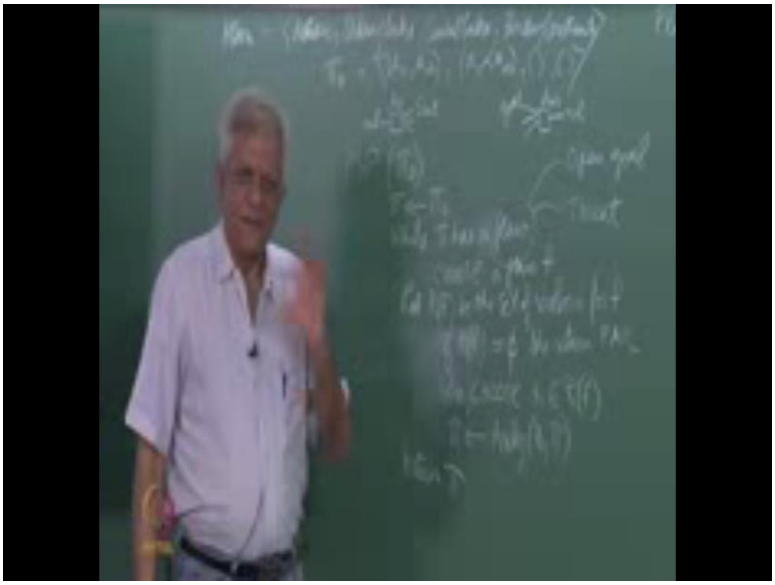
Artificial Intelligence
Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 37
Plan Space Planning

So, we are looking at plan space planning and its known as we discuss in the last class by various names partial out of planning, lease commitment planning. And the different between plan spaces planning, state space planning is plan space planning purchase in the space of all possible plans.

(Refer Slide Time: 00:44)



And a plan in represented as a 4 topple where the first one is actions as they could be operators with variables inside them then ordering links the causal links and then binding constants. So, a plan is basically made of this 4 topple and we have observed that pi 0 is always with a 2 actions A 0 and A infinity. And it as 1 ordering links if says that A 0 happens before A infinity it as know binding constants no causal links. And that is always the starting position for such in plans space planning where A 0 if remember is an action whose pre conditions are nil and post condition are others started. Whatever the dedicates in the start state of everything is a effect of the A 0 actions. And A infinity you

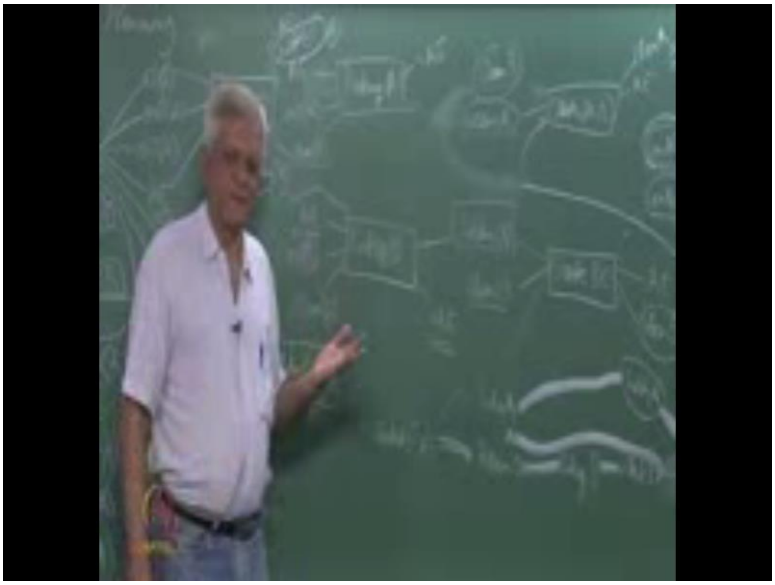
in the action which consumes the goal predicates and has no effect. So the starting node in such space is this pare of 2 actions and the ordinary constrain between them. So, let us first write down on the algorithms and then will discuss for those constants. So, high level algorithms for plan space planning so begins with pi 0 which is a initial plan. And we will use pi to stands of plans in general we initialize pi to pi 0 and then while we discuss flaws that will pi them again and a moment.

So, the basic strategy in plan space planning is to inspect the plan for flaws and try to resolve those flaws one by one essentially. And you have using here what we will call as nondeterministic operator which we choose some flaw from the set of flaw that in the plan is will define flaw the again in moment. And it looks for resolvers as will assume that we have mechanism for fine in of what are the resolvers for those flaws? They may be more than 1 and then again we say choose. If the set of resolvers empty then you cannot you find of plan else choose resolver R belong to set. Again we assume that nondeterministic will make the right choice which is by at this stage we can return empty in practice. Off course, I am sure your know conversion with this idea now, that we talk about nondetermisum algorithms like this. It basically implies at the background the deter mist calculation there search happening essentially. So, you would try 1 flaw from this set try to resolve it and then if will each at then you will back track and try the next flaw and resolve it.

Likewise you would choose one resolver to use of flaw and carry forward. And if you each at then your backtrack and try the next resolver. So, essentially in currently there is a search embedded behind this nondetermisum choose operators which we and a familiar with that here we will assume that somehow a choosing the collect clause of this algorithms is nondeterministic in nature. Then let us a pi 1 or pi gets lets in a apply is a choose is applies operator to pi essentially. So, this so here we have looking for the set of is the dissolvers the flaw then we are picking 1 resolver and applying that 2 pi which means little do something to the plan remove that flaw in some way. And then update the plan essentially and will keep doing that is longest the plan has a flaw essentially. Only place we will exist which failure so of after we exist from this off course we just return on pi. But the other place so exist is when we cannot find the resolver for flaw between is exploding all possible resolver in missing. And then we are discussing that they are 2

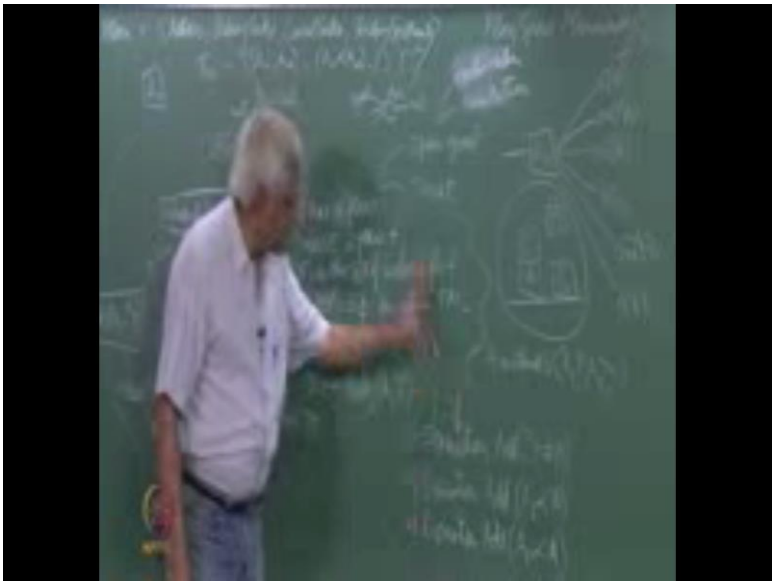
kinds of flaws one is open goal and the other is threat. So, let us discuss this suspense that we at seen earlier remember that we a shown that a algorithms like goals type planning, which does linier planning in the sense at it breaks of the goals one by one trails all the first goal and then trails all the second goal.

(Refer Slide Time: 08:17)



And we are shown that with the problem like this with the goal like on A B and on B C. If this is the goal set which in our system would be represented by A infinity having 2 goals on just you short forms on B C if it is nothing. So if you remember this was the suspense annually the start the goals state was at a s on B and B s on C and the starting state was that C is on A.

(Refer Slide Time: 08:56)

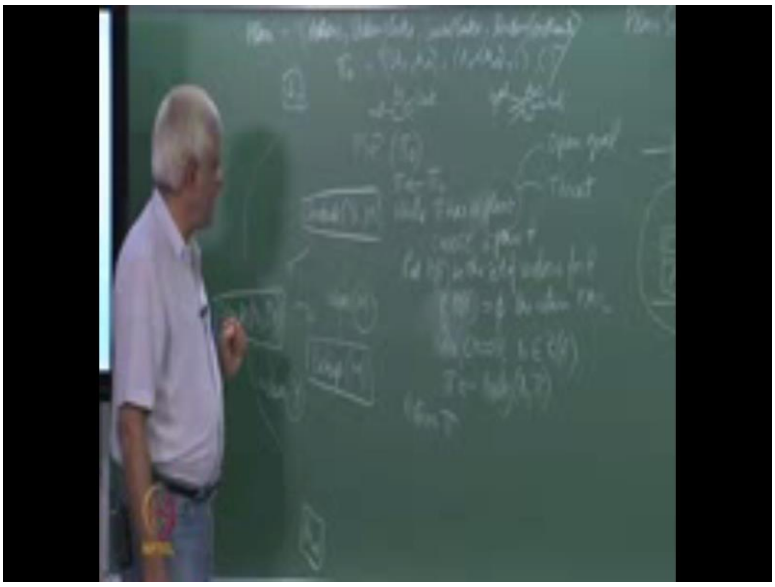


And B is on the table A is on the table which again you will recall you can represent by A 0 with no input and anything that is here which is let us say norms empty global be we a having a dealing with A 1 norms. So, what on take this for this ants for A 1 table B clear B on table A on C A clear A that. So start action an that is end action that is so suspense monopoly problem way wave we have way simple problem of 3 blocks. And what we so earlier was that if you try to say that of the 2 goals that I have on A B on B C I will try to solve them 1 by 1. The first I will completely solve on A B then I will completely solve on B C or in the other order that you cannot do essentially when you do the second goal it enviably disc rubs the first goal that you had achieve earlier. So, these goals are what we set was nonsingular lisable. And what you want see the plans space planning offers the possibility of solving such problems and finding optimal solution.

So, if you work out the solution in your mind for this problem where this is the start state an that is a goal state. The optimal solution is the 6 step solution was you first untaxed from A put it on the table then put B on to C that is 2 more action. Then put A on to C that is 2 more action goals at planning would never find this 6 step plan and plan space planning can find influence that why interest in that. Now, they are 2 kind in flaws as you set in a plan, one is opens of goal or open goals we will use a 2 terms synthetic ignobly. Another is a threat so as of now, this partial plan which as only 2 actions A 0

and A infinity as 2 flaws. And they are 2 open goals which say that there is nothing supplying this predicate what we need is some action which will supply this predicate. While in other words every open goal must have a supporting causal link which means we must have some action here which produces this predicate which is consumed were this actually so threats. So, let us what is the look looked at threats.

(Refer Slide Time: 11:56)



So, let us say some point us on to pick up. Let say pickup M is a block. So I am drawing in this top to down as is open a practice is many books M city some way here. And I have on ordering from top to down which have will not draw. Because we have only few actions with discuss here. And then some way we have the action let say stack n on to y so I will you this question mark to distinguish constants from variables as you also open practice. So, n is a constants with means the block name is n and this question mark y means it is a variable. So, we do not know which block you n stack n 1 to some for some reason you have got this action have an this happy this is the ordering an this is the order. And let us say one of the B condition for this is clear M of course there other B condition. It will remember the strict action that A must B M T and M must b on the table and 1 of this clear M. So, let us so let say this is opens goal that is flaws b addressing so how do you how do you address how do you resolve this open goal flaw? They are 2 possibilities, one is that an existing action in the plan can supply the predicate

which means as already action some way and you can just establish causal link from that action to this action provide it.

It is consistent and by this we mean the ordering links a consistent by and by that we mean that there are no cycles in that essentially. The other option the other way of is ordering open goal list to add an new action. So, for example, in this situation there is no action which can supply on A B or on B C so we are comparing to add a new action. So, for example, we might say stacks A B as on action and this provide this causal link to on A B. That the second way of in solving the open goal which is to add an new action to the plan how do we add an action to the plan? We add the action to this set of actions then we add a causal link from this new action to the predicate that we a try to support. And means certain ordering link here and if there any constant and we for example, you make want to say add stack x y an x equal to A and Y full B. But will skip that part essentially likewise you could say this is achieved by stack B C. So, because of the stack that one other things we are doing in plans space planning is resolve open goals. It still as a flavor of backward is in, because open goals are only in the goals state and the new keep building.

But it split possible that you end of first satisfying some open goals then jump here. So, there is no fixed order of such essentially. The other in case in thing about plans space planning with here observed was that in state place planning both the selection of the action and position in the action was them simultaneously in 1 move. So, for example, forward space place would say this is the next action. So which means the select the action an also set the next position is the next in the plan. Because they are plan is a sequence actions likewise in the backward stage space search you sack construct in the plan from n steps. In plan space planning we are selected this action stack A on B and stack Bon C, but we are not say anything about what is you pending the relating already in between them essentially. So, we separate the task of selecting in action and infusing an ordering on the action essentially so that is then independently. So, in principle off course this allows to first select some action in between. For example, if they have something with as very practical to the plan and we knew it was practical then we could select it in between. And we a choose in select of example of going from place a to place b.

So, if at to go from here to there on for example, I would to say I need to flight to Delhi first and I put that action in between. So plans space planning allows you to do that but, this kind of listing which is more up stack requires still more structure in the plan. And structure is difficultly hypothetical in nature. So your to top of high level actions and the define in to low lower actions which you are not going to considered. Here so we a working as the at the action level which the at 1 plat level essentially which is the I this as the flavor of backward search to some extent. So, we have this clear M here and let us say do make clear M B say up stack something let say it do not give. Now, what that is from x from m and that will produce this. So I will use the dotted line to represented causal link which means this is producing this and this consumedly that essentially. So, this was off course happen before this essentially. So the moment in set this action in to my plan. So, what we that do in this? Already they in my plan this 4 actions the 2 start actions and the this 2 actions at for they in plan.

And then I have added this action up stack something from M and added that. Because I want to resolve this goal of clear M, but the moment I have this plan structure. It has a flaw which is of the type threat essentially and what is the threat? In the threat is that here on the casual link so basically threat is to a casual link. Always that something is trying to disturb to casual link what is the casual link doing? Casual link says this action is producing clear m which is consumed by this pickup M action. So, I which established casual link in my plan essentially. Now, because there are variables involved here, because of fact that we do not know what is ordering of these 3 actions. Off course, we not this once come before this, but we do not my anything about how this is place between take to these to. Then the put essential rate what is the put essential rate that this action will come in between these 2 actions. And this y will be in to M which would mean that for pickup M I needed clear M to be true, but if this stack n on to M happens before that which means it happen between these 2 actions.

So, let us a this is the threat data an considering that this happens first this happens next in this happens third. So, this action happens after this action an before this action and there is the possibility that this y I can we made equal to M or bond to M then you can see the details destroying this casual link. Because for pickup m needed clear m by putting n on to M that clears M is been lost. So, this action is the threat to this casual link

so as we I think discuss in the last class. And action is threat and when you say threat it algorithms potential threat to a casual link. If it can produce so what can it produce? It can produce not clear y as an effect so effects are below and the p condition are that it can produce an effect. Lets us call it not p which can the infinite with a condition p needed by the this action which means is pointed destroyed this condition. If can so in other wards is y can be infinite with M and many say infinite we mean by could M is possible and this action happens after this action and this action happens before this action. So this 3 conditions were to become too at that y equal to M and there as on wonderingly between this and this and on wonderingly between this and this then the threat would materialize.

And my plan with a longer is a valid plan, because is would know this would come on open goal which on a longer be satisfied. Because a casual link was satisfying this, but in certain this action in between destroyed at casual link. So, threat is the other kind of flaw in the plan 1 of 1 is A simply kind that open goal as those support the other kind is that there is a threat to how to be resolve the threat? So, there are 3 west of resolver threat 1 is separation which in this example. So, basically say add of binding constants. And make sure that this cannot unify with is which means add this to might set up binding constants essentially. One side does that they can know that I am not going to stack this n on to M. I will stack known to something else which means M is free for whatever that x is to be stack known to that. That is one way of resolving a threat another way is to demote is called demotion So, let us say in general terms let us say action A is a threat to a link call A one some predicate p in action A 2. So, this is A 1 this is A 2 and this is action A one way so say that p is different which is separation the other 1 is demotion which says that that action A happen after this pickup M which means add the link A 2 happening before A.

So, when I say add added to the relevant place when I binding ordering link I must added to this set when I binding constants I must added to that is set an so on. So, I can pushes action down and so we say it is been demoted it happens later. That were also resolve the threat, because then nothing is disputing this clear M. Because they all this happens some were later another third is as we can a imagine promotion which is you add A 1 to happen before A again to the set of ordering links. So, we have basically 5 resolvers for

open goal we have 2 resolver's goal I just say old action. By this I mean an existing action so may be just right existing actions order new action. To resolver open goal either fines some existing action which can supply the predicate all if you cannot then add a new action. Because taking an old action existing action may actually violated the constant that the should be no cycle in the action. So, this is all the open goal and this for the threat essentially and algorithm essentially goals. So, this cycle of choosing of flaw how a initial plan as 2 flaws those 2 open goals choosing a resolver. So, both of those flaws we are chosen to resolver who is to add the new action. And a keep doing that till there are no more flaws left in the action. So, this is the less a this is the flaw an underline 1 and if have cycle it then it is been resolved. But off course, these actions have their own requirements.

So, stack B C requires holding B it requires clear c it requires on no in that is all and it produces a from on b c produces a empty an produces clear B. So, let us assume that clear B is an effect of stack in B on to C which means 1 unstack something of where if you are to under stack of B from something then B would be your longer clear. So, when you are holding it B is not clear only when it predict down it is clear essentially. Likewise this needs clear B holding A and it produces arm empty and its produces clear A. So, everything A is an open goal every time we added new action all is B conditions are open goal essentially then we look for of flaw and resolve it is essentially. So, let us say we look for this flaw clear B and you say this action is producing clear B. So, I have an existing action which is supplying clear B so I had an ordering link which so I say that that this action or this clear B is consumed by this. So, I anode ordering link which is at their link is between the actions that this action was happen. So, first an must stack B on C which will produce this clear B and then I will consume clear we when I stack A on to B known A B C.

So, you can see that slow logical in some sense that you know you can see that you must first stack B on to c and then A on to B an at least 6 doing in this part correctly essentially. Now, let us at this 1 it to this is an action so an action that we can fine is unstack no holding A. So, action that we can kind is pick up A and the action for this also we can fine is so this here picking care of pickup B. Lets a we are doing this in this order so pickup we will also produce other things likes no arm empty. In that is sense A

little delete A empty it needs arm empty as a predicate A empty is predicate will producing not A empty it needs on table A this needs on table B. So, let us right it here on table B on table A it needs clear A it needs clear B observe that we have and open goal clear B here which could I will supplied by that action stack B on to C. But we cannot choose that, because then we would have to say that that happens before this and you already setup this happen before that. So, something which is a kind of goal with is that the moment you added a causal link this a causal link we are added. Pickup B produces holding B which is consumed by stack B on to C. The moment we add this link we also added ordering link essentially.

So, once you have added an ordering link is this selection we cannot add on ordering link in this selection. So, we cannot choose this that for this essentially. So, let us say that we choose this as an open goal and we say unstack something x from A and that will make it clear. So, become an effect of this action and it also produce holding x and another things and unstack x on A needs x to be on A. So, we can produce it like this that this start action has an effect call on C A and we can say if you say that x equal to C then we can establish this casual link here. So, this is taken care of the which am not drawn here the C condition for this is on C A that is taken care of, because it is will produce by A 0 another C condition is arm empty which is also produce by this. And the third a condition is clear C which must be there somebody should of point is these are this we clear C and that is solving. So, the c conditions need for unshackling c from we which is clear C is true in the initial state, on C A true in the initial state. And arm empty is true in the initial state and they are produce for this so we establish this 3 casual links which will do that essentially.

Now, we can see there if will look at this casual link here that A 0 is producing A empty which is being consume by unstack C from A is being threat arm by this action here pickup B. Because if pickup B we have not set when pickup B is always set is that A 0 happens then we some stack C from A happens. Then pick up A happens notice also that this cannot happen immediately afterwards and once you unstacks C from A we are holding C and it would produce something call mark arm empty and holding C. So, which means we obviously cannot do unstack C from A followed by pick up A, because pick up A also needs arm to be empty. But our plan an as only set that this must happen

before that an nothing else we are not set that no action can be in certain between an as way will see that can be done essentially. So, the only thing set is 1 line is at you unstack C from A and there at some point you pick up A an stack an to B. But before you stack an to B some time you stack B on to C and before you stack B on to C you must pick up B essentially so that is all we have so of essentially. But now, we have a threat this is being is a threat in this is threat in to disturb this casual link here.

So, we have these 3 option available separation is not possible, because there no variables safe. Promotion is not possible, because A 0 is the first action which always the first action an nothing can happen before A 0. The only thing remains is demotion which means we say the this must happen after this. It means we in introduce the casual an ordering link between this and that. Then let us say that so let us say that we address one of these issues they lets a we looking at this arm empty. And we are pushed it be on this so we cannot connect it to the start in action. So, we must produce some action calls pickup our put on something. This put on by will need holding y and this open goal can be might by this goal holding C by seen y could to C essentially. So, what are the actions we have so far? We have what is action unstack C from a pickup A put down so this will become C here put down c is someway so 1 2 3 4 5 6 if got all the 6 actions that we need. And I will set up leave this is the small exercise so you to see that now, you can resolve all the flaws in this by either introducing in ordering constant to remove a threat. All connecting so for example, we can connect this to this which means this must happen after that.

So, the first action that we can C is unstacking C from A and sometime after that you must have pick up A. And sometime after that you must have put down C we do not know we just know that this happens after that. Then we know that after putdown C we must have pick up B from A. After pickup B we have stack B C an sometime after stack B C we have stack A B off course it is not at proper plan, because as you can C you can a do this to action stack B C and stack A B immediately afterwards. It needs A pre condition which is destroyed by this which is should be holding A. So, you can see that this needs holding A and that is provided by this. So, this is a pre condition an some we out to figure out that the sequence of fraction is that you unstack C from A when you putdown C when you pick up B stack B on to C and then this needs to be added. That

this pickup we must be happen after stack B C so why it happens? I will owe it as a small, because it happens. Because some flaw is to be resolved and the flaw could be for example, that arm empty you are something like that is needed for pickup A and stack B on the c producing arm empty so we with that will right the stack.

So, in the end will have one order go from here to here on here to here sorry go from here to here when go from here to here? And we expect earlier plan in this particular example, because of the fact that we a dealing with a one arm brought. And you can do only one thing at the time as oppose to other kind of domains. For example, this shoe tie domain that we discuss earlier to were lefts out were life rights of were left shoe were right shoe there the final plan may not have a leave your order essentially. It will have a parlor order that you first were the 2 socks and then you were the 2 shoes and then you are done essentially. And then you could do between them in any order essentially. So, any linearization of a partial order or any to topological out of a partial order should be a valid plan. That is a original condition is started with, but in set of having to check all linearization we have change this too is condition that which should have known flaws. And the flaw as define is they must be known open goals. It means every open goal or every goal every sub goal must have a supporting casual link.

And they must be no threats that no action must be threat in to dispute casual link essentially. And as on is that condition is satisfied the plan that we produce is going to be a valued plan. In this case it terms out to be linear plan which you are also the optimal plan essentially. The off course we would possibly observe that they little bit of a right of a hand here that as some of chosen the right actions and the right things. In a put some away chosen the right things for this choose a flaw on choose in action. So, notice at this a symmetric situation we want to pick up B on we want to pick up A. And I choose pick up A which let us to the plan if have chosen pickup B then you can see that there know way you could have produce the optimal plan we the produce longer plan. Because if you are going to do pickup B as a action which is the after this then you will have to put it down again. And then pickup C put it down and then pick up A so although extraction would keep in a essentially. So, this is the basic idea of plans space planning this is the algorithms there are variation would be algorithms which people have essentially.

Some people only maintain and is in the, of to be solve and the keep is ordering threat along the way is in the open goals to be solve. This is the more generate algorithms an you can off course right variations this you can put in some heuristic and so on essentially. So I will just like to end with ((refer time 42:14)) about something which and mention on the way which is this notion of so if you have observe what was happening here? We started such resolving open goals from the end essentially. Because x y the open goals for an this produce more open goals and we had this flavor of backward reasoning happening essentially. And that, because we are working as the ground level we are only considering action which are applicable in the do mine but, if you at high level actions like know go from here to Delhi. Now, that is not an which is the ground level action the ground level action would have many other steps that you go from here to the station you go the tree in an all this. If you have this notion of hypothetical actions which means action which can decomposition in to smaller actions then we can talk about high level plan and then refining it in to lower level plan.

And then this strategy that we at spoken about earlier means ends analysis. If it member Simon a Newell the means an analysis strategy problem solving was the another first strategy proposed by in a in the 60's in the last century. It was based on the way thought human beings solve problem. And the basic idea behind means an analysis is that just like we have flaws and we have resolvers of flaws. A more general think is that you have different is and you have resolvers for defenses so what you mean by differences? So, in this travel do mine I would say there are many different is my plan to go from here to deradun. One is at the highest difference is that I am here I am not in deradun so in that is in sense it is like of flaw. But then decompose it in lower level actions the, and then I decide that you know I need these steps I need go from here to station all. Let us say even lower level there I to go from here to the gate and then gate to the station and then getting to a train and then go to Delhi. And so I if realize that and if a can recognize the, these are the difference is that I and it to resolves.

So difference one difference is that I am in Chennai not in Dehradun. And so the largest difference I me have a difference operator table which would say if distance is more than 500 kilometer you must take a train or of flight. If distance is less than between 100 and 500 you can take up bus or something like that we have a operator difference stable like

that. And then the key thing about means and analysis is at if you have difference is and if you a operators you must choose a largest differences first. The largest differences would be that I am in Chennai I am not in Delhi and once says say that I have to resolve that differences first I would first selection action or the plan for doing that. And that let us say that action is taking a train from Chennai central to Delhi. And then I have smaller difference is left am here not in the station that is 1 difference then I will Delhi station. But I am not in lets I am taking to a bus there as soon I am not the bus station on anything likes that essentially. So, you address higher difference is first and the lower difference is later. So differences are the ends and operators are the means to a cheap bus ends essentially. And means at the analysis says that you must analyze the do mine. And look at the larger more important things for first an less them and then look at the less important things essentially.

So, 1 interesting example of this is this tours of ((refer time; 46:18)) which we must be familiar with. So, if you want lets a move 5 dist from location A to location C then there are 5 differences. In that smallest this see that a not at C second smallest dist this at A not in C you are 5. And if we can now, order this difference say the larger the dist the more the difference then the here means. And analysis strategy says that first whatever moving the biggest disk to the definition then many about moving the other this essentially. And then off course, you know the familiar ((refer time 46:56)) algorithms will come automatically out of this that you move n minus those to B then move this largest this to C and so on essentially. So, all those can be seen as specific case is of means analysis and that was propose fight long time in their book for human problem solving. So, all this algorithms that we are have seen of planning so far. These are in some sense ancient they are all developed in the last century sometime. It terms out that whatever however tries to right this algorithms. The problem is hard member we said that even the split domain the simplest planning domain is piece space complete essentially.

So, we terms of that even in simple domains the only kind of plans we can find were of lengths something like 5 to 10 essentially. So, if you are the problem in which the solutions of 5 steps longer 10 steps long then we could use is algorithms for solving them. Then in 1995 of pare of researches care of with a new idea and with an algorithms which look at this whole thing in very different way. In algorithms call graph plan an

there was the variation which came long along that time which increase the length of the plans could be found by an order of magnet to essentially. It means it we could find plans of lengths 10 we these algorithms goes stack planning or any of these algorithms we could find plans of lengths of 100 using these new algorithms essentially. Off course, we would not have time to study those algorithms in details in this course, but in the next class I will just give a games of this algorithms Graph plan which is kind of a land mark algorithms in the world of planning essentially. And will end with planning in this course with that essentially so I will stop here.

Student: This shows A 2 should happen before this shows that A 1 happens before A.

Promotion here promoting so A should happen before A 1 that is right holding is not part of length. Sorry.

Student: Here holding is not part of length here this is we are going to pick up A to after stacking B on C we are pick up in A.

Then you have this holding A not an action it is a predicate that is a I drawn a circle on that for an action. After that after pick in up a stack A on to B in that is a last step in the plan essentially. So, we will stop here.