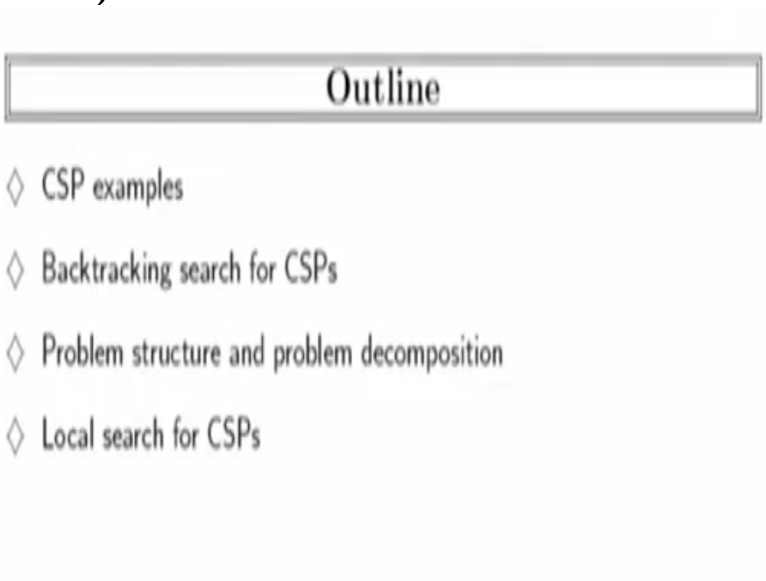
Artificial Intelligence
Prof. Mausam
Department of Computer Science and Engineering
Indian Institute of Technology - Delhi

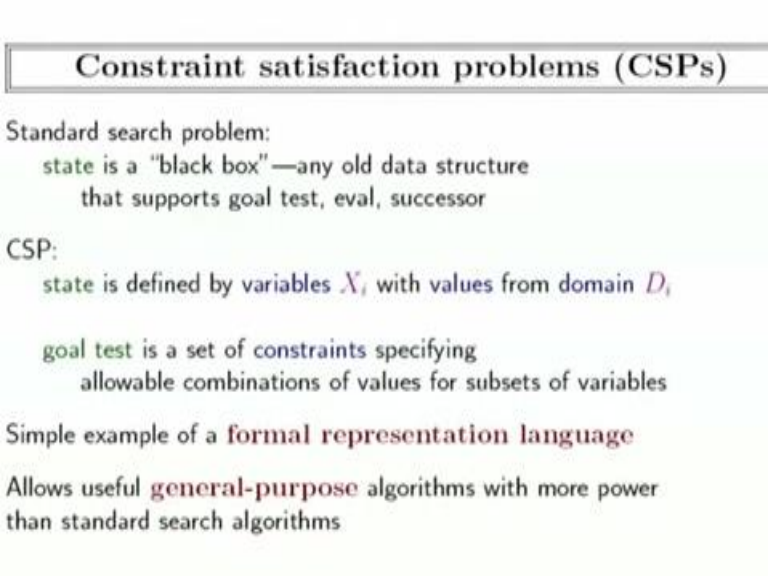
Lecture-6
Constraint Satisfaction Problems: Map coloring and other examples of CSP, Part-2

(Refer Slide Time: 00:22)



So let us get started, so today our goal is to learn about constraint satisfaction problems. We will talk about various examples of constraint satisfaction and the representation which is very simple backtracking search, which is the paradigm to do search in constraint satisfaction then we will talk about how to improve the search by various heuristics by the very important idea of reasoning or inference. We will talk about how different problem structures behave differently and what we can do about them and we will touch on local search, although we will come back to local search more carefully in the context of logic so let us get started.

(Refer Slide Time: 00:54)



A constraint satisfaction problem is one of the standard search problems where instead of saying that state is a black box, we say that state is defined by variables and values. It is extremely simple like in the case all of you is a very all of your variables. Vishwajith is the variable, Kiran is a variable, Vipul is the variable and your values are the chair numbers that you are sitting in, Kiran has value 15, Vishwajith value 35, Vipul value now I have forgotten.

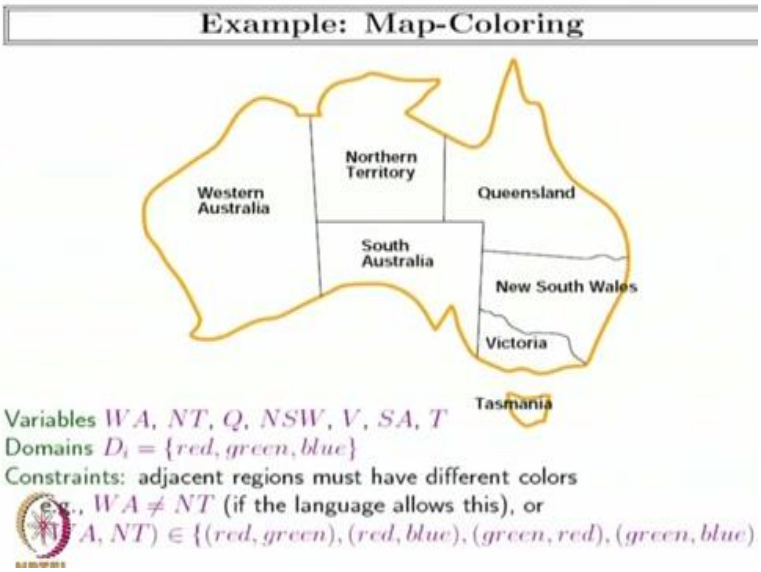
So, the point is that in this way we can represent this class. Let us say we talk about N queens can be think of it as a variable value problem each variable would be variable i would be queen position in ith column and its value would be the position. If you think of n puzzle, you can again say tile position is a variable and its value is the actual position etc., so, now, if you think about it, most problems many problems.

You will be able to cast where you say that each state has a certain set of variables and each variable has a certain set of values and a complete assignment to all the variables creates a final state a complete assignment to all the variables creates a world state and if we are doing local search we will be doing local search in the world state. But if we are doing systematic search where we are adding 1 variable at a time, my node would be as partial assignment.

So, as the slide says this is a simple example of a formal representation language in the field that which we are getting earlier and it allows for general purpose algorithms with more power than an atomic agent that is very important. Because in an atomic agent, there was only a state with

only a number, there is only so much we can do now that state is more than a number, it is a set of variables and values, we can do more we can understand the structure of the state and perform better in our task.

(Refer Slide Time: 03:20)



And for this particular set of slides, we will carefully look at the map of Australia and we will try to solve the graph coloring problem or the map coloring problem. So, first of all it will help to memorize this we have seven states in Australia, Western Australia on the west, Northern Territory on the top, South Australia in the middle on the south. Then here we have Queensland, New South Wales, Victoria, and Tasmania. Tasmania is an island.

And these you have to sort of memorize it will help to memorize because we will use that as an example and let us say I give you 3 colors, red, blue, and green and I want to color, I want you to color the map such that no 2 adjoining states have the same color. So this is a very standard graph theory problem and if we wanted to pass it as a constraint satisfaction problem, our variables would be 7th.

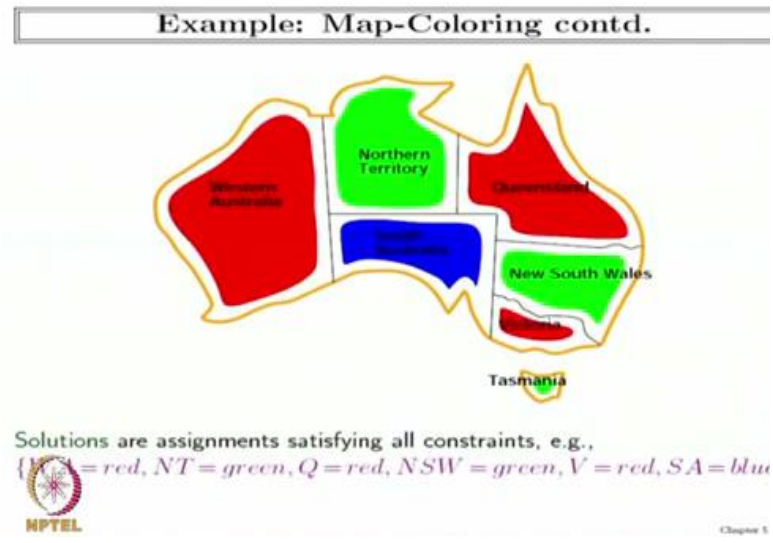
Western Australia, Northern Territory, Queensland 1 variable for each state, like Australia state not to be confused with AI state and the domains would be red, blue, green these are the 3 colors that we are allowed to use for coloring each variable and then what will be the constraint, the constraint would be that Western Australia cannot be equal to Northern Territory. Northern

Territory cannot be equal to Queensland, New South Wales cannot be equal to Victoria and so, these are in this case inequality constraints which represent that 2 states.

Which are adjoining to each other in the map cannot have the same. Now how do we represent this constraint is also interesting you can represent this constraint with a simple inequality if the language allows it. We have not really defined a Constraint Language and different people study different sets of constraint languages and this whole area is called constraint processing and so on so forth it is a very active area of research.

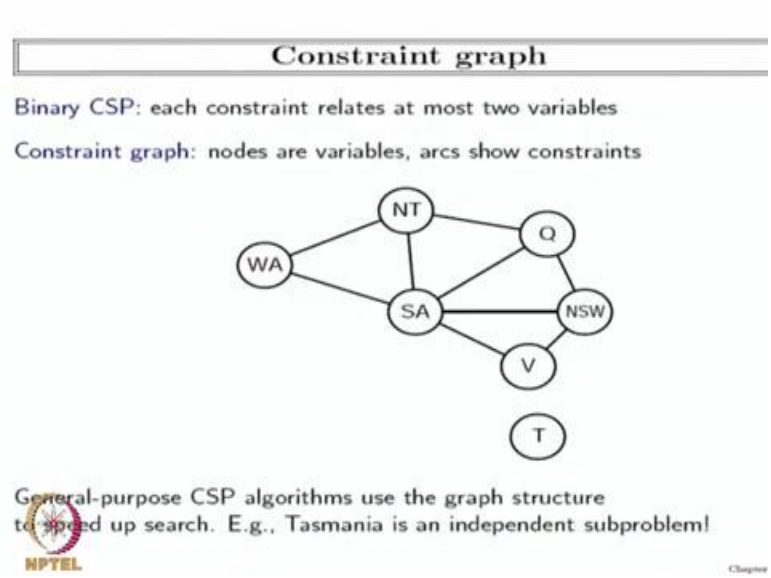
It is a full conference of its own, it is called CP constraint processing that is a good very good conference. But you may say that not equal to is not part of my language, but a numerating the sets of possibilities is part of my language and I can say that my language can say that Western Australia common Northern territory can take values from red green, red, blue, green, red, green, blue, blue, green, blue, red, but not red, red, blue, blue, green, green, so they cannot be similar, but I enumerate all the possible correct pairs of values.

(Refer Slide Time: 06:00)



And this is one possible solution to the problem you know you label color Western Australia red Northern Territory green, so Australia blue and so on and so forth and this is my solution the solution is a specific assignment to each variable such that all constraints are satisfied.

(Refer Slide Time: 06:25)



Now, often people in constraint processing think of it as a constraint graph. In a constraint graph, each node is a variable and each edge determines whether there is a constraint between those 2 variables or not. This kind of a constraint graph is a binary constraint graph, where each constraint relates at most 2 variables and such a constraint satisfaction problems are called binary CSPs now, you may say Binary CSPs to specific.

Because each constraint is only between 2 variables but in fact, you can prove or you can there are procedures which can convert a non binary discrete CSP into a binary CSP. So, you we are just we can just think about studying binary CSPs and not worry about others. Of course, there is some lots of new variables need to be created etc so, you know there is a blower and so we will but in our course, we will not talk about more than binary.

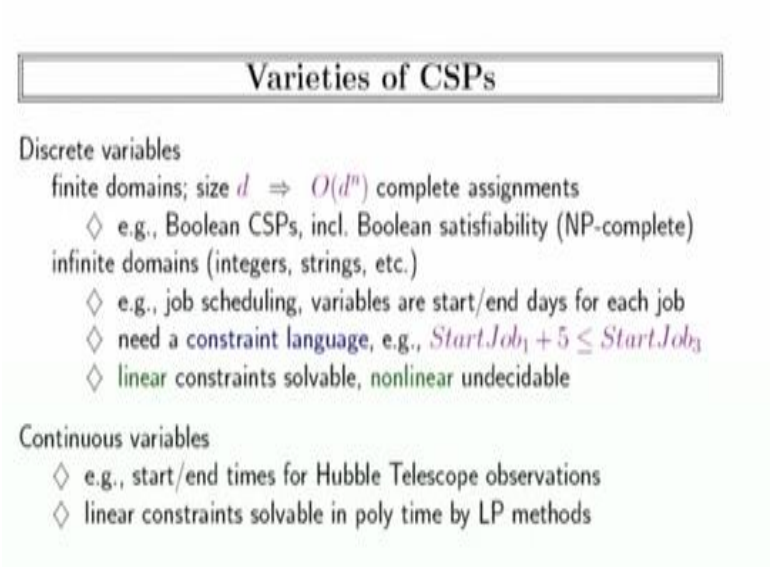
If anything we will talk about converting them to binary CSPs because that is about all the algorithms that you will study will be about constraints which are between 2 variables only and a good general purpose search algorithm might quickly find that there are 2 sub problems in this problem what are the 2 sub problems, whatever 2 sub problems there are 2 independent sub problems the 2 there is a, there are 2 connected components.

Tasmania is in a different component and the other 6 are different component and so it really does not matter how we label just money for the purpose of labeling the rest of the graph. So we can actually convert it into 2 independent problems solve each of those problems separately and

come back, this would be fine. But this we would only be able to do if we looked at this constraint graph or some understanding of the structure of the problem.

Now we can think about the structure of the problem because each state variable is a node, not a state and I hope you see that it difference, earlier each state was a node. Now a state has many variables, let us call them state variables each state variable is a node. So we are operating in the world of state variables at this point not in the world of states only then are we able to recognize that there are 2 independent sub problems.

(Refer Slide Time: 09:09)

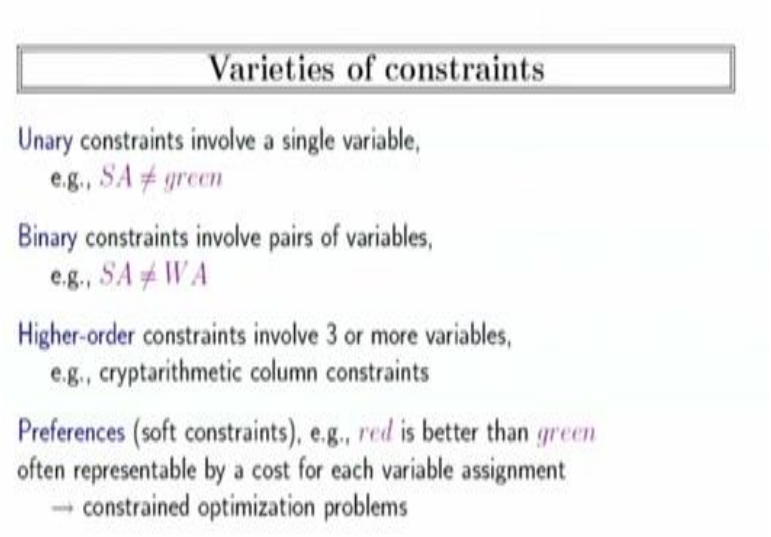


Now, CSP is coming many varieties there are CSPs with finite domains, discrete variables, they are CSPs with infinite domains discrete variables like integers or strings as solutions, there can CSPs with continuous variables, okay. For many, common world, real world problems are constraint satisfaction problems like job scheduling, I have I am a computer, I get many jobs, they all have ending times.

So, I want to satisfy, I want to finish a job up to a certain day before the end time you know, it is a constraint satisfaction problem. I have many, many classes, I have many, many classrooms, I know which classes are happening at the same time. I know which class has how many students now I want to do a scheduling such that we get this classroom and somebody else goes to the 6th floor to teach now, you can think about what is the complete set of assignments.

Suppose I have a discrete CSP and I have no variables and each variable has values which can be taken from a domain of size d, then how many complete assignments do I have? d to the n, the first variable can take d values, the second variable can take d values, the nth variable can take d values. So, we will have d to the power n complete assignment. So, if we could enumerate all the data d to the power assignments, then of course, we will be able to solve our problem really but offcourse that is not possible because it is exponential.

(Refer Slide Time: 10:51)



Similarly, different people study many different kinds of constraints do not label South Australia green, south Australia is a very, it does not like greenery, whatever, be the reason it is too close to water, I do not know. But let us say there is any constraint which says, do not label South Australia green, then that will be presented as a not equal to green, then you have binary, those will be called unary constraints.

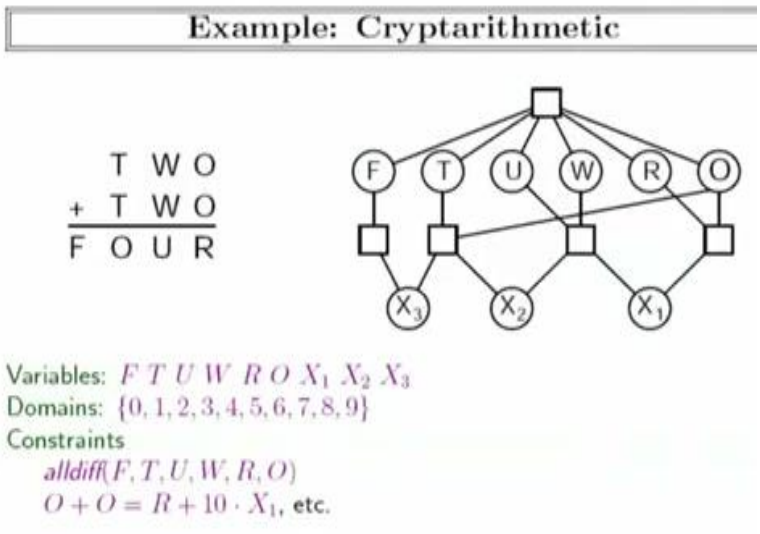
Then you will have binary constraints that you know South Australia should not be equal to Western Australia, you will have higher constraints like 3 or 4 variables come together and define a constraint you may also have what is called soft constraints. Now, soft constraints need not be satisfied, but you get credit for satisfying them. It is like Vishwajith says, you know I like to sit in the front no; can I sit in the back?

I can sit in the back but I rather come early in the class and sit in the front and then another person may have exactly the opposite constraint are sitting in the front is too much pressure and

sit at the back those are soft constraints. You do not always satisfy them, but you know you are happy satisfying them. Now that if you have soft constraints in your problem, then that converts the constraint satisfaction problem into a constraint optimization problem why?

Because now, I have some constraints I need to satisfy them and I have some constraints I may or may not satisfy them, but satisfying each constraint will give me some credit. So, I need to get maximum credit while satisfying all the constraints that said constraint optimization problem because of the maximum part.

(Refer Slide Time: 12:40)

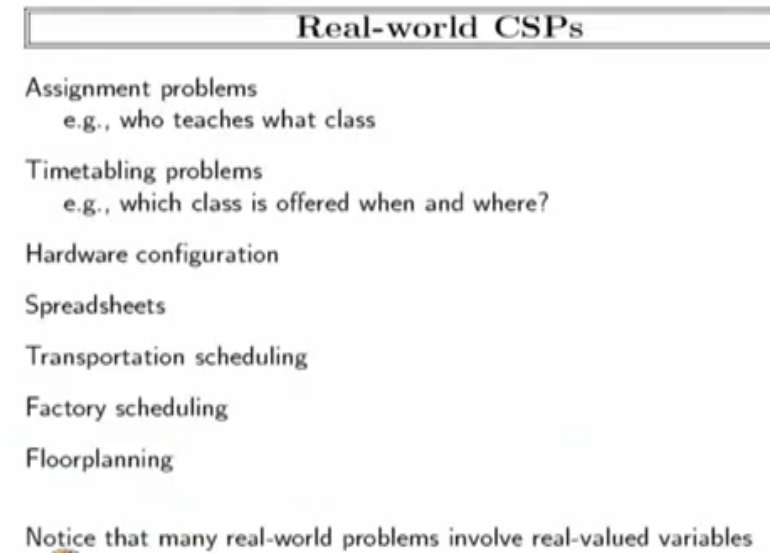


Many interesting problems that you study in real life can be thought of as constraint satisfaction problems. So, this is something that I am sure as children you have done these are called cryptarithmetic puzzles, where if I give you 2 + 2 = 4, where each symbol each letter means a digit and a different digit such that this particular sum is accurately satisfied well, what are my variables the T, W, O, F, O, U, R.

But you will realize that if I want to actually put it as constraints, I will have to define 3 more variables, 1 will be the carry at second step, the carry at the third step and the carry at the fourth step, let us call them X1, X2, X3 what will be the domain will be 0,1,2,3,4,5,6,7,8,9 digits. Now, what will be my constraints so, one of my constraints would be that O + O should be equal to what it should be equal to R + 10 times the carry that is 10 X1 or W + W + X1 should be equal to U + 10 times X2 that will be one set of constraints.

Another set of constraints would be all variables are different digits and such a constraint is written in a syntactic sugar called alldiff. Now, you can always say T 0 = W, W 0 = O, T 0 = O, all pair wise not inequalities, but a syntactic sugar for this is alldiff we just remember this everybody was told that all of these should be different and then this is a constraint graph for it notice that now we have created these square nodes which represent a constraint and multiple edges from variables can come in each constraint but of course we can always convert it into a binary CSP.

(Refer Slide Time: 14:51)



So, let me end the introduction here to say that, who teaches what class which classes offered when where is a class offered how do I put in all the transistors into my configuration so that you know it can fit in the space given I have a factory which employee comes at what time says that you know all machines are running at all times etc., these are all some version of constraint satisfaction problems.