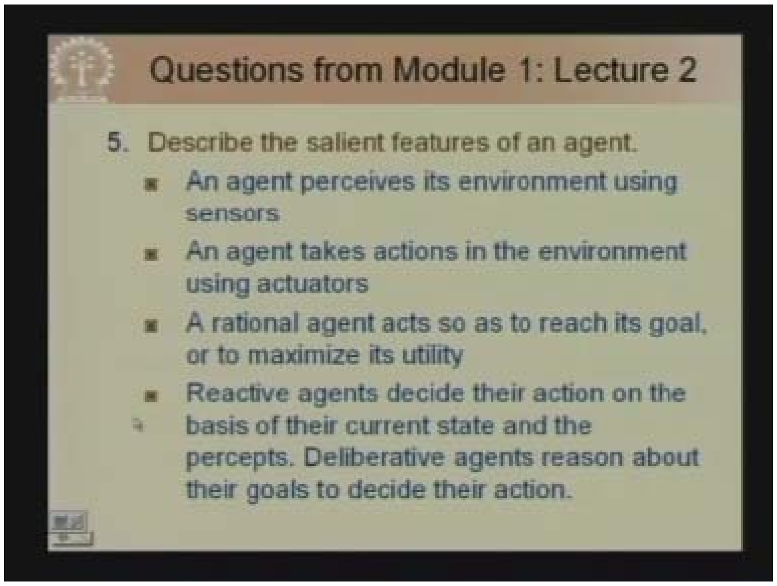
Questions from Module 1: Lecture 2

3. What is bounded rationality ?
   A rational agent that can use only bounded resources cannot exhibit the optimal behaviour. A bounded rational agent does the best possible job of selecting good actions given its goal, and given its bounded resources.
4. What is an autonomous agent ?
   Autonomous agents are software entities that are capable of independent action in dynamic, unpredictable environments. An autonomous agent can learn and adapt to a new environment.

Third question is what is bounded rationality?
A rational agent that can use only bounded resources cannot exhibit the optimum behavior. A bounded rational agent does the best possible job of selecting good actions given its goal and its bounded resources. So, if the agent has only limited time in its disposal and limited memory or other resources the agent should do the best that it can under the circumstances. That is the definition of bounded rationality.

Fourth question is; what is an autonomous agent?
Autonomous agents are those agents that are capable of independent action in environments which are dynamic and unpredictable. An autonomous agent needs to learn and adapt to a new environment. The fifth question is, describe the salient features of an agent.

(Refer Slide Time: 06:43)

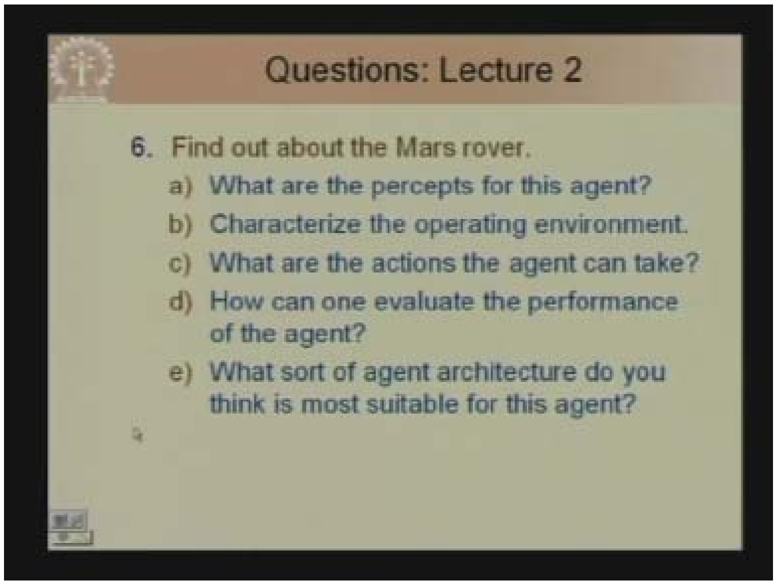


Questions from Module 1: Lecture 2

5. Describe the salient features of an agent.
   - An agent perceives its environment using sensors
   - An agent takes actions in the environment using actuators
   - A rational agent acts so as to reach its goal, or to maximize its utility
   - Reactive agents decide their action on the basis of their current state and the percepts. Deliberative agents reason about their goals to decide their action.

An agent perceives its environment using sensors. An agent takes actions using actuators. A rational agent acts so as to reach its goal or to maximize its utility. Reactive agents decide their action on the basis of their current state and percepts. Deliberative agents reason about their goals to decide their action.

(Refer Slide Time: 07:15)



Questions: Lecture 2

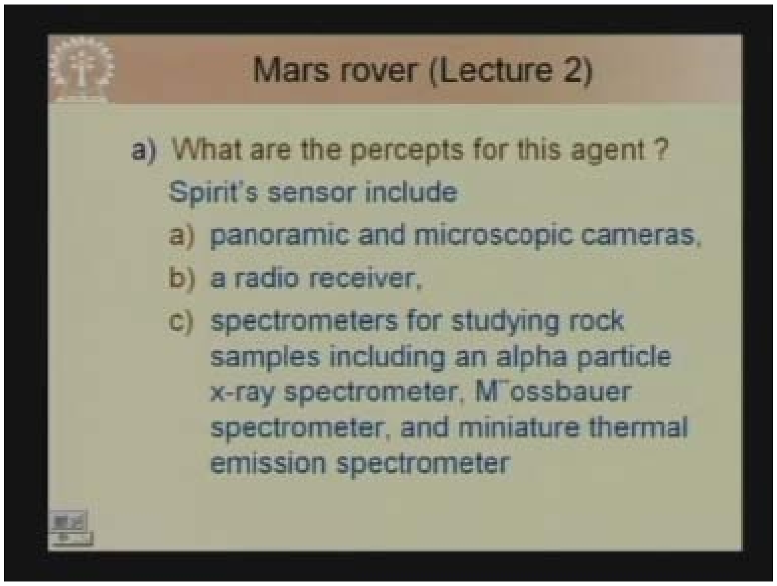
6. Find out about the Mars rover.
   a) What are the percepts for this agent?
   b) Characterize the operating environment.
   c) What are the actions the agent can take?
   d) How can one evaluate the performance of the agent?
   e) What sort of agent architecture do you think is most suitable for this agent?

Question number 6; find out about the Mars rover. These are the questions we want to answer about the rover.

a) What are the percepts of this agent?

b) Characterize the operating environment.
c) What are the actions the agent can take?
d) How one can evaluate the performance of the agent?
e) What sort of agent do you think is most suitable for this agent?

(Refer Slide Time: 07:49)

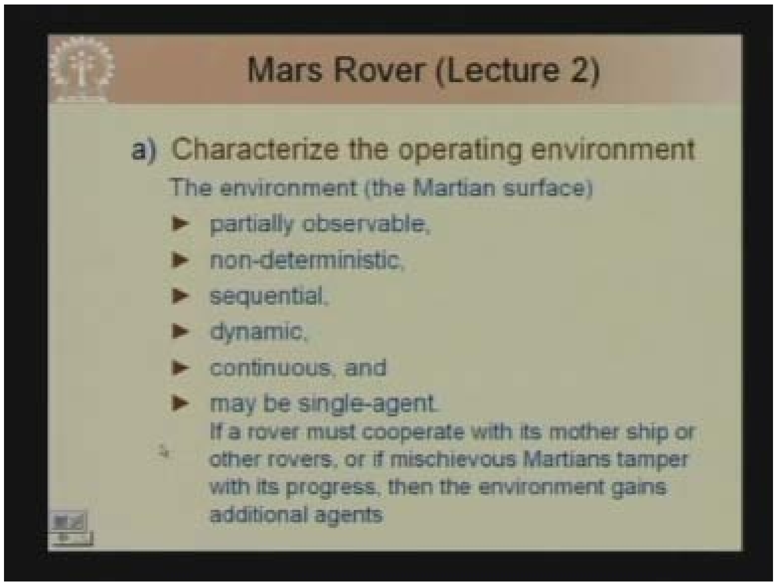


Mars rover (Lecture 2)

a) What are the percepts for this agent ?
Spirit's sensor include
a) panoramic and microscopic cameras,
b) a radio receiver,
c) spectrometers for studying rock samples including an alpha particle x-ray spectrometer, M¨ossbauer spectrometer, and miniature thermal emission spectrometer

Now let us look at the answers to these questions one by one. Firstly, about the percepts that the agent receives we looked at the website of the agent spirit. Spirit is one of the mars rovers that we talked about. Spirit's sensor include: panoramic and microscopic cameras, a radio receiver and various spectrometers for studying rock samples including an alpha particle x ray spectrometer, moss Bauer spectrometer and miniature thermal emission spectrometer.

(Refer Slide Time: 08:26)



Mars Rover (Lecture 2)

a) Characterize the operating environment

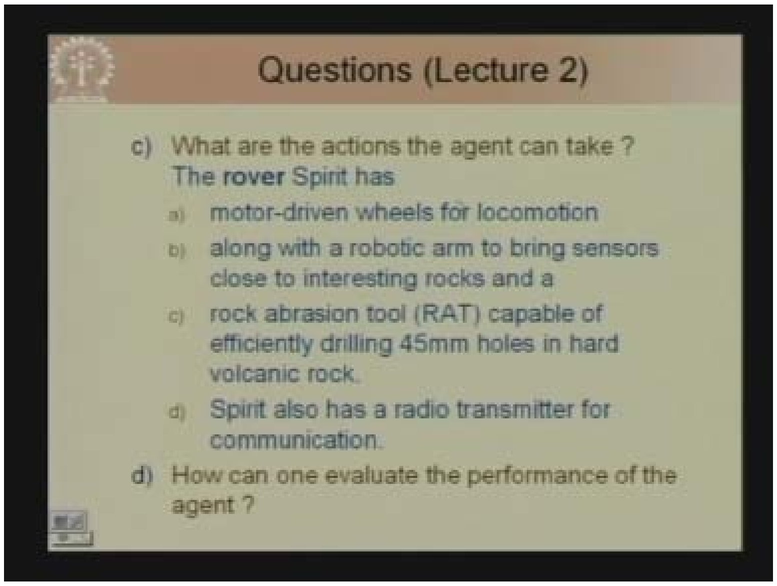
The environment (the Martian surface)

- partially observable,
- non-deterministic,
- sequential,
- dynamic,
- continuous, and
- may be single-agent.

If a rover must cooperate with its mother ship or other rovers, or if mischievous Martians tamper with its progress, then the environment gains additional agents

The operating environment of this rover is supposed to be the Martian surface. This environment is partially observable. This is in fact a very realistic environment. This is a partially observable environment. The environment is non-deterministic. It is sequential, that is, it is not episodic. The environment is dynamic. It is continuous and not discrete and we can look upon it as a single agent environment. However, if we are modeling other agents or the interaction of this rover with the mother ship then we can look upon this as a multi agent environment but otherwise this can be modeled as a single agent environment.

(Refer Slide Time: 09:27)



Questions (Lecture 2)

c) What are the actions the agent can take ?
The **rover** Spirit has

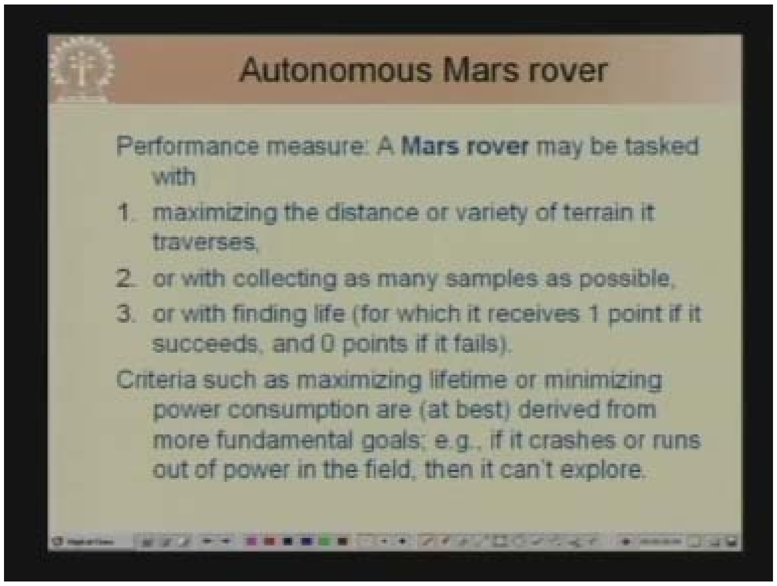
a) motor-driven wheels for locomotion
b) along with a robotic arm to bring sensors close to interesting rocks and a
c) rock abrasion tool (RAT) capable of efficiently drilling 45mm holes in hard volcanic rock.
d) Spirit also has a radio transmitter for communication.

d) How can one evaluate the performance of the agent ?

What are the actions this agent can take?
Now, this rover spirit has the following actuators. It has motor driven wheels for locomotion. It has a robotic arm with which it can bring sensors close to interesting rocks. It has a Rock Abrasion Tool RAT which is capable of efficient drilling holes in hard volcanic rocks. Spirit also has a radio transmitter for communication.

The final question was or rather question number d) was; how can one evaluate the performance of his agent?
Now let us look at the possible performance measures that we can design for this agent.

(Refer Slide Time: 10:20)



Autonomous Mars rover

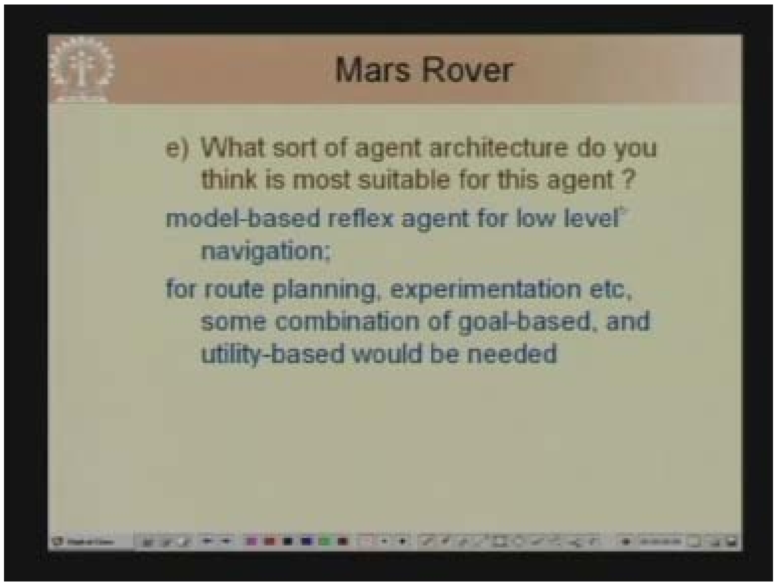
Performance measure: A **Mars rover** may be tasked with

1. maximizing the distance or variety of terrain it traverses,
2. or with collecting as many samples as possible,
3. or with finding life (for which it receives 1 point if it succeeds, and 0 points if it fails).

Criteria such as maximizing lifetime or minimizing power consumption are (at best) derived from more fundamental goals; e.g., if it crashes or runs out of power in the field, then it can't explore.

A Mars rover may have the task of maximizing the distance that it has traversed or the diversity of the exploration that it has carried out or the variety of terrain that it has traversed. Its performance could be measured by seeing how much variety of samples it has been able to collect or whether it has been successful in finding water or finding life. We already know that the Mars rovers' were successful in detecting traces of water in mars. There could be other criteria such as maximizing the lifetime or minimizing the power consumption. And there are other goals like, if it crashes or runs out of power it cannot explore. Therefore there are various ways in which we can define the performance of this agent.

(Refer Slide Time: 11:30)



Mars Rover

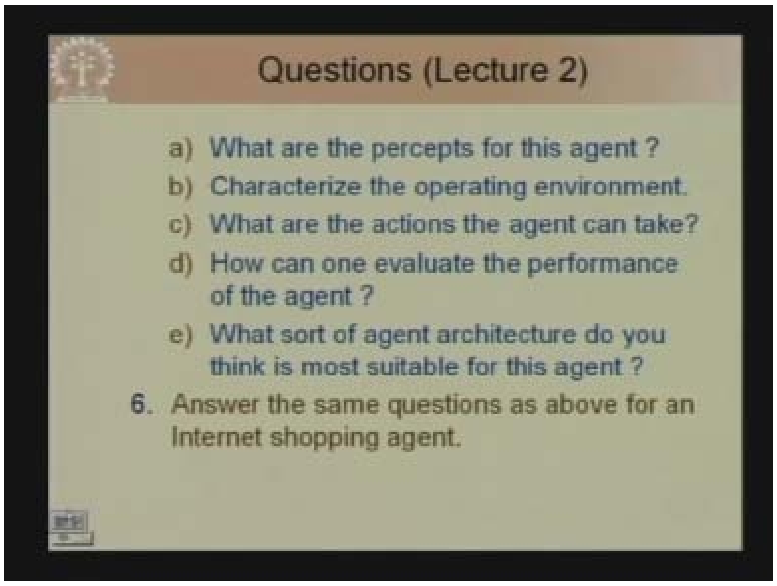
e) What sort of agent architecture do you think is most suitable for this agent ?

model-based reflex agent for low level navigation;

for route planning, experimentation etc, some combination of goal-based, and utility-based would be needed

The final question on the Mars rover was what sort of agent architecture you think is most suitable for this agent?
In my mind a hybrid architecture which has the components of both stimulus response and reactive type of architecture as well as deliberative architecture would be ideal. For example, I would visualize a model based reflex agent for low level navigation and a goal based or utility based agent for other tasks like route planning, experimentation etc. And the next question was the same set of questions about internet shopping agent. Let us look at some of the answers.

(Refer Slide Time: 12:17)



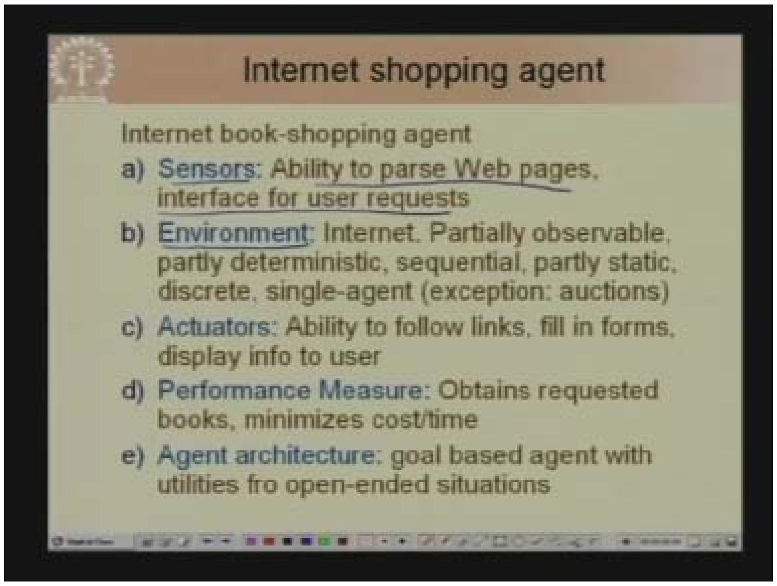
Questions (Lecture 2)

a) What are the percepts for this agent ?
b) Characterize the operating environment.
c) What are the actions the agent can take?
d) How can one evaluate the performance of the agent ?
e) What sort of agent architecture do you think is most suitable for this agent ?

6. Answer the same questions as above for an Internet shopping agent.

For the internet book shopping agent the sensors would be the ability to parse web pages or interface for user requests. Getting requests from users and being able to parse web pages would be looked upon as sensors for internet shopping agent.

(Refer Slide Time: 12:43)



Internet shopping agent

Internet book-shopping agent

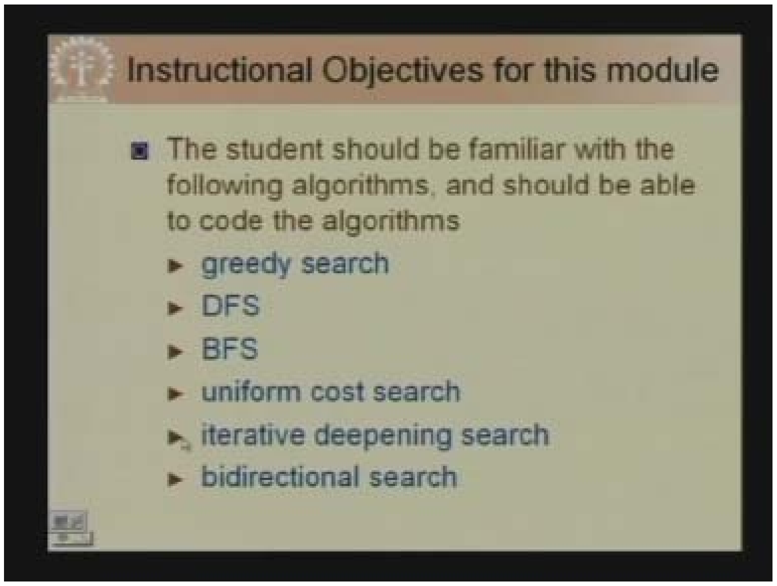
a) Sensors: Ability to parse Web pages, interface for user requests
b) Environment: Internet. Partially observable, partly deterministic, sequential, partly static, discrete, single-agent (exception: auctions)
c) Actuators: Ability to follow links, fill in forms, display info to user
d) Performance Measure: Obtains requested books, minimizes cost/time
e) Agent architecture: goal based agent with utilities fro open-ended situations

The environment it acts in is the internet. This environment is partially observable. It is partly deterministic, sequential, it is static but not quite because the web is changing. The environment is discrete and it is usually a single agent unless we are trying to model auctions and other such things in which case we go for a multi agent environment.

The third question was; what are the actuators of this agent?
The actuators is the action that this agent can take is the agent can follow links, it can fill in forms, it can give information to the user. These are the actions the agent can take.

d) We wanted to look at the performance measure of the agent.
This agent’s performance is whether it has been able to obtain the requested items or books, whether it can minimize the cost of the books or time to fetch the books and so on. And the agent architecture that we visualize is we have a goal based agent with utilities for open ended situations. Therefore, in situations which are not very well specified we can have a utility based agent but mainly we can have a goal based agent which has a particular goal in mind for which it has to minimize the cost and the time.

(Refer Slide Time: 14:39)

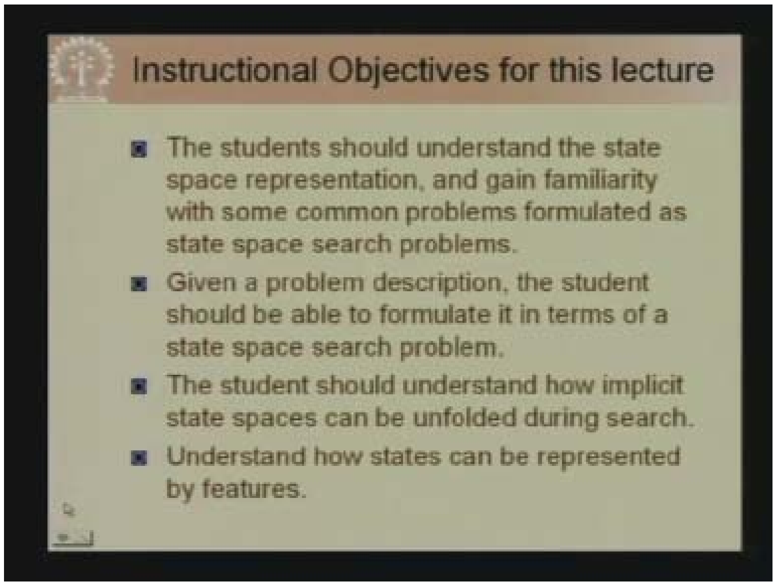


Instructional Objectives for this module

- The student should be familiar with the following algorithms, and should be able to code the algorithms
  - greedy search
  - DFS
  - BFS
  - uniform cost search
  - iterative deepening search
  - bidirectional search

Now we come to our second module. This module is called problem solving using search. We will introduce the concept of State Space Search in the first lecture today. Instructional objectives for this module are as follows:
Our objective is that the student should be familiar with various search paradigms like depth first search, breadth first search, uniform cost search, iterative deepening search and bidirectional search etc which we will deal with in the subsequent lectures. In today's lecture we will look at the basic formulation of search problems as State Space Search.

(Refer Slide Time: 15:15)



Instructional Objectives for this lecture

- The students should understand the state space representation, and gain familiarity with some common problems formulated as state space search problems.
- Given a problem description, the student should be able to formulate it in terms of a state space search problem.
- The student should understand how implicit state spaces can be unfolded during search.
- Understand how states can be represented by features.

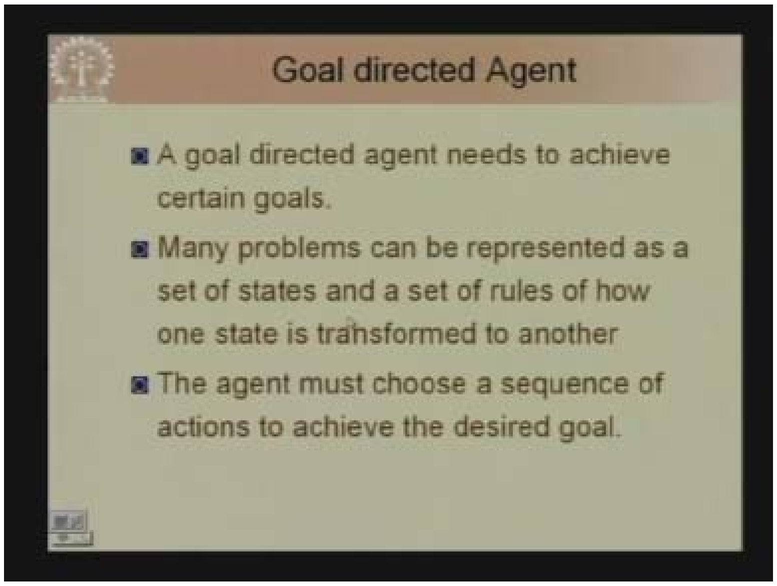
At the end of today's lecture the student should understand the state space representation and gain familiarity with some common problems which can be formulated as State Space Search problems.

The second objective is, after the student has learnt this material he should have this following capability. Given a problem description he should be able to formulate it in terms of a State Space Search problem. He should be able to do the state space decomposition of the problem, decide how a state should be represented, how state change takes place model actions and formulate the State Space Search problem.

Thirdly, the student should understand how implicit state spaces are described and how implicit state spaces are unfolded during search by making the generated state explicit.

Fourthly, the student should understand how states can be represented by some features of a state. Now, before we come to the state space search problem let us review the definition of an intelligent agent and let us see how search is important or why search is important to an intelligent agent. Now to recapitulate we have this agent which is working in this environment. The agent receives the percepts from the environment which the agent visualizes and the agent takes actions using its actuators which change the state of the environment.

(Refer Slide Time: 17:18)

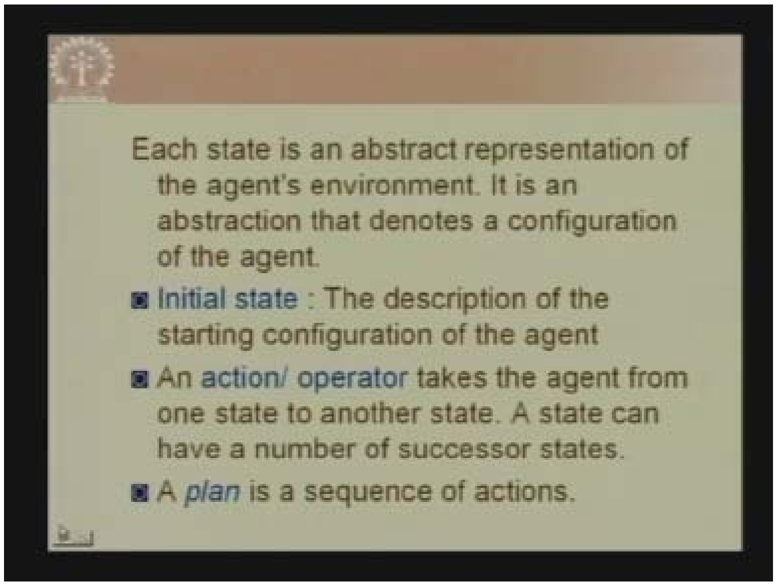


Goal directed Agent

- A goal directed agent needs to achieve certain goals.
- Many problems can be represented as a set of states and a set of rules of how one state is transformed to another
- The agent must choose a sequence of actions to achieve the desired goal.

A goal directed agent as we have seen needs to achieve certain goals. Today we will see that many problems can be represented as a set of states and a set of rules of how one state is transformed to another. And in these problems the agent must choose a sequence of actions in order to achieve its goal. And we will see that such goal based agents can be modeled using state space search.

(Refer Slide Time: 17:56)

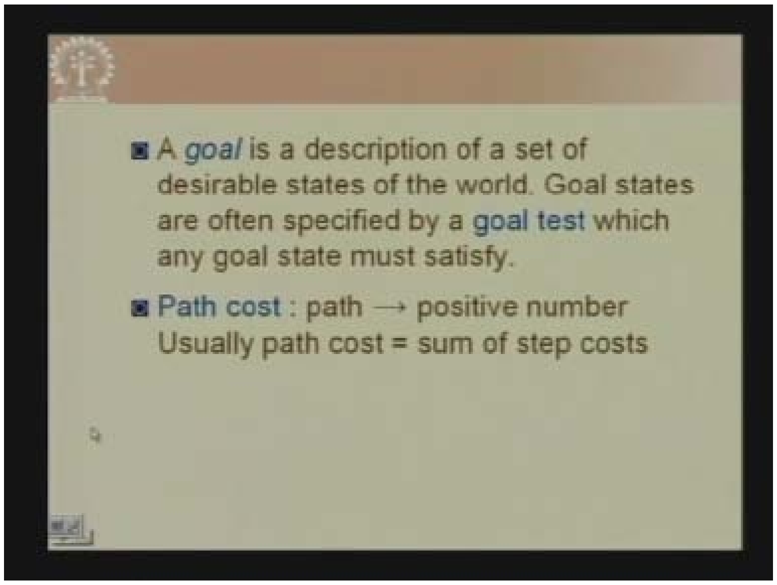


Each state is an abstract representation of the agent's environment. It is an abstraction that denotes a configuration of the agent.

- Initial state : The description of the starting configuration of the agent
- An action/ operator takes the agent from one state to another state. A state can have a number of successor states.
- A *plan* is a sequence of actions.

Each state is an abstract representation of the agent's environment. Each state is an abstraction that denotes a configuration of the agent. A state denotes the current configuration of the agent. Each state is an abstract representation of the current configuration of the agent which is sufficient for solving the problem that the agent needs to solve. The initial state is a description of the starting configuration of the agent. An action or an operator takes an agent from one state to another state. By taking an action the agent moves from a current state to its successor state. A plan is a sequence of actions that the agent can take. So a plan is a sequence of actions and the agent starts from the initial state executes the plan that is takes a sequence of actions so that it reaches a goal.

(Refer Slide Time: 20:24)

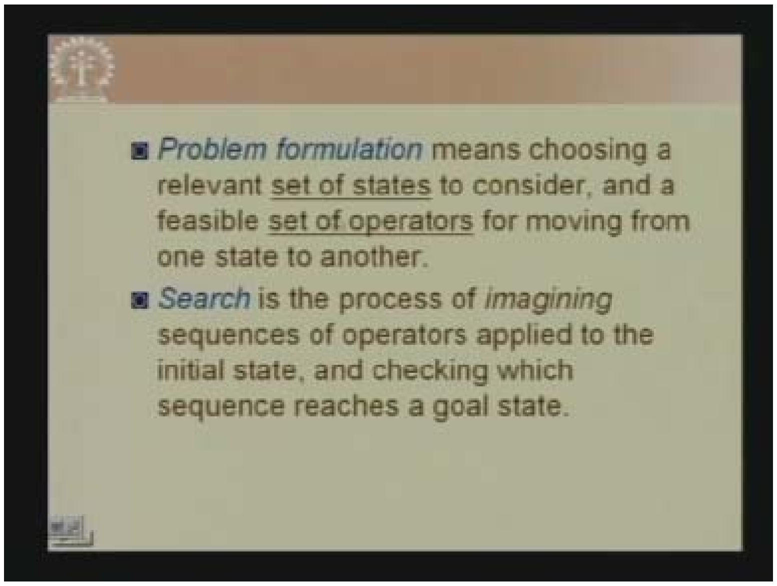


- A *goal* is a description of a set of desirable states of the world. Goal states are often specified by a goal test which any goal state must satisfy.
- Path cost : path $\rightarrow$ positive number
  Usually path cost = sum of step costs

A goal is a description of a set of desirable states of the world. Goal states are often specified by a goal test. We can say that the following are the possible goal states. Or we can say that any state which satisfies the goal test is a goal state. We will take up examples to illustrate this. So a plan or a path is a sequence of actions. And the cost of a path is a positive number which is usually the sum of the costs of the different steps or different actions which are part of the path. Now, in this class we are going to talk about problem formulation.

What is problem formulation?
Problem formulation means choosing a relevant set of states to consider and a feasible set of operators for moving from one state to another.

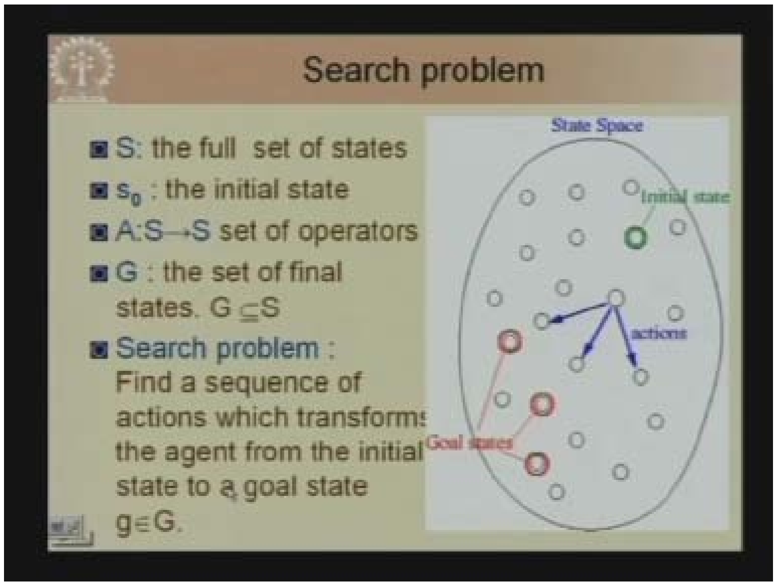
(Refer Slide Time: 20:16)



- *Problem formulation* means choosing a relevant set of states to consider, and a feasible set of operators for moving from one state to another.
- *Search* is the process of *imagining* sequences of operators applied to the initial state, and checking which sequence reaches a goal state.

So, given a problem we want to formulate the problem in terms of a set of states and a set of operators or actions. Search as we will see is the process of imagining sequences of operators applied to the initial state and checking with sequence reaches a goal state.

(Refer Slide Time: 20:50)



Search problem

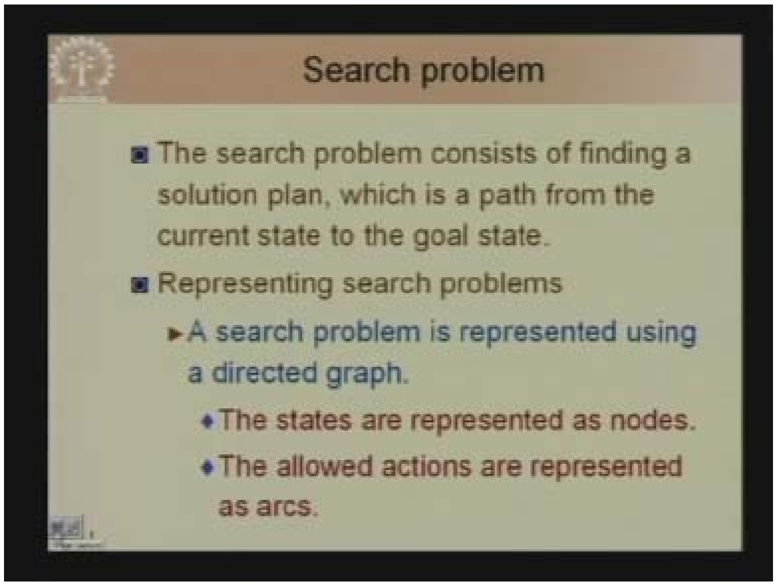
- S: the full set of states
- s_0 : the initial state
- $A:S \to S$ set of operators
- G : the set of final states. $G \subseteq S$
- Search problem : Find a sequence of actions which transforms the agent from the initial state to a goal state $g \in G$.

State Space
Initial state
actions
Goal states

Now let us look at what constitutes a search problem. S is the complete set of states and s_0 is the initial state. So s_0 is the member of the set of states S. Suppose this is the representation of a state space and these circles here are the different states. This is the set of states or S. And s_0 is the initial state. This green circle here is the initial state s_0. A is the set of actions. A is a mapping from S to S. An action takes the agent from one state to another state. And g is the set of final states. These red circles (Refer Slide Time:: 21:50) are the goal states in this case.

A search problem consists of finding a sequence of actions which transforms the agent from the initial state to a goal state. For example, from this initial state the agent might have three actions. If he takes action a he goes to this state, if he takes action b he reaches this state and if he takes action c it reaches this state. So, from this again the agent can take another action, another action, another action and then the agent can take a sequence of actions to reach a goal state. These are the possible actions available from this state. The agent can take one of the available actions from a state. The search problem is, given an initial state find a sequence of actions which transforms the agent to a goal state. Therefore, a search problem was represented as a directed graph.

(Refer Slide Time: 23:11)

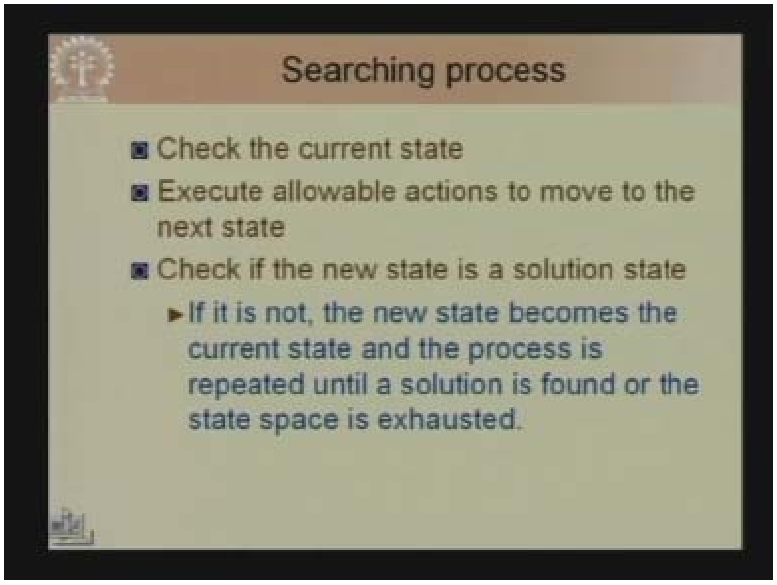


Search problem

- The search problem consists of finding a solution plan, which is a path from the current state to the goal state.
- Representing search problems
  - A search problem is represented using a directed graph.
    - The states are represented as nodes.
    - The allowed actions are represented as arcs.

In this graph the states are represented as nodes. The states are the nodes and the operators are the edges between the nodes. The operators are the edges between the nodes and the states are the nodes. The searching process is grossly the following. The agent checks the current state and it can execute allowable actions to move to the next state.

(Refer Slide Time: 23:41)

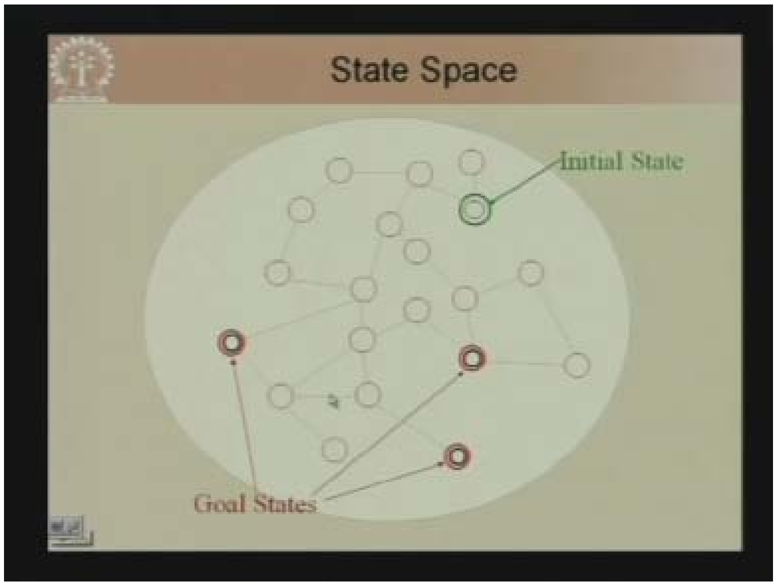


Searching process

- Check the current state
- Execute allowable actions to move to the next state
- Check if the new state is a solution state
  - If it is not, the new state becomes the current state and the process is repeated until a solution is found or the state space is exhausted.

Suppose this is the current state and suppose from this state there are two possible operators, this is a state a, operation a takes the agent to the state x and operation b takes the agent to the state z, so a and b are the executable operators in state a. Now the agent is at a. It takes one of the executable operators. Suppose the agent selects the action b and moves to the state z the agent checks if this state is a goal state. If it is a goal state the agent has reached a solution and if it is not a goal state the new state may become the current state of the agent and this process is repeated. That is the agent finds out the possible operations that it can execute in this current state and it continues this until the goal state is reached.

(Refer Slide Time: 24:58)

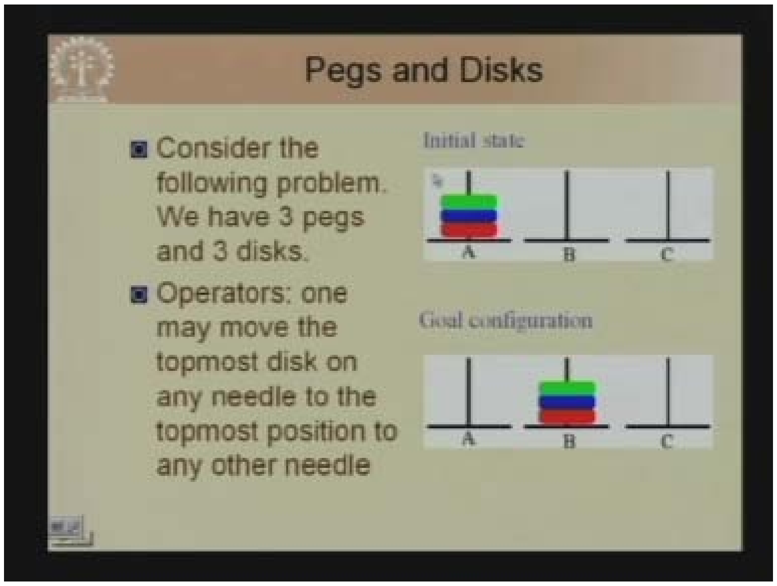


State Space

Now, this is an example of a search space. This is the initial state here and these three are the goal states. These other circles are the other states in the system. Now, the objective of the agent is to start from this initial state and reach one of these goal states. Now, from this state the agent can take two possible actions to reach either this state or this state. From this state again the agent can take two possible actions and then these are the other nodes the agent can reach and so on until the agent reaches a goal state. So this is the sort of process that the agent carries out. From here the agent has a choice of going either in this direction or this direction, from here you can go here or here and from here it can go here and here and so on. So the agent can start from the initial state and explore the state space and its objective is to reach one of the goal states.

Now let us look at another problem. This is a pegs and disks problem which is a variation of the block problem or these various towers Hanoi problem. In this particular problem we have three pegs and we have three disks red, blue and green.

(Refer Slide Time: 26:40)



Pegs and Disks

- Consider the following problem. We have 3 pegs and 3 disks.
- Operators: one may move the topmost disk on any needle to the topmost position to any other needle

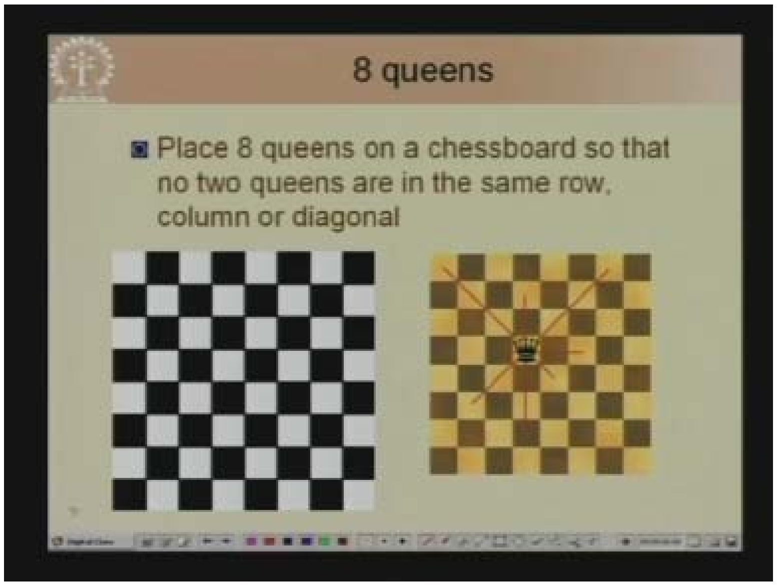
Initial state
A
B
C
Goal configuration
A
B
C

This picture shows the initial configuration of the problem. We have three pegs and we have red, blue and green on peg a. The following operators are allowed. One may move the top most disks or any peg to the top most position of any other peg. These are the only allowable operations. And our objective is to reach this configuration where the red, blue and green or disks are in this same position on peg b and these are the allowable operators. We will see how we can find a solution to this problem.

This is the starting configuration. Now what I will do is I will outline a sequence of steps by which we can move from the starting configuration to the goal configuration and later we will see how we model the different states. The actions we will take here is move from a to c, that is, move the top most disk of a to c. The next action is move from a to b, next action is move from a to c, next action is move from b to a, then move from c to b, then move from a to b, then move from c to b and finally we have this desired goal configuration.

Now, the next problem we will look at is the well known 8 queens problem. The problem constitutes the following:
We have a chess board and we have 8 queens. And according to the rules of chess a queen can attack any other piece which is situated either in the same row or the same column or on one of the diagonals.

(Refer Slide Time: 28:36)

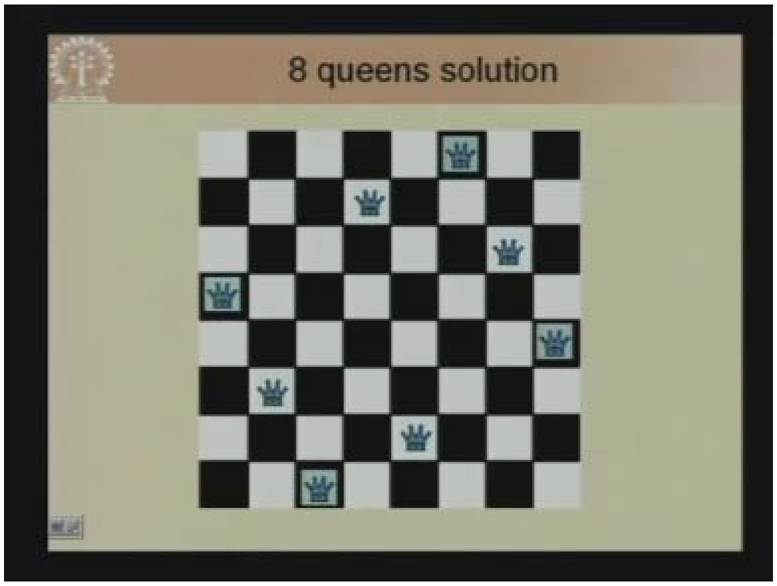


8 queens

- Place 8 queens on a chessboard so that no two queens are in the same row, column or diagonal

If we have any piece in these positions they can be attacked by the queen. The 8 queens problem is the following:
Place 8 queens on the chess board such that no queen is attacking any other queen. Now how can we formulate the state space for this problem and then we will look at how to solve this problem as a State Space Search problem. There are different ways in which we can formulate the 8 queens problem.

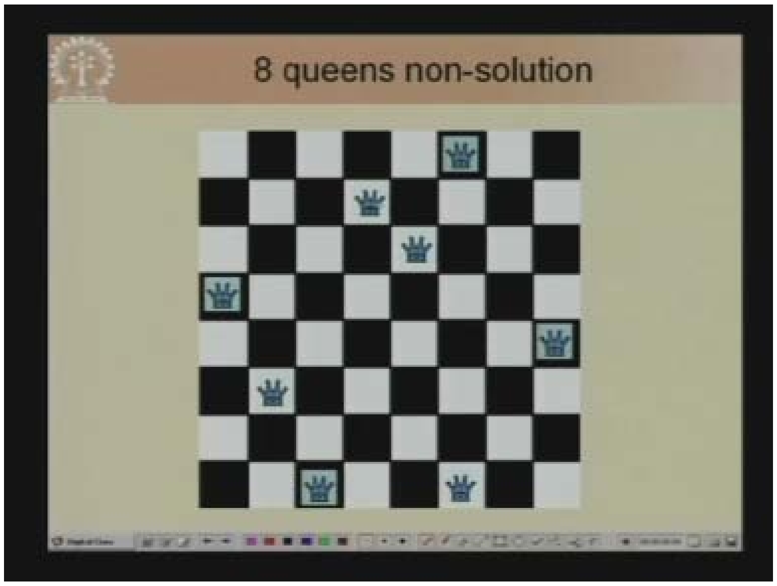
(Refer Slide Time: 29:29)



8 queens solution

So, before we look at this let us look at a typical solution of the 8 queens problem. This is a valid solution of the 8 queens problem and you can verify that none of the queens are attacking each other. This is an example of a non solution to the 8 queens problem.

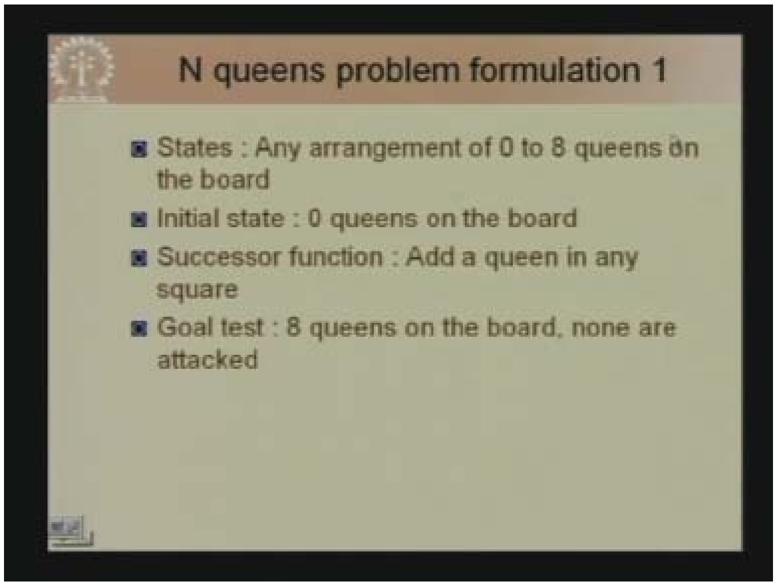
(Refer Slide Time: 29:47)



8 queens non-solution

Why is it not a solution because these two queens are attacking each other and these two queens are attacking each other and so on. So this is not a correct solution to the 8 queens problem. Now let us look at some possible formulations of the 8 queens problem.

(Refer Slide Time: 30:09)



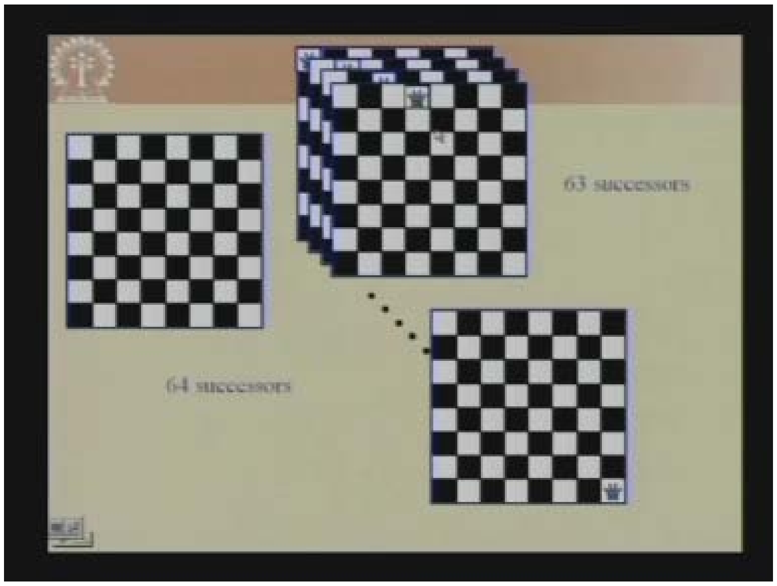
N queens problem formulation 1

- States : Any arrangement of 0 to 8 queens on the board
- Initial state : 0 queens on the board
- Successor function : Add a queen in any square
- Goal test : 8 queens on the board, none are attacked

In the first formulation we can say that a state is an arrangement of 0 to 8 queens on the board. A state is any arrangement of less than or equal to 8 queens on the board. A state is specified by the position of the queens which are currently in the board. The initial state is no queens on the board. The possible successor function is add a queen in any square.

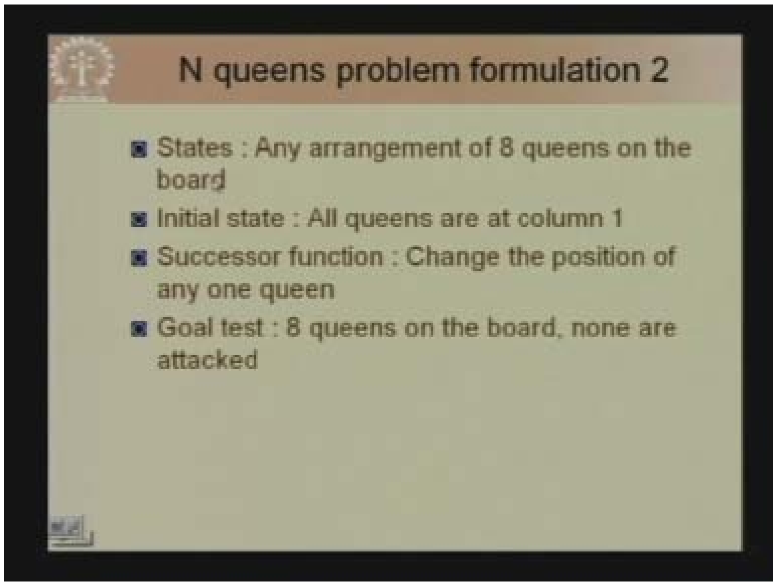
Suppose in this configuration we have five queens in the next configuration we can add any queen to the boards that it does clash with the same position. That is, from this board we can have 64 minus 5 is equal to 59 possible successor states. The goal test is that 8 queens are on the board and none are attacked. So in this formulation this is the initial state. This state will have 64 successors depending on the placement of the first queen that we introduce. So this is one successor where we have placed the first queen in this square, this is the second successor, this is the third successor, this is the fourth successor and so we have 64 successors. And each of these states will again have 63 successors. So if we start from this state there will be 63 successors.

(Refer Slide Time: 32:02)



So, that was one possible way of formulating 8 queens as a State Space Search problem. We can look at other alternative formulations of the 8 queens problem.

(Refer Slide Time: 32:23)



N queens problem formulation 2

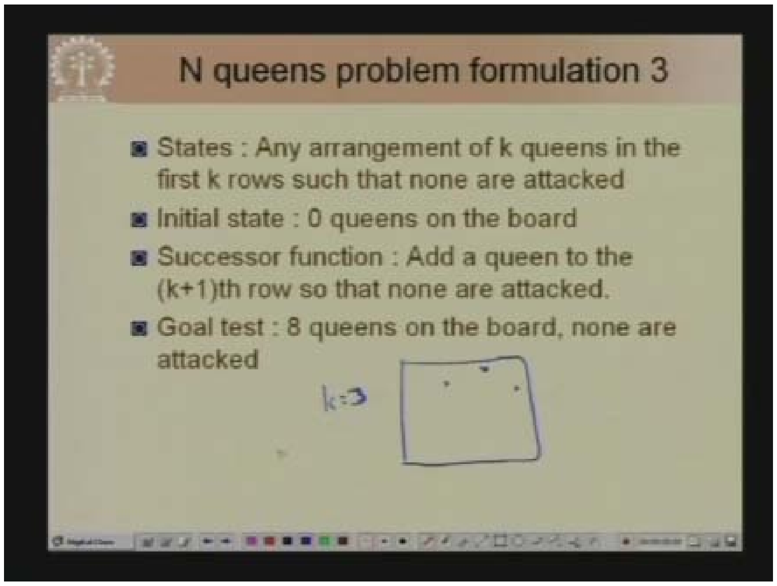
- States : Any arrangement of 8 queens on the board
- Initial state : All queens are at column 1
- Successor function : Change the position of any one queen
- Goal test : 8 queens on the board, none are attacked

In the second formulation a state is any arrangement of 8 queens on the board. So we always have queens and the state is any arrangement of 8 queens. In the initial state all queens are at column one. So this is the board, in the initial state all the queens are at the column one and the successor function has changed the position of one queen. The goal test is the same where 8 queens are on the board and none are attacked. So this is the initial state.

Now the possible operators are, we can take this queen and move it to another column. So this particular queen we can move to any of these seven columns or we can take this queen and move to one of these seven columns. These are the possible successor states. Now this is an intermediate state for this formulation. At this intermediate state also we can take one of the queens and move it around. So this is one formulation of the 8 queens problem.

A third formulation is; we describe a state by any arrangement of k queens and the first k rows. So at an intermediate point we would have a current value of k. Suppose k is equal to 3 means that one queen is placed somewhere in row one, another queen is placed somewhere in row two and the third queen is placed somewhere in row three. So, for k is equal to 3 this is a possible state of the system. The successor function is, add a queen to the k plus 1th row so that none are attacked. That is, the partial solution does not violate any constraints. So we take the fourth queen and put it in one of the rows such that the queens are not attacked by each other.

(Refer Slide Time: 34:55)

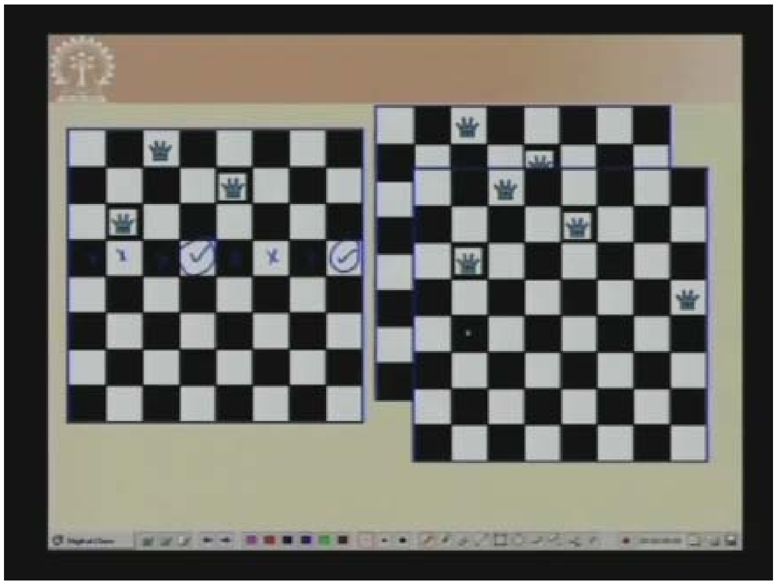


N queens problem formulation 3

- States : Any arrangement of k queens in the first k rows such that none are attacked
- Initial state : 0 queens on the board
- Successor function : Add a queen to the (k+1)th row so that none are attacked.
- Goal test : 8 queens on the board, none are attacked

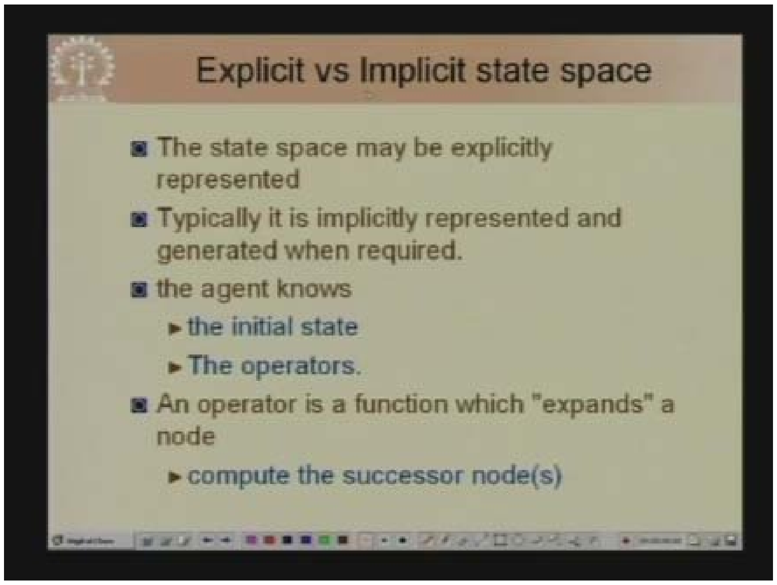
Suppose for k is equal to 3 this is a possible starting configuration and this is possible then let us see in this configuration what are the positions in which we can keep the queen. We cannot keep the queen here because it will be attacked, we cannot buy this queen we cannot keep the queen here, we cannot keep the queen here because it will be attacked by this one, we can keep the queen in this position, we cannot keep the queen here, we cannot keep the queen here because it will be attacked by this queen, we cannot put the queen here because it will be attacked by this queen, we can put the queen here. So this state has two successor states.

(Refer Slide Time: 35:46)



What we have seen is that, given the same problem there are different ways of abstractly representing the state space of the problem. When we look at the different search algorithms we will try to see whether a particular formulation is suitable or good for obtaining a solution of the problem in an efficient manner and in a correct manner. Now we come to the concept of implicit state space.

(Refer Slide Time: 36:28)



Explicit vs Implicit state space

- The state space may be explicitly represented
- Typically it is implicitly represented and generated when required.
- the agent knows
  - the initial state
  - The operators.
- An operator is a function which "expands" a node
  - compute the successor node(s)

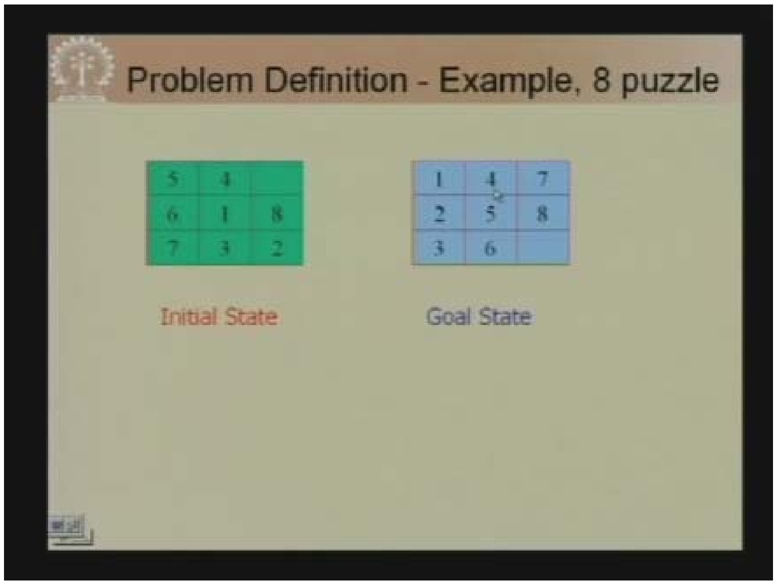
The state space may be explicit or it may be implicit. In an explicit state space we model explicitly all the states of the system and we model all the possible operations on the system. This is an example of an explicit state space where every state is explicitly modeled. However, in this course we were mainly dealing with very huge state spaces. And it may not be possible or easy to model all the states of the system explicitly neither is it necessary. It may not be necessary to spell out all the states of the system in order to find a solution. So we will look at mainly problems where the state space is implicit.

That is, we need not explicitly in the beginning keep all the states rather we give a way of representing the states and we give methods so that given a state and given an operator we can generate the next state. We give tests for testing whether the agent has reached a goal state. So our state space will mostly be implicitly represented. And in this implicit state space representation the agent will know the initial state that the agent is in but he will not initially need to keep all the states that the system has. He will know the initial state and he knows the operators and the effects of these operators so that the agent can unfold the state space. So he knows that if these operations are applied at this state these are the states that he can reach. Then if he selects the state for expansion we can obtain the different other states that he can reach from this state.

An operator is a function which expands a node. That is, it computes the successors of the node. Given a state given an operator the agent will be able to generate the successor node. So let us take the 8 puzzle problem as an example. Many of you may be familiar with either the 8 puzzle or its cousin the 15 puzzle.

In this 8 puzzle problem there is a boat consisting of 9 squares and there are 8 blocks numbered 1 2 3 4 5 6 7 8. These blocks are in a particular configuration in this boat and one position is empty. Now in this sliding tile puzzle we can slide these blocks which are adjacent to the blank position. That is we can slide this tile four to this position or slide tile eight to this position. And the objective of this 8 puzzle problem is to reach the goal state. So the goal state is another configuration of this system. There could be different configurations that we decide for the goal state. Given a particular goal set configuration the objective of the puzzle is to start from the initial state and slide the blocks around so that the agent reaches the goal state.

(Refer Slide Time: 40:25)



Problem Definition - Example, 8 puzzle

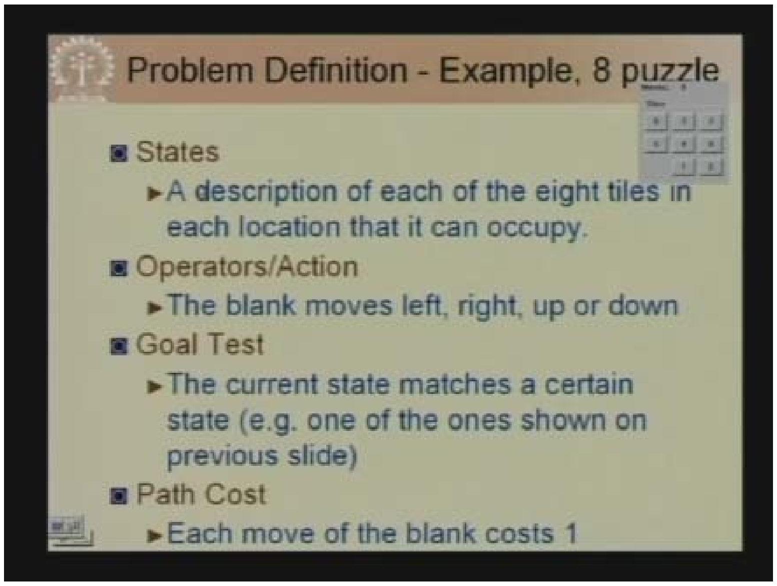
5	4	
6	1	8
7	3	2

Initial State

1	4	7
2	5	8
3	6	

Goal State

(Refer Slide Time: 40:36)



Problem Definition - Example, 8 puzzle

- States
  - A description of each of the eight tiles in each location that it can occupy.
- Operators/Action
  - The blank moves left, right, up or down
- Goal Test
  - The current state matches a certain state (e.g. one of the ones shown on previous slide)
- Path Cost
  - Each move of the blank costs 1

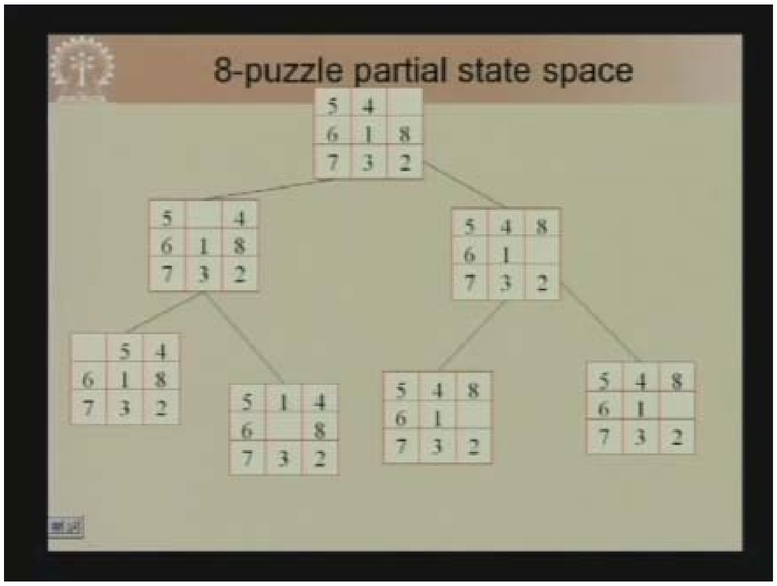
What is the state space representation of 8 puzzle?
A state is a description of each of the eight tiles in each location that it can occupy. The operators are; the blank can move left right up or down.

Goal test: If the current state matches the goal configuration then we say that the system has reached the goal state. The path cost is the number of steps or the number of times one has to slide a block in order to go from the start state to the goal state. So the objective of the eight puzzle problem is to start from the start state and reach the goal state taking the minimum number of steps. So let us look at how we represent eight puzzle and how we can expand the state space of the eight puzzle. This is our initial state. From this initial state two possible actions we can take.

We can either move four to the right or eight up or rather we can say we can move blank to the left or blank downwards. If you move blank to the left we come to this configuration and if you move blank downwards we come to this configuration. Now from this state again blank can move left or down. We do not move the blank right because then we will go back to this state. So we can get these two as the successor states of this state.

Similarly, we can expand this state and we can move the blank either to the left or downwards to get either this configuration or this configuration and so on. Likewise we can unfold the state space of eight puzzle. This is a very simple example of an implicit state space where we describe the result of applying the operator. The operators are moved up, down, left or right. There are four possible operators and we also give the pre conditions under which an operator can be applied. We can move blank up only if the blank is not already at the top most rows.

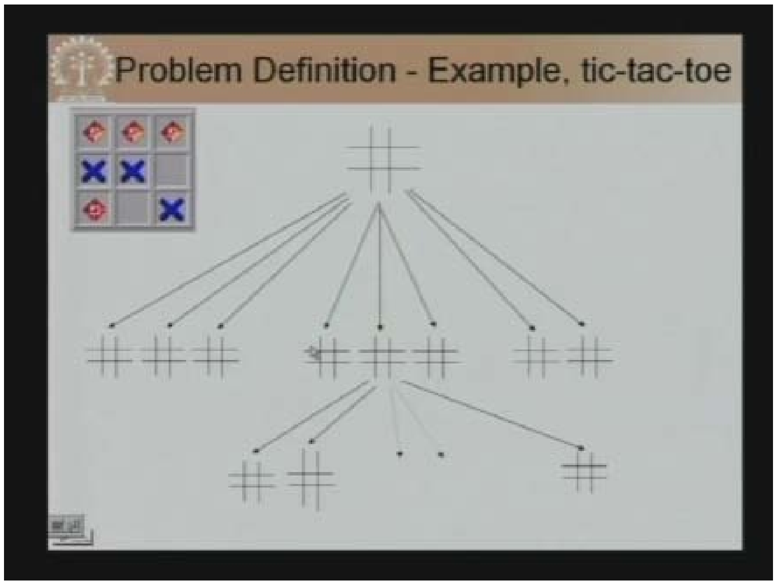
(Refer Slide Time: 42:57)



8-puzzle partial state space

Similarly, for each of the four operators we can lay down the pre condition and we can give the effect or describe the effect of these operators so that we can unfold the state space. As another example let us take the game Tic Tac Toe or Knots and crosses which many of you are familiar with. This by the way is a two person game which cannot be modeled as a search for a single agent but as that of two competing agents.

(Refer Slide Time: 43:30)

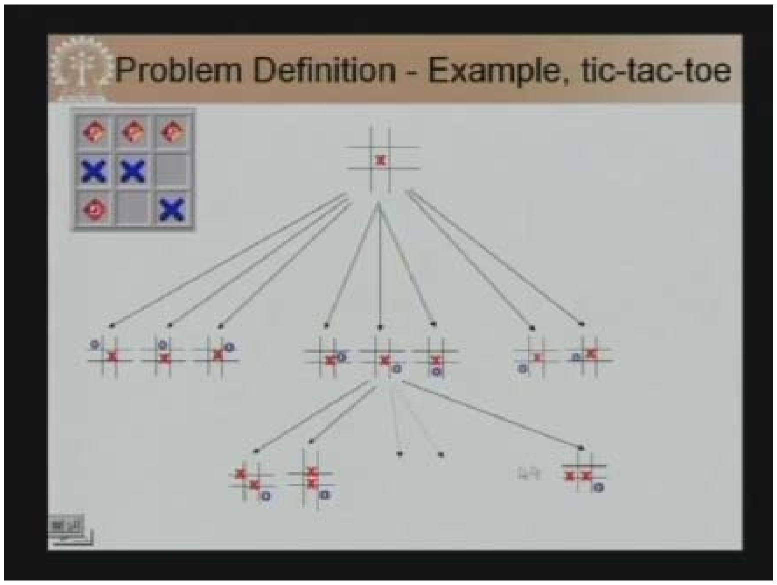


Problem Definition - Example, tic-tac-toe

In a Tic Tac Toe we have a board configuration like this and in the first instance the first player plays a cross suppose the first player puts a cross at the center location. So, in the search tree the different successors will correspond to cross at the central location and not placed at any of the other eight locations. So, knot can be placed either here or here or here or here or here or here or here or here. These are the eight possible places in which the player can place a knot.

Suppose a knot was placed in this location what are the possible successors?
Now in the next move the cross can be placed at any of the other seven locations. For example, the cross can be put either here or here and so on may be here. This way we can take the Tic Tac Toe problem to represent each state by the position of the knots and crosses.

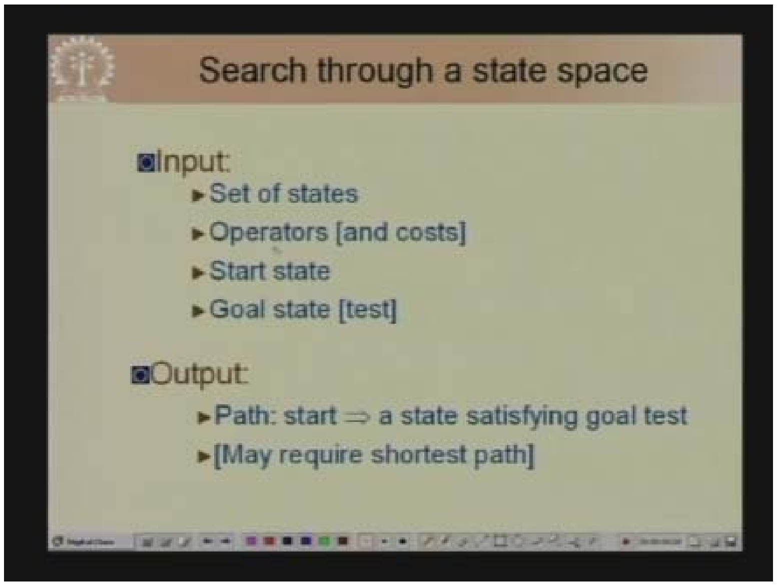
(Refer Slide Time: 44:52)



Problem Definition - Example, tic-tac-toe

We can represent each state and we can say the current operator is either a cross or knot and depending on whose turn it is to move whether it is a odd level move or even level move and the cross or the knot can be placed in a blank location.

(Refer Slide Time: 45:34)

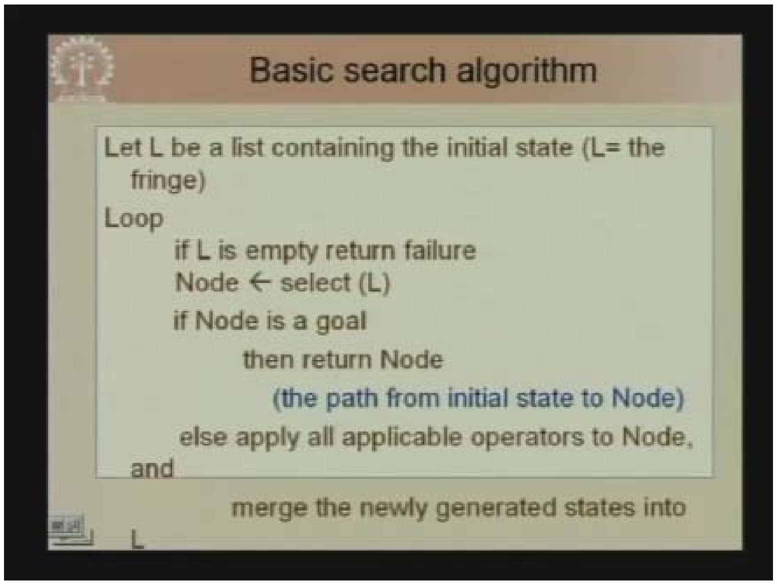


Search through a state space

- Input:
  - Set of states
  - Operators [and costs]
  - Start state
  - Goal state [test]
- Output:
  - Path: start $\Rightarrow$ a state satisfying goal test
  - [May require shortest path]

Now in a state space search problem the input is a set of states, a set of operators and may be costs associated with the operators. There is a distinguished state called the start state and there is a goal test which tells us whether we have reached a goal state g which is one of the goal states. A path starts from the initial state and it is a sequence of actions or sequence of states and the last state is the goal test. And the objective of the search problem is either to find a solution that is find a sequence of moves to take the agent from the start state to the goal state or the objective would be to find the shortest possible such path.

(Refer Slide Time: 46:28)



Basic search algorithm

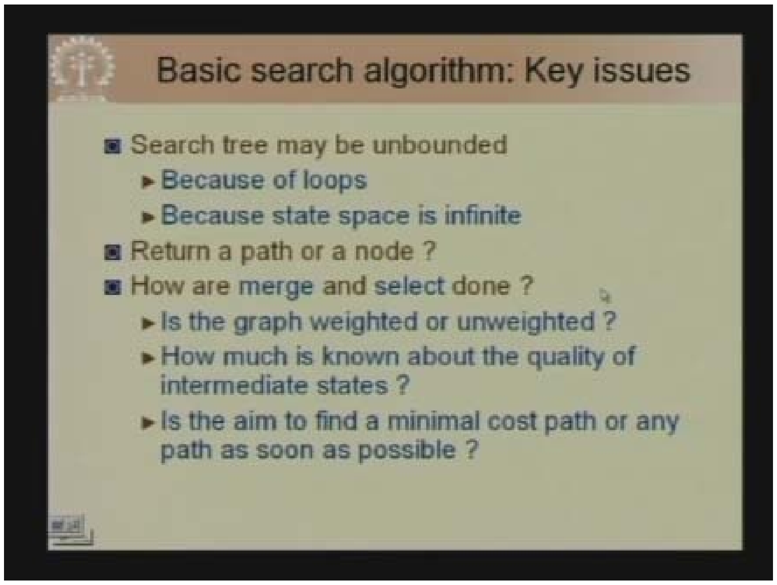
```
Let L be a list containing the initial state (L= the
  fringe)
Loop
     if L is empty return failure
     Node ← select (L)
     if Node is a goal
          then return Node
             (the path from initial state to Node)
     else apply all applicable operators to Node,
  and
          merge the newly generated states into
  L
```

Now let us look at the description of the basic search algorithm. Let us look at the basic idea of a search algorithm. Let l be a list containing the initial state. So l is called the fringe. So l is a list containing the initial state and l in the intermediate state will contain the set of states we have generated so far and that we need to expand. Then the search algorithm executes in a loop. If l is empty that is if the fringe is empty return failure. Otherwise we select the first node from the fringe.

Select is a function which selects a node from l. It may be the node with the minimum cost. If this node happens to be a goal then we return the node or return the path from the initial state to this node. So if this node is a goal then we return the node that is return the path from initial state to node. Otherwise we apply all applicable operators to node to generate all the successors of node and these successors we merge into l. So l is l union the successors of node.

Now, as we will see, the various search algorithms differ as to how we position the different successors into the fringe and consequently which node we select for expansion from the nodes in the fringe.

(Refer Slide Time: 48:47)



Basic search algorithm: Key issues

- Search tree may be unbounded
  - Because of loops
  - Because state space is infinite
- Return a path or a node ?
- How are merge and select done ?
  - Is the graph weighted or unweighted ?
  - How much is known about the quality of intermediate states ?
  - Is the aim to find a minimal cost path or any path as soon as possible ?

Now let us discuss the key issues about search. In state space search we start from an initial state and we expand this state to get other possible states. Now these states we get constitute go into the current fringe. We select one of the nodes of the current fringe for expansion. So this node goes out of the fringe and its successors are placed on the fringe. That is, if we look at this search tree the fringe constitutes those nodes which are at the leaf of the search tree. So here there are five nodes in the fringe. The search strategy will determine which of these five nodes I will expand next.

Suppose we decide to expand this node then we generate the successors and place this in the fringe. We decide which of these six nodes we will expand.

Now one thing we will notice is that, if we model out search problem like this the search tree may become unbounded because of loops in the state space representation. Suppose there is an edge, there is an operator which takes you from a to b, b to c and c to a and now if we say that this is a, this is b, and this is c then the successors of c will contain a and then again under a we will have this entire tree. So, in effect even if the number of states is finite the search tree may become unbounded if we unfold it like this tree. So a finite state space may become unbounded because of presence of loops and in some cases we will see the state space can itself be infinite.

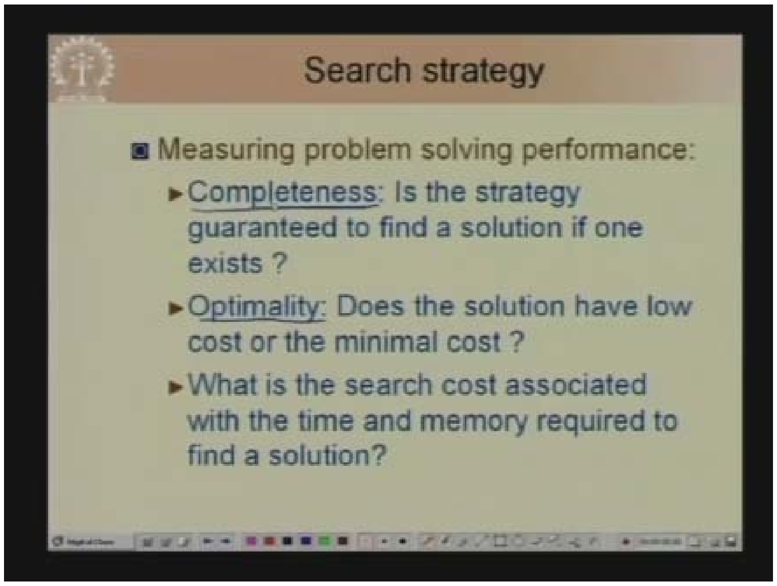
When we talk about a search algorithm we have to handle these issues. Another issue we need to handle is, whether depending upon the problem whether the search problem returns the path or whether it returns a node. For example, in the 8 puzzle problem the solution to the search problem is a set of operations whether it is up, down, up, left, right, left, left, left, down. It is a sequence of operations which takes us from the initial state to the goal state. However, for problems like the 8 puzzle we really do not need to spell out the operators. If we just give the goal configuration the operators are evident. If you say that this is an initial state and this is the goal state then such a thing is sufficient.

Given the goal state we can find the sequence of operators very easily which takes us to this goal state. Therefore depending on the problem definition we will either decide to return a path or a node. A very important aspect for the design of a search problem is to discuss how merge and select are to be done. That is, when we put nodes in the fringe, when you put the successors of the current generated node in the fringe where should we put the successors? Is it in the beginning or in the end?
Hence we should know how to put the nodes on the fringe and also how the select program is carried on.

We should know whether to select the first node, select the cheapest node and so on. And to consider these things we need to consider certain factors. Is the graph weighted or un weighted?
If all steps have same cost we can model the state space as a un weighted graph. But if different operators have different costs we can associate weights with the edges corresponding to the cost of that operator and we will deal with weighted graphs. We must also find out how much is known about the quality of the intermediate states.

Do we have any information or heuristics to guide the search process?
So, if we know that out of the nodes available in the fringe, there are 1 2 3 4 5 6 7 8 so 8 nodes available in the fringe, if we know that this node has very high promise and it is likely to take us to a goal solution fast we might use that knowledge to expand this node before expanding the other nodes. So, if we have some information about the quality of these intermediate states we can have better search strategies so that we can find a solution fast. And also we need to consider whether our objective is to find a minimal cost path or any path as soon as possible. When we discuss problem solving strategies we will evaluate each problem solving strategy by the following parameters whether it is complete, whether it is optimal and what is the cost associated.

(Refer Slide Time: 55:08)

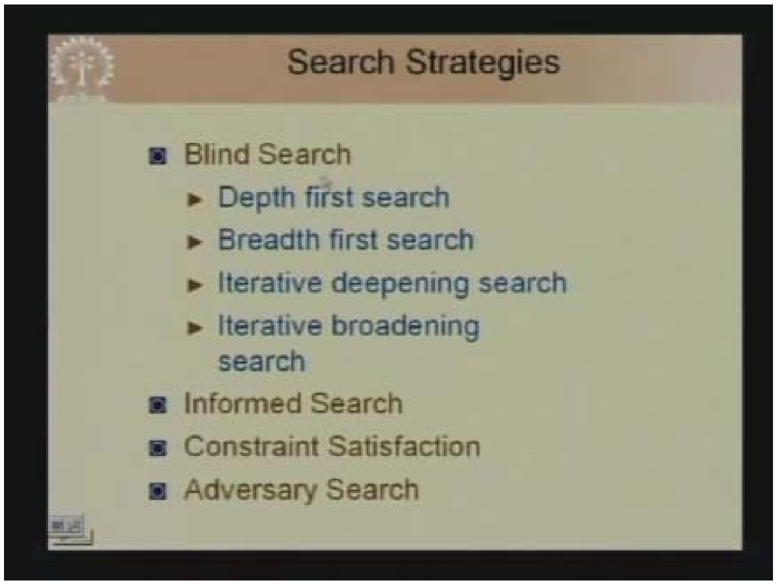


Search strategy

- Measuring problem solving performance:
  - Completeness: Is the strategy guaranteed to find a solution if one exists ?
  - Optimality: Does the solution have low cost or the minimal cost ?
  - What is the search cost associated with the time and memory required to find a solution?

A strategy is complete, if the strategy is guaranteed to find a solution to the search problem if a solution exists. If a solution exists, the strategy must guarantee that one solution will be found. A strategy is optimum if it has the least cost. The search cost associated with the strategy is the time and memory resources required to find a solution. The time and space complexity of the algorithm is a very important thing that we need to keep in mind and that we need to measure. The various search strategies that we will study are blind search without heuristics. In those we have depth first search, breadth first search, iterative deepening search, iterative broadening search etc.

(Refer Slide Time: 56:04)

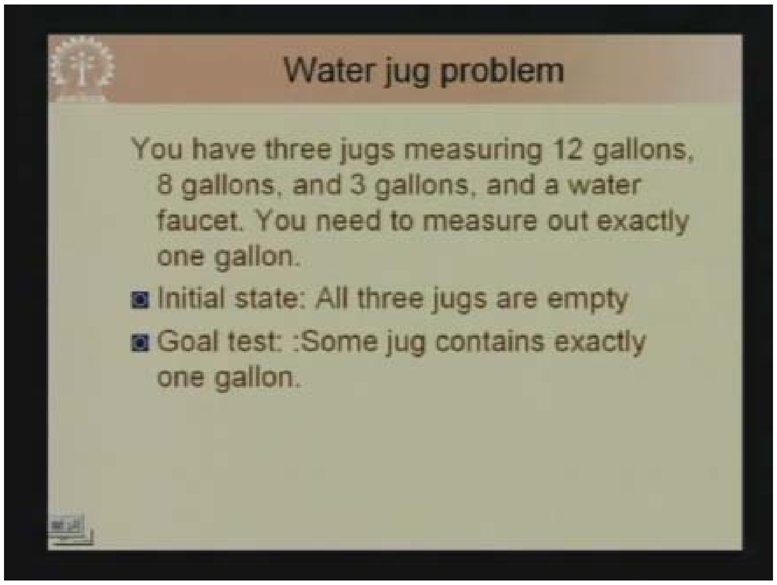


Search Strategies

- Blind Search
  - Depth first search
  - Breadth first search
  - Iterative deepening search
  - Iterative broadening search
- Informed Search
- Constraint Satisfaction
- Adversary Search

We have informed search that makes use of heuristic information. We will also study constraint satisfaction problem and we will also look at adversary search involving two person games.

(Refer Slide Time: 56:33)



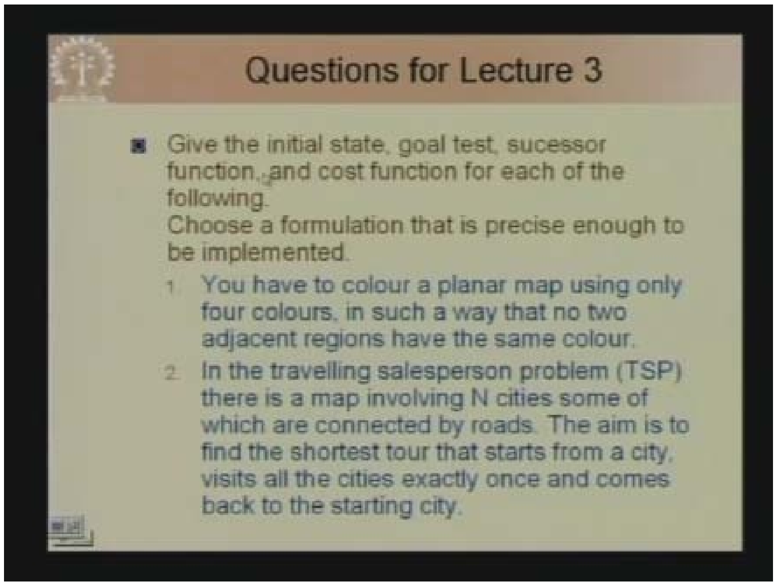
Water jug problem

You have three jugs measuring 12 gallons, 8 gallons, and 3 gallons, and a water faucet. You need to measure out exactly one gallon.

- Initial state: All three jugs are empty
- Goal test: :Some jug contains exactly one gallon.

Here is a problem to think of: You have three jugs measuring 12 gallons, 8 gallons and 3 gallons and you have a water faucet or a pipe from which you can get water. And your objective is to check out exactly one gallon of water. In the beginning you have three empty jugs and you objective is to get one gallon of water in any one of the jugs. Think of how you can formulate this as a state space problem, how you can represent a state and how you can get a solution to this problem. So think about this problem and you will see that representation is extremely important for search problems.

(Refer Slide Time: 57:34)



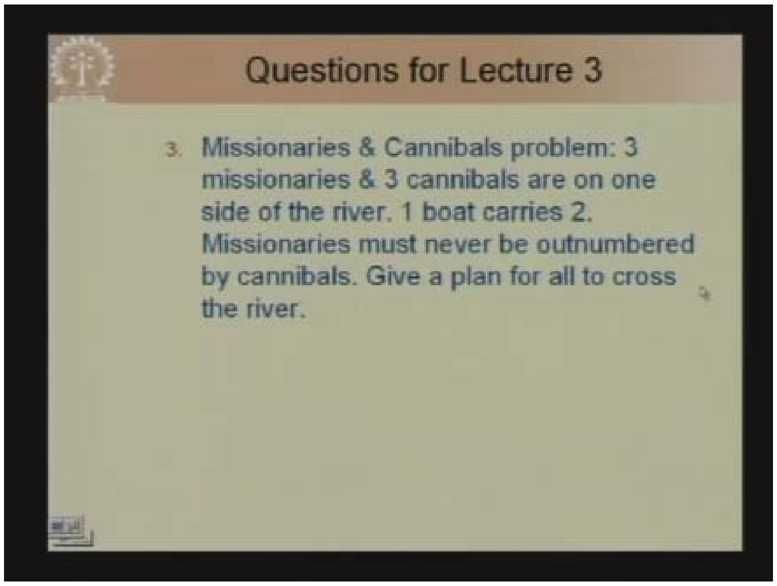
Questions for Lecture 3

- Give the initial state, goal test, sucessor function, and cost function for each of the following.
  Choose a formulation that is precise enough to be implemented.
  1. You have to colour a planar map using only four colours, in such a way that no two adjacent regions have the same colour.
  2. In the travelling salesperson problem (TSP) there is a map involving N cities some of which are connected by roads. The aim is to find the shortest tour that starts from a city, visits all the cities exactly once and comes back to the starting city.

I will describe a few problems. For each of these problems you have to represent them as a State Space Search problem and specify the initial state, the goal test, the successor function and the cost function. You must choose a formulation. You must also of course give the state space formulation and discuss how a state is represented. You must choose a formulation that is precise enough to be implemented.

First problem is you have to color a planar map using only four colors in such a way that no two adjacent regions have the same color. Second problem, the traveling salesperson problem, there is a map involving n cities some of which are connected by roads. The aim is to find the shortest tour that starts from a city visits all the cities exactly once and comes back to the starting city.

(Refer Slide Time: 58:35)



Questions for Lecture 3

3. Missionaries & Cannibals problem: 3 missionaries & 3 cannibals are on one side of the river. 1 boat carries 2. Missionaries must never be outnumbered by cannibals. Give a plan for all to cross the river.

Problem three is the missionaries and cannibals problem. There are three missionaries and three cannibals on one side of the river. There is one boat which carries two at a time. The missionaries must never be outnumbered by cannibals. Give a plan for all to cross the river.