Artificial Intelligence: Search Methods for Problem Solving

Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Chapter – 7 and 10
A First Course in Artificial Intelligence
Lecture - 58

Automated Domain Independent Planning

So, welcome to a new topic and a new module in our course here. And we are going to spend the next week or so, looking at this area of planning or as we sometimes say Domain Independent Planning. And essentially planning is a task of something that we have addressed earlier that, we had classified search problems as two kinds; one was the planning problem and the other was a configuration problem.

So, in a way we are coming back to the planning problem. And we are going to focus only on planning problems and study what is been happening in this field. So, as I said in the last 20 odd years a whole community has emerged to work on planning, and they have their own conference they have their own competitions and a lot of work has been going on in this area essentially.

Partly this work has been motivated by the fact that people have started building autonomous systems. Systems which will take decisions on behalf of their owners or users, they will perceive the environment, they will have goals to achieve and they would have to decide by themselves as to what are the actions that they need to pick to achieve those goals.

So, this is broadly speaking what happens in the world of planning, there have been some very impressive examples where planning has been used and one of them has been in the area of space exploration. So, quite a long time ago there was a experiment called deep space one, in which they deliberately introduced some faults in a spacecraft. And then they wanted to see whether the spacecraft can monitor detect and plan its activity in this in the light of this deliberately introduced defect.

Another example from the space domain has been space exploration. So, for example, NASA had sent mars rovers about 15 odd years ago and the distance from mars to earth is so, significant that it takes light which would be the carrier for the signal to many minutes to reach the earth. Not only that so that means; obviously, you cannot remote control a robot from earth, because by the time you send the signal say turn left the robot would have gone straight and fallen over a cliff or something like that.

So, clearly it needs to be autonomous and not only, because of the time lag between earth and the mars that the time it takes for the signal to receive. But, also because at times a spacecraft might be out of signal zone, it may be on the other side of mars and you may not be able to give a signal or receive a signal.

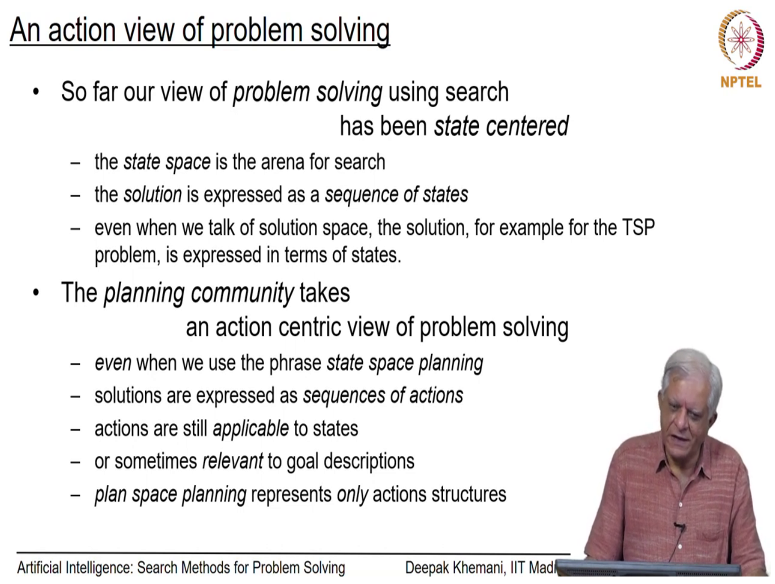
There have been similar applications closer to earth specifically I remember for example, Indian space research organisation was looking at the possibility of introducing autonomy into its low earth satellites. So, satellites which are visible to a ground station only for 20 minutes in a day or maybe a bit more than that. And otherwise you know they are orbiting the earth and you cannot communicate with them.

Of course you can communicate with them, if you have ground stations in other places. So, India has had ground stations in other countries, but the lure of having autonomous satellites is quite there. And especially if you want to you know do things like remote sensing and all you have to manage, when to take images when to upload images when to download images whatever the case may be and planning again has been useful in that.

Nowadays of course, you know that autonomous vehicles do a fair amount of planning, they do mostly road planning, but nevertheless its a planning activity they also have to plan when to accelerate, when to decelerate, when to stop all that is part of deciding what actions to do. And obviously, there is a world of robotics in which you want to have autonomous robots which are doing things for you. So, they amongst other things will need to be able to plan.

So, for these many reasons planning has emerged as a active area of research and what we are going to do is to look at the basics in this next week or maybe 2 weeks. And look at what are the kind of algorithms that have been developed.

(Refer Slide Time: 05:38)



An action view of problem solving

- So far our view of *problem solving* using search has been *state centered*
  - the *state space* is the arena for search
  - the *solution* is expressed as a *sequence of states*
  - even when we talk of solution space, the solution, for example for the TSP problem, is expressed in terms of states.
- The *planning community* takes an action centric view of problem solving
  - *even* when we use the phrase *state space planning*
  - solutions are expressed as *sequences of actions*
  - actions are still *applicable* to states
  - or sometimes *relevant* to goal descriptions
  - *plan space planning* represents *only* actions structures

Artificial Intelligence: Search Methods for Problem Solving   Deepak Khemani, IIT Mad

So, one thing that we will start with is that, we will change our perspective of problem solving. So, far what we have said is that our perspective of problem solving using search has been state centred. So, we have said that we have a set of states and you have a move gen which allows you to move from one state to another.

And we have been deliberately vague about what this move gen function looks like; we have also been vague about how the representation of states has done. So, our focus has been on the fact that given that there is a move gen function and given that, there is a space to search what are the best strategies for searching through that space.

So, in this state centric approach of things, the solution that we found also was a sequence of states. So, we said that the path of formed by an algorithm for example, like a star is the sequence of nodes that it goes through essentially even, when we talked about solution space searched for example, when we talked about TSP.

The solution was still expressed in terms of states for example, we said that this is a candidate tour and you can alter this tour or you can extend this tour partial tour and things like that and everything was centred around states.

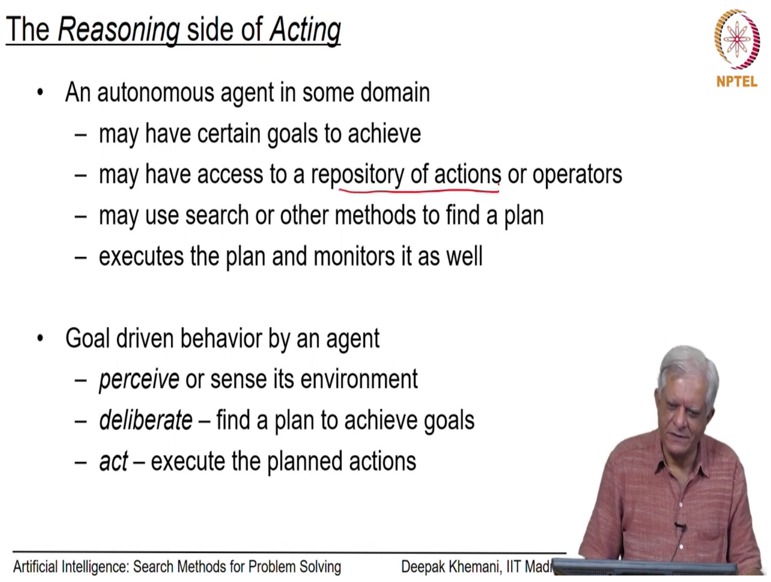
The planning community takes a different viewpoint; they say that we want to talk about actions. And we want to look at an action centric view of problem solving, even when we plan in the state space we are going to talk about actions. So, solutions in the planning community are expressed as sequences of actions rather than a sequences of states.

Obviously actions will be applicable in some states. So, we will have to consider as to when they can be applied. And they will be relevant to goal states because, our task is to achieve some kind of a goal state or a goal description. And they we will also look at a little bit an area of which is called plan space planning or partial order planning, in which we do not even represent states. The space that we search through is entirely that of plans essentially.

Now, in our study so far we have on and off introduced the notion of actions. In particular if you remember, we when we talked about the 8 puzzle kind of a problem. We said that actions can be named as left right down or up which we called as LDRU. And we kind of talked a little bit about it at that this is that you have to choose between those actions.

But that was only in some sense adding to the basic search mechanism which was state based. Essentially that you had a move gen function, which gave you a set of neighbouring states and you have to choose one of them. Now, we want to consider a view point, where we take an action centric view. And in even if we are talking of state space planning as we will start by doing, we will still consider that in a given state what are the actions that we can apply and then choose between those actions.

(Refer Slide Time: 08:56)



The *Reasoning* side of *Acting*

- An autonomous agent in some domain
  - may have certain goals to achieve
  - may have access to a repository of actions or operators
  - may use search or other methods to find a plan
  - executes the plan and monitors it as well
- Goal driven behavior by an agent
  - *perceive* or sense its environment
  - *deliberate* – find a plan to achieve goals
  - *act* – execute the planned actions

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Mad

So, let us get on with this thing there have been a couple of books by Malik Ghallab and Dana Nau and traver so for example, they have written a very popular book on automated planning. And more recently they have come up with a book which is called planning and acting, in which they focus also on how to monitor the actions that you are doing and what do you expect of actions and how do you verify whether those expectations are met or not.

And they gave us this kind of a definition or a slogan you might say, that planning is the reasoning side of acting that an autonomous agent has to act in some world. But, before acting the autonomous agent has to deliver it essentially decide as to what are the actions that it should do. This is different from what we can call as reactive systems, which simply sense and react in some predetermined kind of a way.

When, we talk about planning we have goals in mind and we have an explicit process of reasoning or deliberation that we expect the agents to do. So, an autonomous agent in some domain would have certain goals and that is either given to it by some kind of a user or a manager or it may derive goals of its own, if it is a truly autonomous agent.

Then, its planning has to be done in terms of a repository of actions or operators that are accessible to it. So, a planner will have certain set of actions available to it and we will see how they are described, it could be any method in particular we will focus on search for the moment.

But in general it could be other methods as well and the task would be to find something called a plan. So, a plan would be like a first class object in programming terms, something which can be passed around and manipulated and things like that. And; obviously, the agent will have to execute and monitor the plan as well.

Now, this goal driven behaviour of agent in the sense that the agent has certain goals to achieve and is deliberating how to achieve, those goals is can be seen as a cycle of three steps. First you perceive what is the environment what is true in the environment and in terms of what is the given state and so on?

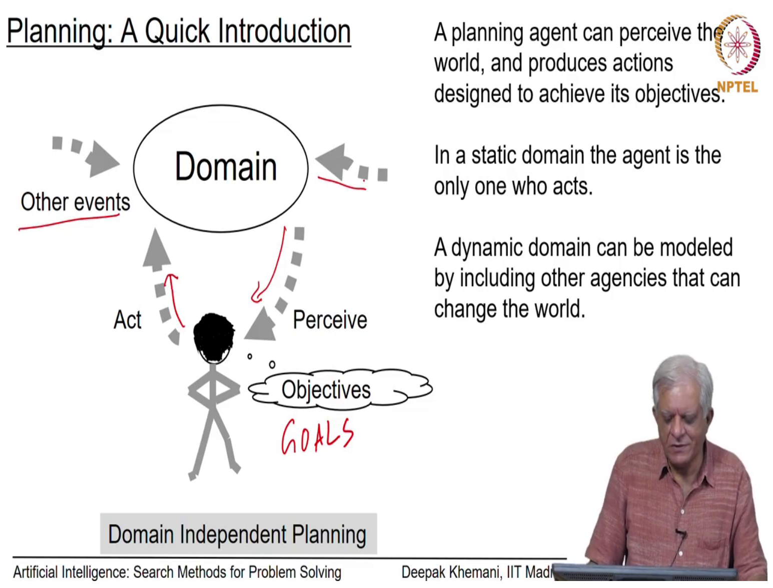
Then, you deliberate as to how do you do something that would move you towards closer to your goal or. And the third is that having chosen the action you need to actually act in the world, which means you will actually change the environment you will for example, move a chair around or whatever put some clothes out to dry in the sun. Whatever the actions are you have to actually do them.

Then again see what the new world is like and what you need to do in terms of the original goals that you had and again continue deliberation. So, it may happen that deliberation may happen once and for all, and then you simply execute the whole plan, but it could also happen in some situations. Where the world is not entirely in your control and you have to keep monitoring and perhaps replanning as you go along essentially.

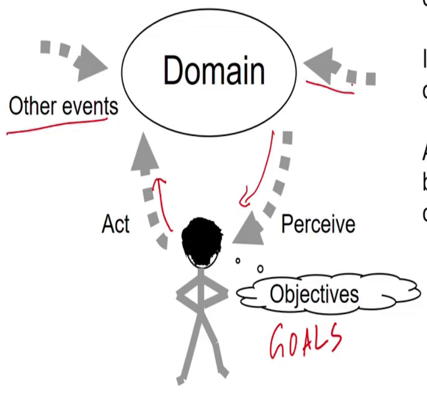
For example, if you are planning a trip to the mountains and you are going along a particular path and or example in Himachal and then, you see that this road that you wanted to take has had a landslide and you there may be a delay of 4 hours or something like that.

Then you have to re plan should you wait there for 4 hours and maybe have a meal or something, or should you backtrack and try a different path. So, in the real world everything is not in your control and you may have to monitor and maybe re plan and so on. But, the basic cycle of a planner is perceive deliberate act or sense deliberate act essentially.

(Refer Slide Time: 13:12)



Planning: A Quick Introduction



A planning agent can perceive the world, and produces actions designed to achieve its objectives.

In a static domain the agent is the only one who acts.

A dynamic domain can be modeled by including other agencies that can change the world.

Domain Independent Planning

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Mad

So, we can model we can think of planning as follows, our view of planning is going to be domain independent ok. In the sense that we will not say that we are writing a programme to solve for example, the 8 puzzle of the Rubik's cube or let us say the missionaries and cannibals problem or whatever the case may be.

In the spirit of what we have been doing in AI that we will write general purpose algorithms and we will expect that somebody will give us the what we said. So, far was that somebody will give us a move gen function and somebody would give us a goal test function and maybe somebody would give us a heuristic function.

Now, we are saying or we are going to say as you will see that we have a well defined way of describing a domain. And we would expect a user to describe the domain in the language that we will specify. These languages are called planning domain description languages and we will kind of have a passing look at them as we go along.

But, the idea of planning the research in planning is largely domain independent and it says that given a set of planning operators accessible to agent, described in the language that we are talking about. Given a start state which is also described in the language that we are going to soon discuss and a goal description, also expressed in the same language. What is the sequence of actions that I would construct to achieve the goal that has been given to us?

Now, you can see that there is a domain here and there is an agent, which is kind of anthropomorphised here. Looking like a human being and this agent is the one which is acting in the world. The agent can perceive what is happening in the domain? And the agent can act upon the domain. So, this perceive and act are the two parts that, we have been talking about we said perceive deliberate and act. Deliberation we have not shown here and that is what we are going to study essentially.

Now, I have drawn in this figure the agent as somehow being outside the domain, but that is only for the case of a schematic diagram. Now, in practice of course, the agent would be in the domain for example, if there is a robot moving around things on the shop floor. Then the robot would be in the domain itself essentially, but it is just for the case of drawing a neat diagram essentially.

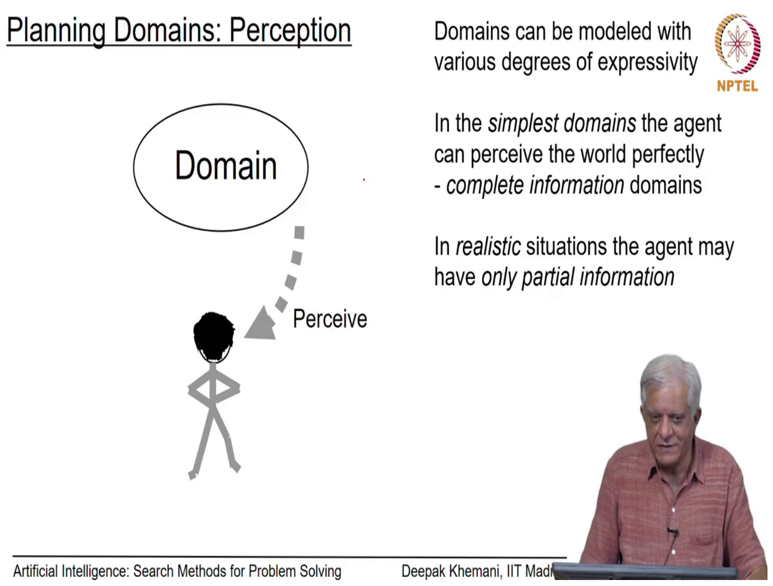
The agent of course, will have a objectives or goals to achieve and deliberation is oriented towards achieving those goals essentially. Either the agent is acting solo by itself or there may be other events which are happening essentially. So, other events could be for example, they may be other agents also or it could be events happening in nature or what is sometimes called as exogenous events. For example, rain may suddenly fall or something like that and that may also change the domain.

So, the domain is manipulated either by the agent or by other agencies, but for our case we will confine ourselves to the case where, the agent is the master of the world and the only one who can make changes in the domain. So, that those are the simplest kind of planning domains.

So, this is what planning is about and a plan can perceive the world. And it produces actions designed to achieve its objectives or goals. Then, in a static domain the agent is only one who acts, there is nobody else which can change the world. And in a dynamic domain, we can have other agencies like other agents or nature for example or for example, market rules as soon as the market opens let us say, at 10 in the morning and closes at 7 in the evening or whatever the rules of that particular city might be.

If you want to go shopping somewhere you have to be aware of the fact, when will the shop be open essentially and thus whether the shop is open or closed is not within your control, but somebody else or some other agencies doing that.

(Refer Slide Time: 17:44)



Planning Domains: Perception

Domain
Perceive

Domains can be modeled with various degrees of expressivity

In the *simplest domains* the agent can perceive the world perfectly - *complete information* domains

In *realistic* situations the agent may have *only partial information*

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Mad

Let us look at these different aspects one by one, first is the perception. We have said that an agent can perceive the world essentially. Now, perception may or may not be complete it can be modelled domains in general can be modelled with various degrees of expressivity. And one of the parameters is going to be how much can the agent see the world essentially.

Now, for example, if you treat let us say playing a game of chess like a planning problem, then from the planning perspective that you are deciding your moves, you can see the entire chessboard. And you can see where the pieces are what are the moves possible for you, what are the moves possible for your opponent and everything is known to you. And that is why if you recall such a situation as a complete information game essentially.

Likewise a domain may be complete information or it may be with partial information. So, in the simplest domains the agent can perceive the world perfectly essentially, but of course, such domains are rare, but they are good for the purpose of study.

And it has been shown by a group of researchers Guptha and Nau more than 20 years ago. That even in the simplest of domains that we are going to be talking about planning is the planning problem is a hard problem; it is a problem which is called in this in the class of complexity called p space which basically says that it may take exponential time and polynomial space essentially.

So, its a hard problem to solve and that is why we would begin our baby steps with the simplest of domains. And in the simplest of domains the agent can see the world perfectly. Now; obviously, in the real world you cannot do that if you for example, are staying in your college campus or in your home in a city and you want to let us say go to the train station. Your plan may be that you would go out and catch a bus at some point to the to the station essentially.

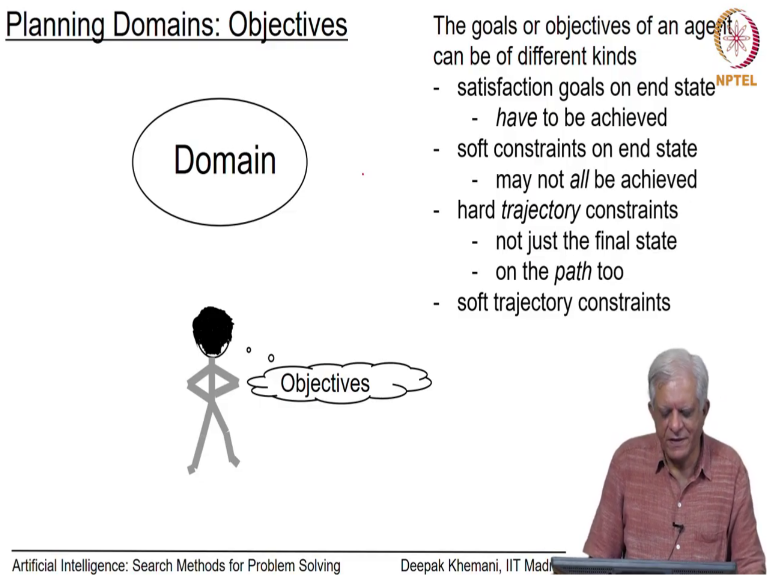
But of course, you do not know what is happening in the world out there? For example, you do not know whether its raining or not you do not know whether there are traffic jams, you do not know whether there is a bus strike all kinds of things could happen essentially. And yet you have to somehow plan in the face of this uncertainty, but that is a little bit too much for us at this moment.

And we will assume that the domains that we are working on are simplest. And in which the agent can see the world perfectly the agent, knows exactly what is true in the world and what is not true in the world. So, such domains are called complete information domains.

As I said in realistic situations the agent may have only partial information, you may not know whether the particular for example, you may be going to a movie theatre. And you may not know whether the tickets are available or not of course, in the modern age you can always do all this stuff online. But, assuming that you are in a small town in the remote part of the country and that particular theatre owner has not created an online system. Then you have to work with the partial information you have to go to the theatre and hope that you will get a ticket essentially.

Or if you are travelling to let us say train in a travelling to a train station in a bus or an auto, you do not know whether there is going to be a traffic congestion and whether you will reach on time and things like that. And again you have to work with partial information, but we as I have been saying we will stick to simple domains.

(Refer Slide Time: 21:39)


Planning Domains: Objectives

The goals or objectives of an agent can be of different kinds
- satisfaction goals on end state
  - *have* to be achieved
- soft constraints on end state
  - may not *all* be achieved
- hard *trajectory* constraints
  - not just the final state
  - on the *path* too
- soft trajectory constraints

Now, what about the goals of the agent or goals of the objective? What kind of goals that we will give to our agents to achieve? They can also be vary in degrees of expressivity the simplest and that is what we will assume, is satisfaction goals on the end state. By this we mean that we will specify certain conditions and which should be true in the end state. And all of those conditions have to be achieved for you to say that the plan has worked or the plan is successful or the plan is a valid plan.

So, I might I might say that you know this should be true this should be true and this should be true. For example, I should have bought a toothpaste and I should have bought a book and I should have eaten in the restaurant by the time I finished my activity, then all of those should have been done essentially.

On the other hand you may have soft constraints on the end states we say ok. These are the things that you want to do you want to buy a toothpaste, you want to buy a book you want to eat in a restaurant, you want to take a stroll on the beach. And you would like to time allows you to go and buy some other groceries that you need for your home, you may have any of these goals.

And yet all of those goals may not be feasible, time may be limited jobs may be open for only a short period of time. In such situations we can model the planning problem as problem of soft constraints, in which we say that achieve as many goals as you can. And then we evaluate the plan based on how many goals have been achieved.

So, for every example that you cannot achieve you can impose a penalty on the plan. And then the task becomes a task of optimization to say that I want a plan, which has the least number of penalties essentially. In the previous case where the goals are only satisfaction goals, the task is a satisfaction task, either the goals have been achieved or they have not been achieved. So, either the plan is successful or its not successful.

Whereas for soft constraints you can have a degrees of success and say that ok. The plan is completely successful or partially successful and things like that. But then; obviously, optimization is a harder problem than satisfaction and algorithms, which try to plan with soft constraints are more complex than that.

Then why should we have only constraints only on the end state we may have constraints on the trajectory also. So, you might say for example, in Chennai if you are going let us say from

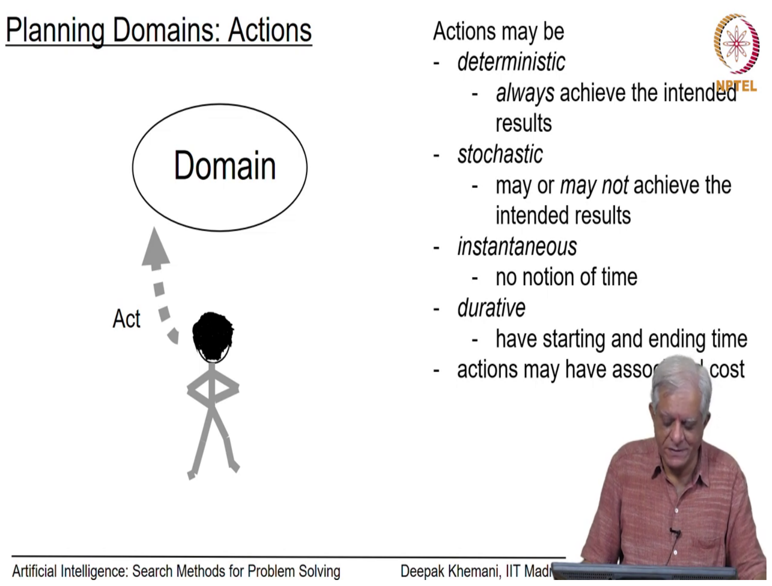
IIT Madras to the central station. You might say I want to go along a path from which I can get a good view of the sea essentially. Now, this is not part of the end goal, end goal is you have to be at the central station maybe at a particular point of time.

The trajectory constraint says that I would like a trajectory to be of a certain nature essentially or for example, if you are transporting a patient from one city to another. Then you may have trajectory constraints that choose a path in which at all points of time, I am within 20 kilometres of a hospital or I am within maybe 20 kilometres of a petrol pump essentially.

So, that you are sure that if you have to revise your plan, you would be in a situation where you can revise it. Because, suddenly there is an emergency and you have to take the patient into an emergency ward, there must be a hospital to start with essentially ok.

So, you may impose a constraint that find me a path from city a to b, but in such a way that I am never too far from a hospital essentially. So, we have constraints on the path to and these are called as trajectory constraints. Again trajectory constraints may be hard or soft that you may prefer that certain things are met, but if they are not met, then you may be willing to live with it essentially.

(Refer Slide Time: 26:04)



Planning Domains: Actions

Actions may be
- *deterministic*
  - *always* achieve the intended results
- *stochastic*
  - may or *may not* achieve the intended results
- *instantaneous*
  - no notion of time
- *durative*
  - have starting and ending time
- actions may have asso l cost

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Mad

Now, let us talk about what the agent can do what are the kinds of actions that agents can do and they there is a whole variety of them that, we can think of the simplest is that actions are always deterministic that they always achieve the intended result essentially.

So, for example, if I have a pen lying on the table and I plan to say pick up the pen like this, then when I execute the plan I am holding the pen. So, it has achieved the intended consequence of the action. On the other hand there may be actions, which are stochastic which may or may not achieve the intended results essentially.

So, for example, if I for example were to go and try to play basketball and then I would say here is an action I am doing and throwing the ball into the basket. And I may chuck the ball towards the basket, but as you can probably guess, the ball may or may not go into the basket.

So, such actions are called stochastic actions, where you intend to do something, but that may or may not happen essentially.

So, it could happen for example, if you are trekking in the mountains and you are walking on a glazier, which is a kind of inclined surface of hard snow. You may want to say that I am going to take five steps in the forward direction, but you might find that yourself that, you have slipped down and fallen a little bit on the wayside essentially

So, so the world of course, is always full of stochastic actions, we will assume that actions are deterministic that. If you say pick up a pen, then at the end of that action the pen would be picked up, if you say take five take five steps in the direction of the door. Then by the time those five steps have been taken, you are that much closer to the door.

That actions can have duration normally in the real world actions are never instantaneous in the real world nothing is instantaneous. Because, there is energy involved, but in the simplest of domains we will assume that actions are instantaneous and there is no notion of time essentially, which means that if I say pick up a pen it happens at time t 1 essentially.

We can model time as a discrete sequence at t 1 I can pick up the pen at t 2, I can give it to a friend at t 3 the friend can start writing with the pen. All these things are modelled as happening instantaneously. And the goal is to find the correct sequence of actions we are interested in the sequence of actions, we are not talking about time at all even though in the real world we may often do that essentially.

So, again if you are doing root finding for example, you might say go from location a to location b, then location b to location c and. so on without worrying about time. In which case you are simply trying to find a path essentially, when we talk about optimal paths its a different matter.

On the other hand actions may be durative; they may have a starting time or an ending time. So, for example, if you want to make a cup of tea you might say take a pan fill it with water keep it on the stove, light the stove, wait for the water to start boiling, add the tea leaves, close it for a few minutes. And then strain it out each of these actions has duration.

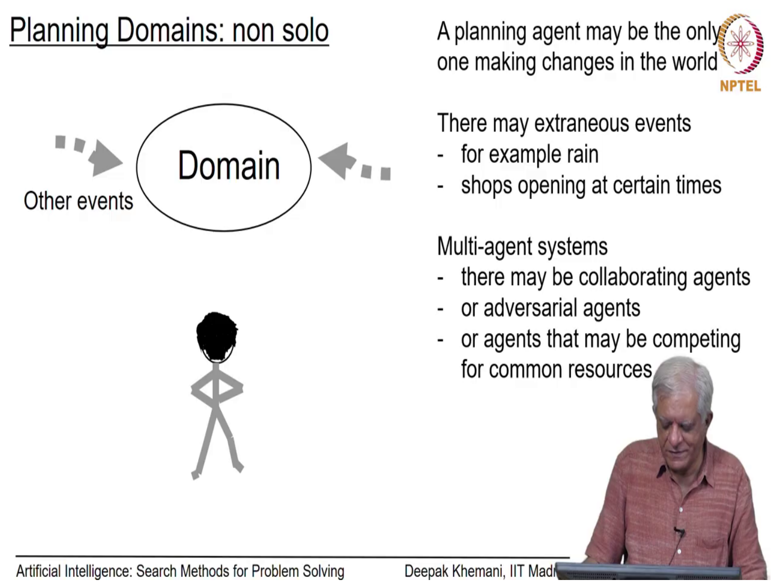
Its not as if you can boil the water in one instant or something like that it takes a certain amount of time. And there have been domain descriptions, where people have worked with durative actions, which take time essentially. That is particularly important, if you are talking about situations where you can do many actions in parallel.

So, typically if you see the way cooking happens in an Indian household, people have gas stoves most people have gas stoves and they have 2 or 3 burners on them. And something is going on 1 burner maybe some dhal is being cooked and on the other burner, you may be stir frying some vegetable, and these things happen in concurrently. But, it may take the each may take a different amount of time.

So, for example, the dhal may take 20 minutes or 30 minutes to cook and if you are stir frying, then you may be in the last stages and you may finish in 5 minutes. And then you may start making chapatis and each chapati may take 2 or 3 minutes to make. So, durations count essentially. If you want to finally, say that I want to prepare my meal as quickly as possible and I am allowed to do concurrent actions, then you know planning becomes a much harder problem. We will stick to deterministic instantaneous actions in our study here.

Finally actions may have associated cost and then, you may worry about. What is the cost of the plan that you have found? We may start off or we will rather confine ourselves only to the simplest of cost, which is a number of actions that you have to do and we would say that shortest plan is the one in which the number of actions is the smallest.

(Refer Slide Time: 31:32)



Planning Domains: non solo

A planning agent may be the only one making changes in the world

There may extraneous events
- for example rain
- shops opening at certain times

Multi-agent systems
- there may be collaborating agents
- or adversarial agents
- or agents that may be competing for common resources

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madr

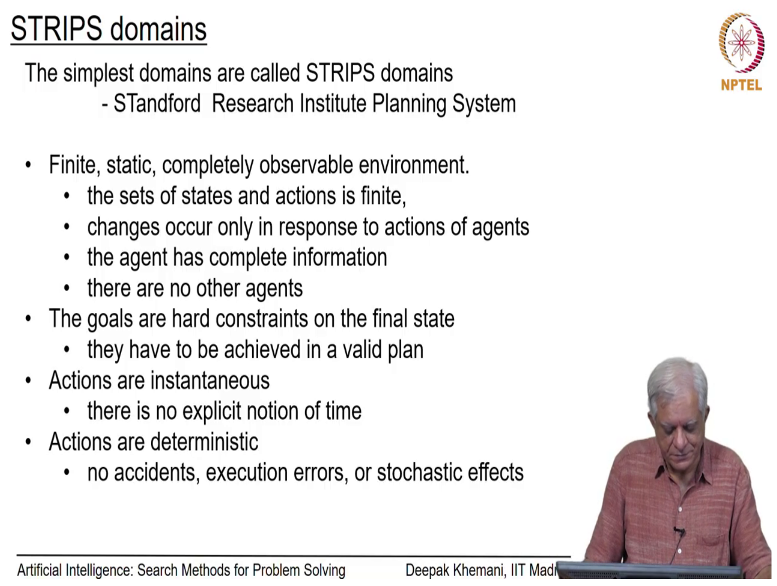
So, I said that in static domains agent is only one whose acting and so, you can entirely predict what the domain will be like after you have done let us say 6 actions or 10 actions. But, in dynamic domain that may not be the case, in the static domain the planning agent is the only one that makes changes in a dynamic domain, there maybe extraneous events for example, they may be rain or the shop may be open or shop may be closed and things like that essentially.

There maybe other agents so, we often model agents and nature in some sense in a different way. The one could think of nature as some huge agent. But, when I talk about agents with in some sense say that you know let us say there are 2 robots and 3 robots or whatever two people or three people. And if you have to move for example, a cot from one room to another, then the two robots have to work in a coordinated fashion. So, both of them should plan their action. So, that both of them lift the cot at the same time.

All these kind of situations can happen in multi agent situations. So, you may have agents which are collaborate, collaborating agents or they may be agents who are adversarial agents who are trying to stop you from achieving your goals. Or they may be agents who are competing with you for common resources essentially.

So, for example, you are standing in line to buy a ticket and there may be other agents also standing in line to buy tickets for themselves. And of course, as happens in India lines are never well formed. So, they are competing and you know you may have to kind of be careful not to let them barge into your line all that kind of stuff essentially.

(Refer Slide Time: 33:19)



STRIPS domains

The simplest domains are called STRIPS domains
- STandford Research Institute Planning System

- Finite, static, completely observable environment.
  - the sets of states and actions is finite,
  - changes occur only in response to actions of agents
  - the agent has complete information
  - there are no other agents
- The goals are hard constraints on the final state
  - they have to be achieved in a valid plan
- Actions are instantaneous
  - there is no explicit notion of time
- Actions are deterministic
  - no accidents, execution errors, or stochastic effects

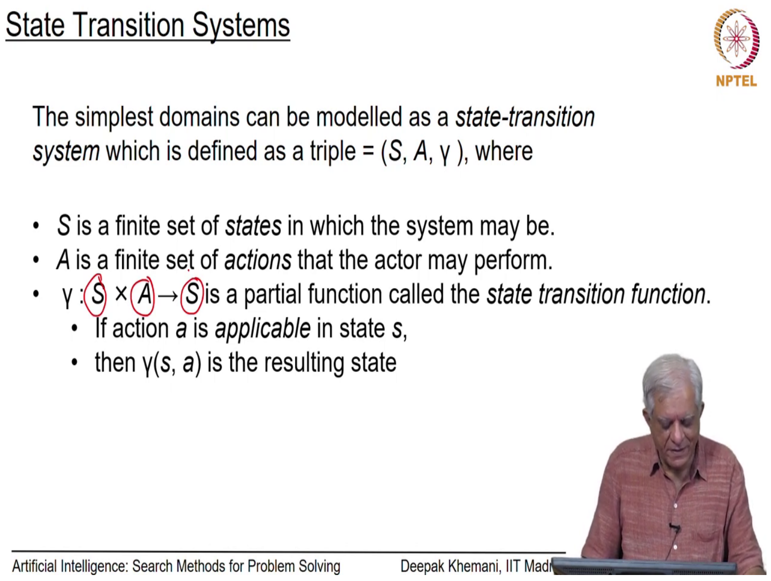
Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Mad

The simplest kind of domains, that we are going to be dealing with are called STRIPS domains. And these are the oldest planning formal systems that were devised. And, many people say that STRIPS stands for Stanford Research Institute Planning System. And it was a planning system, which was devised for one of the earliest robots called Shakey, which apparently roamed the corridors of Stanford at that time.

And the STRIPS domains are the simplest their actions are instantaneous and all these things that we said the domain is finite static completely observable environment. The set of states and the set of actions is finite, changes occur only in response to the actions of the agent they are static. Agent has complete information you can completely see the world and there are no other agents.

Then the goals are hard constraints on the final state we do not bother about what happens during the plan. And they have to be achieved to form a valid plan actions are instantaneous, there is no notion of time. And, we just want to find a sequence of actions that will achieve the goal, actions are deterministic in the sense that what we plan will happen essentially. No accidents, no execution errors and no stochastic effects. We are not talking about for example; rolling a dice in a game of backgammon or snakes and ladders and hoping that you will get a five or something like that you know may or may not happen.

(Refer Slide Time: 34:56)



State Transition Systems

The simplest domains can be modelled as a *state-transition system* which is defined as a triple = (S, A, γ), where

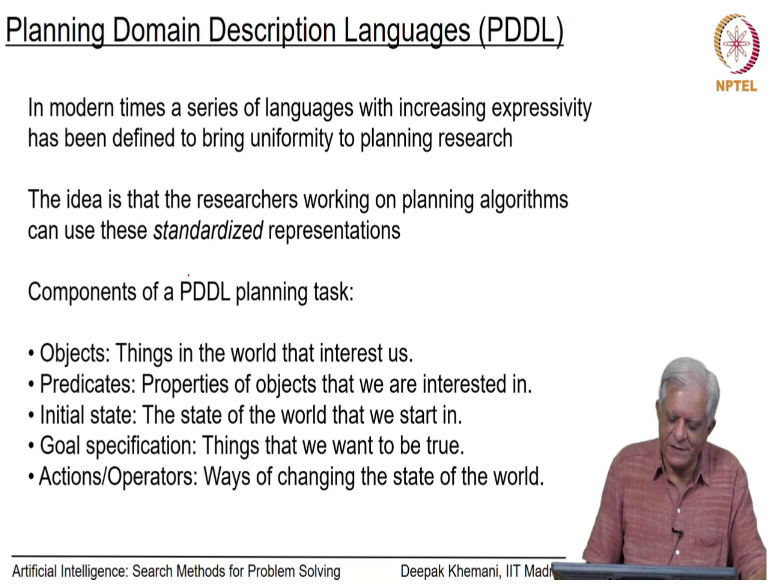
- S is a finite set of *states* in which the system may be.
- A is a finite set of *actions* that the actor may perform.
- $\gamma : \bigcirc\!\!\!\!S \times \bigcirc\!\!\!\!A \rightarrow \bigcirc\!\!\!\!S$ is a partial function called the *state transition function*.
    - If action a is *applicable* in state s,
    - then $\gamma(s, a)$ is the resulting state

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Mad

Such simple systems can be modelled as state transition systems, which can be defined as a triple ah of set of states a set of actions and a transition function essentially. So, S is a finite set of states which the systems may be and the system moves from state to state its like a finite state machine. A is a finite set of actions that the actor may perform. And of course, certain actions may be performed in certain states and all that kind of stuff, we will see shortly.

And there is a transition function let us call it gamma, which takes a state and takes an action that you do in the state. So, you take a state and you do an action in the state and it results in the system or the agent going into a new state essentially. So, its a partial function called the state transition function. If the action a is applicable in the state S, then gamma S A is the resulting state essentially.

(Refer Slide Time: 35:58)



Planning Domain Description Languages (PDDL)

In modern times a series of languages with increasing expressivity has been defined to bring uniformity to planning research

The idea is that the researchers working on planning algorithms can use these *standardized* representations

Components of a PDDL planning task:

- Objects: Things in the world that interest us.
- Predicates: Properties of objects that we are interested in.
- Initial state: The state of the world that we start in.
- Goal specification: Things that we want to be true.
- Actions/Operators: Ways of changing the state of the world.

I have said that see there has been a considerable amount of work in the last 20, 30 years in the domain of planning and people have devised a sequence of series of languages which have been all called as planning domain description languages.

These are languages in which any user can specify a domain, specify the predicates in the domain, specify the operators in the domain, and specify a planning problem in the domain which means that you specify the start state the goal description and the set of operators available to you.

One thing that has been happening in the world of planning in the planning conferences is that every time, there is a conference they also have competitions. So, they say our domain will have these kind of characteristics, you write a planner and we will see whose planner does the best essentially.

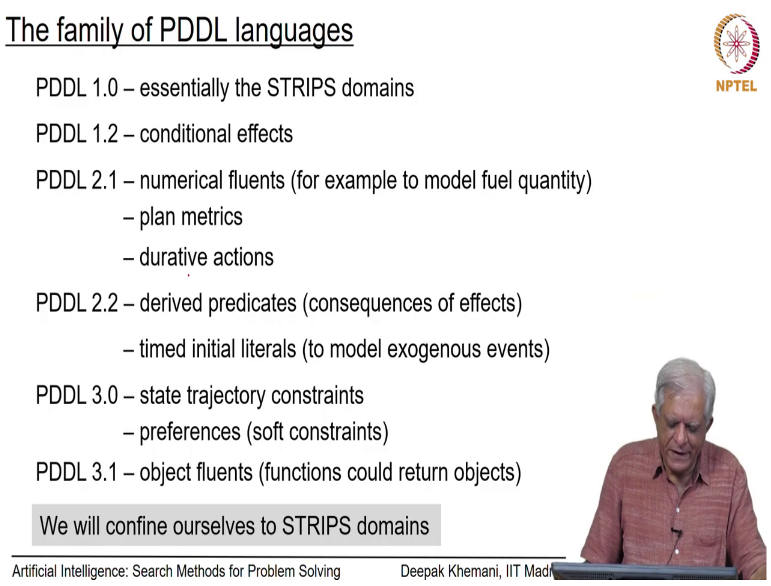
So, starting with a (Refer Time: 36:50) of languages which was the STRIPS like language that we have just mentioned, there have been a sequence of new languages, which are more expressive as you go along. The idea is that the researchers working in planning algorithms can use this standardised representations.

Which basically means; that that if two different groups research groups are writing planners for a particular domain, let us say a robot sorting out things in a shop floor. They do not have to devise their own representations; they will work with the same value representations. And then, they will also be able to compare their algorithms performances essentially.

So, components of a PDL PDDL planning task are as follows, you have to describe what are the objects in the domain that the planner has to work with you have to describe the predicates. That talk about the properties of the objects that we are interested in we have to describe, what is the initial state that the world is in we have to describe the goal specification as to what do we want to achieve.

And we have to say, what are the actions of the operators? That are available to the planner or to the agent for changing the world. So, if you describe all these things you have described a domain in a planning domain description language essentially.

(Refer Slide Time: 38:11)



The family of PDDL languages

PDDL 1.0 – essentially the STRIPS domains

PDDL 1.2 – conditional effects

PDDL 2.1 – numerical fluents (for example to model fuel quantity)
– plan metrics
– durative actions

PDDL 2.2 – derived predicates (consequences of effects)
– timed initial literals (to model exogenous events)

PDDL 3.0 – state trajectory constraints
– preferences (soft constraints)

PDDL 3.1 – object fluents (functions could return objects)

We will confine ourselves to STRIPS domains

Artificial Intelligence: Search Methods for Problem Solving   Deepak Khemani, IIT Mad

So, as I said there has been a family of languages. Let us quickly run through them and we will see that the simplest was PDDL, which was essentially the STRIPS domain that we spoke about. The next language introduced something called conditional effects PDDL 1.2.

Conditional effects basically say that the effects of an action would be conditionally true essentially. So, for example, if I am sitting in my office and I take a book and put it in my backpack and then I go home, then is the book at home or not. Now, the book could be at home if I go home provided that it was in the backpack where I put it.

So, my action of going home would also transport the book to home provided the book was in the backpack. If the book I forgot to put it in the backpack, then the same action would not achieve the effect of taking that book home. So, such actions are called conditional effects.

Then, in 2.1 we introduced numerical fluents so for example, how to model how much fuel do you have in your let us say, car or whatever you are driving. So, that you can plan your trip accordingly and maybe plan that some point you would go to refuel your vehicle.

You can have plan metrics which will say, how much fuel you have spent, how much cost you how much has it costed you and that kind of stuff. They also introduced durative actions we have spoken about them, that actions which have a start time and an end time. Then in PDDL 2.2 we had derived predicates.

So, derived predicates basically say that if something is true, then something else is also going to be true as a consequence of that essentially. So, if you achieve some something, then you may also end up achieving something else which is directly related to that. So, in the sense that this the derived predicate has do not have to be explicitly planned for. If you plan to achieve the goals on which they are based on then they would automatically become true.

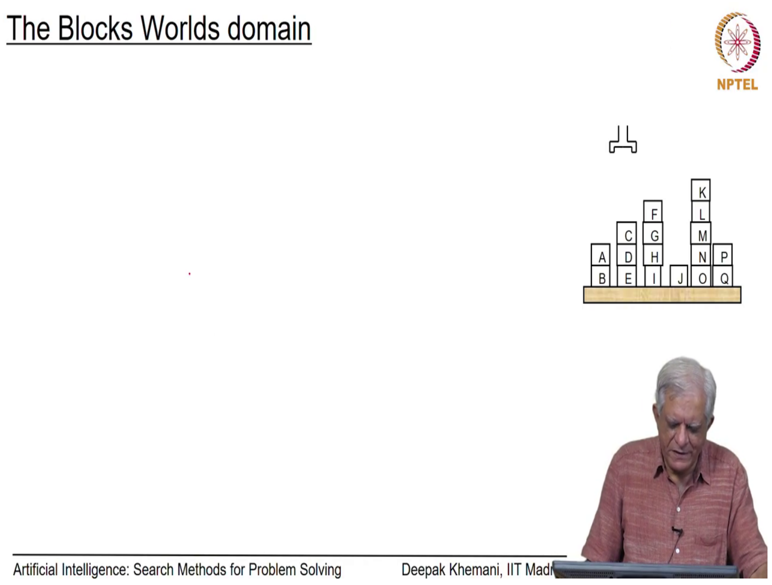
They also introduce a notion of time initial literals these are literals, which become true at certain points of time pre decided points of times and become false at certain points of time. So, for example, night and day every day in the morning the sun rises and every day in the evening the sunsets and we can model things like exogenous event.

So, if you are planning an outing you may have to take into account the fact that you want to do it only during daylight. Then, in PDDL 3.0, they introduced state trajectory constraints we have spoken about that. That not only are we interested in what we what is true in the final state, but also what was true along the path that took us to the final state.

We also added preferences of soft constraints that we have spoken about that goals which do not necessarily have to be achieved, but they are desirable they are preferred it is preferred that you will achieve those goals.

And then we introduced object fluents that if you have functions inside your language. Then, the function could not return this values for example, how much petrol you have consumed, but also certain objects has to you know what is the object that you have bought for example. We will confine ourselves to the STRIPS domain as I have been repeatedly saying.

(Refer Slide Time: 41:50)



The Blocks Worlds domain

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madr

And we will take a break at this point and come back in the next session. And start by describing the STRIPS domain and the problems we can solve in that. So, see you in the next session.