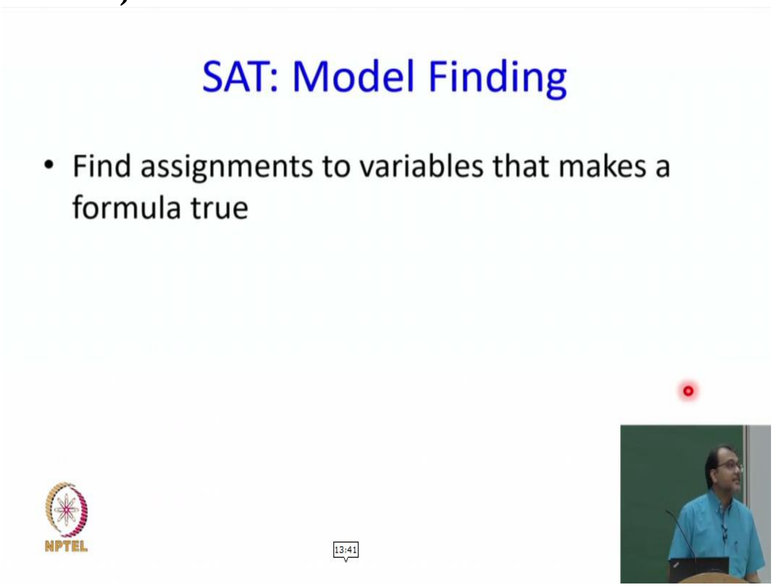
Artificial Intelligence
Prof. Mausam
Department of Computer Science and Engineering
Indian Institute of Technology Delhi

Lecture-47
Logic in Artificial Intelligence: Reduction to Satisfiability Problems, Part - 6

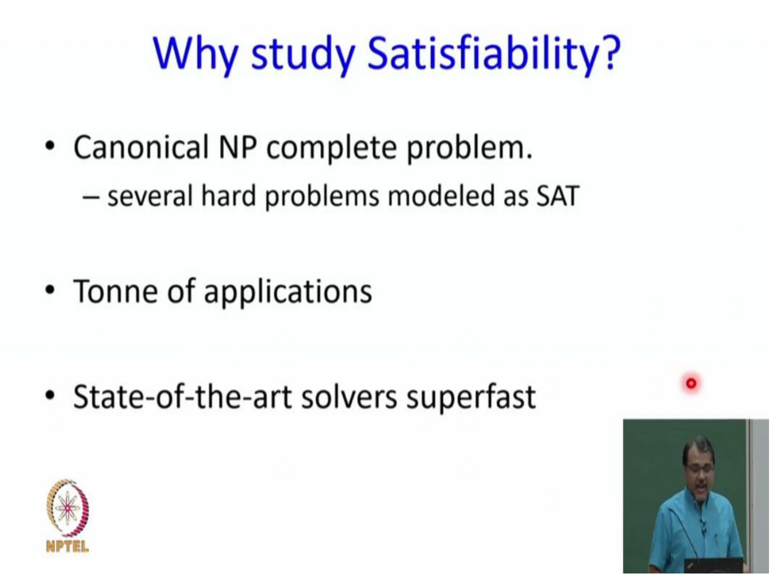
Now I am going to move on to a different set of problems. So up until now, we were talking about given a set of formula, what can I prove? Can I prove a particular formula? This was the inference procedure we were working on. And again, that is beautiful. We know we have got 2 strategies they are many, many other strategies. But now we will talk about model finding.

(Refer Slide Time: 00:40)



Model finding means find the assignment to variables that make the formula true. So I give you a set formula, like A and B, and I want you to find as a model for it. Any one assignment of variables that makes the formula true if my formula is A and B which model can make this formula true equal to 1 = 1, if my formula is A or B how many possible models do I have? 3 And I can give anyone of that. That is the model finding problems. Now, why do we want to study it is also called the satisfiability problem given a formula find up as an assignment of variables that makes this formula satisfiable. It is the canonical NP complete problem.

(Refer Slide Time: 01:25)



Now, we have discussed this before. What is NP? NP stands for? Yes, what is your name? Yes Abisheik is a non deterministic polynomial time algorithm. It is a polynomial time algorithm, but in this non deterministic world now, it will take you some time to appreciate what does NP really mean? But here is your homework serious homework. Go home and read about NP completeness.

Even if you do not understand it, read about it. The next time somebody asks you what is NP you should not say not polynomial, because that is not true. In fact, if you know that it stands for not polynomial, you will be the most famous person in computer science today. Because the biggest question in computer science today is P versus NP, because P means polynomial time and if NP means not polynomial time.

Then we are done basically as a computer scientist, the AI folks are still useful. But that is the question for theory of computer science, the entire theory of computer science that is the I would not say million dollar question. No it has to be at least a billion dollar question if not more. But what we know so far is these NP complete algorithms you can verify their solution in polynomial time but so for you cannot compute the solution in polynomial time.

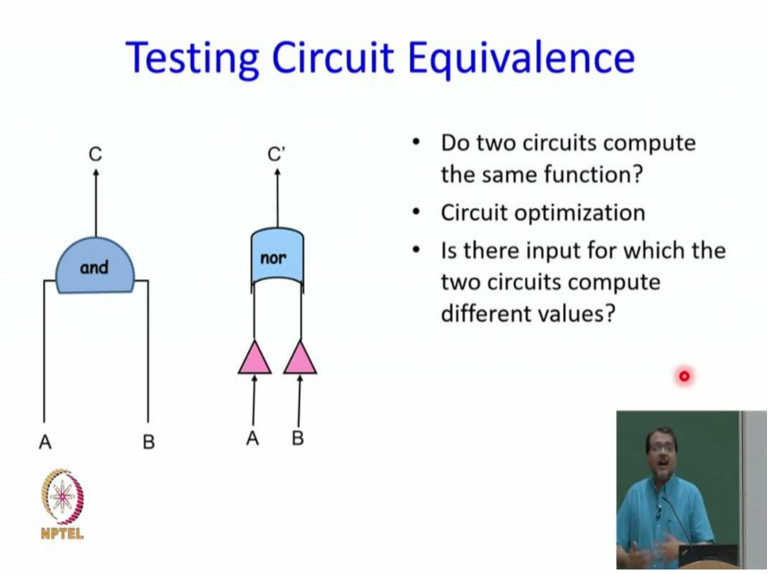
But you can compute the solution in polynomial time if you were given this computer, which could take the decision at every step, none deterministically and such a computer is called abstraction of that is called a non deterministic Turing machine. That goes into a certain level of theoretical depth which you will study in a theory of computation or the complexity goes, we will not talk about this, but we will say that we love solving NP complete problems and for that you need to have some idea about what a non NP complete problems.

They are those problems which you have not been able to find a polynomial time solution so far so, therefore, only solutions that exist for them are exponential time right now. And this is talking about NP. But what is NP complete? NP complete says that if you solve this problem, any NP complete problem that any other problem in NP can be converted into this problem and then you can use a solution of this problem to solve that problem.

And we will show you examples because it is such a canonical problem, because everything can be converted to this problem. Therefore, if you have a good solution for the SAT algorithm SAT is the short form for satisfiability then you have a good solution for a lot of problems in the world. And therefore, it has a tonne of applications. Now, what is beautiful from an AI point of view this is all theory so far but what is beautiful from an AI point of view is the state of the art solvers.

The current solvers for satisfiability problems are amazingly fast, like blazingly fast and how did they get there? We will talk about some insights. But let us first see why satisfy so important. So, let us take some examples and try to convert those problems into a CNF formula or a logical formula such that if I can give a satisfying assignment to that formula, I have solved my original problem.

(Refer Slide Time: 05:29)



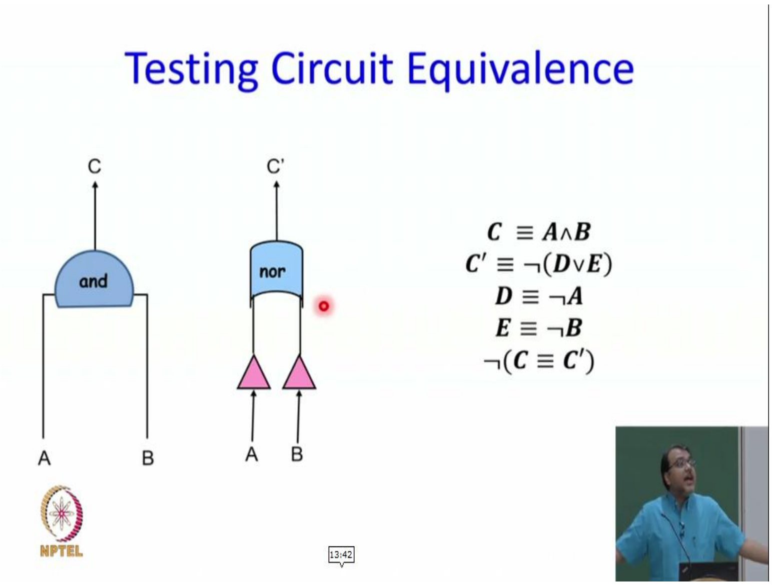
And here is one such example I give you 2 circuits to complicated circuits. These are not complicated, but let us say I give you 2 complicated circuits. In this case, you know the first circuit is A and B are anded to give C in the second case A and B are knotted and the nored to give C prime. And now I asked the question, do circuits compute the same function? This is very important in circuit optimization.

Suppose I have a formula that somebody gave me let us say I want to optimize it with minimum number of you know gates and so on and so forth. And then after I done all this, I want to make sure that I did not make a mistake, the original circuit and the circuit are actually equivalent, how do I do it? I do it with what is called in this case where I do it with satisfiability problem. So, now I will turn it around so when I want to ask the question, are they equivalent.

I will instead ask the question, is there some assignment which gives different values to C and C time? Now, why do I ask this question? Why cannot I ask are the equivalent somebody has an answer, who has an answer. Yes, you are Ansh. In that case, I will have to check all the possible as I mentioned, what can satisfiability give me? Any one possible assignment see satisfiability is the it is limited, you give it a problem, it will give me one possible assignment, you can always use theorem proving to prove that C = C prime.

If it is a horn clause or if you can use resolution you can also use resolution here, nothing wrong with it but satisfiability if you want to use satisfiability, you will have to use a problem formulation in which one assignment does the job. So let us do this. So, the first step we will do is we will input the current logical formulas.

(Refer Slide Time: 07:34)

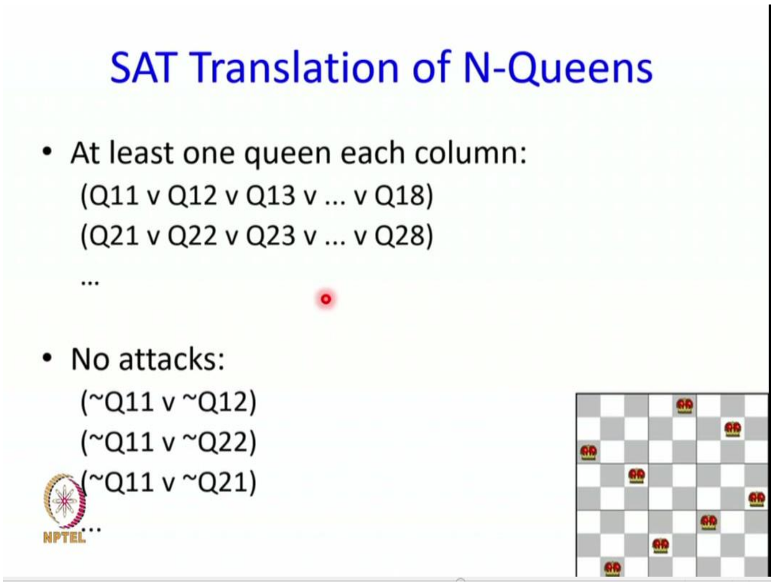


So, you will say C is equivalent to A and B C prime is equivalent to not D or E D is equivalent to not A E is equivalent to not B I have defined intermediate variables D and E for this point. This is good so far. Now, the next question you want to ask for the satisfiability problem to check circuit equivalence is what else do I add to my formula? Should I add C equivalent to C prime? If I add C equivalent to C prime and ask for a satisfying assignment, what will happen?

It will give me one satisfying assignment It is possible that there is one satisfying assignment, but it is also possible there is some assignment which makes CNC false I mean C not equal to C prime and we will not know about it. So, we have to somehow reverse it. So, we will add not of C = C prime of C not equal to C prime which when you use all those tricks to convert it into a clause. Now, if I have a satisfying assignment what have I done?

I have got as an assignment where they are not equivalent, and that means that they are not equivalent in the however, if I do not get any assignment then I know that there let us take another example we will do several of these so that you can learn the importance of SAT satisfiability problem. So I will let you think. So we need to create a SAT formula such that it is satisfying assignments gives me and solves the N Queens problem.

(Refer Slide Time: 09:23)



Solve the 8 queens problem in this case. Now, whenever I have such a problem, the first thing I need to do is to figure out what will be the guess, logical form representation CNF no they will always have a CNF form eventually, but a CNF form is over a set of variables. So, the first thing you need to know we have to model the world see, this is this is a real world N queens problem there are real chess board there is a real queens whatever.

But now we are creating an abstraction in the logical representation we are giving it this information to the agent in logic. So the first thing we need to know is define symbols. What is each variable? What does it mean? Now, remember, in logic, a variable is always binary it is only true or false. So what true false variables can be defined, which might help us in solving the N Queens problem? So this is going to be interesting? One assignment of those variables should tell me exactly where each queen is placed.

And it says a 01 assignment. Can somebody think of such variables, not clauses, just variables where queen is present or not present as in its present in the clause in the chessboard at a specific position. So, you say that I will define XI or QIJ such that QIJ is true if the queen is present in IS row JS column or IS column JS row I do not know exactly how this thing is written. So, we will figure it out.

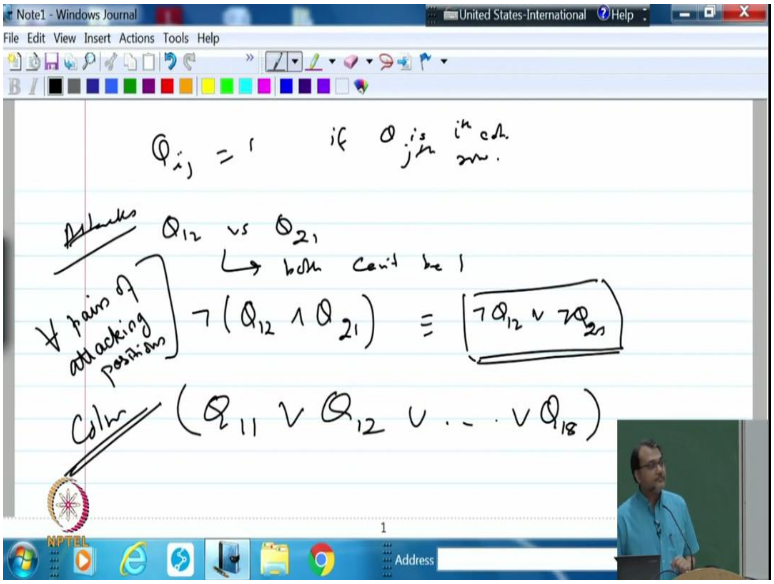
That we see how this thing written is IS column and JS row. I think this is how it is it. So we define this let us say we define this or how many variables will be defined for 8 queens problem 64 variables. So fast able. Now I need to somehow encode the constraints of my problems such that one satisfying assignment gives me the solution for the N queens problem. Should not tell me what should these constraints be? And how do we represent them in logic?

People who have not spoken let us you know get towards this is not too hard to think about it, you can get it. So let us see somebody who has not spoken so far. Somebody who is feeling brave enough to suggest an answer you want to say something? Not yet. This is not very hard. What is the major problem with N queens? What do we want to make sure no 2 queens are attacking each other? How do we say this? Now we may not be able to state in general, but we can say it for pairs of positions.

So now that was that is your hint. So now somebody, raise your hand. Yes. What is your name? Asha yes Asha so difference between I 1 and I 2 and J 1 and J 2 should not be equals you are trying to model it in general for a diagonal, ? That is what you are trying to do very good. But remember that our language is and all implications of whatever we say, we have to say it in that language, we are not given the minus operator. And we cannot even go to the index of the variables, the variables, we have 64 variables now.

It is much simpler than what you did. You did something very complicated in your brain, you have to do something much simpler. Let us take 2 positions. Let me take some positions. Let us say 1 2 and 2 1 very simple forget general diagonal. We want to say that what do you want to say about Q 1 2 and Q 2 1. Both cannot be 1 very good. That is it. That is the only thing we have to say we have to say both Q 1 2 and Q 2 1 cannot be 1 at the same time. What logical formula gives me that? Not off. Let us do this together so that this is actually really important. So what formula tells me that?

(Refer Slide Time: 14:40)



So we defined Q ij = 1 if queen is in is column, js row. Now we are talking about attacks. And let us say we talk about Q 1 2 versus Q 2 1. So we want to say that both cannot be 1. And how do we say it logically? Not of. Perfect not of and of Q 1 2 and Q 2 1 basically it says, if you cannot have Q 1 2 and Q 2 1 at the same time which is equivalent to saying not of Q 1 2 or not of Q 2 1 and then let us think about what this means.

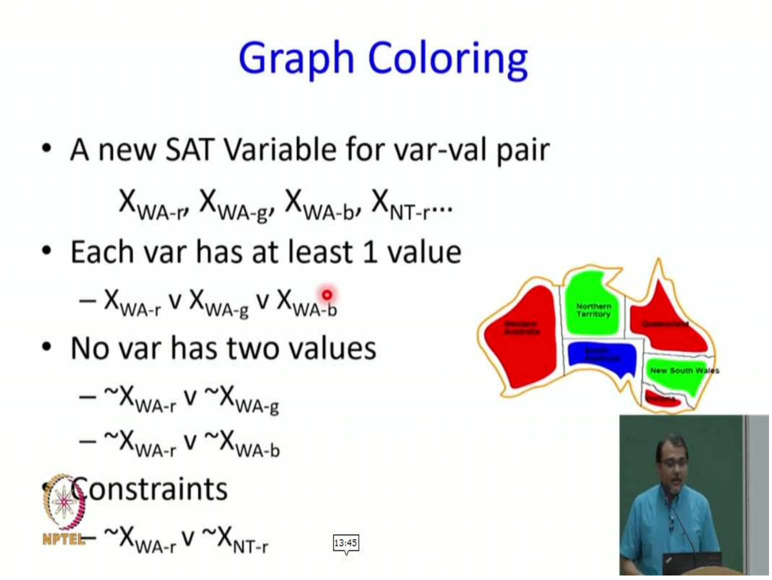
It basically says either Q 1 2 is false or Q 2 1 is false, which is equivalent to saying both cannot be true. Now, let us suppose I wrote this formula for all such pairs of attacking positions. For all pairs of attacking positions, I wrote this is it sufficient? Is this enough? This is the most fundamental thing this makes SAT a little complicated and but this is very important. Suppose I gave this problem to my SAT solver and gave me a solution will it always be the correct solution?

Whatever I not captured here or what solution could it give which will be meaningless? No Queen on the board excellent. I am so happy you got this. I put my SAT formula could say do not put anything. You are saying do not put Q 1 2 or do not put Q 2 1 let me not put anything. I have not said that. Please put queens. How do I say please put queens all of the all the rows should be 1 or all of all the columns should be 1.

So, for example, I want to say that they should be at least 1 queen in the column. So, how can I say it first column, either Q 1 1 should be true or Q 1 2 should be true or Q 1 3 should be true or Q 1 8 should be. And now, if we are able to solve this problem, that means it would satisfy all these clauses also. So, that means it would get a queen in every column and it would make sure that no 2 pairs of queens are attacking because of the previous formula and so, overall we would be able to solve the problem at home.

One more example this is fun stuff by the way. It is possible that in some exam we give you a new problem and say give me a SAT encoding for this. This is really saying I have a physical problem I want to model it computationally. This is my model. It may be a good model not good model, but it is a model. And because these are formal problems, this is an accurate model. In the real world, it is an approximation in a formal world. It is no there is no approximation. Let us do our favorite map of Australia problem.

(Refer Slide Time: 18:20)



And this is also very interesting, because so, here is a simple thing is a SAT problem or CSP problem. See what is in a CSP, you have a set of variables, a set of domains and a set of constraints, what is the SAT of a set of variables? The domains are true false, and you are set of clauses which are like constraints. So eventually every SAT problem is a CSP problem. But is every CSP problem or SAT problem?

Well, every discrete CSP every discrete finite, CSP can be converted into a SAT formula. And here is how this is called proof by example. So, do not use that in your exams. If I asked you to do something and then you say, let us say for example, and then that is it. That is not a proof? But in classes I am allowed. So I am giving you the map of Australia I my goal is to create a SAT formula such that the solution of the SAT problem is the solution of the CSP problem, what will be my variables?

Now you have a hint already from the previous slide. What should I do my variables? I have 3 colours, 7 states. How many variables do I have 21 variables. I have a variable a new SAT variable for every variable value pair. In this case, my variables are Western Australia is red or Western Australia is green or Western Australia is blue and Northern territory is red, Northern territory is green and so on. Now, I need to say something some basic things about CSP, and then how to model constraints.

Let us talk about the basic things in the CSP. What are the basic things I need to say? Somebody raise your hand, not constraints, just basic so that, you know I do not get into all falls and things like that. What are the obvious things I need to say so that it is a well formed SAT problem? Every state needs to have at least one colour. So give me one example. Let us say Western Australia, what will be the clause X WA r or X WA Green or X WA blue. There is one other obvious constraint that I need to add.

So it needs to have one colour and at the back, yes, what is your name? Kiran Yes, Kiran it should (())(21:12). Because up until now, we have only said it should have one. But we have not said it cannot have more than one. It could decide to have all colours in Western Australia then things will become crazy? Because it is a SAT formula you have to think like SAT you are saying that we are giving no initial constraints. So now you have to add all the constraints but these are general constraints, you will learn to add this general constraint.

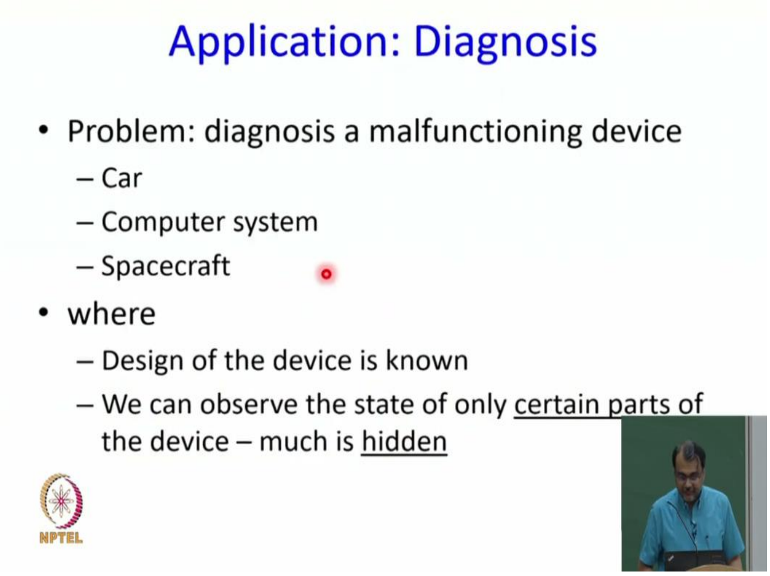
So how do we add it a variable should not have 2 colours? Pairwise ends and the negation of that or alternatively, the same idea that we had last time that you know Q 1 2 and Q 2 1 both cannot be true. In the same way we will say WA r and WA g both cannot be true WA r and WA b both cannot be true WA g and WA b both cannot be true. And now, finally, last but not the least, we should add the neighborhood constraints.

And give me any one neighbourhood constraints somebody not of X Western Australia red and X Northern Territory or red very good. So, this is what you have to start getting used to. And the reason I am spending time on this is because this is an important skill they will possibly come a time in your later life when you are given a very complicated constrained problem without uncertainty and these are all deterministic problems actually it is a formal problems there is no notion of uncertainty probability and so on and so forth.

This would not come up in in robotics, but let us say you thought you wrote a software and you want to check whether the software is accurate or not. It is a mission critical software it is going onto a plane on NASA or the you know the rover on NASA. It is going to be used by Boeing, what will you do? You will use SAT solvers, believe it or not, and you will have to convert your problem, the formal problem model checking problem into a SAT problem and then give it to a SAT solver, this is exactly what you will have to do.

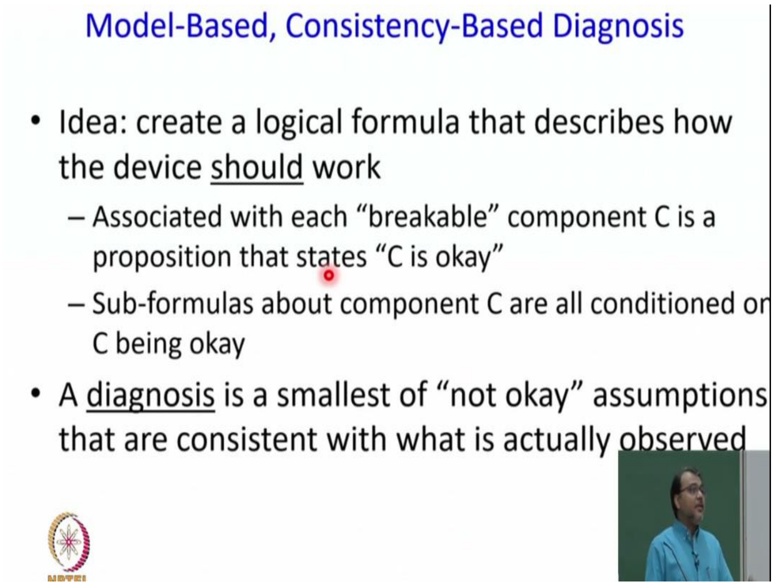
That is why I am pushing on you to learn this skill. So, the suggestion that was given, what is your name Abi uday by Abi uday is that not Western Australia red or not Northern Territory red, and you do it for all pairs of states in all colours. And this allows us to formally model the graph colouring problem in SAT.

(Refer Slide Time: 23:39)



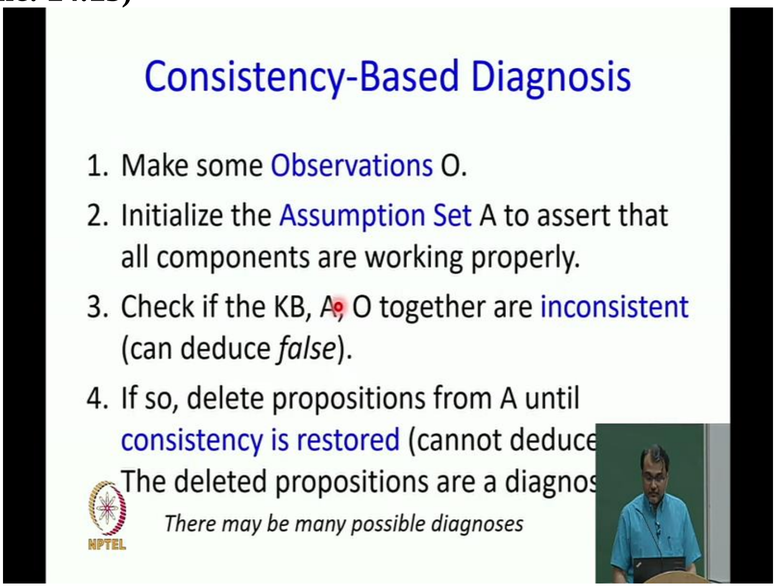
There are many other applications. For example, a diagnosing a malfunctioning device is called model based diagnosis. If the design of the device is known and already expressed in the form of a logical representation, and what we do is we can observe, unfortunately, only observe states of certain parts of the device a lot of them are hidden.

(Refer Slide Time: 24:00)



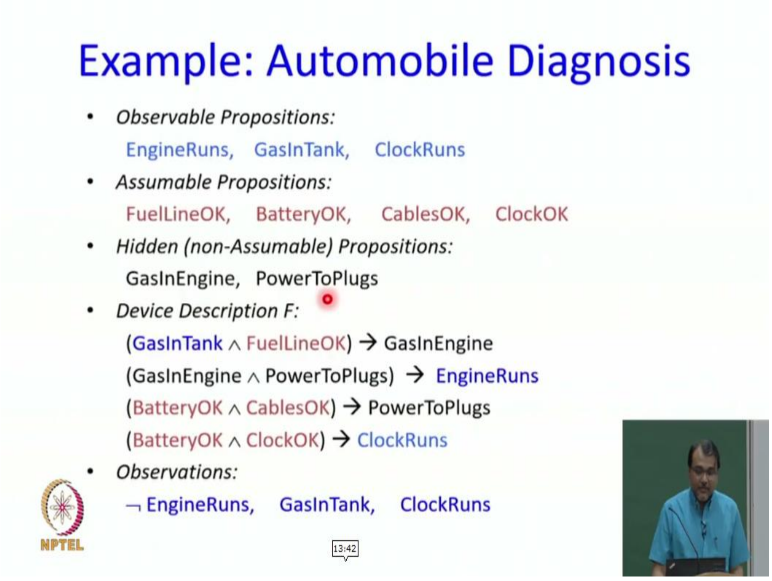
And that leads to this idea of model based diagnosis where we create a logical formula in which how the device works. And then we have some components that can break and each formula says whether this particular component is okay or not. And the diagnosis is the smallest set of not okay assumptions that are consist consistent with what is actually observed.

(Refer Slide Time: 24:23)



I can skip some of the details, but highly at the highest level, you make some observations. You initialize a set to assert it all components are working properly you first check that, that is not possible. Once you check that SAT is not possible, you have been able to deduce false, then you delete propositions from your assertion one at a time to check if this was related, will that now be consistent if it will be consistent, and that is A diagnosis and then you go into the car and you actually check that component somehow, and then you try to fix it.

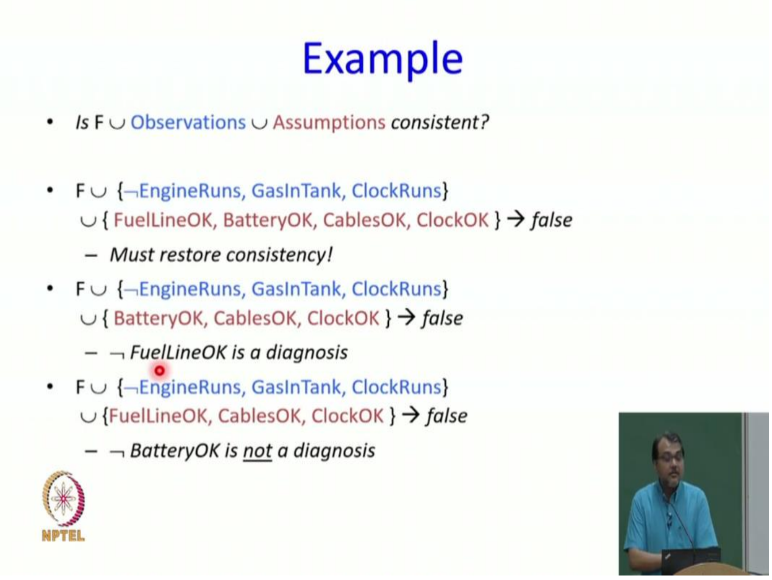
(Refer Slide Time: 24:56)



So for example, if I have propositions like engine is running gases in tank clock is running this you can observe but then there are some internal propositions like whether fuel line is working OK whether battery is working OK whether cables are OK whether clock is I do not know about this and then there is some things you might not even be able to you know check and that they are somewhere in the middle they are not even the diagnosis there is something in the middle like gases in the engine or not power is coming to the plugs or not.

And then you have a model for the device that if gas is in tank and fuel line is OK then gases in the engine and if gases in the engine and plugs are getting power then engine is running and if batteries OK cables are OK and power is coming to plugs and if battery is OK and clock is OK then clock is ready. So, now you this is your model for the problem. And you suddenly observed that engine is not running but there is gas in the tank and clock is running. And now you have to hypothesize what must have happened in the car to figure out we have to look we have to try to fix the problem.

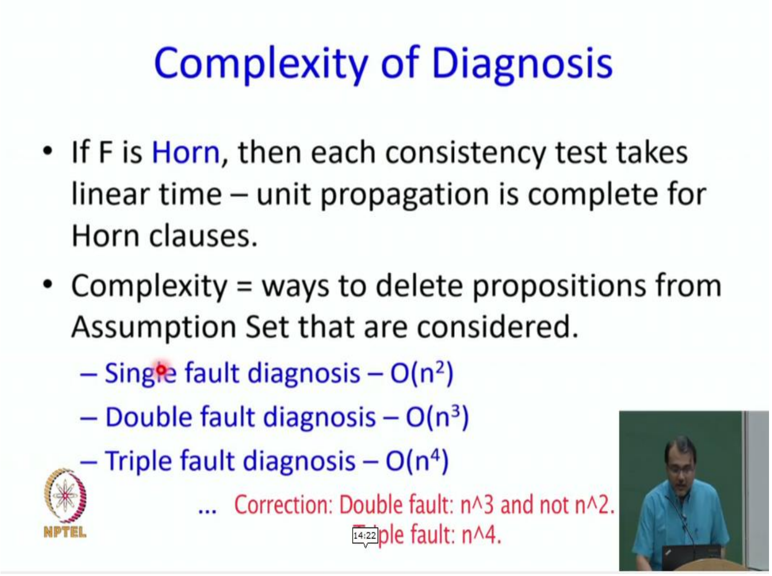
(Refer Slide Time: 26:03)



So, the first thing you say is F union observations union assumptions consistent of course that is not consistent. Because if every assumption was true assumption server, fuel line is OK battery is OK cables are OK clock is OK then engine should have been running engine is not running. So now you say, let me think that maybe fuel line is not OK. So let me remove fuel lines of my assumptions and then check that is it a diagnosis and if I can have a satisfying assignment there, then it is a diagnosis.

Similarly, I see it is battery the problem? And does it give me to false and if that gives me false then it is also it is not a diagnosis. So in the first case this becomes true because fuel line OK can be a possible reason for why engine is not running. So it is a diagnosis in the other case battery OK it does not lead me to battery not OK it does not lead me to engine is not OK running so that is not a diagnosis.

(Refer Slide Time: 27:12)

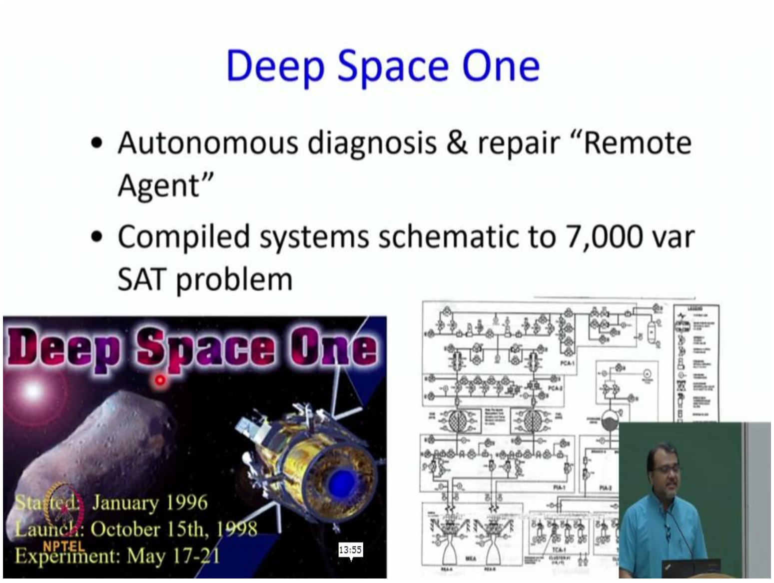


And so typically F is a horn clause then consistency text test takes linear time and if you only remove one so that remember it was linear time on clause (())(27:29) time. So, if you only remove one diagnosis then the total time taken is order n square so, you are saying only one thing went wrong at a time? I do not know if you realize if you see how is MD you will see. Have you seen how is MD? Some people have seen how is MD you should see how is MD is an absolutely fantastic.

See really talks about rare medical conditions and how they diagnose it by making mistakes. So he always believes that there is one problem that is causing everything. You cannot say that you know there is a problem in his nerve and there is also a problem in his lung and also there is a problem in his brain. That does not work there is one underlying diagnosis which is really the problem due to which the particular symptoms are coming out.

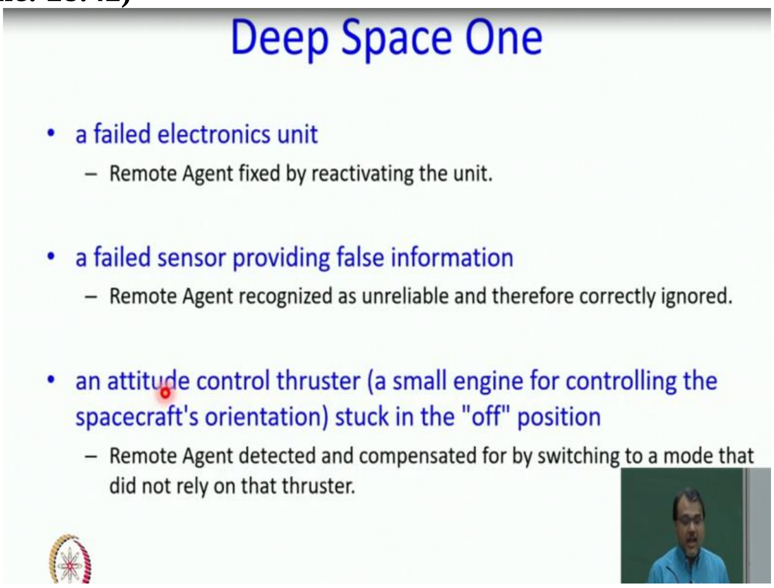
So, in the same way, if you are looking for only one fault, then it will take order n square time because you will ever try n of those problems and every instance will take order n time and double faults will be N square and triple fault will be n cube.

(Refer Slide Time: 28:24)



I think we have discussed that earlier in deep space one. This spacecraft we they used, NASA used a satisfy SAT solver, they compiled the whole system the schematic into a 7000 variables SAT problem and use a SAT solver to do diagnosis.

(Refer Slide Time: 28:42)



And of course, we have discussed this earlier, but they did not they were not able to test the diagnosis system in the real world because nothing went wrong. And so the simulated faults and when they simulated false the diagnosis system was able to diagnose it and the system was able to repair it these are the 3 faults that they simulated, they failed an electronics unit. The system reactivated it. It was called remote agent.

They failed a sensor it was starting to provide false information remote agent recognized it and stopped using that particular sensor. And then some thruster was stuck in an off position it was not turning on. So the remote agent detected it and compensated for it by switching into a mode that did not rely on that thruster. SAT solvers have had an amazing impact on our community and in the real world. And in the next class, we will talk about how to solve satisfiability problems. In this class we learned about how to use how to model a problem as satisfiability.