COL:750

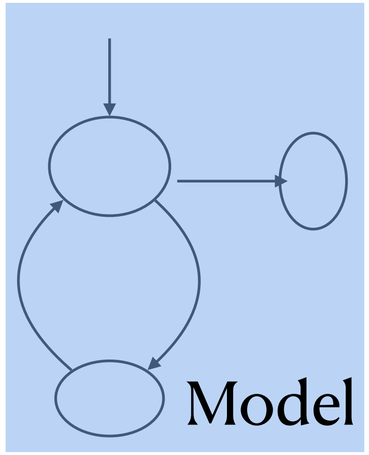
Foundations of Automatic Verification

Instructor: Priyanka Golia

Course Webpage 

https://priyanka-golia.github.io/teaching/COL-750/index.html

Does Code Satisfy Requirements ?

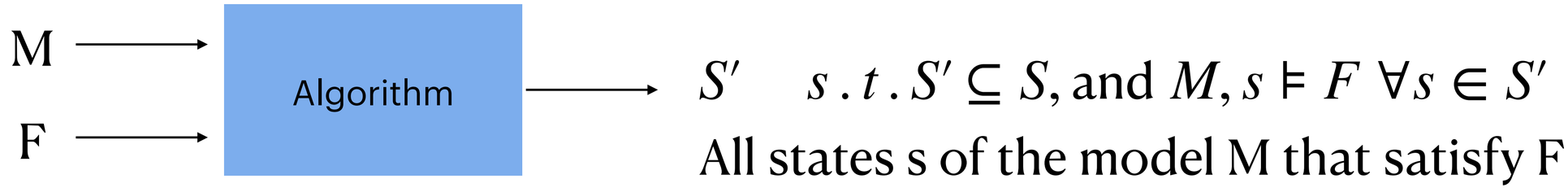


Does Model Satisfy Logical formulation: LTL/CTL Formula ?

Model Checking

Model Checking Algorithm

$$M, s \models F?$$



Note that not necessarily $I \subseteq S'$

Labelling Algorithm —

1. Does not scale well to large systems due to state explosion.
2. Memory-intensive as it maintains explicit labels for each state.

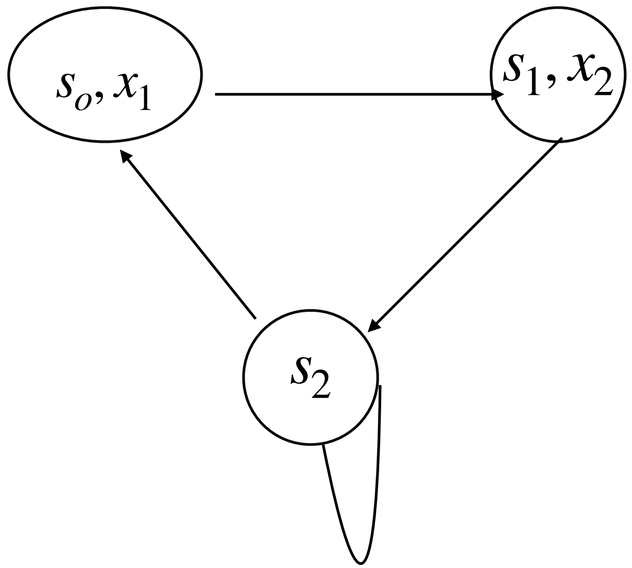
We need better data structure.

Implementing CTL Model Checking using BDDs

CTL model checking computes a set of states $[F_i]$ for every sub-formula F_i of the original formula F.

Sets of states will be represented using ROBDDs

That describes characteristic function of the set

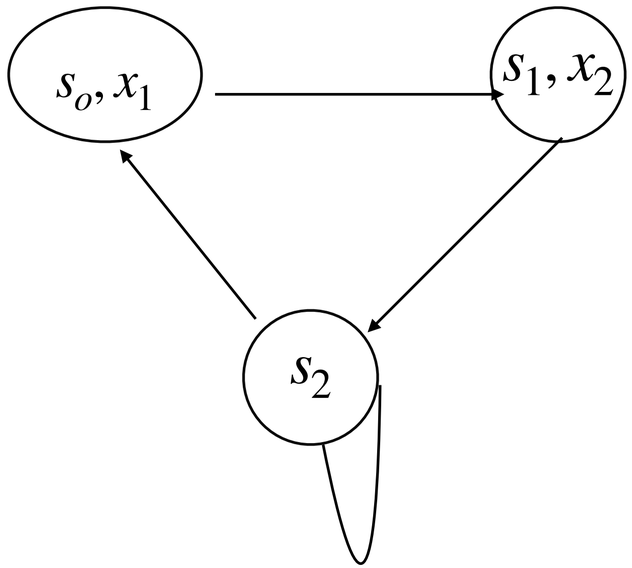


Implementing CTL Model Checking using BDDs

CTL model checking computes a set of states $[F_i]$ for every sub-formula F_i of the original formula F.

Sets of states will be represented using ROBDDs

That describes characteristic function of the set

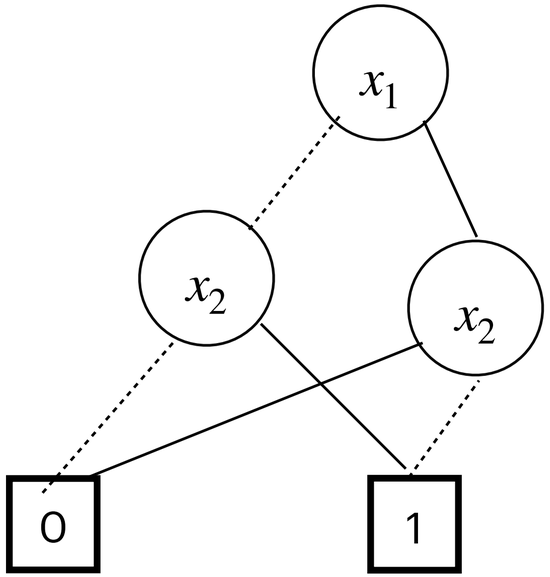


Implementing CTL Model Checking using BDDs

CTL model checking computes a set of states $[F_i]$ for every sub-formula F_i of the original formula F.

Sets of states will be represented using ROBDDs

That describes characteristic function of the set



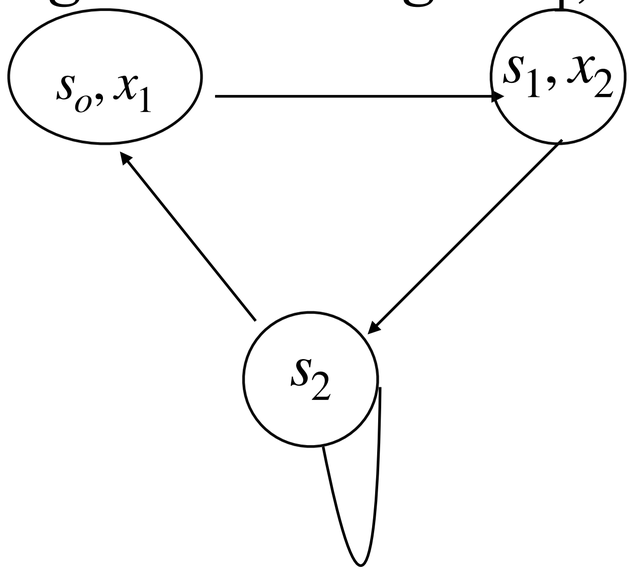
ROBDD for the set $\{s_o, s_1\}$

Set of states	Representation by	Representation by Boolean
∅		0
{so}	(1,0)	$x_1 \wedge \neg x_2$
{s1}	(0,1)	$\neg x_1 \wedge x_2$
{s2}	(0,0)	$\neg x_1 \wedge \neg x_2$
{s0,s1}	(1,0),(0,1)	$(x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$
{s0,s2}	(1,0),(0,0)	$(x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge \neg x_2)$
{s1,s2}	(0,1),(00)	$(\neg x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_2)$
{s0,s1,s2}	(1,0),(0,1),(0,0)	$(x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_2)$

Implementing CTL Model Checking using BDDs

Representing the transition relations.

- Transition relations ($\rightarrow$) $\subseteq$ S $\times$ S are represented by ROBDDs on 2n variables.
- If the variables $x_1, \ldots, x_n$ describe the current state, and the variables $x'_1, x'_2, \ldots x'_n$ describe the next state.
- The good ordering is $x_1, x'_1, x_2, x'_2, \ldots, x_n, x'_n$ (interleaving).



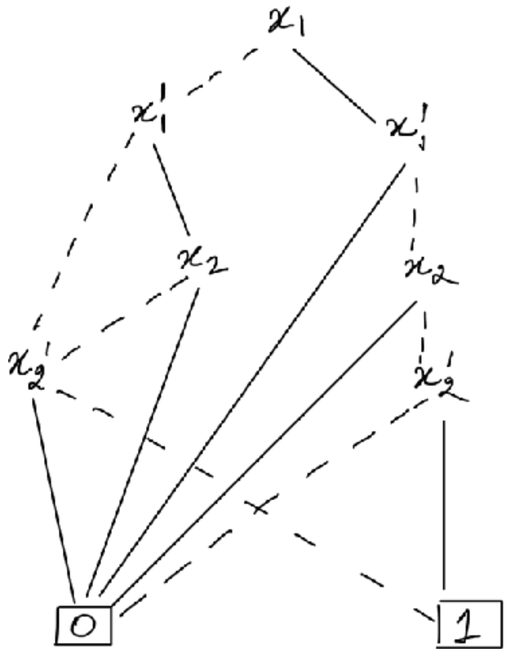
X1	X2	X'1	X'2	->
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
1	0	0	1	1
0	0	0	1	0
..	..	..	..	..

Implementing CTL Model Checking using BDDs

Representing the transition relations.

- Transition relations ($\rightarrow$) $\subseteq$ S $\times$ S are represented by ROBDDs on 2n variables.
- If the variables $x_1, \ldots, x_n$ describe the current state, and the variables $x'_1, x'_2, \ldots x'_n$ describe the next state. The good ordering is $x_1, x'_1, x_2, x'_2, \ldots, x_n, x'_n$ (interleaving).

ROBDD of $F^{\rightarrow}$



X1	X2	X'1	X'2	->
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
1	0	0	1	1
0	0	0	1	0
..	..	..	..	..

Implementing CTL Model Checking using BDDs

Representing the transition relations.

- Transition relations $(\rightarrow) \subseteq S \times S$ are represented by ROBDDs on 2n variables.
- If the variables $x_1, \ldots, x_n$ describe the current state, and the variables $x'_1, x'_2, \ldots x'_n$ describe the next state. The good ordering is $x_1, x'_1, x_2, x'_2, \ldots, x_n, x'_n$ (interleaving).

But exploring Truth table will be expensive.

Can we learn $F^{\rightarrow}$ without Truth table?

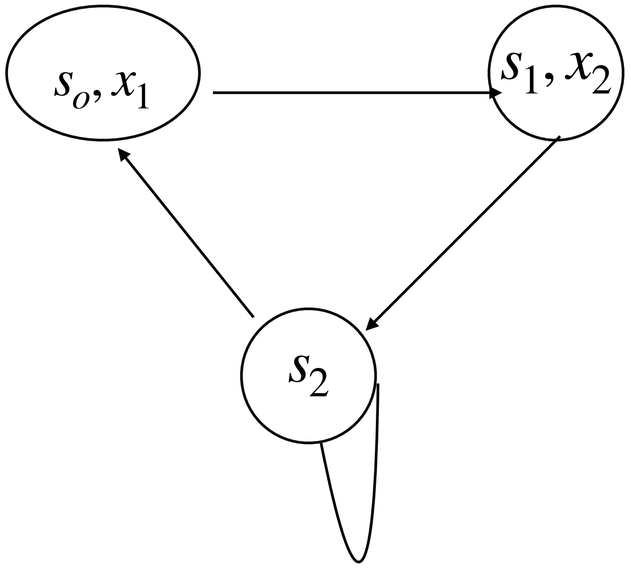
X1	X2	X'1	X'2	->
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
1	0	0	1	1
0	0	0	1	0
..	..	..	..	..

Implementing CTL Model Checking using BDDs

Representing the transition relations.

- Transition relations $(\rightarrow) \subseteq S \times S$ are represented by ROBDDs on 2n variables.
- If the variables $x_1, \ldots, x_n$ describe the current state, and the variables $x'_1, x'_2, \ldots x'_n$

describe the next state. The good ordering is $x_1, x'_1, x_2, x'_2, \ldots, x_n, x'_n$ (interleaving).



Can we learn $F^{\rightarrow}$ without Truth table?

$$F^{\rightarrow} := (x_1 \wedge \neg x_2 \wedge \neg x'_1 \wedge x'_2) \vee (\neg x_1 \wedge x_2 \wedge \neg x'_1 \wedge \neg x'_2) \vee (\neg x_1 \wedge \neg x_2 \wedge \neg x'_1 \wedge \neg x'_2) \vee (\neg x_1 \wedge \neg x_2 \wedge x'_1 \wedge \neg x'_2)$$

Convert $F^{\rightarrow}$ to ROBDD.

Implementing CTL Model Checking using BDDs

Symbolic Model Checking — it represents and manipulates sets of states and transitions using symbolic expressions or formulas (like Boolean functions or Binary Decision Diagrams) rather than explicitly enumerating each state.

Specification — $F = \exists \mathbf{N} p$

Pre([p]) same as Pre(Y)

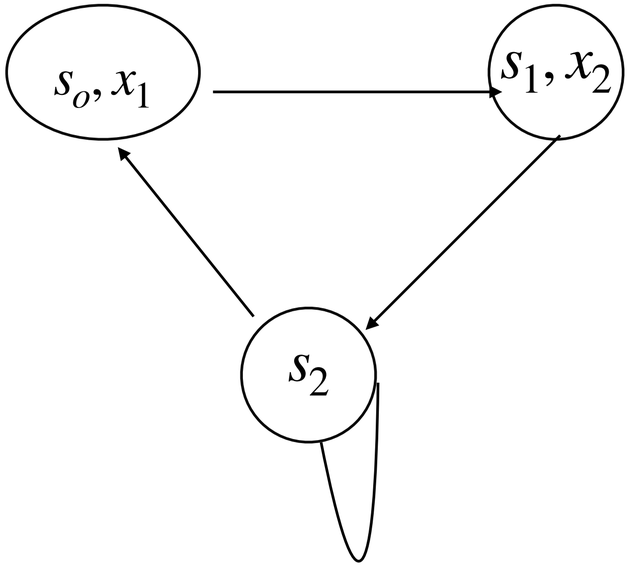
$B_{Pre(Y)} =$ exists $(X', \text{apply}(\wedge, F^{\rightarrow}, F_{Y'}))$

Where X' is set of next state variables.

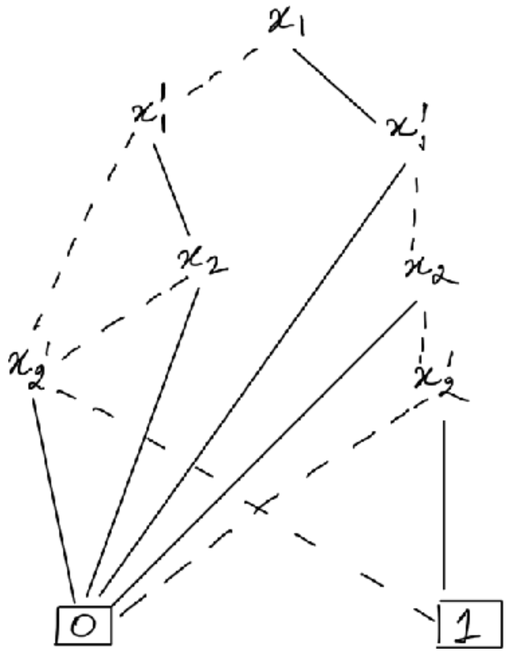
$F^{\rightarrow}$ is the ROBDD representing the transition relation.

$F_{Y'}$ is the ROBDD representing the set Y with variables $x_1, x_2, \ldots, x_n$ renamed to $x'_1, x'_2, \ldots, x'_n$

Symbolic Model Checking



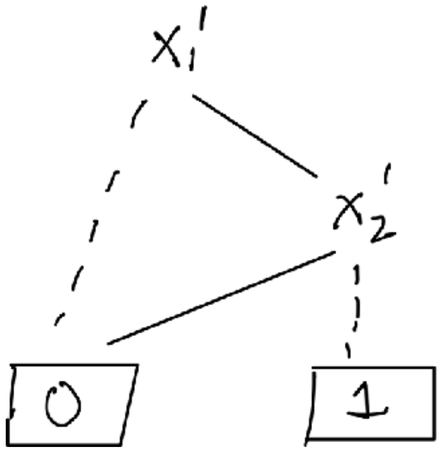
ROBDD of $F^{\rightarrow}$



$\exists \mathbf{N} x_1$

$B_{Pre(Y)} =$ exists $(X', \text{apply}(\wedge, F^{\rightarrow}, F_{Y'}))$

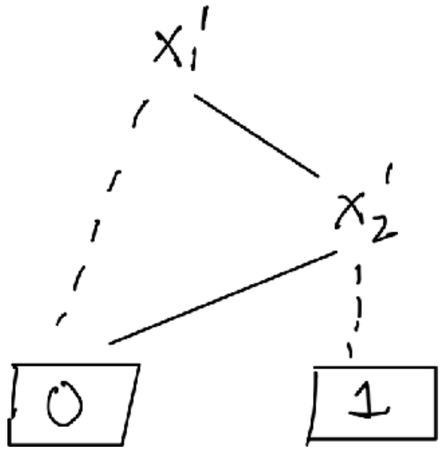
$F_{Y'} = ROBDD(s_0)$



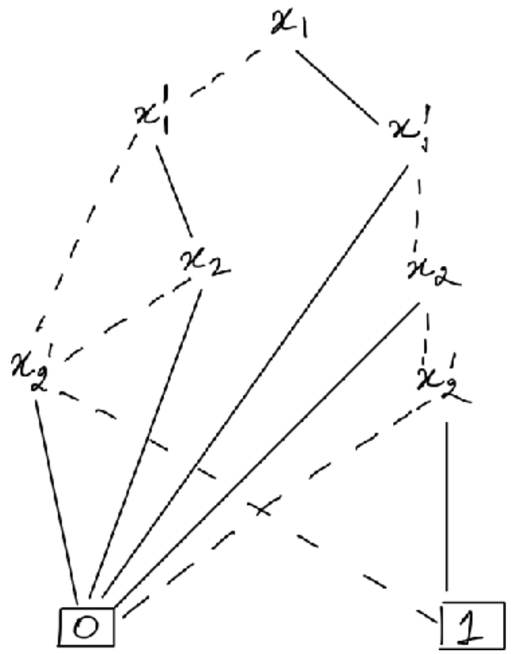
Symbolic Model Checking

$B_{Pre(Y)} =$ exists $(X', \text{apply}(\wedge, F^{\rightarrow}, F_{Y'}))$

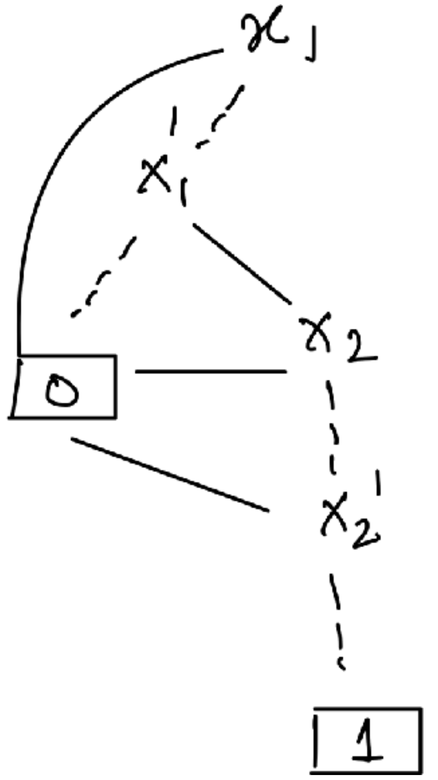
$B_{Pre(Y)} = exists(x'_1, x'_2, apply(\wedge, F^{\rightarrow}, F_{Y'})$



$F_{Y'} = ROBDD(s_0)$



ROBDD of $F^{\rightarrow}$



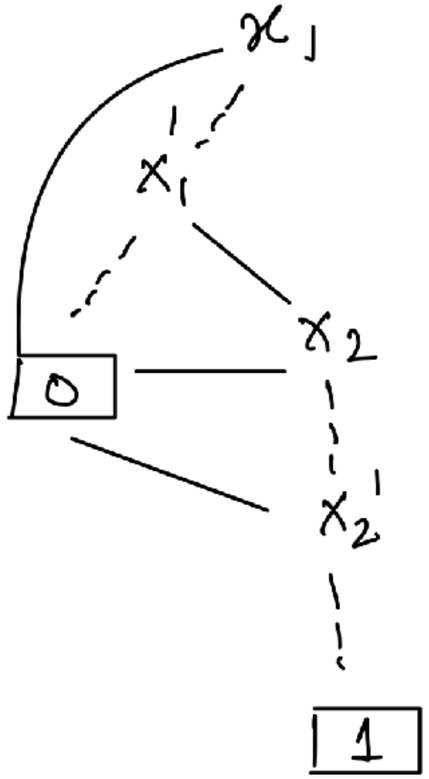
$apply(\wedge, F^{\rightarrow}, F_{Y'})$

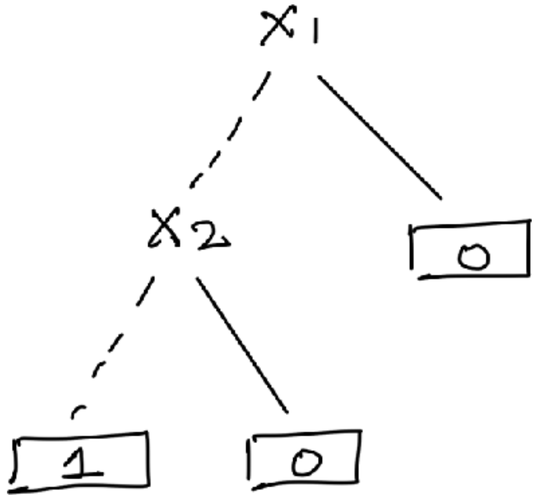
Symbolic Model Checking

$$B_{Pre(Y)} = exists(x_1', x_2', apply(\wedge, F^{\rightarrow}, F_{Y'})$$

$B_{Pre(Y)} =$

$restrict(x_1, x_2, ,0,0) \vee restrict(x_1, x_2,1,0) \vee restrict(x_1, x_2,0,1) \vee restrict(x_1, x_2,1,1)$





$restrict(x_1, x_2, F_1,1,0)$

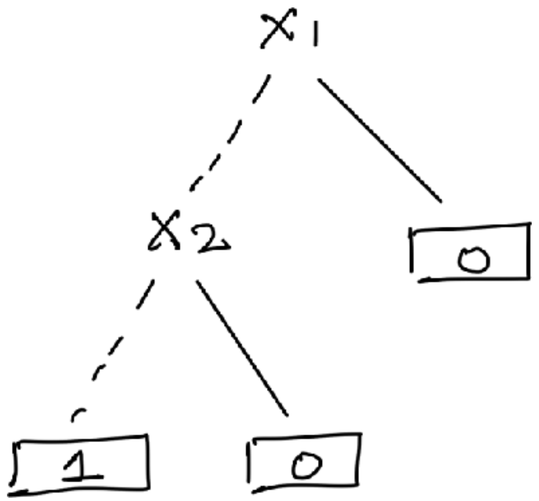
$restrict(x_1, x_2, F_1,0,0) \vee restrict(x_1, x_2, F_1,0,1)$

$restrict(x_1, x_2, F_1,1,1) = 0$

$$F_1 = apply(\wedge, F^{\rightarrow}, F_{Y'})$$

Symbolic Model Checking

$$B_{Pre(Y)} = exists(x'_1, x'_2, apply(\wedge, F^{\rightarrow}, F_{Y'}))$$



ROBDD of s_2

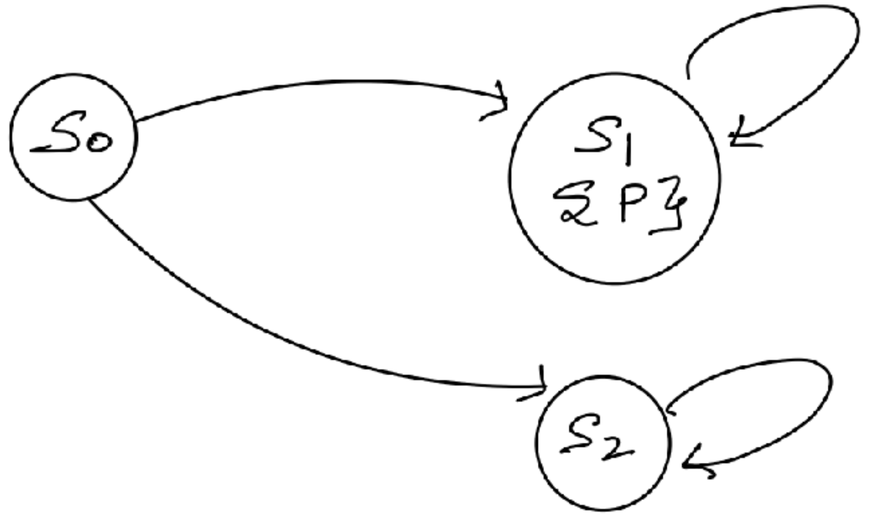
CTL Model Checking Algorithm —Symbolic Model Checking

Function Label(F, M){

Case F *of* :

True	return S
False	return {}
p	return $\{s \in S \mid p \in L(s)\}$
$\neg F_1$	return $\neg ROBDD$ *of* F_1
$F_1 \wedge F_2$	return $apply(\wedge, ROBDD(F_1), ROBDD(F_2))$
$\exists \mathbf{N} F_1$	return $\mathrm{pre}(ROBDD(F_1'), ROBDD(F^{\rightarrow}))$
$\exists \Box F_1$	return $Label\_EG(ROBDD(F_1'), ROBDD(F^{\rightarrow}))$
$\exists F_1 \mathbf{U} F_2$	return $Label\_EU(ROBDD(F_1'), ROBDD(F_2'), ROBDD(F^{\rightarrow}))$

End Case



$\exists \Diamond P$

$\cong \quad \exists [\text{true} \cup p]$

$x_1 x_2 \rightarrow$ for states

$S_0 = \quad \neg x_1 \wedge \neg x_2$

$S_1 = \quad \neg x_1 \wedge x_2$

$S_2 = \quad x_1 \wedge \neg x_2$

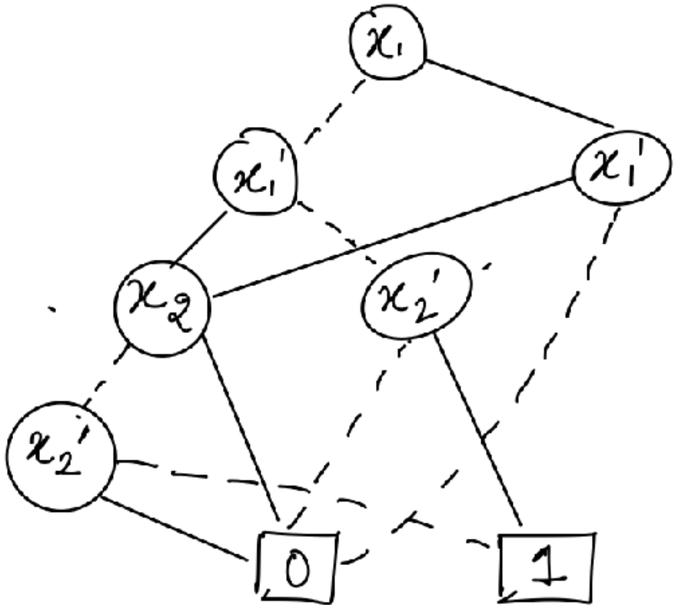
$P(x_1, x_2) = \neg x_1 \wedge x_2$

$$T: \quad (\neg x_1 \wedge \neg x_2 \wedge \neg x_1' \wedge x_2') \vee (\neg x_1 \wedge x_2 \wedge \neg x_1' \wedge x_2') \vee (\neg x_1 \wedge \neg x_2 \wedge x_1' \wedge \neg x_2') \vee (x_1 \wedge \neg x_2 \wedge x_1' \wedge \neg x_2')$$

$(S_0 \to S_1)$ $(S_1 \to S_1)$ $(S_0 \to S_2)$ $(S_2 \to S_2)$

$O = x_1, x_1', x_2, x_2'$

order



$\exists F_1 \cup F_2$

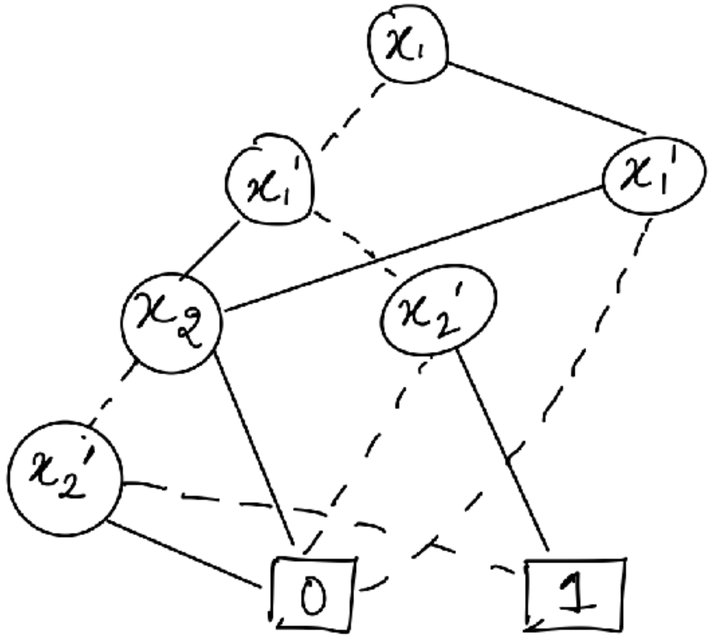
```
Label_EU([F1], [F2])
{
    X = [F2]
    Y = S
    while X ≠ Y do:
        Y = X
        X = X ∪ ([F1] ∩ pre(X))
    Return X
}
```

$\exists [true \cup p]$

pre([p])

$$\rightarrow \exists x_1' x_2' \cdot \left(T(x_1, x_2, x_1', x_2') \wedge P(x_1', x_2') \right)$$

$$\rightarrow \exists x_1' x_2' \cdot$$



$\wedge$