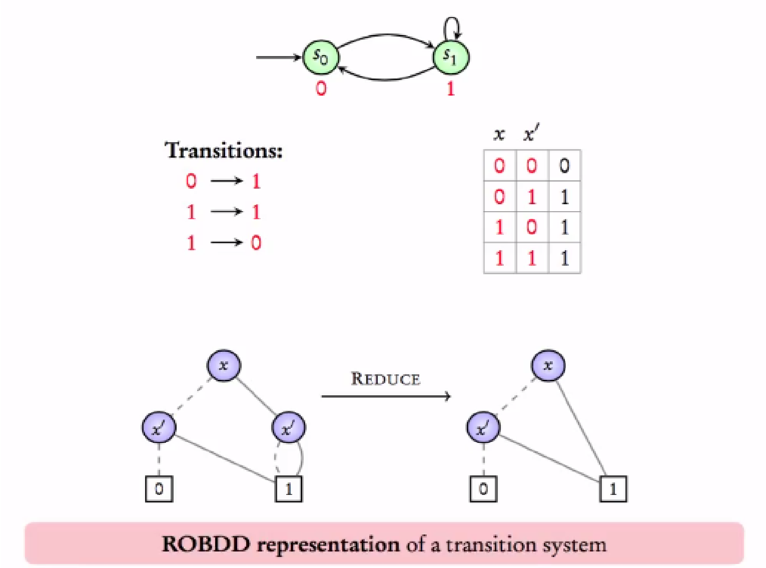
Model Checking
Prof. B. Srivathsan
Department of Computer Science and Engineering
Indian Institute of Technology – Madras

Lecture – 52
Representing Transition Systems as OBDDs

We are in unit 11 of this course; this unit is on binary decision diagrams. In the last two modules, we have given an introduction to binary decision diagrams. They are a data structure for representing Boolean function. Now why did we talk about Boolean functions and BDDs, we will explain this in this module, we want to represent transition systems using ordered BDDs.

(Refer Slide Time: 00:41)

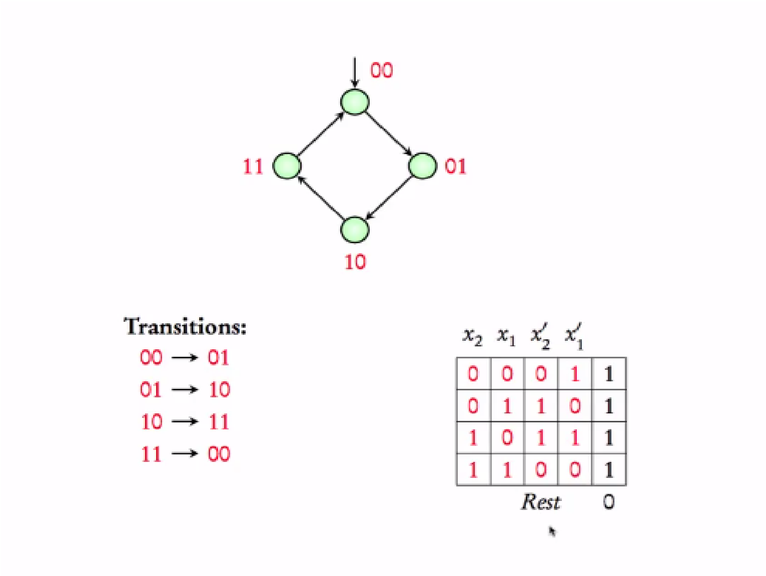


Let us start with an example, so this is a simple transition system, it has two states, S knot has a transition to S1, S1 has a transition to S knot and S1 again has a self loop on S1. There are two states, let us call this state, state 0 and this state as state 1. So you can use 1 bit to represent two states. Now what about the transitions, 0 to 1 is the transition, 1 to 1 yet another transition and 1 to 0 is one more transition.

Let us now look at the transitions as a Boolean function. Assume that there are variables, x and x prime, x is the source variable and x prime is the target variable. So when the source is 0 and the target is 0, there is no transition. So 0 0 goes to 0, on the other hand 0 1 goes to 1, 1 0 goes to 1 and 1 1 goes to 1. This truth table is an encoding of this transition system. So we say that since there are two states, the states are 0 on 1 and the transitions are given by this truth table.

This contains the entire information about the transition system. Now once we see a transition system as a Boolean function, we can represent it using an OBDD. So we start with x 0 0 goes to 0, 0 1 goes to 1, 1 0 goes to 1 and 1 1 goes to 1. You can see that this is not reduced; you reduce it because this is an unnecessary point. So once you reduce it, you get this ROBDD. So this ROBDD is a representation of a transition system. Let us see one more example.

(Refer Slide Time: 03:05)



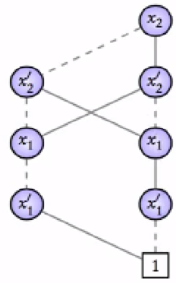
So here is a transition, whose transitions are in the form of a ring and it has four states. Since it has four states, you need two bits, if you want to name the states uniquely. So this is 0 0, 0 1, 1 0 and 1 1, call these using these identifiers. Now, how does the transition relation looked like. 0 0 goes to 0 1, 0 1 goes to 1 0, 1 0 goes to 1 1 and 1 1 goes back to 0 0. So this is the transition relation. Now, how does the truth table look like.

Note that the source already has two bits, so the target will also be two bits, call this x2, call this x1, this will be x2 prime and this will be x1 prime. So 0 0, 0 1 is 1, 0 1, 10 is 1, 1 0, 1 1 is 1 and 1 1, 0 0 is 1 and the rest of the entries map to 0 okay. So let me repeat, this column, I mean, this bit is represented by x2, this part is represented by x1, this is x2 prime and this is x1 prime okay. Denoted using two bits, then the transition relation will look at four variables, x2 x1 and x2 prime, x1 prime.

(Refer Slide Time: 04:59)

x_2	x_1	x'_2	x'_1	
0	0	0	1	1
0	1	1	0	1
1	0	1	1	1
1	1	0	0	1

x_2	x'_2	x_1	x'_1	
0	0	0	1	1
0	1	1	0	1
1	1	0	1	1
1	0	1	0	1

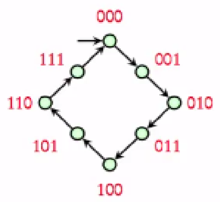


ROBDD with ordering $[x_2, x'_2, x_1, x'_1]$ for ring with 4 **nodes**

Now we can construct the OBDD given this truth table, however, let us consider a different ordering instead of looking at x2, x1, x2 prime, x1 prime, let us look at it as x2, x2 prime, x1, x1 prime. So you see 0 0 goes to 0 1, 0 1 goes to 1 0, 1 0 goes to 1 1 and 1 1 goes to 0 0. So we just changed the ordering of this truth table and this is the OBDD representation of this truth table.

I have just not given 0 for clarity, however, all nodes, I mean each node should have two children and if the child is not mater, it just goes to 0, assume that it goes to 0 okay. Let us have a look it, this says that 0 0 0 1 goes to 1. See 0 0 0 1 goes to 1, 0 1 1 0,.0 1 1 0 goes to 1, 1 1 0 1 goes to 1 and then 1 0 1 0 goes to 1. So this is the ROBDD representing this ring with four nodes, four states and the order of this x2, x2 prime, x1, x1 prime for the ring transition system with four states.

(Refer Slide Time: 06:44)



x_3	x_3'	x_2	x_2'	x_1	x_1'
0	0	0	0	0	1
0	0	0	1	1	0
0	0	1	1	0	1
0	1	1	0	1	0
1	1	0	0	0	1
1	1	0	1	1	0
1	1	1	1	0	1
1	0	1	0	1	0

- Either fully 10 – 10 – 10
- Or 01 occurs:
  - after 01 only a sequence of 10
  - before 01 only sequences of 00 or 11

Let us now look at a ring with a ring with eight states. Since there are eight states, you need three bits to denote the state 0 0 0, 0 0 1, 0 1 0, 0 1 1, 1 0 0, 1 0 1, 1 1 0 and 1 1 1 okay. So there are eight states, so you needed three bits so that each state can be given a unique number. Now let us look at the transition relation, it will be similar. Since, we are dealing with the shifted ordering, so this will be x3, x2 and x1 and the transition relation will be from x3, x2, x1 to x3 prime, x2 prime and x1 prime.

However, we want to look at it in a different order. So first, when x3, x2, x1 are 0 0 0 then this will be 0 0 1 like this. When x3, x2, x1 is 0 0 1 then this will be 0 1 0. When this is 0 1 0 the target will be 0 1 1 and so on. 0 1 1 will go to 1 0 0, 1 0 0 goes to 1 0 1, 1 0 1 goes to 1 1 0, 1 1 0 goes to 1 1 1 and finally 1 1 1 goes to 0 0 0. Now I want to observe certain patterns.

So these are the ones which will be going 1, if you look at the truth table of this huge thing, the rest of the entries will be going to 0. Let us try to observe certain patterns. Either in this sequence there is no 0 1, everything is 1 0 1 0 1 0 otherwise it will contain a 0 1 somewhere, so this bit shifts to 1. So this is the successor okay. What mode once the sequence sees a 0 1 after that it can see only a sequence of 1 0.

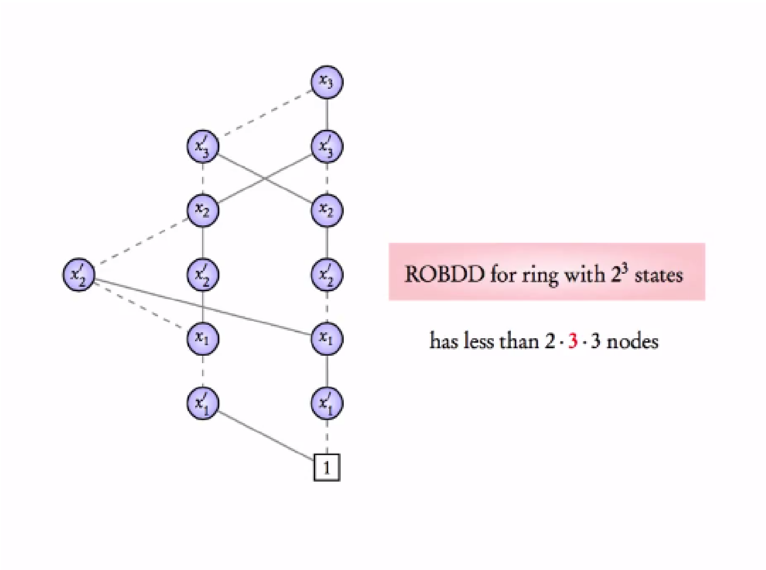
See, this is because x3 prime, x2 prime, x1 prime is the successor of x3, x2, x1. The moment you have done 0 1, it means that you have added 1, that means in the vector x3, x2, x1 all the numbers after this would be 1, so that is what, see this 0 and this is 1, that means from 0 0 1, you wanted to add 1 because this was 1, this became 0 and since this was 0, this became 1, that is what this means.

So whenever you see 0 1, the subsequent digits will be 1 0 1 0, subsequent bits will be 10 10. Moreover, till you see 0 1, x3 and x3 prime, I mean the prime variable will be equal to the unprime variable. See, this is 0 0 1 1, this 1 1 0 0, 1 1, 1 1 1 1. You will not see a one 0 here okay. So let me repeat the points.

When what are the sequences for which the truth table goes to 1, either fully 1 0 1 0 1 0 like this because this was 1 1 1 when you add at 1, everything became 0. So either 1 0 1 0 1 0 or somewhere 0 1 occurs, after 0 1 only a sequence of 1 0 like after you see 0 1, you see only 1 0 1 0. Before you see 0 1, you have to see either 0 0 or 1 1. This just says that there is no change in the prime variable. Note that this is going to be, this rule is going to be the same, no matter how many variables we take.

Instead of eight, if we had taken, see eight is 2 power 3, instead if we had taken 2 power 4, we would have had x4, x4 prime, x3, x3 prime, x2, x2 prime, x1, x1 prime and the same rules would hold. Either fully 1 0 1 0 1 0 or you hit 0 1. Once you hit 0 1, you will see only 1 0 1 0 1 0 and before you hit 0 1, you will see either 0 0 or 1 1 okay. Let us now use these rules to construct the OBDD corresponding to this Boolean function.

(Refer Slide Time: 12:10)



Firstly, 1 0 1 0 1 0 goes to 1 okay. This is there. Now, we need to add the other paths, how do we do it. From x3 if you see a 0, suppose you go to x3 prime, see 0 0 and 1 1 come here. So this is the node which has seen a sequence of either 0 0 or 1 1 so far. It is yet to see a 0 1. You can still continue from here you can see a 0 0 or a 1 1 but then at x1, you have to see a 0 1.

See, x2 you can see 0 0 or 1 1. Again, from here, so this is the first sequence.

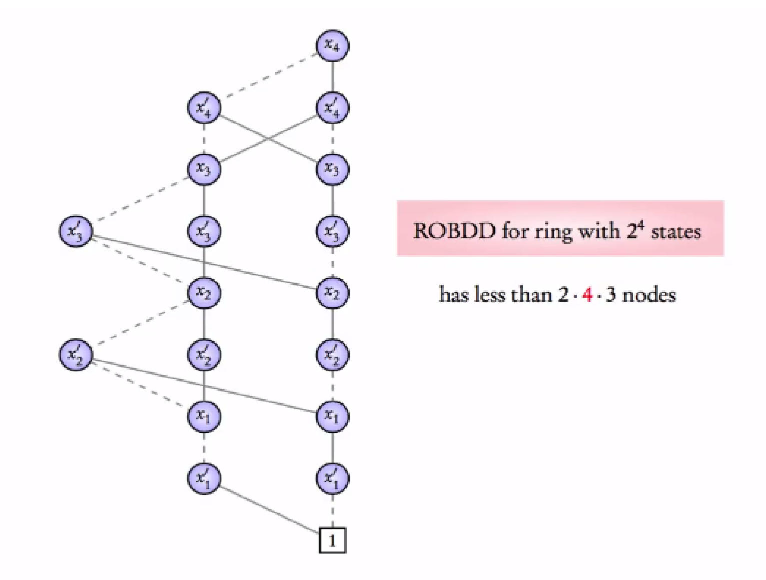
This one we are reading, either 0 0 or 1 1 you can see and after that still you can see either a 0 0 or 1 1 but then you should definitely see a 0 1 from here, that is why this is. Moreover, you could have also seen a 0 1 here, the first thing could have been a 0 1 but then once you see a 0 1 you should see only 1 0 after that right. That means, if you see a 0 1 here, you go to this path, you continue to this path, 0 1, 1 0, 1 0.

Similarly, if you see a 0 1 in the second bit, I mean in the second pattern, then from 0, you go to 1, and then you have to see only 1 0. So this will be the ROBDD which will correspond to the Boolean function that we saw in the previous slide. So just get the pattern clear 1 0, 1 0, 1 0 should go to 1 otherwise you should see a string of equivalent equal pairs, 0 0, 1 1, 0 0, 1 1.

Finally, you have to see a 0 1 or you could see a 0 1 somewhere before but once you see a 0 1, you should continue to see only 1 0. So this is the ROBDD for the ring with 2 power three states. How many nodes does it have, if it has three states, I mean 2 power 3 states, it means that we have used three bits hence, for each bits we have two variables.

So there are at least 1 2 3 4 5 6 rows and each of them there is utmost 3. So it has less than 2 into 3 into 3 so far. So where 2 is because of the primed and unprimed variables and this 3 is for this width okay. So it has less than 2 into 3 into 3 nodes. So far we have not seen any spectacular game.

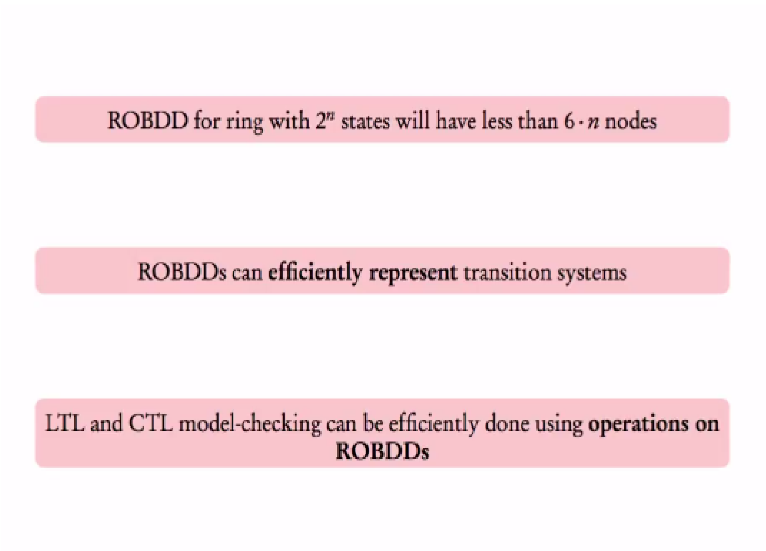
(Refer Slide Time: 15:17)



What if we had a ring with 2 power 4 nodes, we will have the same pattern for the transition relation. So 1 0, 1 0, 1 0, 1 0 will go to 1 and this node is seeing 0 0 or 1 1, the moment you see a 0 1, you go to this stretch and from here if you see a 0 1, you go here otherwise you see 0 0 or 1 1. Similarly, either you see 0 0 or 1 1, the moment you see a 0 1, you go here and finally if you come to this node, this node so far has not seen 0 1.

So 0 0, 1 1, 0 0 then here it has to see a 0 1. So this is the ROBDD for the ring with 2 power 4 states and how many nodes does it have, it has less than 2 times four because there are 2 times 4 many variables and each variable comes utmost 3 times in the BDD. Now what does this lead us to.

(Refer Slide Time: 16:27)



If you have a transition system in the form of a ring with 2 power n states, we will have less than 2 into n into 3 nodes in the ROBDD representation for that transition system. Note that there are exponentially many states but there are only linearly many nodes in the BDD. So this is one example where an ROBDD representation of the transition system gives us an exponential game. Gives us a lot of game.

It is an efficient representation of the transition system. So I have just given you one example in the practice for huge transition systems there are automatic ways of synthesising ROBDDs and in practice the ROBDDs efficiently represent these transition system and most of the cases. Moreover, LTL and CTL model-checking can be efficiently done using ROBDDs. Once you have the transition system as an ROBDD you can apply CTL and LTL model-checking methods.

I have not talked about it in this unit, we would not have a look at in this course, but you can refer it in the book that I have given Logic and Computer Science by Huth and Ryan. So CTL model-checking can be efficiently done using ROBDDs. So ROBDDs are one way of tackling the state space explosion problem.