**Foundation of Cyber Physical Systems**

**Prof. Soumyajit Dey**

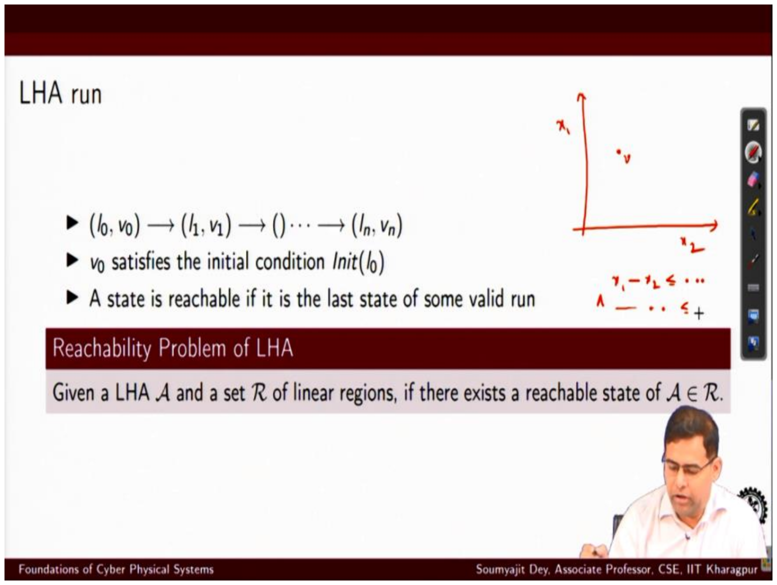
**Department of Computer Science and Engineering**

**Indian Institute of Technology, Kharagpur**

**Lecture - 34**

**Reachability Analysis (Continued)**

**(Refer Slide Time: 00:33)**



## LHA run

- $(l_0, v_0) \longrightarrow (l_1, v_1) \longrightarrow () \cdots \longrightarrow (l_n, v_n)$
- $v_0$ satisfies the initial condition $Init(l_0)$
- A state is reachable if it is the last state of some valid run

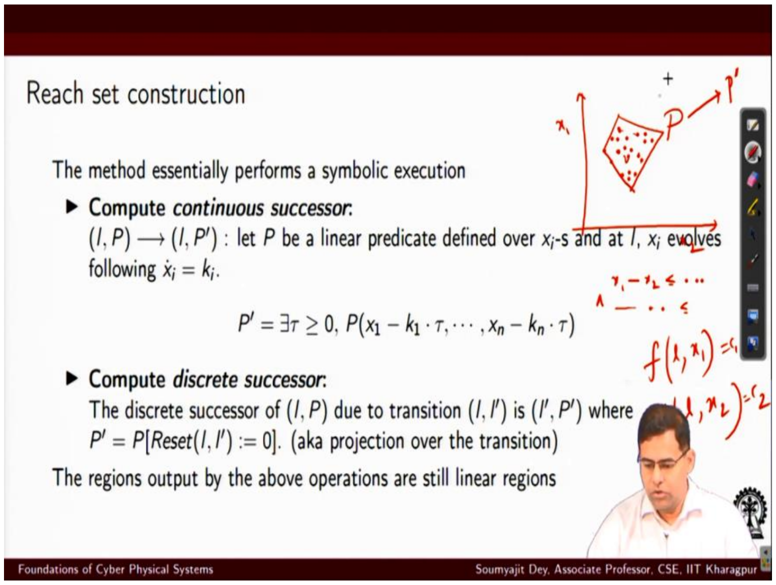
**Reachability Problem of LHA**

Given a LHA $\mathcal{A}$ and a set $\mathcal{R}$ of linear regions, if there exists a reachable state of $\mathcal{A} \in \mathcal{R}$.

Foundations of Cyber Physical Systems | Soumyajit Dey, Associate Professor, CSE, IIT Kharagpur

Hi welcome back to our lectures on foundations of cyber physical system. So, in our previous lecture we defined what is the reachability problem of linear hybrid automaton. So, if you remember this is what we define that you are given that automaton, you are given a set of linear regions where the question is whether this region is at all reachable, whether I can start any of the valid initial states of the automaton and I can have a valid run of that automaton which would reach a point inside this region.

**(Refer Slide Time: 00:56)**



## Reach set construction

The method essentially performs a symbolic execution

- **Compute *continuous successor*:** $(l, P) \longrightarrow (l, P')$ : let $P$ be a linear predicate defined over $x_i$-s and at $l$, $x_i$ evolves following $\dot{x}_i = k_i$.

$$P' = \exists \tau \geq 0, P(x_1 - k_1 \cdot \tau, \cdots, x_n - k_n \cdot \tau)$$

- **Compute *discrete successor*:** The discrete successor of $(l, P)$ due to transition $(l, l')$ is $(l', P')$ where $P' = P[Reset(l, l') := 0]$. (aka projection over the transition)

The regions output by the above operations are still linear regions

Foundations of Cyber Physical Systems

Soumyajit Dey, Associate Professor, CSE, IIT Kharagpur

So, let us see how this can be done. So, what we do is we compute the sets that are reachable starting from some initial location. Now for that we will define some operations. So, this is what we call essentially is a symbolic execution of an automaton or any formal system in general. So, first thing we will talk about is how to compute a continuous successor. So, a continuous successor that means that I am in a state and the only thing that is happening is time elapsing.

So, in my state let us say I have l the location and a valuation v, this is how we talked about a step. Now you see I am not talking about valuation v but rather I am writing something like p which is like a linear predicate. Now what does that mean? So, let us take an example. So suppose I am talking about a two dimensional systems there are two variables $x_1$ and $x_2$. So, when I talk about state here is basically a point that means I am in a location l let us say.

And a valuation v is giving me a value of $x_1$ and value of $x_2$ is a point. Now let us say I start writing $x_1$ - $x_2$ less than equal to something and similarly another constant less than equal to something like that. So, using such a combination of linear inequalities I will get convex linear predicate like one which we show here that is $P_1$. So, that means now we are not really talking about a point but maybe we are talking about a region like this.

So, the question is suppose I am in any of this states which are inside this region. So, there is an infinite set of such states and they are succinctly captured by this linear predicate P. So, that is a way that we capture it symbolically and we are talking about well what is the continuous successor of this collection of states. So, this is the region as you can see and we are talking about what happens to this region with elapsing of time.

So, with elapsing of time this region's shape would change. Because I mean for the simple region saying that these variables $x_1$ $x_2$ their values are going to change based on their flow dynamics at l. So, based on these two constants these variables values are going to change and accordingly P would get modified to some prime. So, this region's shape would change. So, the question is how do I characterize this symbolical?

So, this is it. P is the linear predicate. It is defined over this exercise and $x_i$'s are evolving following some relation like this given for a given l. So, P' would be defined by a predicate which I can construct given P so I am given P that means I know what is the structure of P. So, whatever is the definition of P in that you make a substitution. That is you introduce an existential quantifier here which is there exist some $\tau$ greater than 0. Essentially you are talking about the elapsed time.

So, $\tau$ is basically the amount of time elapsed starting from this from the initial states I mean at l. So, suppose tau amount of time has elapsed so that it mean $x_1$ value has altered by this much $k_1$ times tau because $k_1$ is the rate of change, the constant rate of change and tau is the amount of time for which this change has happened. So, $x_1$'s value has altered by this much, similarly some exchange value is altered by this much.

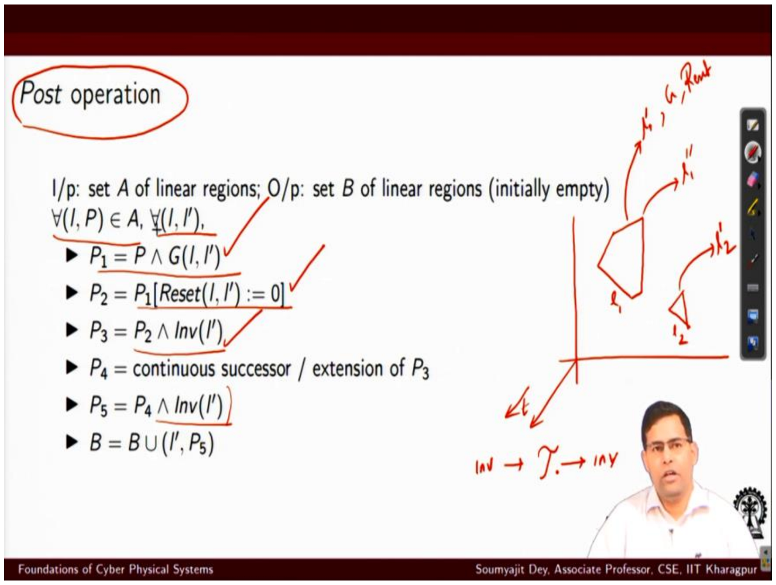
So, in this condition the modified structure of the predicate would be given exactly by P but with this variables $x_i$'s getting substituted by $x_i$ - $k_i$ times $\tau$. And in general, when I write P’, I am not telling that $\tau$ equal to this it is a continuous successor, that means tau is elapsing. So, tau is symbolic, so for any tau I have this P and this is the predicate which is capturing the continuous successor construction for the initial region that I have.

Now the next thing is just like in this way I can create a continuous successor of collection of states I should also be able to construct a discrete successor. That means there has been a location switch. So, suppose I was in that same region but now a location switch has happened. So, the transition l to l prime took place and the resulting region will be modified to some l prime P prime. Now what is P’?

Well, P’ is that same linear predicates which define P with the modification that all the variables which are in this reset set of l’ they will become 0. So, this is also known as projecting the region over the transition. So I was in a region I was in an n dimensional region like this. And I have just projected it that means I reset some of the variables here. So, essentially this n dimensional thing you are projecting to a subspace where some of the variables have been reset to 0.

That means you have projected it to that plane. You have projected this n dimensional setting to a plane where those variables are 0. But one thing we are sure that even after doing these operations that the region that will be constructed by these operations, they are still linear regions because we did not violate linearity by any of these operations all the operations done has been linear. So, the resulting regions that have been constructed using these symbolic executions are also linear.

**(Refer Slide Time: 08:02)**



## Post operation

I/p: set $A$ of linear regions; O/p: set $B$ of linear regions (initially empty)

$\forall(l, P) \in A$, $\forall(l, l')$,

- $P_1 = P \wedge G(l, l')$
- $P_2 = P_1[Reset(l, l') := 0]$
- $P_3 = P_2 \wedge Inv(l')$
- $P_4 =$ continuous successor / extension of $P_3$
- $P_5 = P_4 \wedge Inv(l')$
- $B = B \cup (l', P_5)$

G, Reset
Inv → T → Inv
Foundations of Cyber Physical Systems
Soumyajit Dey, Associate Professor, CSE, IIT Kharagpur

Now we will define another operation which is known as the post operation. Now let us understand what it means, we are trying to figure out. Suppose I am given a set of such linear regions not one but multiple such linear regions. And what I want to figure out is what will be the set of linear regions once I start in these regions the initial set at some location l and then I jump to another location l' and I spend some time there then what is my set of linear regions?

So, due to a jump what does it happen there and how does my set of regions change that is what I want to figure out. So, what we do is I am given this set of linear regions. So I figured out inside this set of linear regions what are the convex linear predicates. So, if you remember these are set, so that means it is containing multiple such regions. So, let us pick up in each of these regions. So, let us say it one is given by l p another is given by some $l_1$ $p_1$ like that.

So, we have such linear regions. Previously whatever we defined was with respect to one region. A region again I will repeat a region is nothing but a collection of states and a possibly infinite collection of states and here we are talking about a set of such regions. So, inside each region inside this set you pick up each region like this. Now what we want to do is we want to figure out that starting from these regions what are the all-possible transitions that exist.

So, let us say we are talking about these two regions along with the associated locations $l_1$ and $l_2$ and from $l_1$ there are switching's possible to locations $l_1$' and $l_1$" and from $l_2$, there is a switch that is possible to location $l_2$'. So, that means I will pick up each of these regions. For each of these regions I will consider all the outgoing transitions and I will do the following.

So, that is why for all the regions, for each of the regions for all the outgoing transitions what am I doing. I will compute this which is nothing but well I can take this transition from $l_1$ to $l_1$' only if the guard condition is satisfied. That means the moment I take the transition it is not that for all the points or all the states in this region, my guard will be satisfied. That means the moment I jump to $l_1$' this region will be intersected with the guard.

And the sub region will emerge. So, what are we really doing here? We are trying to figure out that assuming that my system starts from this set of linear regions, what are the all possible regions where I will land up after one discrete step. This is what we define as the post operation. We are saying that suppose my initial regions $l_1$ and $l_2$. These are the discrete steps that are possible $l_1$', $l_1$'', $l_2$'.

We are trying to figure out after this one discrete transition what is this collection of regions where I may reside so that is the post operation, now that is a definition here. So, pick up this one transition from $l_1$ to $l_1$' it has this guard which is again a linear predicate. So, you will intersect these regions mathematical condition with the guard condition. That will give you possibly a sub region based on which sub part of $l_1$ is satisfying the guard, so you get $p_1$.

Not only guard there would be some reset also. So after constructing $p_1$ you have to apply the reset condition here to generate some $p_2$. Not only that when I get to the target location it will have an invariant. So, let us understand what is going on we have picked up a region. Not all the states in the region are states from which I can really take the guard condition. I can really take the transition.

So, the first operation gave me that sub part of the region from which the guard is enabled and I take the guard. Then after taking the guard my region will change further because certain variables have been reset. Now again even if these are done, I may not enter my I am not allowed to enter l prime if the invariant is not satisfied. So, that means I will go to a further sub region of this from which the invariant will actually be satisfied.

So, that is how I get $p_2$ intersection invariant of l'. Now the next thing is the continuous successor or the extension of $p_3$. So that is basically this computation the continuous successor. So, that captures that what are the possible regions in which I can go with elapsing of time. So, that is how this variable will now come in there will be an existential quantifier and that will capture that how this region can change as a function of time elapsing.

There is a continuous successor. Now then with continuous successor you will do intersection again of this. So, what does this represent? I mean let us understand what did we just do. We took an intersection or with invariant set but again we are taking an intersection with invariant set. We are trying to create a region definition which is symbolic in time. That means I started from a region in $l_1$, I took the discrete switch to $l_1$'.

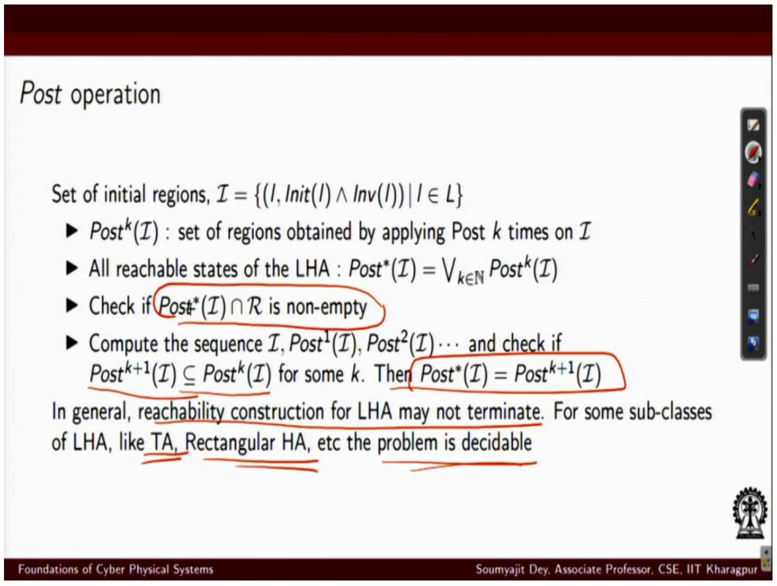
I satisfied the guard condition, I reset suitable variables, I satisfy the invariant of $p_2$ and then I am trying to create a symbolic definition of the region as a function of time. That means with change of time till this definition of that reachable region is holding. And then it should again after this elapsing let us say time has elapsed up to some $\tau$. So, here the invariant was satisfied after this the invariant will again be satisfied.

So, you can see that this is a symbolic definition. For some value of $\tau_1$ it will be true for some value of $\tau_2$ it will be false. So, in that way what is happening is we are adding another variable in your region definition with respect to time here. So, in that so that is what is happening, I hope it is understood. So, in this way we added the continuous successor operation here you took the invariant intersection earlier.

And again, after the continuous successor operation to filter out those states which will be part of the region at any time $\tau$. And then what we do is you get this final set p and store it in your collection of final sets. So, this is initially empty, this is what you want to compute, output set of initial regions, that is what you want to compute initially it is empty. So, what we say is we are saying that suppose I started from $l_1$ with region p.

If I took the transition to $l_1$', I would land up in a region or a collection of states that is $l_1$ l' $p_5$ and that is of now a part of this set of output linear regions. Now you take again $l_1$ and do the same computation for another possible discrete switch, you take $l_2$ do the possible computation for the corresponding discrete switch and all those regions you take union here. So, this tells you that if I had a set of initial linear regions with one discrete switch what will be my set of output linear regions. So, this is the post operation.

**(Refer Slide Time: 16:47)**



## Post operation

Set of initial regions, $\mathcal{I} = \{(l, Init(l) \wedge Inv(l)) \mid l \in L\}$

- $Post^k(\mathcal{I})$ : set of regions obtained by applying Post $k$ times on $\mathcal{I}$
- All reachable states of the LHA : $Post^*(\mathcal{I}) = \bigvee_{k \in \mathbb{N}} Post^k(\mathcal{I})$
- Check if $Post^*(\mathcal{I}) \cap \mathcal{R}$ is non-empty
- Compute the sequence $\mathcal{I}, Post^1(\mathcal{I}), Post^2(\mathcal{I}) \cdots$ and check if $Post^{k+1}(\mathcal{I}) \subseteq Post^k(\mathcal{I})$ for some $k$. Then $Post^*(\mathcal{I}) = Post^{k+1}(\mathcal{I})$

In general, reachability construction for LHA may not terminate. For some sub-classes of LHA, like TA, Rectangular HA, etc the problem is decidable

Foundations of Cyber Physical Systems

Soumyajit Dey, Associate Professor, CSE, IIT Kharagpur

So, how do I use this post operation? Well, it is very useful. So let us see. So, symbolically this is my set of initial regions that for every location l what my set of initial regions is that I take those which are I mean I have a set of possible initial states which is the Init l. And they must also be. So, basically what is Init l if you remember previous lectures, it just tells me what are the set of allowed valuations.

What are my set of allowed valuations and states for the start of the system and along with that for those locations there are invariant conditions which also have to be satisfied. So, in that way Init l intersection in well that gives me the set of possible starting the set of possible initial states of the system. So, overall, I have a set of initial states. So, as an example let us say I have an hybrid automata which can either start here or it can start here and here it has some Init $l_1$, here it has some you can refer to the two-tank automaton example in the previous lectures in del 2 and inside there are the invariants, here also there is an invariant. So, Init l 1 intersection in l 1 union I mean along with that I have this Init $l_2$ intersection Init $l_2$ along I mean. So what I have is $l_1$, this Init $l_1$ intersection this that is one possible initial region. Similarly for $l_2$ if there is any that means if these are both valid initial locations then I can enter Init $l_2$ with the initial set of value valuations Init $l_2$.

But not all of them may be valid not of them satisfy the invariance. So again Init $l_2$ intersection Init $l_2$, so and so forth and their collection so that is my set of initial regions. So, the next thing we do is we apply this post operation k times on this set of initial regions. What does it give me? It gives me the set of regions which are reachable by that automaton starting from the set of initial regions and with k number of discrete switches.

So, with one switch whatever are my reachable regions that is given by one post operation, apply post operation k times that I get, well I start in these initial regions and there are k possible discrete switches what are the set of regions I can go. So, that is this set post $I_k$. Then what are my all set of reachable states for the for the hybrid automata? It is defined by this. So, we are we are basically giving mathematical definitions of reachability here.

So, my reach set for the hybrid automata is simply given by this, that you apply post one time, two-time, three time, k time up to all the natural numbers and take their union that is your post star. So, then what is the reachability problem? The reachability problem was that. Well, if there exists some initial state from which there exists a valid run through which this give some given reach set is some.

I mean this reach set is at all reachable or not. So if you remember that was our reachability problem. If we go back here given A and given R if there exists a reachable state of A that belongs to R that means whether there is a valid run in a starting from some initial state inside I that will eventually be a member of R which means post star I intersection the given reach set R is non empty.

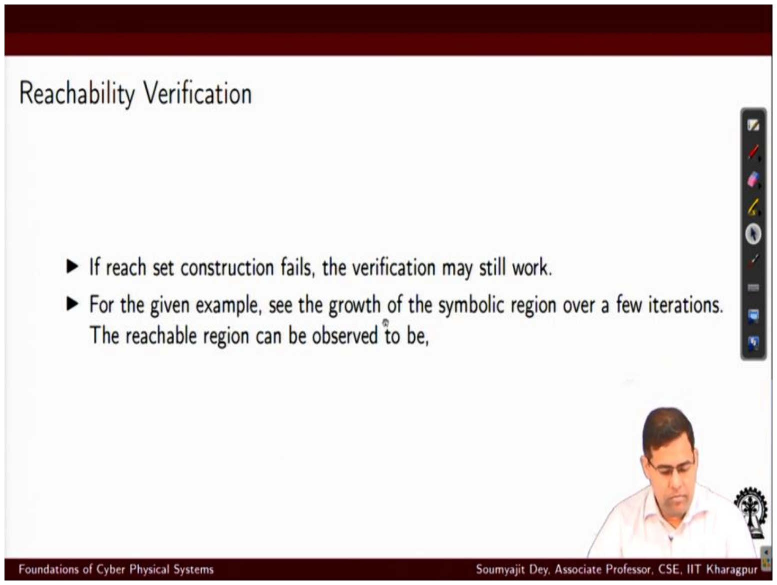
So, that is the verification problem here and the reach set computation problem is this, that whether I can compute this that is my reach set computation problem. And if I can intersect the reach set with the given reachable set and check that the intersection is non empty. So that is the verification problem that whether reach set R, I can reach at all. So, figuratively speaking, suppose this is my initial set of states where there exists A path which reaches this R from I.

Now how can I compute this? Well, I can go on computing $post^1(I)$, $post^2(I)$ like that and check for some k, that whether this happens that after doing the post operation the region that I get does not have more number of I mean points or states which are not part of the previous post set. That means if I can get something like this that $post^{k+1}$ is contained inside or is equal to $post^k$, if I can get this.

Because if I can get this then eventually I can always I can apply this inductively and I can say that well this holds. That means well this is my final set I can say that. Why? Because this is kind of a monotonic construction going on here. However, this construction in most cases may not terminate for linear hybrid automata, but for some sub classes so that makes the problem undecidable itself but for some sub clauses of linear hybrid automata.

For example, timed automaton timed automaton, rectangular hybrid automaton, the problem is decidable. We will see what it means, it means essentially that there are computational procedures or well-defined algorithms through which I can actually figure out whether this I can actually compute these two sets and I can actually check whether this condition is non empty or not, for certain subclasses the problem is decidable.

**(Refer Slide Time: 23:36)**



Reachability Verification

- If reach set construction fails, the verification may still work.
- For the given example, see the growth of the symbolic region over a few iterations. The reachable region can be observed to be,

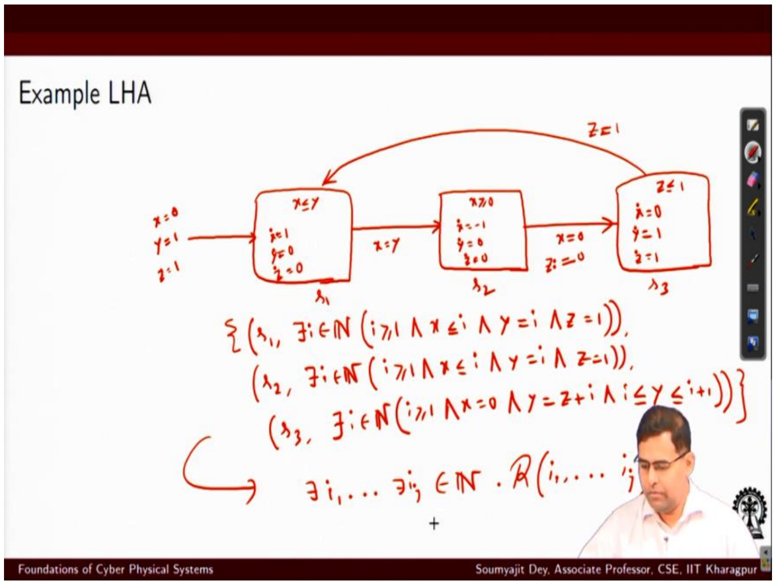
Foundations of Cyber Physical Systems Soumyajit Dey, Associate Professor, CSE, IIT Kharagpur

So, the issue is that well in general constructing this reach set may be difficult for the given initial region but let us understand what is my problem. I mean I finally want to see whether this intersection is non empty. That is what I want to see, that I start from here whether I mean. So, this is the overall problem that means from here if I have a reach set which is this from. So this is I and let us say this is post*(I) and let us say this is so that is how I went and this is R whether there is a nonempty intersection or not that is what we wanted to check.

Now while this may be difficult if I am trying to see if it is non empty or not. Essentially, I am looking for one run, one run which starts there and ends in this intersection. So, while doing this may be difficult it may be easier to guess some states from where I can apply some heuristics to make a good guess and find out a valid run which lands up here. So, I take one state which is a good guess and I see what is the valid run which eventually goes to a point which is both a member of this set and this set then also I am done.

So, while the overall problem may be undecidable the verification problem may still work because I can I mean eventually I am looking for some run or in or there is another thing we can do also we can see how the reach set is getting constructed. That means how this post one, post two, sets are getting constructed and from that if I can come up with a closed form expression of this post operation. So, let us understand what it means through an example.

**(Refer Slide Time: 25:44)**



## Example LHA

$$\{(s_1, \exists i \in \mathbb{N} (i \geq 1 \wedge x \le i \wedge y = i \wedge z = 1)),$$
$$(s_2, \exists i \in \mathbb{N} (i \geq 1 \wedge x \le i \wedge y = i \wedge z = 1)),$$
$$(s_3, \exists i \in \mathbb{N} (i \geq 1 \wedge x = 0 \wedge y = z + i \wedge i \le y \le i + 1))\}$$

$$\hookrightarrow \exists i_1 \ldots \exists i_j \in \mathbb{N} . R(i_1, \ldots i_j$$

This is the invariant, this is an example we are drawing and now I am writing the flow relations. So this is taken from the same paper we talked about earlier. Again, the invariant again I am writing the flow relations this is a transition guard. So, see this is the initial valuation x = 0, y = 1, z = 1, invariant for location one let us x less than y let us call this $s_1$, this $s_2$, this $s_3$ and the gradients are x dot is 1 and the others are 0.

And this is the transition guard x = y then I will switch to $s_2$ where the invariant condition is this. And again, the flow relations are so you can see x will, now decrease the others are remaining same and then when the guard condition is x = 0 I have a reset of z to 0 and here is invariant is z less equals 1 and x does not have any flow value I mean its zero and y and z have time lapsing I mean they have a constant flow value of one.

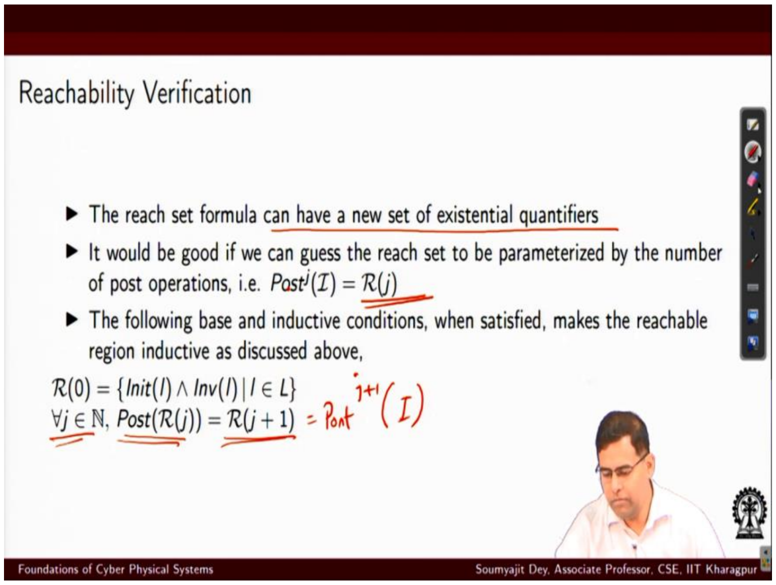
And in state $s_3$ I am and then there is a guard condition of z = 1. So when z was reset and z increases up to one then I make a transition to s 1. So, the idea is that well I can see how this reach set is growing and if I am able to write a closed form expression for this post operation. Now if you can see for this example automaton, we can actually write reachable region definition which will be given like this.

So, I can either be s in s 1 with the valuation sets I mean they have been captured by a predicate looking like this. So, i is a natural number which satisfies i greater than 1 and x less equals i and y = i and z = 1. So, I can be at s 1 and I will have these conditions being satisfied here or I can be at $s_2$ or I can be at $s_3$. $s_3$ will have x = 0 and as you can see, I mean the guards I mean I get into $s_3$ only when x = 0 and here x does not change.

So, x remains at 0. So that is how this reach set would look like after a few iterations I can write it in this kind of a form that the reachable region is described by this relation. So, in general what I am really looking for? I am trying to look for a heuristic which will kind of guess this reachable relation and write it in the form of a linear predicate where I could bring in, I mean the general form here is like this that you bring in some new variables.

Let us say here it was just I so in general it can be some $i_1$ some there exists some $i_j$ etcetera belonging to the naturals and you will have a predicate defined over this variables. In addition to the automata variables which are already there I mean the x y z etcetera that I have. So, overall, I am looking for this kind of expression and whether I can create a algorithm a heuristic which can do a few iterations of the reset construction that means the creation of the post relation and create this kind of an expression. So, that is what I am looking for here.

**(Refer Slide Time: 33:30)**



## Reachability Verification

- The reach set formula $\underline{\text{can have a new set of existential quantifiers}}$
- It would be good if we can guess the reach set to be parameterized by the number of post operations, i.e. $Post^j(\mathcal{I}) = \underline{\mathcal{R}(j)}$
- The following base and inductive conditions, when satisfied, makes the reachable region inductive as discussed above,

$$\mathcal{R}(0) = \{Init(l) \wedge Inv(l) \mid l \in L\}$$
$$\underline{\forall j \in \mathbb{N}}, \underline{Post(\mathcal{R}(j))} = \underline{\mathcal{R}(j+1)} = Post^{j+1}(\mathcal{I})$$

Foundations of Cyber Physical Systems

Soumyajit Dey, Associate Professor, CSE, IIT Kharagpur

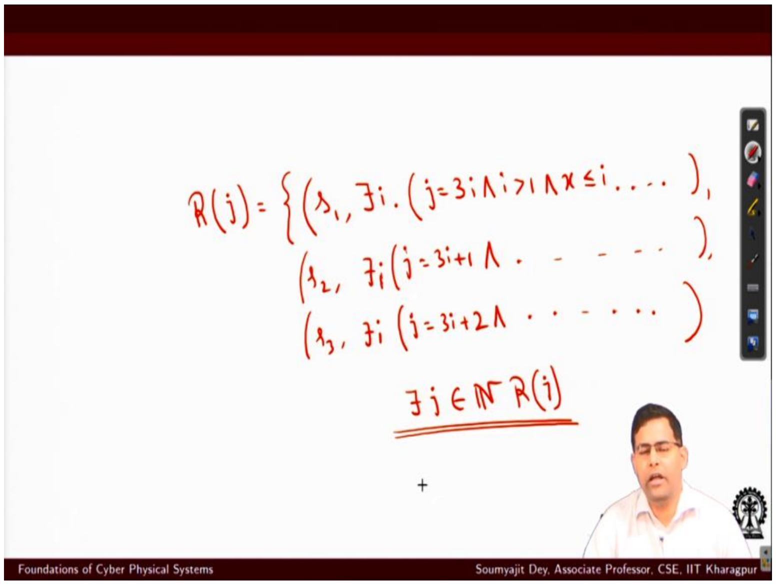
So, that is what we discussed that the reach set formula can have a new set of existential quantifiers along with the automata variables. Now the best possible situation can be that if we can guess the reach set to be parameterized by this number of post operations. What it means is if I can directly write convex relation a predicate which is parameterized by the number of post operations I am doing.

That means starting from the initial region i, if I take j number of discrete transitions my reachable set is given by this predicate R which has j as a parameter itself. Now what is the advantage of that? The advantage is the; I mean I can then just use some method maybe a theorem proverb or some other automata theoretic method to figure out whether the following base and inductive conditions are satisfied.

So, I will just check that will R 0 is given by this condition that Init l intersection Init l like this and for all j the I mean any natural number I mean post R j is defined like this I mean because if this holds then post j is R j. And post 1 post applied over R j is nothing but post j + 1 over I which is R j + 1, I can just write this. So, then I have a straight forward close form expression which gives me the reach set. Well even if that is not possible.

Well in this case we show that it can be possible, even if that is not possible it may help if I can define such an inductive invariant and we will see that how such close form expressions and discovery of invariance in this way can help me to verify the safety of the hybrid automata. We will see that.

**(Refer Slide Time: 35:43)**



$$R(j) = \{ (s_1, \exists i . (j = 3i \wedge i > 1 \wedge x \leq i, \ldots ),$$
$$(s_2, \exists i (j = 3i+1 \wedge \ldots ),$$
$$(s_3, \exists i (j = 3i+2 \wedge \ldots )$$
$$\underline{\exists j \in \mathbb{N}\ R(j)}$$

Foundations of Cyber Physical Systems

Soumyajit Dey, Associate Professor, CSE, IIT Kharagpur

Now coming back to this example, we can actually see that I can have such an expression like we just described. So let us just define such an expression. So, what we will do here is we will bring in another variable j and we will add constraints like this. So, this essentially characterizes the three states I mean let us see that how I can define this reach set as a parameter of the number of transitions I am just taking.

Let me just rewrite this. So this remains same and then I have. So this is forcing the value of i to be of a specific form and along with that I will have all the previous constraints like i greater than 1 and x less equals i and the other constraints which I just wrote earlier or we will have for $s_2$ similarly, we will have j = 3 i + 1 that again makes this as a specific constraint here along with whatever we had earlier and s 3.

So, it is the same i keep on repeating here with j = 3 i + 2 and all the other constraints that we wrote there. So, what happens is now this guess that I have that there exists some j which belongs to the set of natural numbers R j. Now this is basically representing the number of iterations. So, overall, this is how we can create a reach set definition parameterized by the post operation and that can help me in the verification process. So, we will end our lecture here and in the next lecture we will continue from this point, thank you.