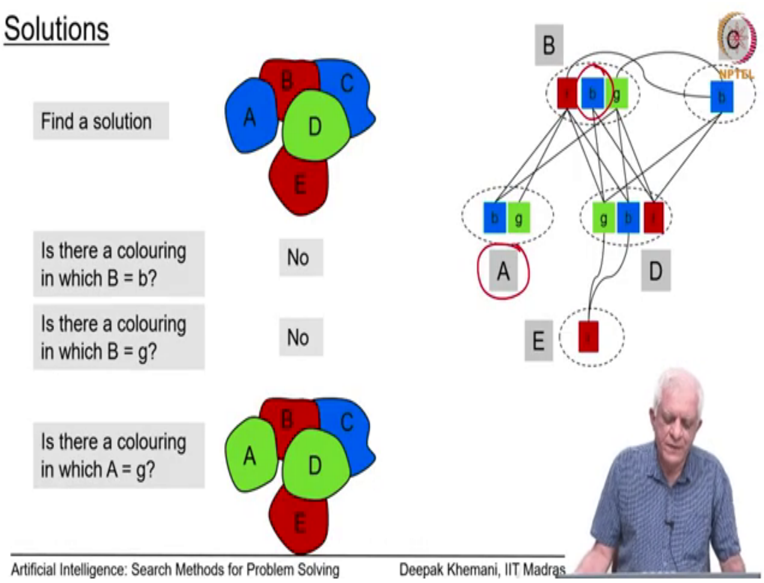
Artificial Intelligence: Search Methods for Problem Solving
Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Chapter – 09
A First Course in Artificial Intelligence
Lecture – 89
Constraint Processing
Binary Constraint Networks

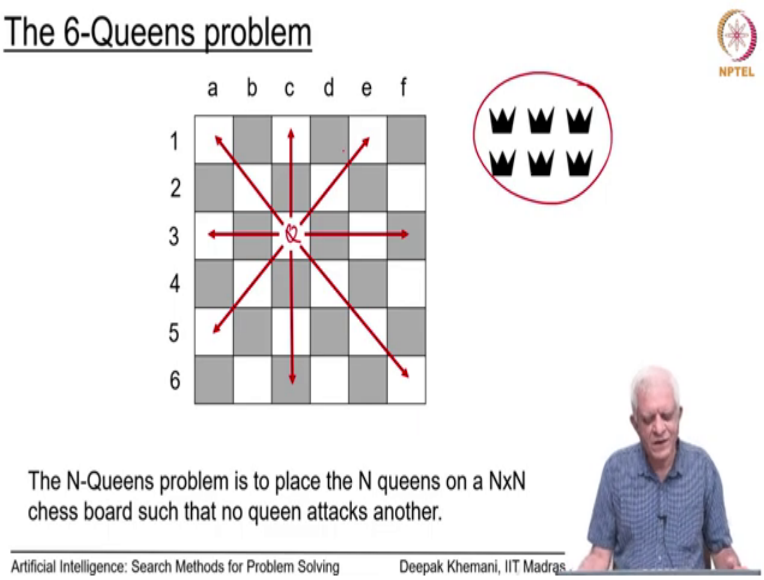
(Refer Slide Time: 00:14)



Let us continue our study of constraint satisfaction problems. At the moment we are focusing on looking at examples as to how we pose problems as constraint satisfaction problems. And once we are done with a few examples, we will look at some algorithms for solving CSP's which are independent of the problems that you are trying to solve.

We ended in the last lecture with this problem of map coloring and we said that here this is how you represent the problem. You have this thing called the matching diagram and, then there is some process sitting behind which tries to find values for all the variables. And in this example, we saw that out of the four questions we asked; in two of them we got a solution and two of them the system came back and said no.

(Refer Slide Time: 01:07)



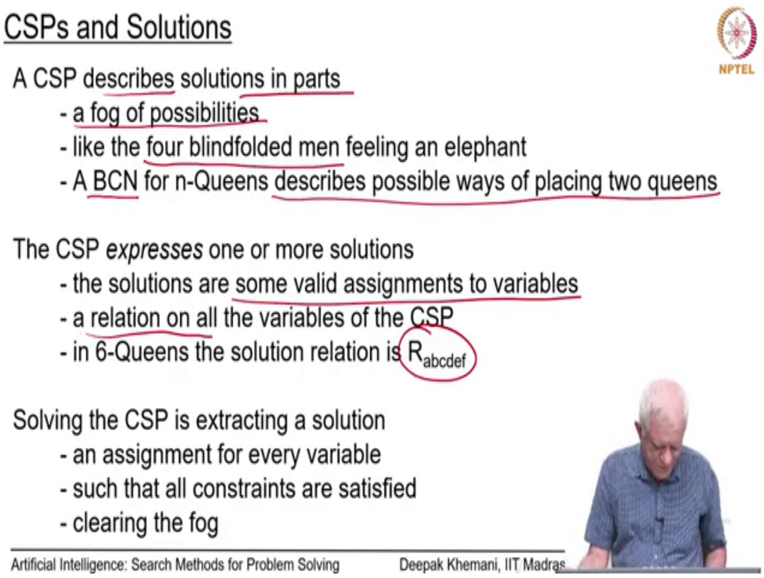
So, let us look at another nice example, because it helps illustrate some of these ideas well and that is a N-Queens problem. In particular we are starting by looking at the 6-Queens problem and as you if are no doubt familiar, the problem is to place these six queens which are at the moment outside the board on the 6 by 6 chess board if you can call it a chess board such that; no queen attacks any other queen.

So, as you know queens attack in a particular direction as we will show in this thing here. So, if you had a queen sitting here, then no other queen can sit on any of these red lines that we have drawn essentially, because queens attack in vertical, horizontal and diagonal directions. So, you cannot place a queen in any other place essentially.

And now, you have to place all these 6 queens. Of course, it is a problem which is familiar to us and often you might do it in a data structures course and you can use that first search to solve it and so on and.

In fact, we will start by using that first search when we try talk about solving it, but we will see that more can be done; which is what I was trying to hint at that you can combine reasoning with blind search essentially. So, in general the problem is for N-Queens on a N by N chess board and it is an interesting problem which serves as an interesting programming exercise essentially.

(Refer Slide Time: 02:40)



Now, we have been talking about constraint satisfaction problems and their solutions. When we express a problem as a constraint satisfaction problem, we are specifying the constraints essentially. And what are constraints? Constraints are constraints or relations on a subset of variables essentially.

So, a CSP describes solutions in parts only. So, in the map coloring problem for example, you can say that that region a cannot have the same color as region b, region c cannot have the same color as region d or whatever the map may be based on that, but you are only specifying a subset of the solution relation. The solution relation is a relation which is on all the variables essentially. But we are not specifying that; that is we want to find that we want to extract a solution essentially.

So, a CSP describes this only a part of a problem. And I am sure you must be familiar with this story of the four blindfolded men and they are put in front of an elephant and each of them feels the elephant touches the elephant and they get of course, different responses to their sense to their sensing. Somebody senses the leg, somebody senses the tail, somebody senses the trunk, somebody senses the tusk and so on.

But they are each having knowledge or information about a part of the solution essentially and of course, they want to figure out what is it that they are all looking at essentially. So, so like. So, in the binary constraints network problem that we are looking at if we specify the constraint as a binary constraint.

So, BCN stands for binary constraint network in an n-Queen problem, then each constraint is describes possible ways of placing two queens; that queen 1 can be here, queen 3 can be there, queen 2 can be here, queen 4 can be there. So, it is like pairs of things.

So, like we did for courses. We said that AI can be taught by DK and ML can be taught by MK essentially. So, like this combination of things. Eventually, you want to find out for all the queens in this example where can they be placed essentially. So, the CSP over not over specify it specifies more constraints then will participate in a particular solution essentially.

So, in some sense as we will see when you look at this example of binary constraint network, the set of possibilities is huge essentially and it is a task of the solver to somehow shift through those possibilities and come up with the solution.

So, let me call that as a fog of possibilities; that there are so many possibilities and somehow, we have to unearth as to what is the real values of the variables essentially. The number of possible assignments to variables is many, the number of constraints which are partial in nature. Because for example, we are mostly looking at binary constraints here is only tells you that this cannot go with this and that kind of thing exactly it does not talk about the whole set of variables.

So, it is a fog of possibilities. Then we have said that a CSP expresses one or more solutions that somewhere hidden in the fog is are solutions. It may be one solution or it may be more than one solution.

But if you were to solve the CSP, then you could get one of those solutions. So, in some sense, the CSP the original CSP also expresses those same solutions, because those are the only solutions to those CSP essentially.

And if you were to do a more analytical study of constraint networks and so on, then you would start talking about that this network is equivalent to this network, because both express the same solution, but we will not have time to go into that.

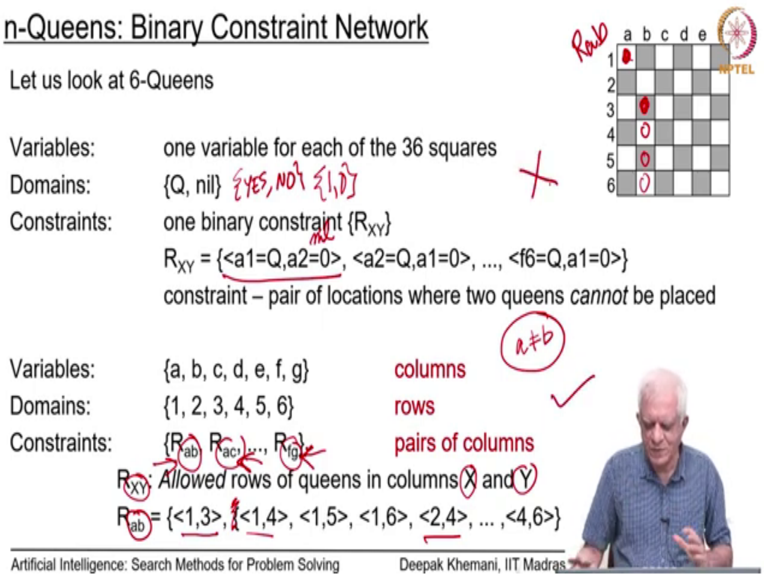
We want to simply focus on how search plays a role in solving constraints and how search can be combined with reasoning while solving CSP's. So, a CSP expresses those solutions and as we have said, there are some valid assignments to variables which means all the constraints must be satisfied.

The solution is a relation on all the variables of the CSP we have said that it is called row and in the 6-Queens problem that we just described, it is the relation on the five six variables a b c d and e, which we have not specified how we will specify the variables, but one of the ways that we will do it is that we will have 6 variables which we will call as a b c d e f essentially and what is the task.

Solving a CSP means extracting a solution from all that set of possibilities which means an assignment to every variable such that all the constraints are satisfied essentially. So, you can think of it as clearing the fog and getting to see the solution essentially. So, let us look at the binary constraint network to get an idea of the kind of complexity that can sit behind even simple CSP's essentially.

So, if you had instead of five regions that I showed if you had a map of 20 color 20 regions and you were doing line search, then the number of possibilities that the algorithm would explore is quite huge essentially. So, we want to emphasize that fact and say that this is a problem that solve CSP's solvers have to address as to how to cut through all the haze and the fog and get to the solution.

(Refer Slide Time: 09:04)



So, if we express the n-Queens problem as a binary constraint network which basically means that we tell you what is the relation between placing two queens. So, let us say the queens are labeled by let us say a, b, c, d, e as I said and how can you place a and b how can you place c and d and that kind of thing.

So, it is a binary relation. Now, in the 6-Queen problem that we are looking at here, I have used a, b, c, d, e like is common practice as column names and 1, 2, 3, 4, 5, 6 as row names.

So, any combination of a row name and a column name identifies a particular square and then the task is to find out whether you are going to place a queen on that square or not. Now, this simple intuition can be directly translated into CSP where we have one variable for each of the 36 squares and the domain of every variable is that either it has a queen or it has nothing. So, you could have said something else; you could have said yes no for example. It does not matter it is just a way of expressing it some binary.

Then, we are talking about binary relations. Here, you could have expressed relation between three queens, four queens, five queens, six queens if you express it on 6-Queens then you have expressing the solution directly. So, there is nothing to be done. So, we are talking about binary constraints here binary constraint networks.

And how do we specify the constraints now. So, remember there are 36 variables here and you want to say something about where you can place two queens. So, the explicit way of saying that is to do something like this. It says that you can place a queen on a 1.

So, what is a 1? a 1 is this one here. If you do that then on then, on a 2 you cannot place a queen. So, I have written 0 here which is a kind of a mix up. So, let me say this can be 1 or 0. So, I could have written nil here according to the formulation I did. I hope it does not cause the confusion.

So, essentially this says that if you place a queen on a 0, then you cannot place a queen on a 2. Now, I would have to do that for every pair of squares essentially that you can place two queens here or you cannot place two queens here. It is kind of the more intuitive thing, because you are thinking of the variables as the at the board as the squares on the board and the values as the presence or absence of a queen, but as you can see it leads to a cumbersome formulation formulating the problem itself is a lot of work essentially.

So, as we all know as we have all studied n-Queens I am sure. The popular representation is to choose the variables as the column names we can also call them queen names ok. So, not the fact not square names essentially and the values on.

So, essentially when you when you say that the variables are a b c d e, you are essentially saying that I will place one queen in every column one queen in column a one queen in column b one queen in column c and so on.

And the value for the variables which is column names is the row names; that in this column I will place the queen at this row and so on. And now, we are talking about binary constraint networks. Now; obviously, we can do this in an intentional form. So, for example, you could say that a is not equal to b. In fact, you can say all different here.

And say that you cannot place two queens two different queens; a and b represent two different queens in the same row. Remember the value of a is a row and the value of b is also a row and you are saying that the rows cannot be same so; that means, that this intentional expression simply says that you cannot place queens in any row in the chess board essentially.

Likewise, you can try and express other constraints. So, for example, diagonals are not so hard to express, but we will not get into that. We are looking at explicit representations. So, let us focus on that. So, our variables are the columns, our values are the rows. So, column stand for queens and rows stand for which row the queen will be placed and the relations are binary relations between every pair of queens essentially ok.

So, a, b, a, c; a d b c and so on and so forth every pairs essentially. And what the relation expresses explicitly is the allowed rows of queens in columns X and Y. So, a relation between X and Y is that in column X in column Y what rows are allowed. So, if you look at the relation between two queens; the first two queens a and b. This one says that you can place a queen on row 1 and the second queen on row 3.

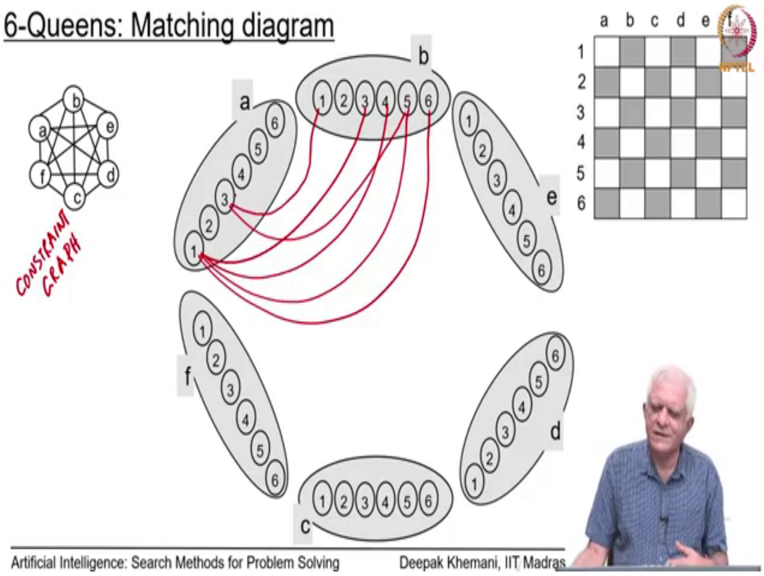
So, let me rub that part off. So, you can place the second queen here essentially, but you could have also placed the second queen on 1 4 ok. So, 1 this thing is crept up here. You could have placed it here you could have place it here you could have place it here. So, this is R a b essentially.

These are the allowed combinations and likewise you would do it for queen 2 sorry you can place it in row 2 and 4 and so on and so forth. And then likewise, you would do it for other variables. So, we talked about a b here, you can talk about ac next, and you know eventually you can talk about f g.

So, you can specify those constraints since they are not. So, hard to specify you can write a program to do that. All you have to do is to check that if a queen is in; let us say this location a 1 and if it is a queen is on b 3 that it not attacking each other. So, which means that you can you cannot draw a line which is the attack line if you want to call it between those two queens.

So, that is the formulation. So, this is the preferred formulation and this is not the preferred formulation even though it is more intuitive it tends to think about the chess board as a set of variables.

(Refer Slide Time: 16:20)



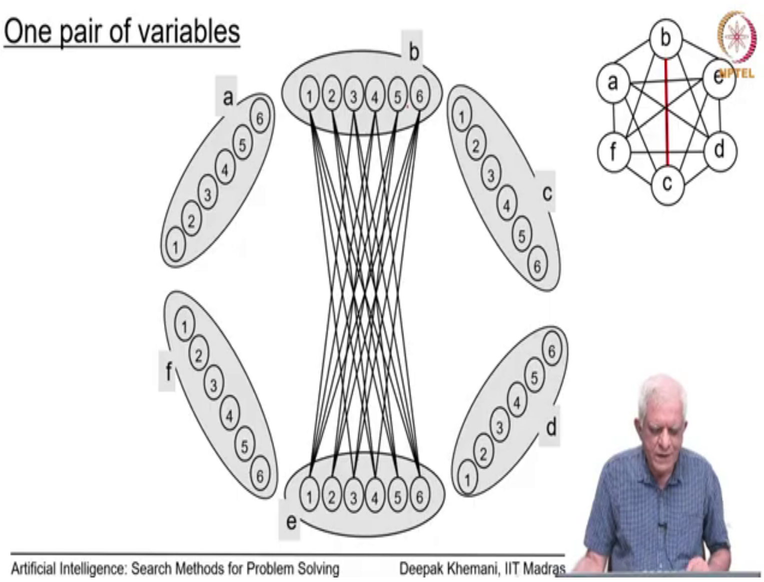
So, what about the matching diagram. So, now, we have the 6-Queen problem here and here you have the constraint graph. As you can see it is a fully connected graph which means you want to specify this relation between any pair of queens and what will the matching diagram look like.

Remember that the matching diagram gives us the edges between allowed values essentially. So, for example, if you are looking at a and b, we just said that 1 can be placed with 3 or 1 can be placed with 4 or 1 can be placed with 5 or 1 can be placed with 6. So, that is the first four examples we see, but there are other possibilities; 1 can be placed 2 you can place a let us say just to take another example.

You can place queen a on row 3 and queen b on row 1 or you can place queen a on row 3 and you can place this in 1. So, all kinds of possibilities are there and the matching diagram will express this. Now, you see that there are the six variables and we want to express the relation between all combinations of variables.

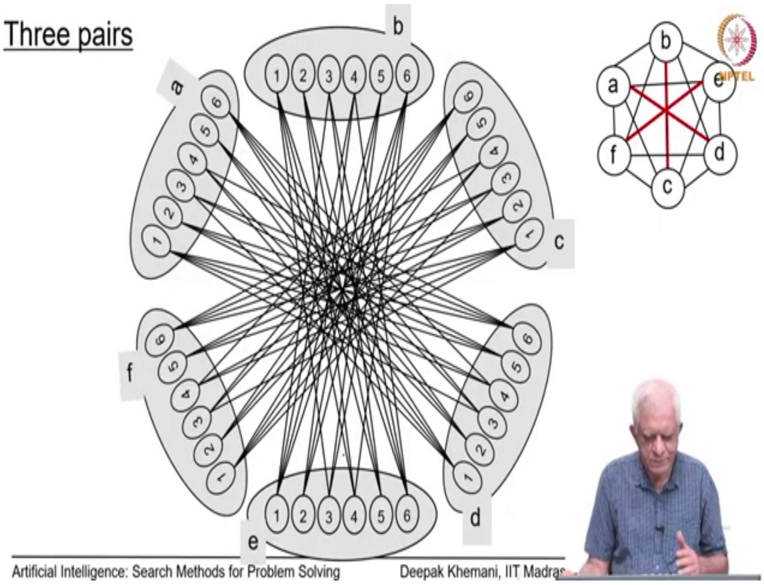
So, the reason why we are doing this so explicitly is to get a feel of the fact that this may be a small problem of 6-Queens, but if you think of it from fundamental principles, then the space that you want to look at is quite huge essentially. And that is what will lead us a little bit later into saying that how can we cut down on the space of possibilities even before doing the trial and error essentially.

(Refer Slide Time: 18:18)



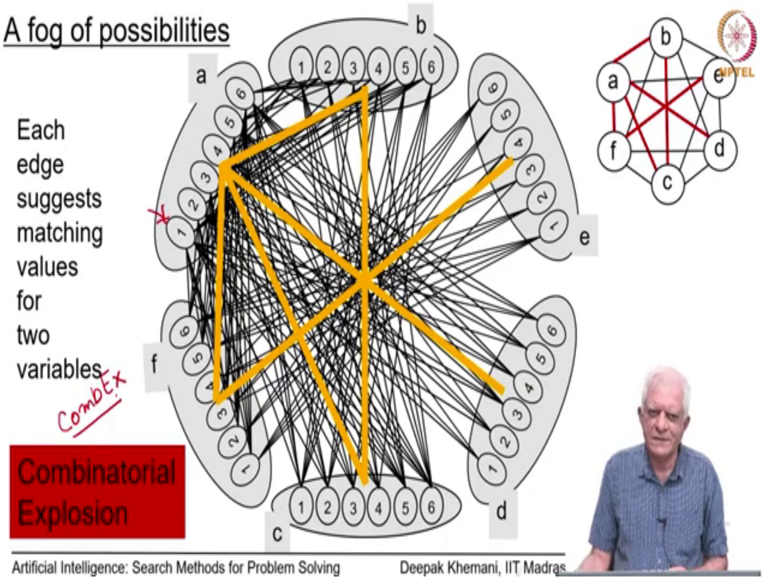
So, if you look at a pair of variables in this it is the case of b and c that you are looking at, then you can see that each of them has three or four partners on the other column. So, queen b if it is on 1 row 1, then it can be placed on 3 4 5 6 e can be placed on 3 4 5 6 and likewise, for each value of b, we have a set of values. So, this matching diagram if you remember, an edge says that this combination is allowed. You can place queen b on row 1 and you can place queen e on row 3 that is perfectly fine essentially.

(Refer Slide Time: 19:18)



Now, if you look at three pairs, you can see that the number of edges is growing.

(Refer Slide Time: 19:29)



If you look at more and I had to stop after this it was a bit cumbersome to draw all these edges. You can see from the constrained graph that the edges in red are the ones that we have depicted in the matching diagram, which I have kind of copied the constraint graph on the matching diagram as well. And we have done roughly half and we have to still do another half essentially.

So, my intention of doing this in so much detail was to say that. Now, if you have to write a search algorithm ok. How will that search algorithm work? We will see that algorithm shortly. It will say take the first variable. Let us say there is some particular order of looking at these variables. Let us say it is a b c d e that you are placing the queen in the first column, then in the a column, then in the b column, then in the c column and so on.

So, it will say let me see let me try this one. Let me place the first queen in the first row of column a, then this matching diagram will tell me what where can I place it in b. So, I will choose one value. Place the queen on b and I will ask where can I place the queen on c and I will choose a value which is consistent with both the queens a and b and so on. So, I will keep doing that I will keep.

So, I will keep exploring edges in this graph essentially. And the number of edges is what is growing up is something which should be a cause of worry for us. So, each edge suggests matching values for two variables and here, we are running into our old four the combinatorial explosion. Remember that we had called it combEx; the monster if you want to call it that we are fighting.

So, when you get down to algorithms and this whole field of constraints satisfaction problem is trying to figure out what are the best ways of solving this CSP's and solving this CSP's mean giving values to every variables. So, the first set is that you have to pose the CSP carefully.

Then, you have to find values for that carefully essentially. We look at one or two more examples after a short break before we move on to a glimpse of the algorithms. We do not have more than a week here I think ok. So, let us come back after the short break.