COL:750

Foundations of Automatic Verification

Instructor: Priyanka Golia

Course Webpage

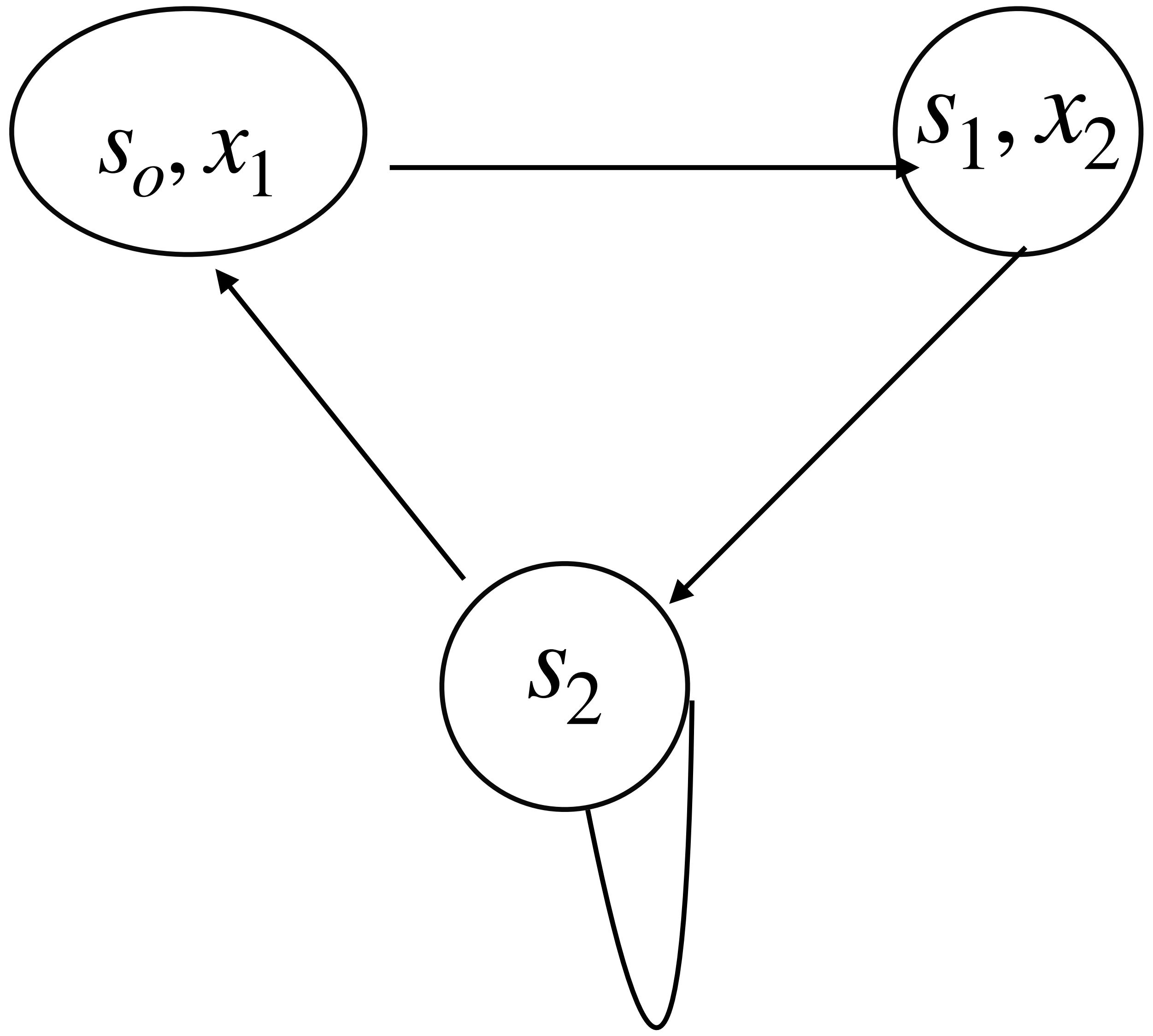
https://priyanka-golia.github.io/teaching/COL-750/index.html

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

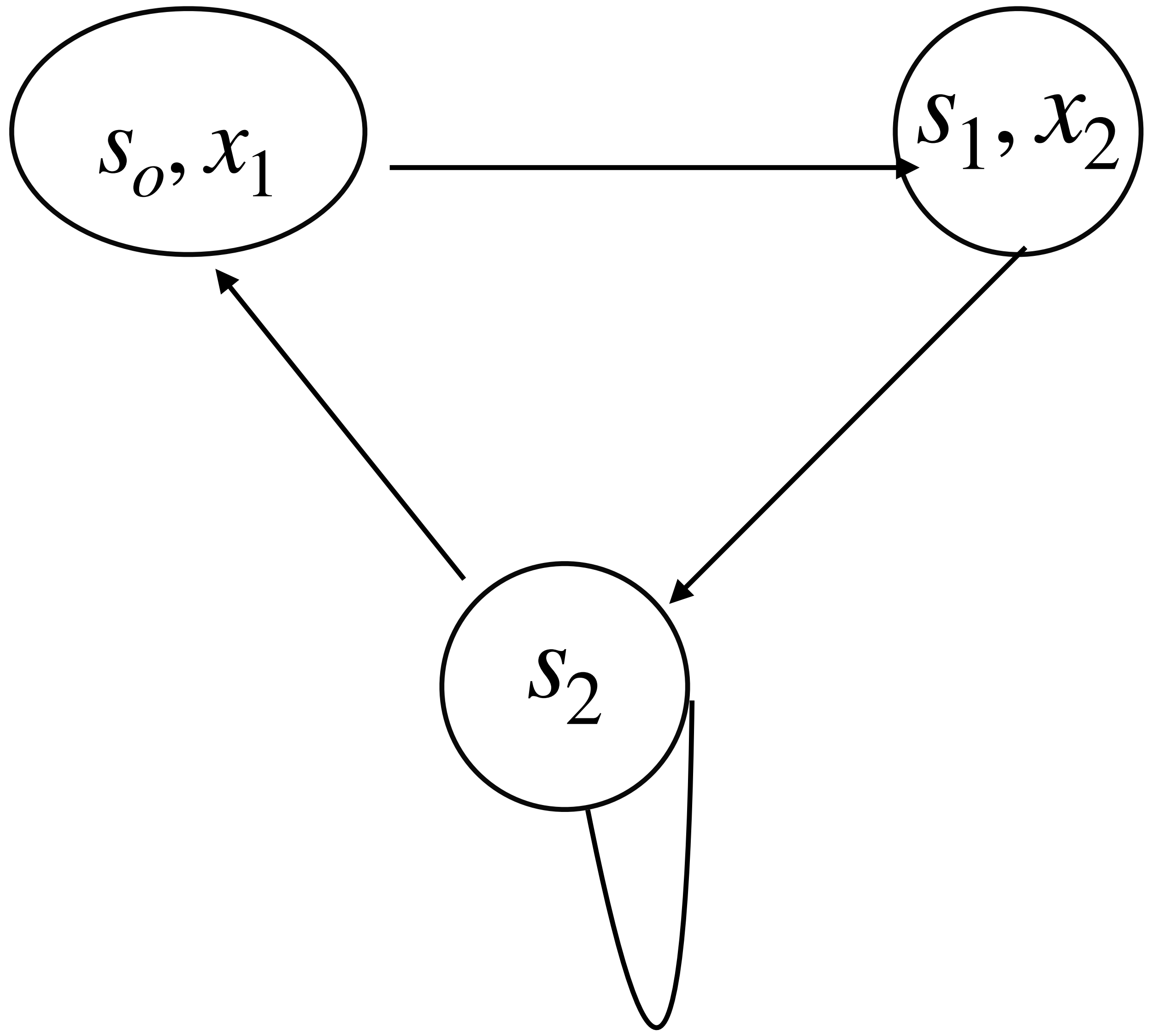


Implementing CTL Model Checking using BDDs

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



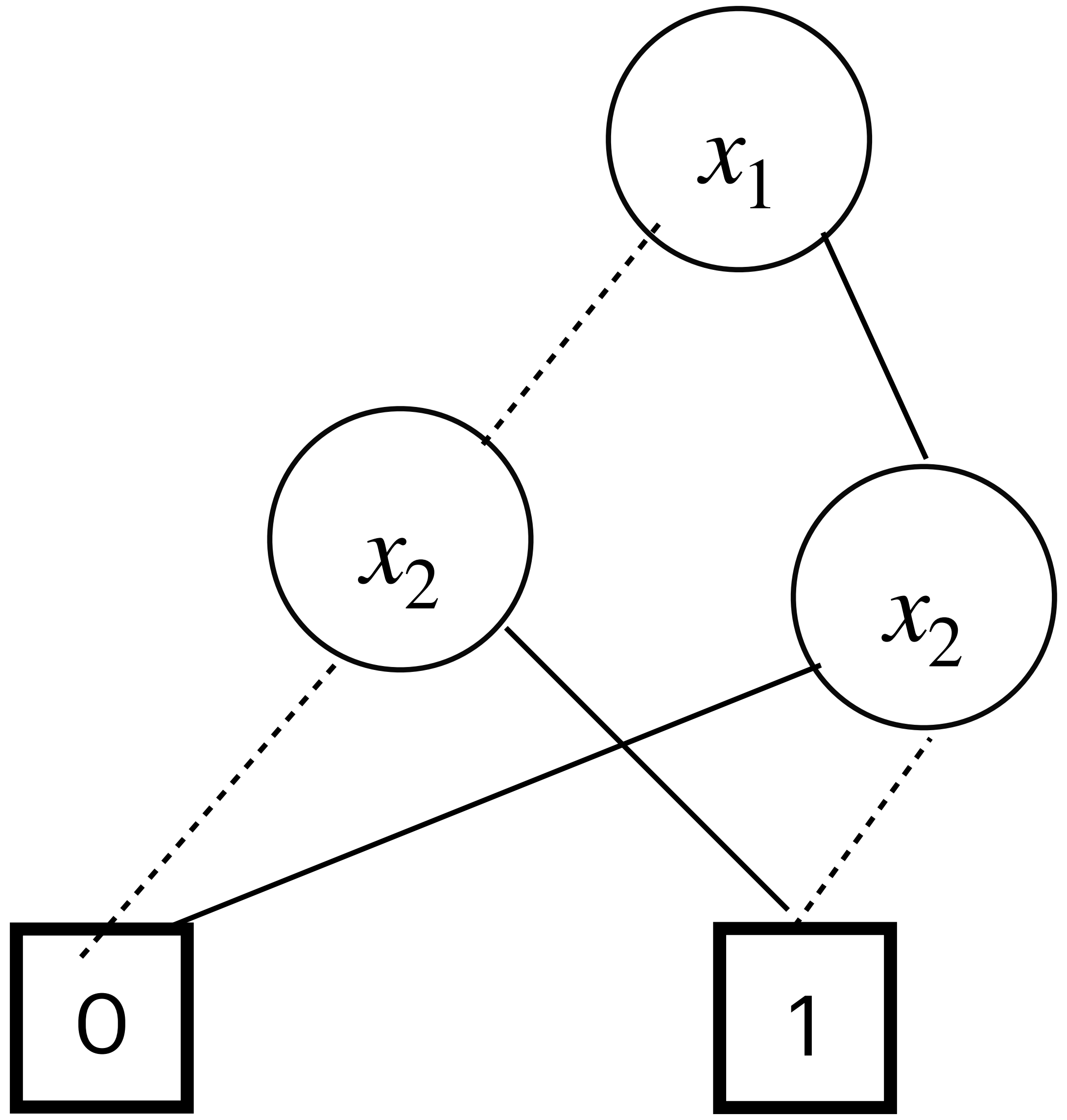
Set of states	Representation by	Representation by Boolean
∅		0
{so}	(1,0)	$x_1 . \neg x_2$
{s1}	(0,1)	$\neg x_1 . x_2$
{s2}	(0,0)	$\neg x_1 . \neg x_2$
{s0,s1}	(1,0),(0,1)	$x_1 . \neg x_2 + \neg x_1 . x_2$
{s0,s2}	(1,0),(0,0)	$x_1 . \neg x_2 + \neg x_1 . \neg x_2$
{s1,s2}	(0,1),(00)	$\neg x_1 . x_2 +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

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)	$x_1 . \neg x_2$
{s1}	(0,1)	$\neg x_1 . x_2$
{s2}	(0,0)	$\neg x_1 . \neg x_2$
{s0,s1}	(1,0),(0,1)	$x_1 . \neg x_2 + \neg x_1 . x_2$
{s0,s2}	(1,0),(0,0)	$x_1 . \neg x_2 + \neg x_1 . \neg x_2$
{s1,s2}	(0,1),(00)	$\neg x_1 . x_2 + \neg x_1 . \neg x_2$
{s0,s1,s2}	(1,0),(0,1),(0,0)	$x_1 . \neg x_2 + \neg x_1 . x_2 + \neg x_1 . \neg x_2$

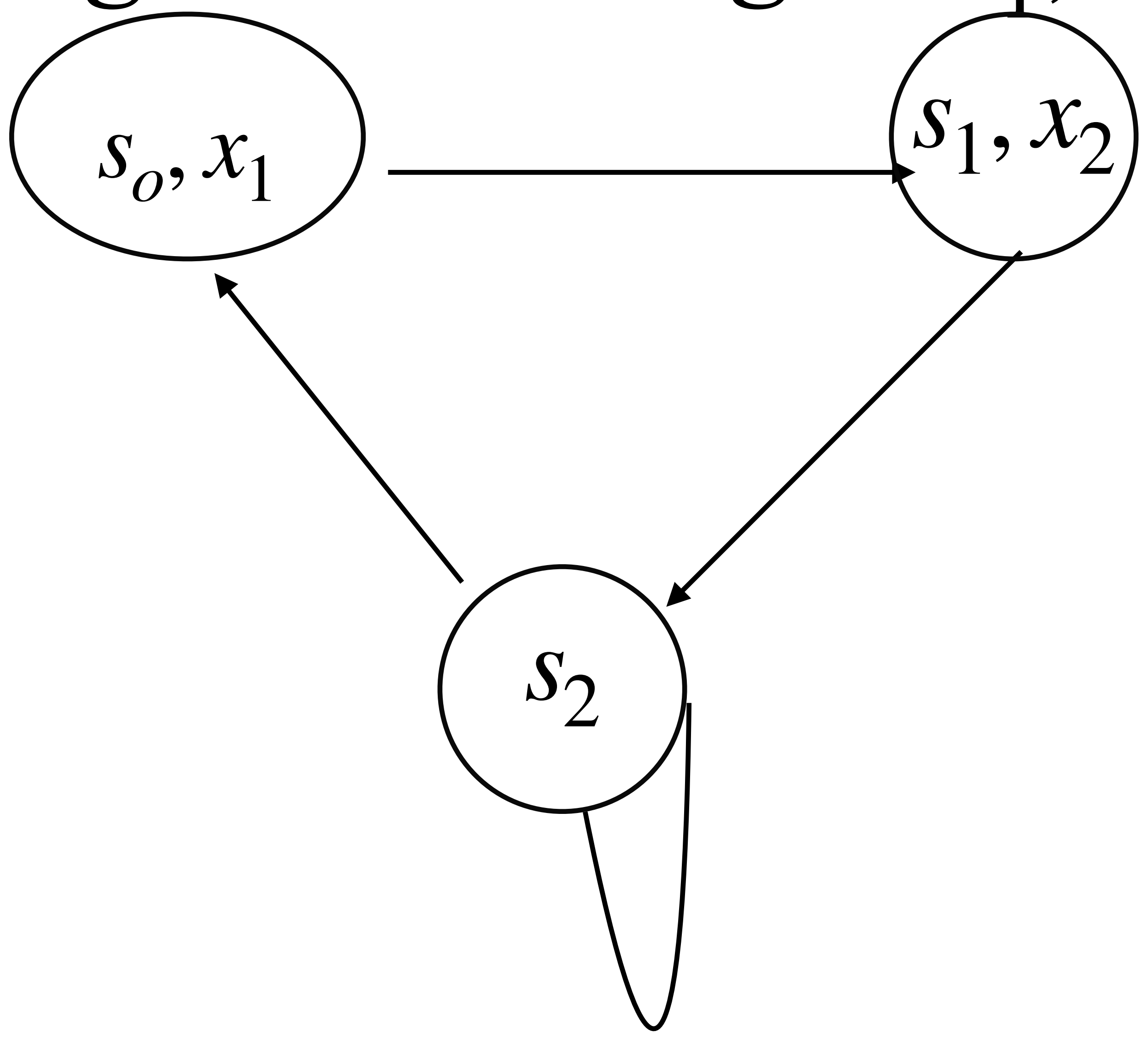
Implementing CTL Model Checking using BDDs

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
..	..	..	..	..

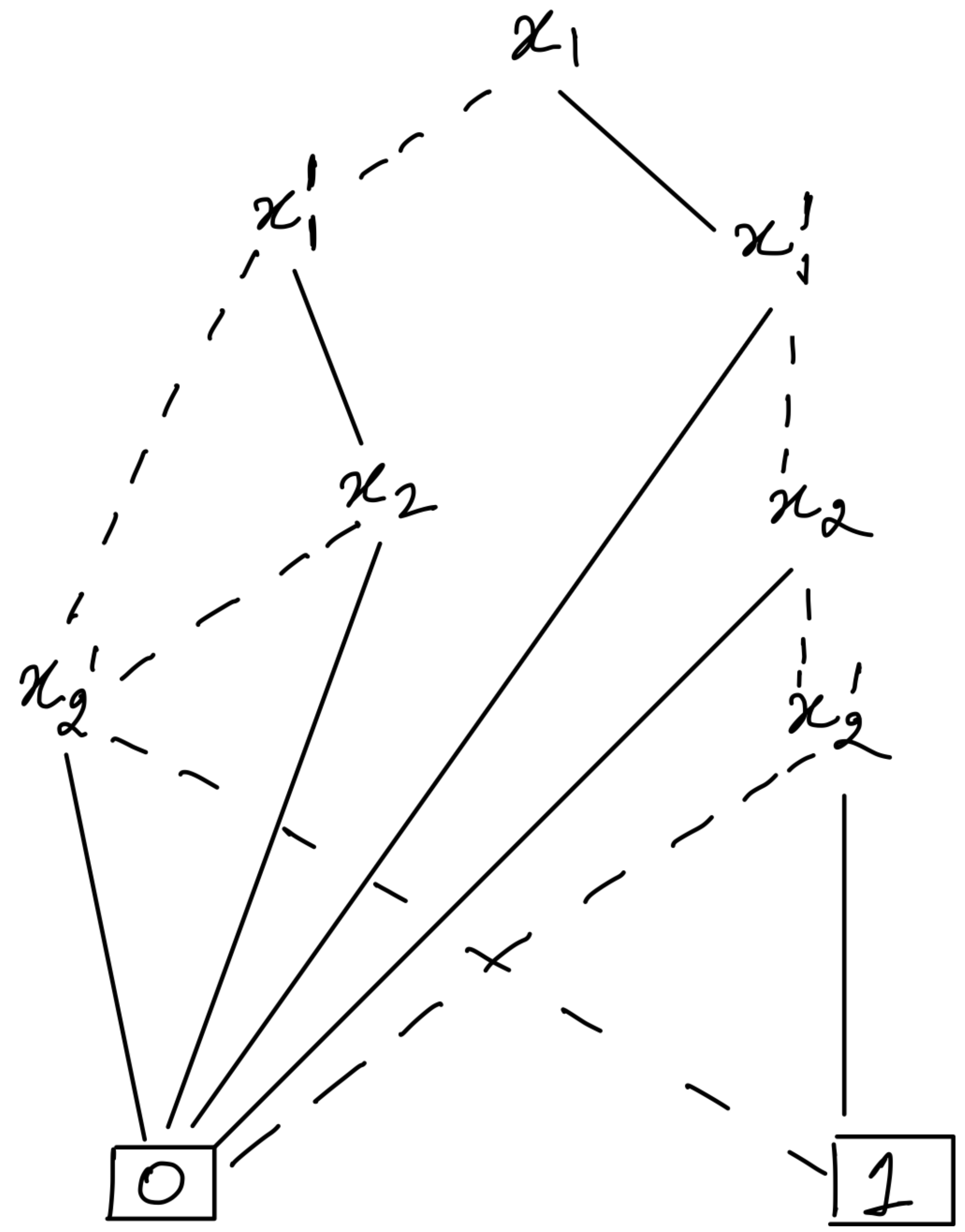
Implementing CTL Model Checking using BDDs

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
..	..	..	..	..

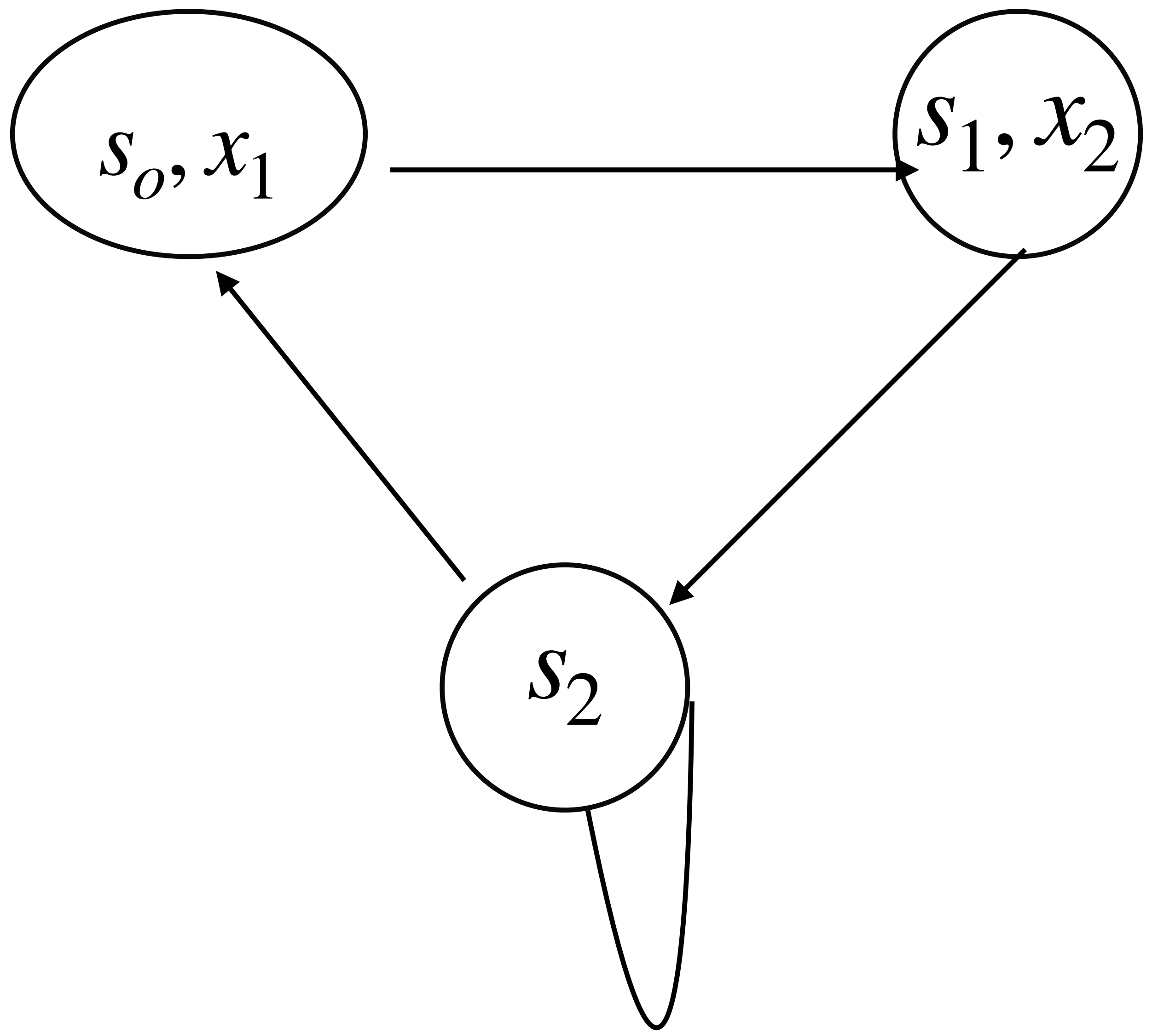
Implementing CTL Model Checking using BDDs

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)

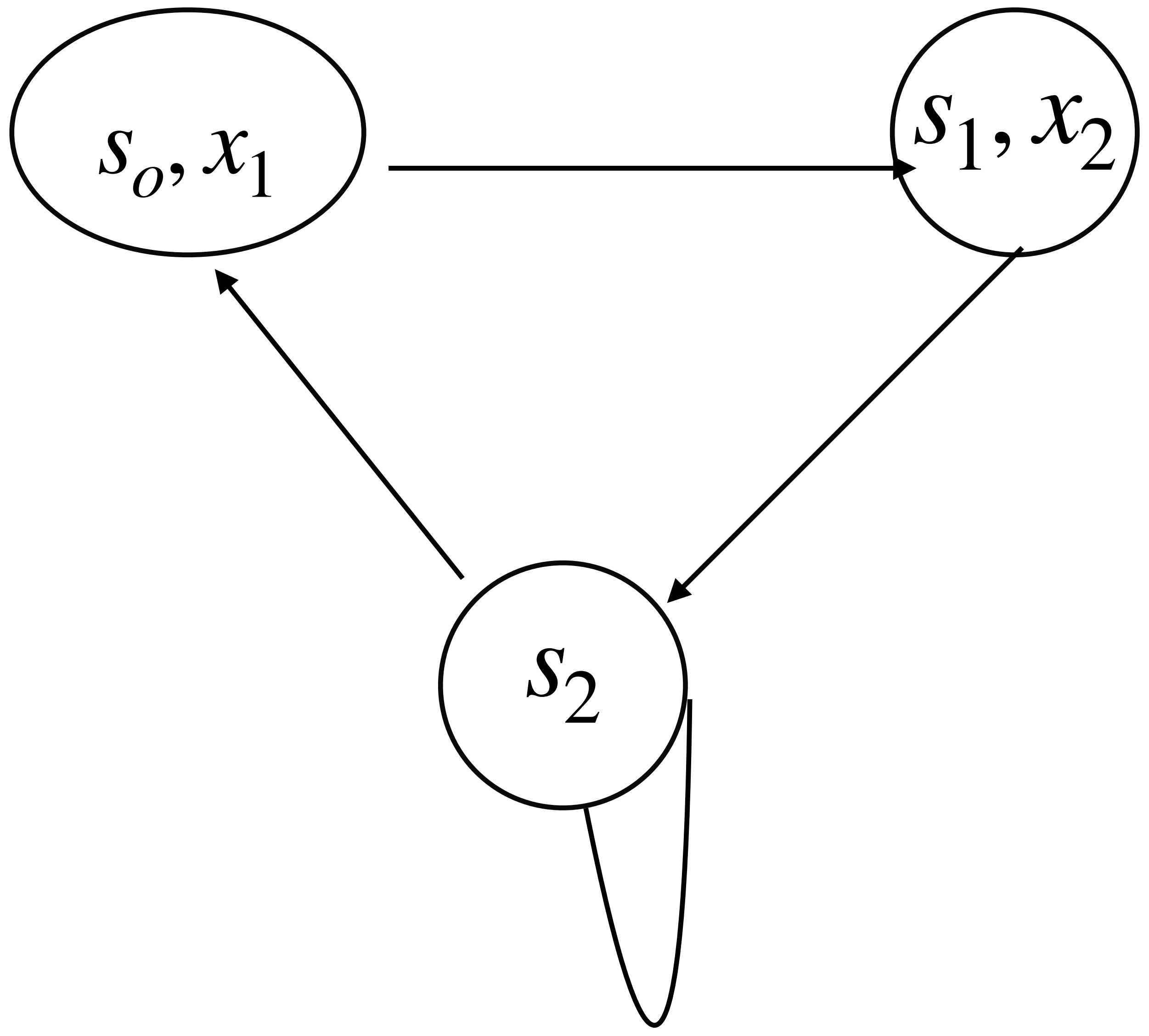
$$B_{Pre(Y)} = \text{ exists } (X', \text{apply}(\wedge, F^{\rightarrow}, F_{Y'}))$$

Where X' is set of next state variables.

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

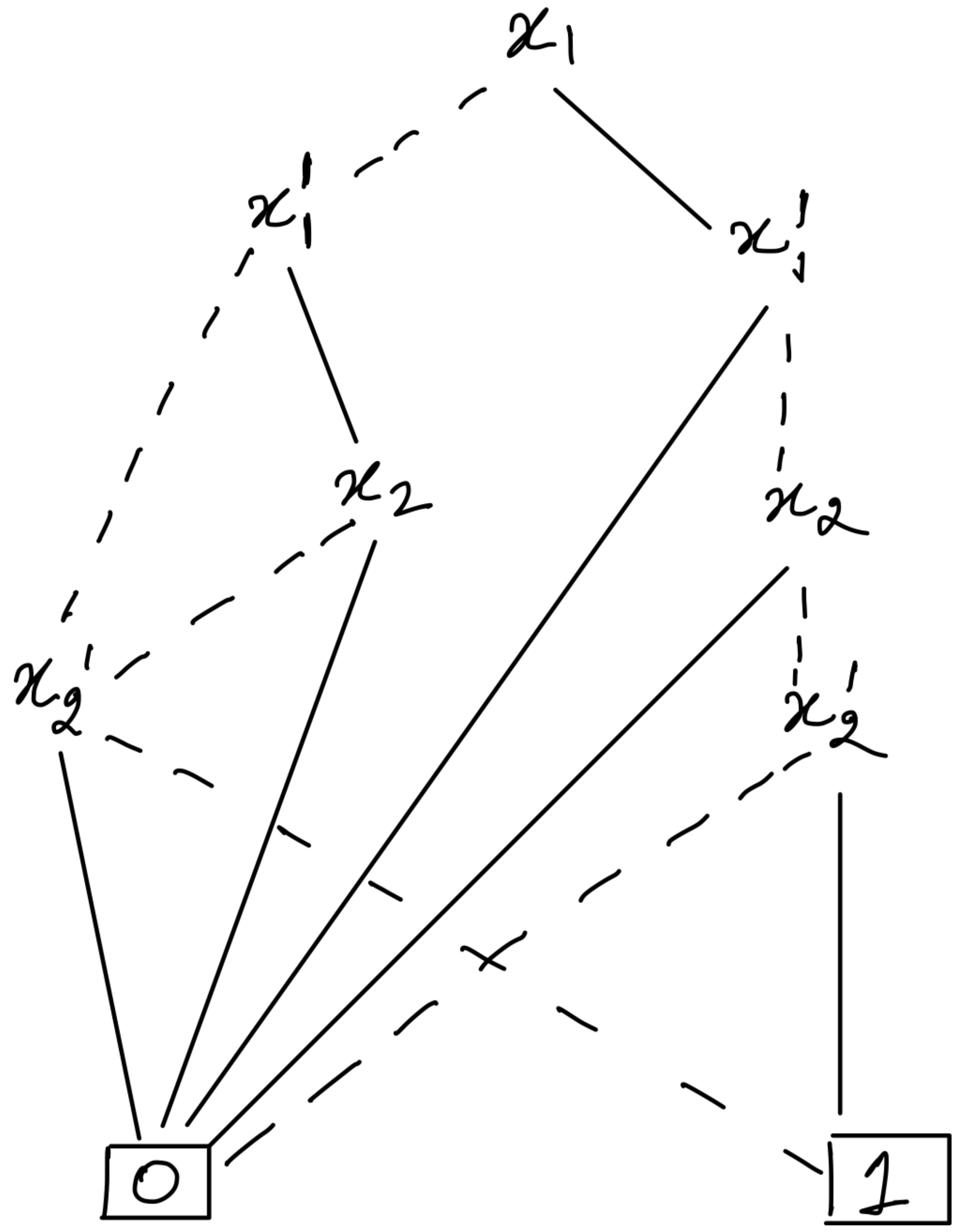
$F_{Y'}$ is the ROBDD representing the set Y with variables $x_1, x_2, \ldots, x_n$ renamed to $x_1', x_2', \ldots, x_n'$

Symbolic Model Checking



$S = x_1 . \neg x_2 + \neg x_1 . x_2 + \neg x_1 \neg x_2$

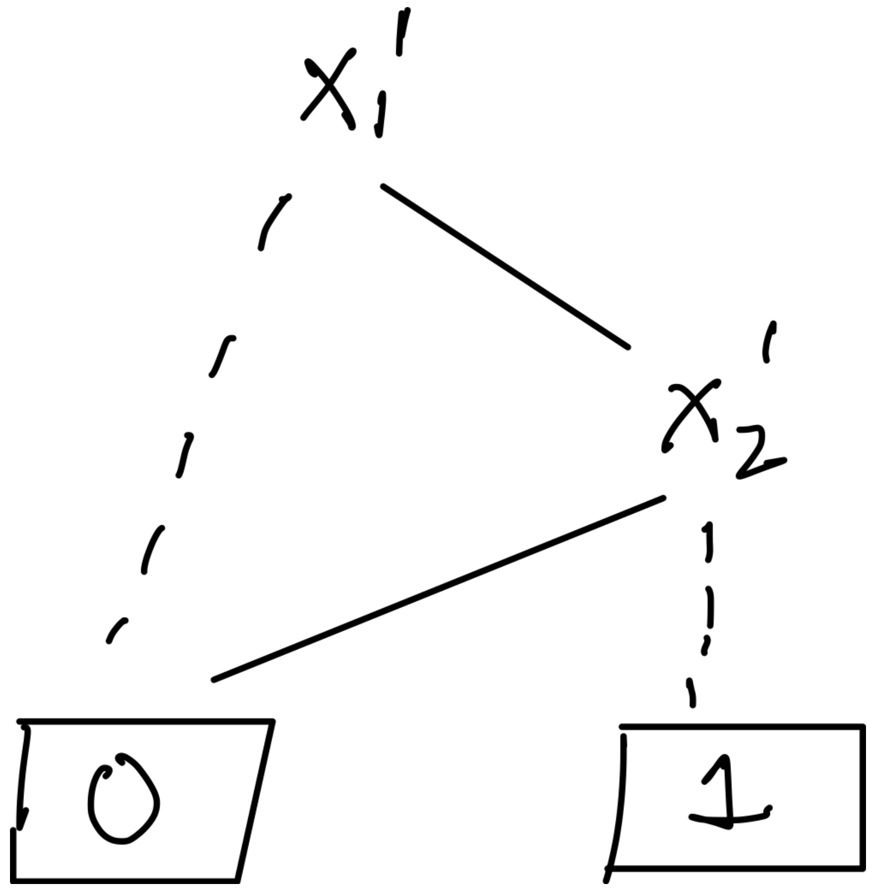
ROBDD of $F^{\rightarrow}$



$\exists \mathbf{N} x_1$

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

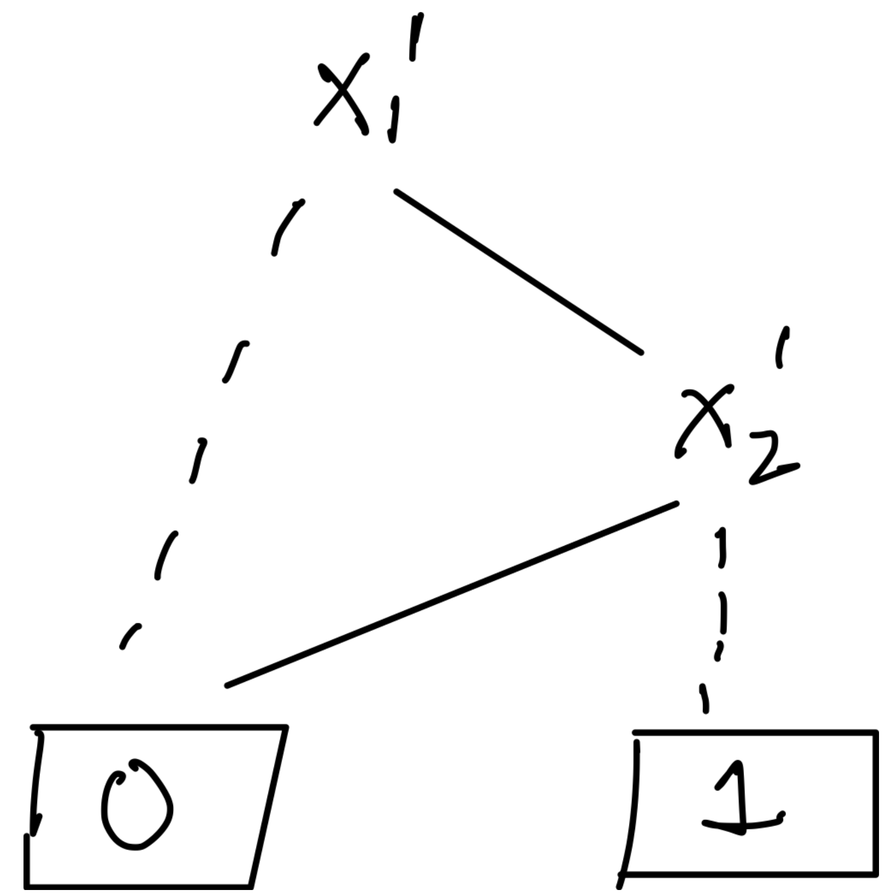
$F_{Y'} = ROBDD(s_0)$



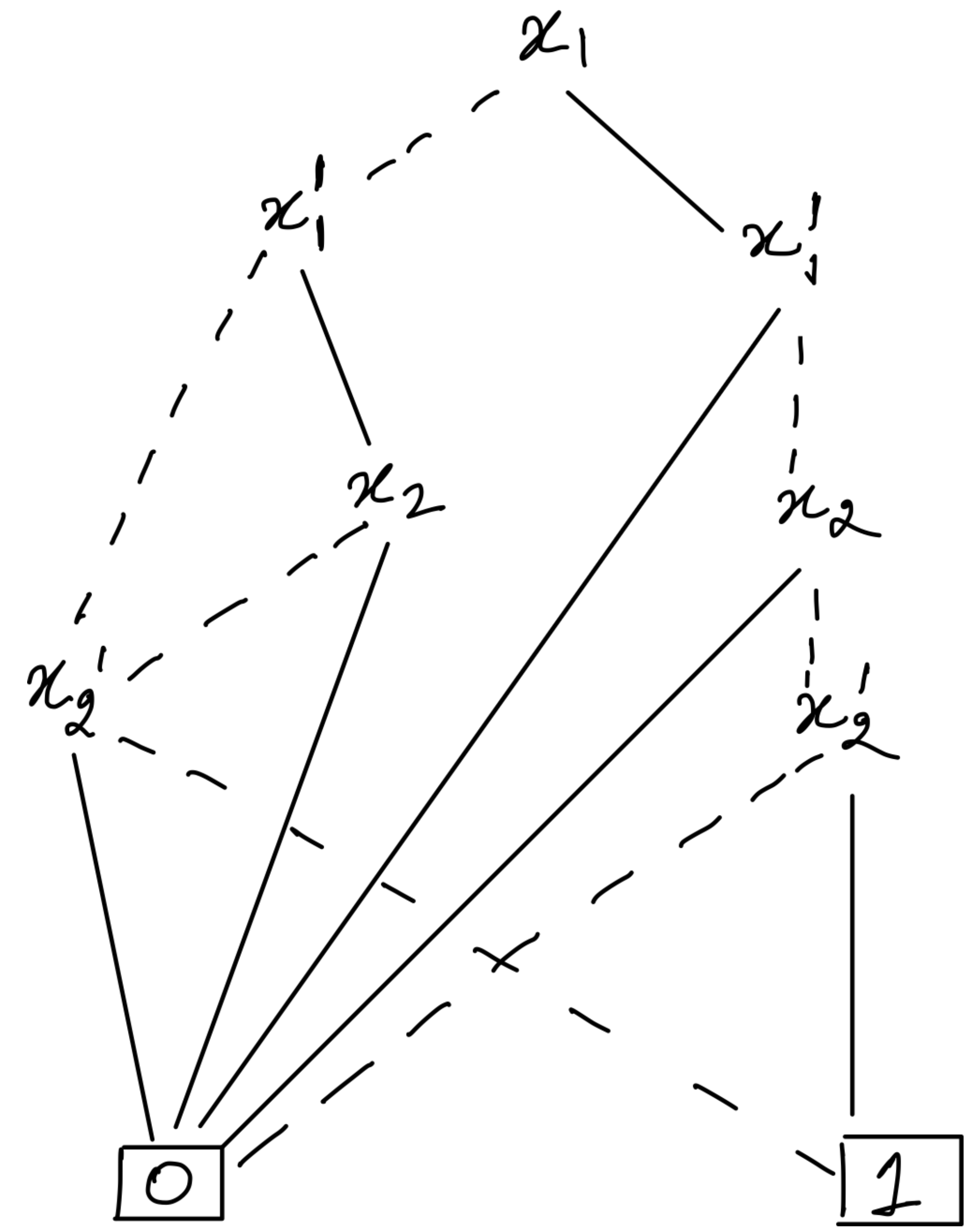
Symbolic Model Checking

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

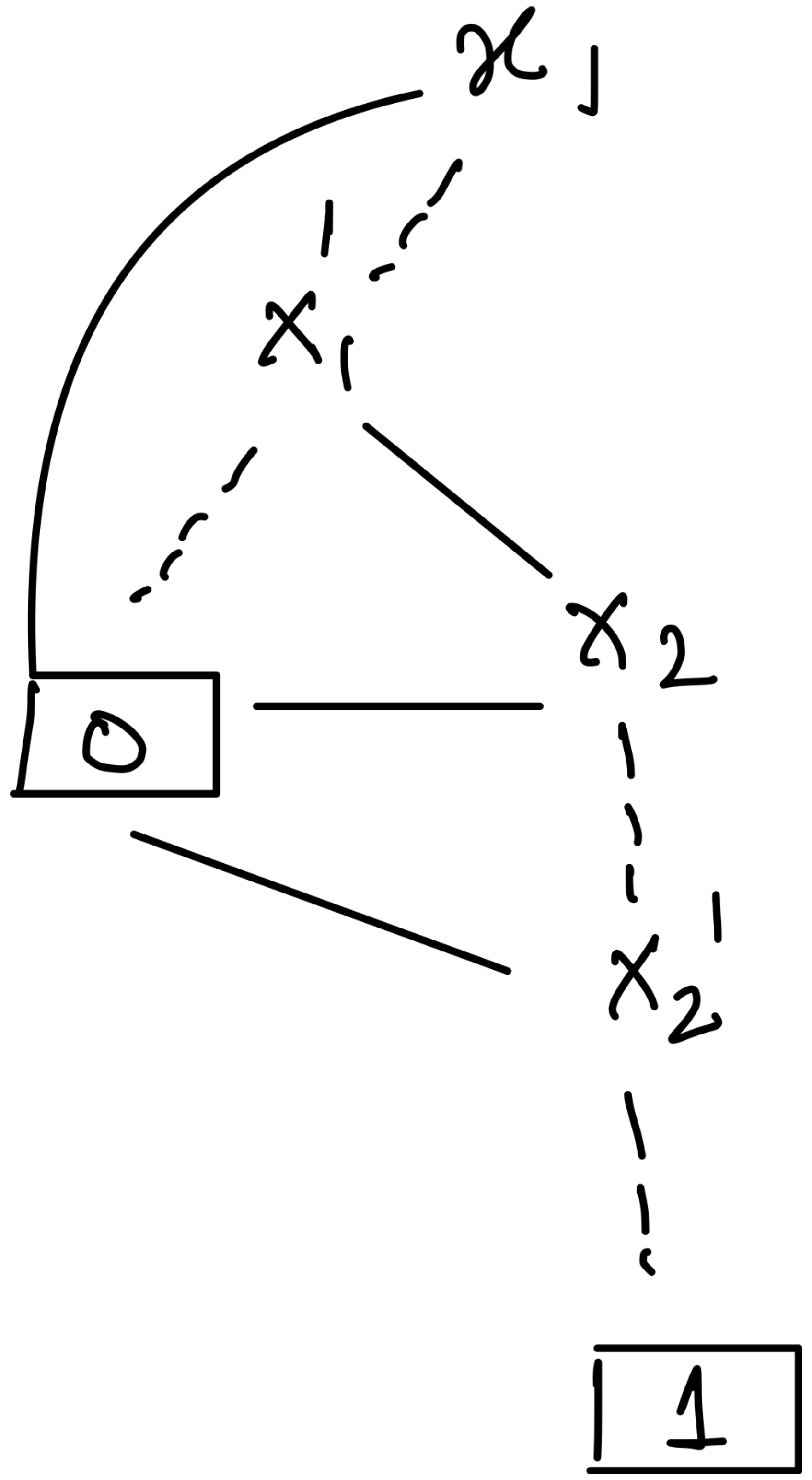
$$B_{Pre(Y)} = exists(x_1', x_2', apply(\wedge , F^{\rightarrow}, F_{Y'})$$



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



ROBDD of $F^{\rightarrow}$



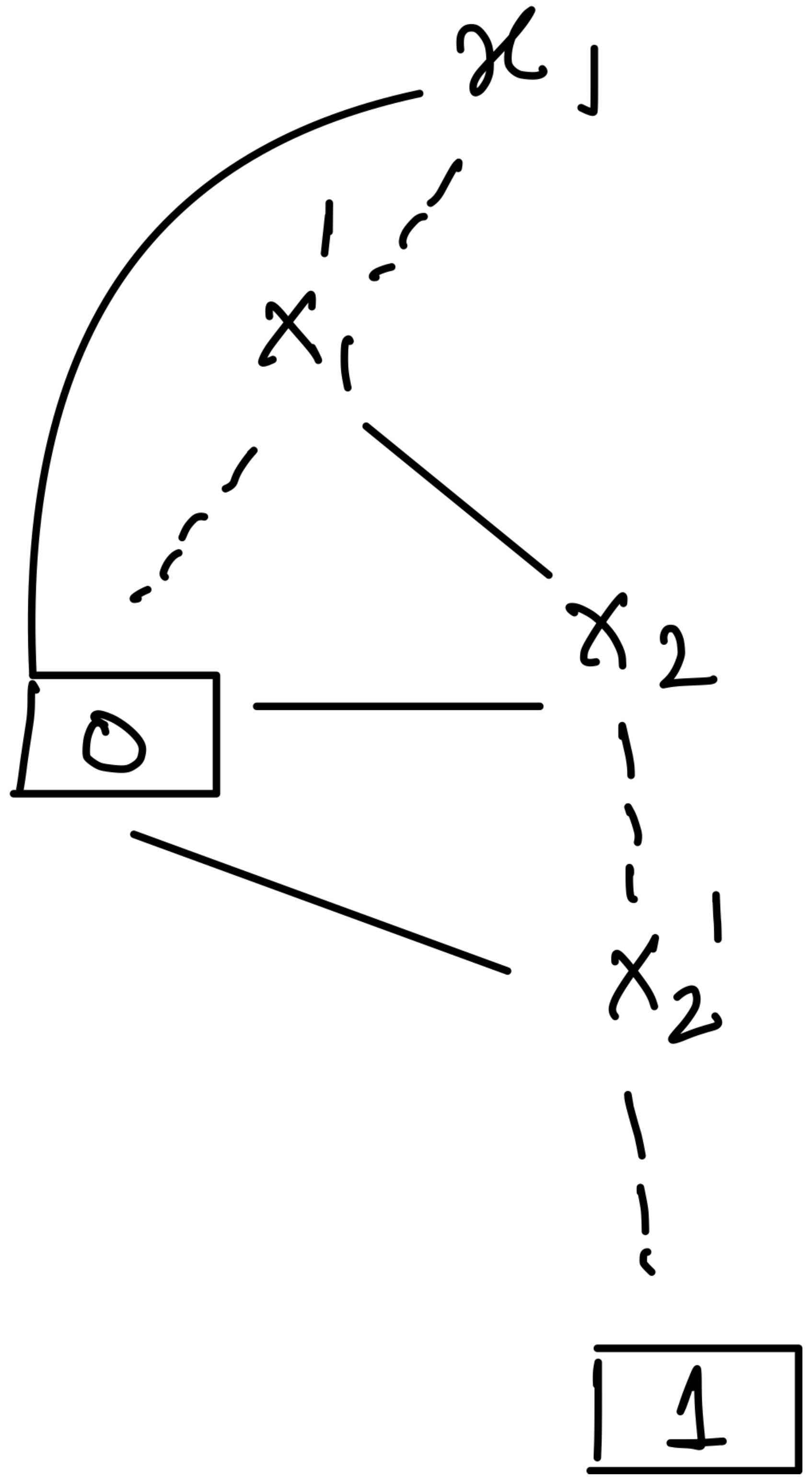
$apply(\wedge , F^{\rightarrow}, F_{Y'})$

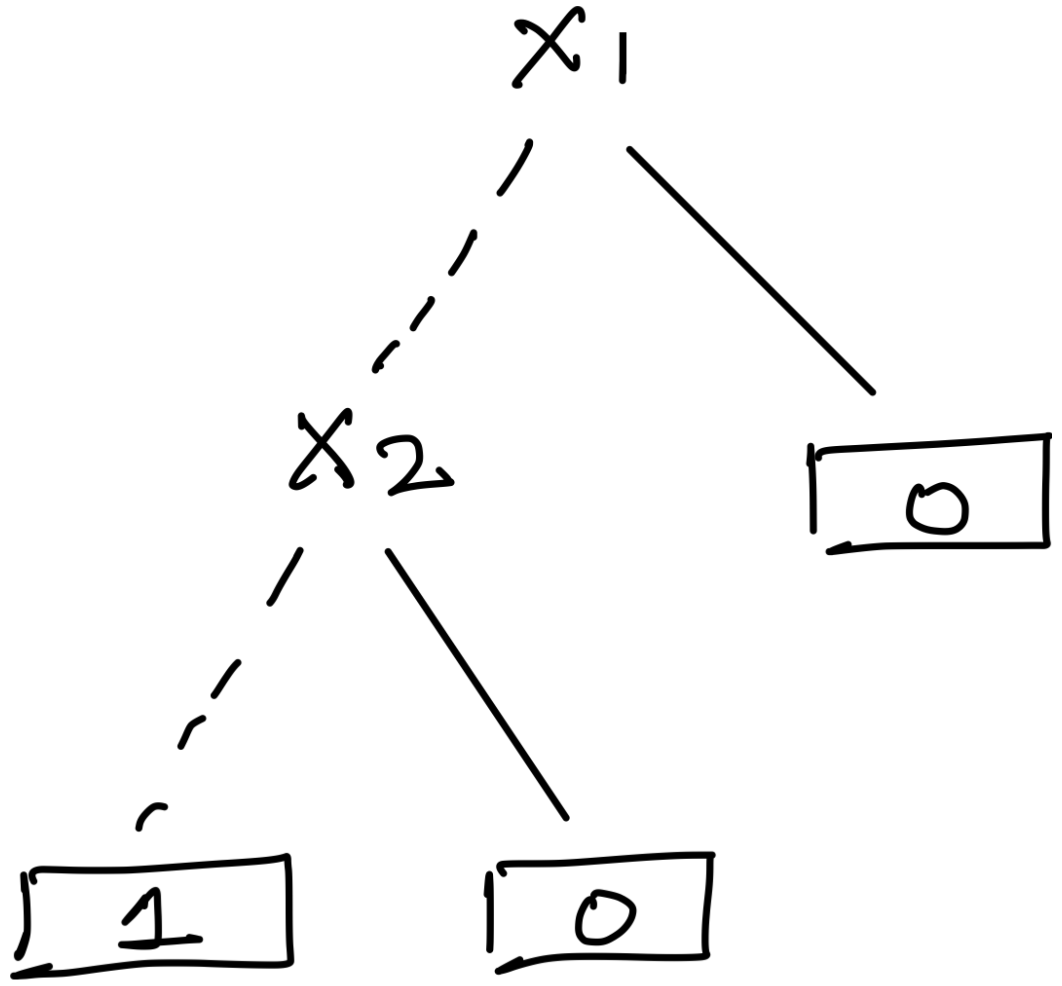
Symbolic Model Checking

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

$B_{Pre(Y)} =$

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





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

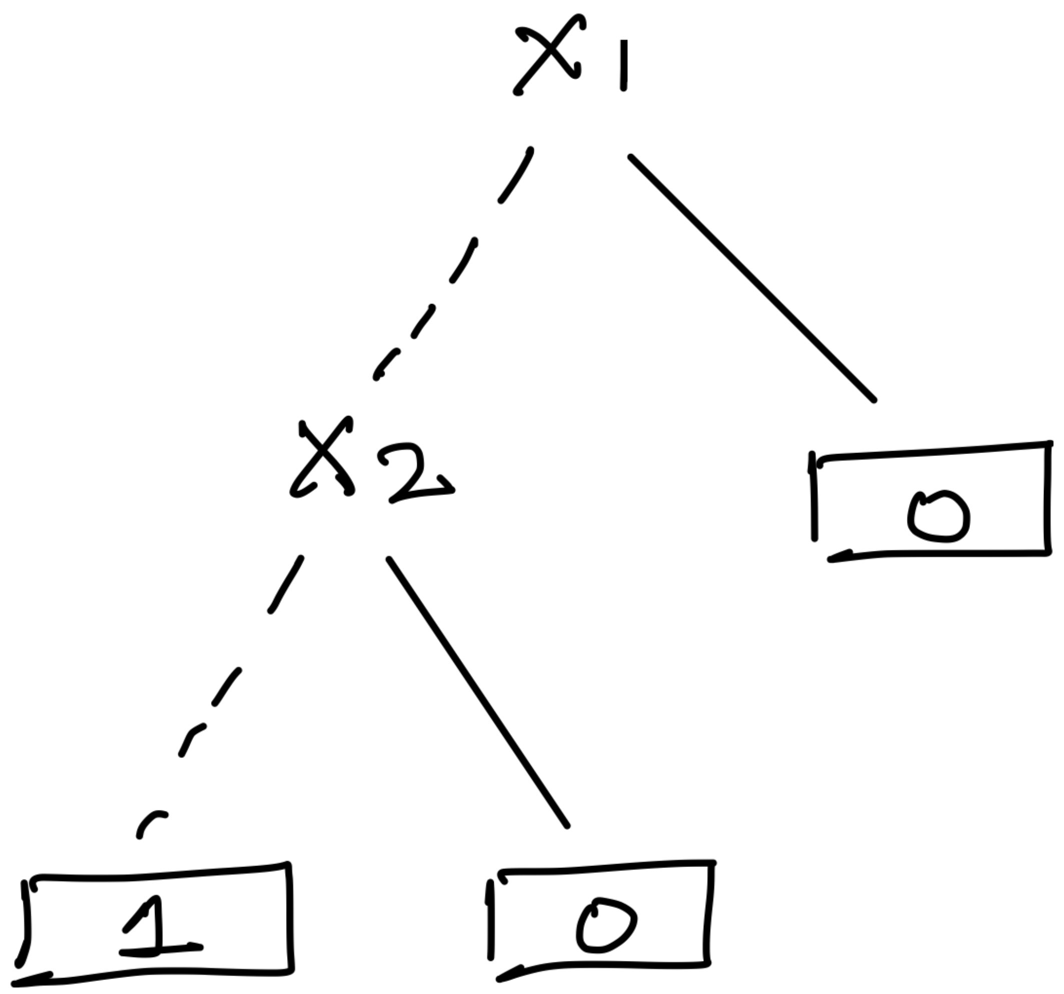
$restrict(x_1, x_2, F_1, 0,1) = 0$

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

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

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 $\mathrm{pre}(ROBDD(F_1'), ROBDD(F^{\rightarrow}))$
$\exists \square F_1$	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

CTL Model Checking Algorithm —Symbolic Model Checking

Problems with BDD:

1. BDDs are canonical representation — often become too large.
2. Variable ordering must be uniform along paths.
3. Selecting right variable ordering for small BDDs.
    1. This itself is an open problem, and can consume too much time to predict a right ordering for given instance.
    2. Sometime, no space efficient variable ordering exists.

Bounded Model Checking with SAT (BMC)

- Method used by most “industrial” model checkers.
- BMC uses SAT procedure instead of BDDs.
    - Uses Boolean encoding for transitions and set of states.
- Can handle much larger design.

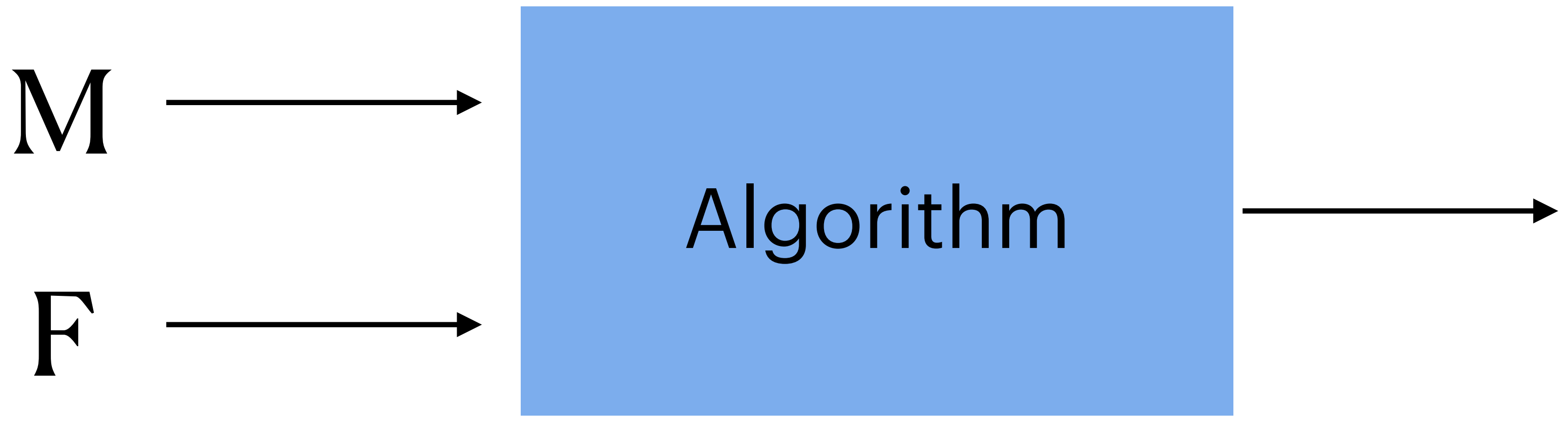
Can we reach a desired state in k steps?

Verification of safety properties — can we find a bad state in k steps?

Verification — can we find a counterexample in k steps?

Model Checking Algorithm — so far

$$M, s \vDash F?$$



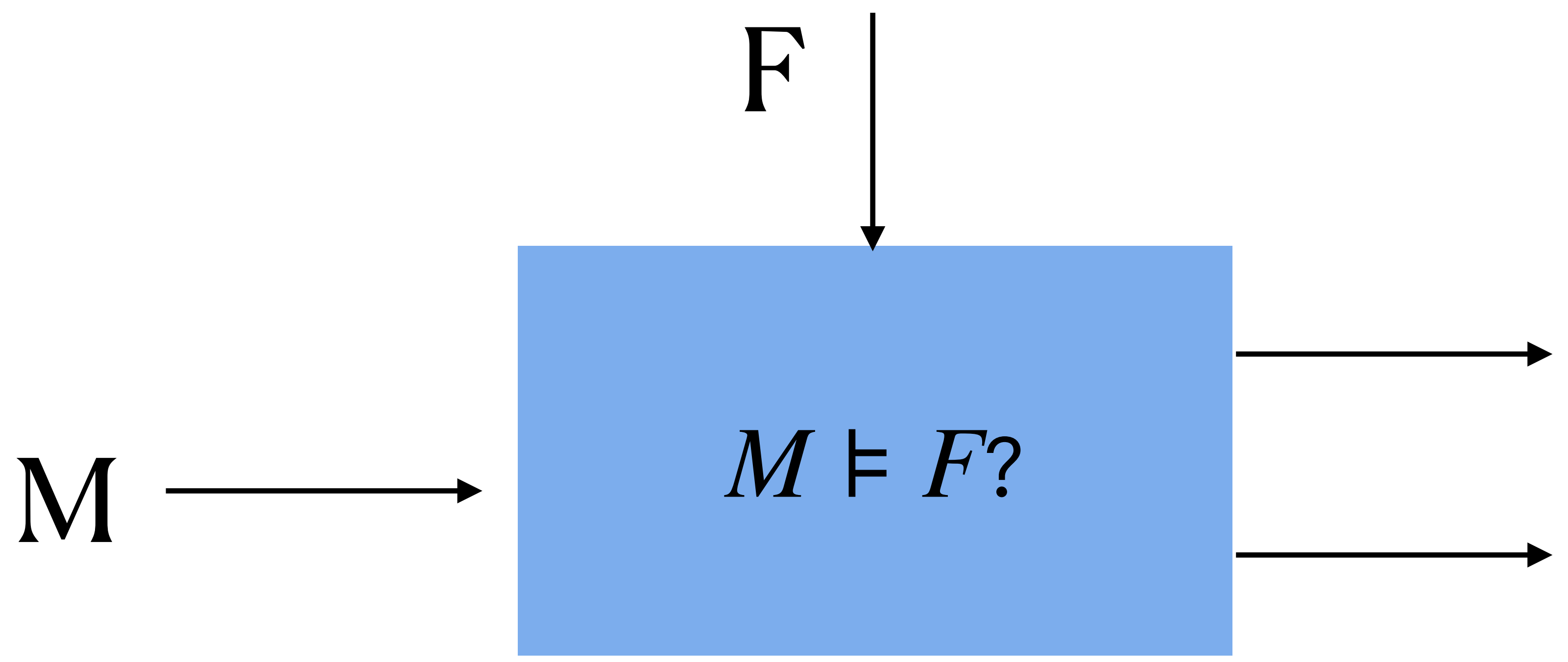
$S' \quad s.t.\ S' \subseteq S$, and $M, s \vDash F\ \forall s \in S'$

All states s of the model M that satisfy F

Note that not necessarily $I \subseteq S'$

Model Checking Algorithm

$M, s \vDash F?$



Yes, if $M, s \vDash F, \forall s \in I$

No, if $\exists s \in I\ s.t.\ M, s \nvDash F$,

A path of the system M demonstrating that M can't satisfy F

Bounded Model Checking with SAT (BMC)

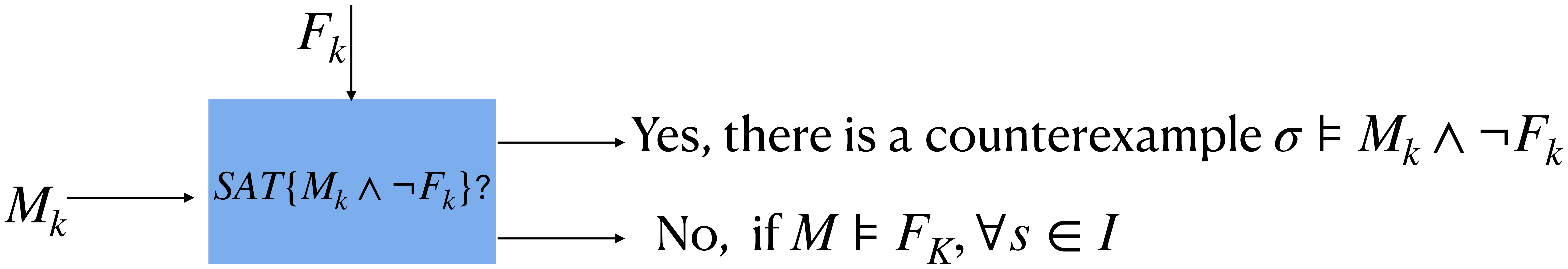
Given: Transition system M, Temporal logic formula F, and a user-supplied time bound k

Output: UNSAT, if M unrolled upto k satisfies F

A counterexample if M unrolled upto k don't satisfy F

Model Checking Algorithm

$M, s \models F?$



Bounded Model Checking with SAT (BMC)

Given: Transition system M, Temporal logic formula F, and a user-supplied time bound k

Output: UNSAT, if M unrolled upto k satisfies F

A counterexample if if M unrolled upto k don’t satisfy F

Bounded Model Checking with SAT (BMC)

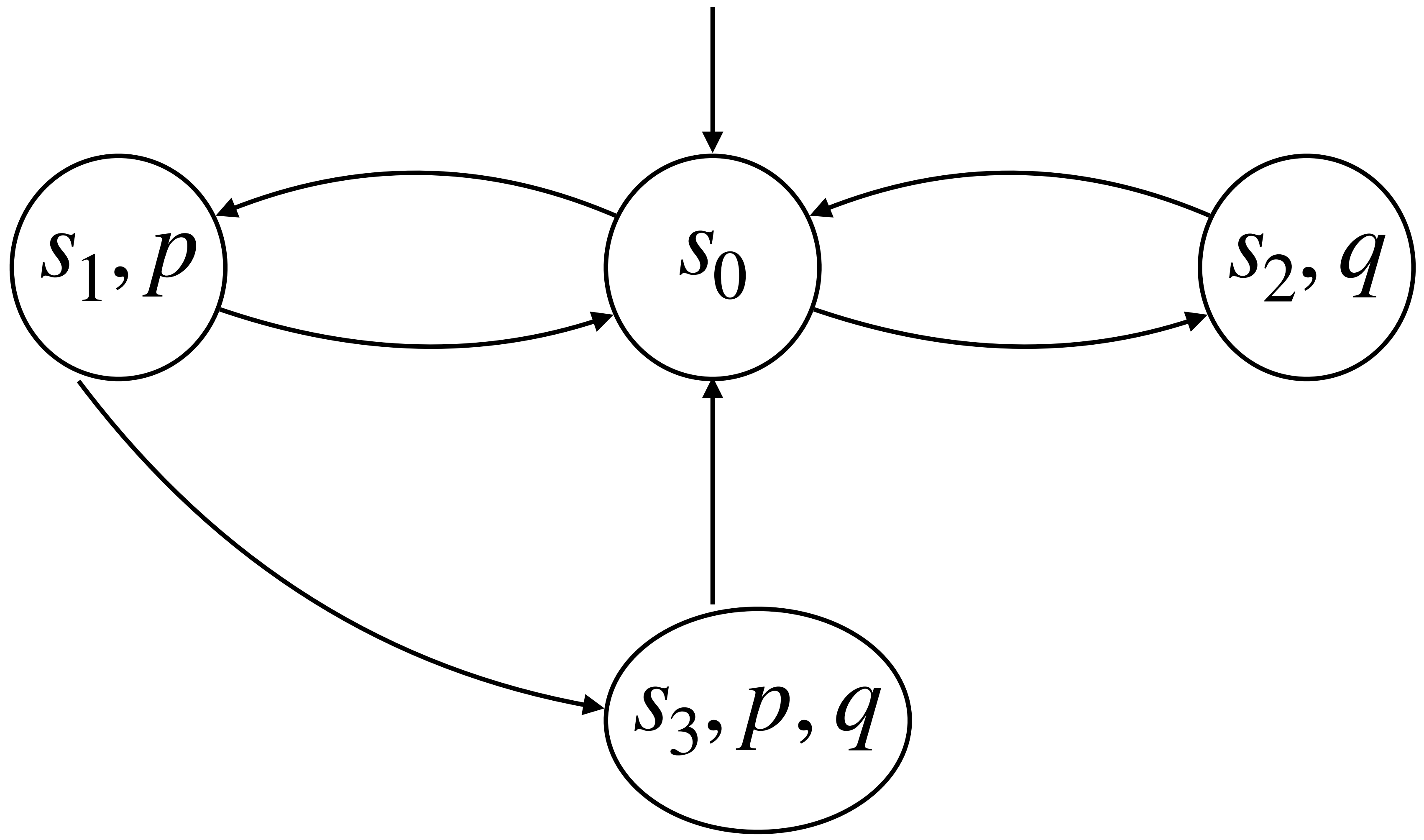
General idea:

1. Convert transition system to propositional encoding — unroll for path length k
2. Convert temporal formula along the states to propositional encoding for k length.
3. Using SAT Solvers look for counterexamples

Bounded Model Checking with SAT (BMC)

Given two processes P and Q which share a resource R.

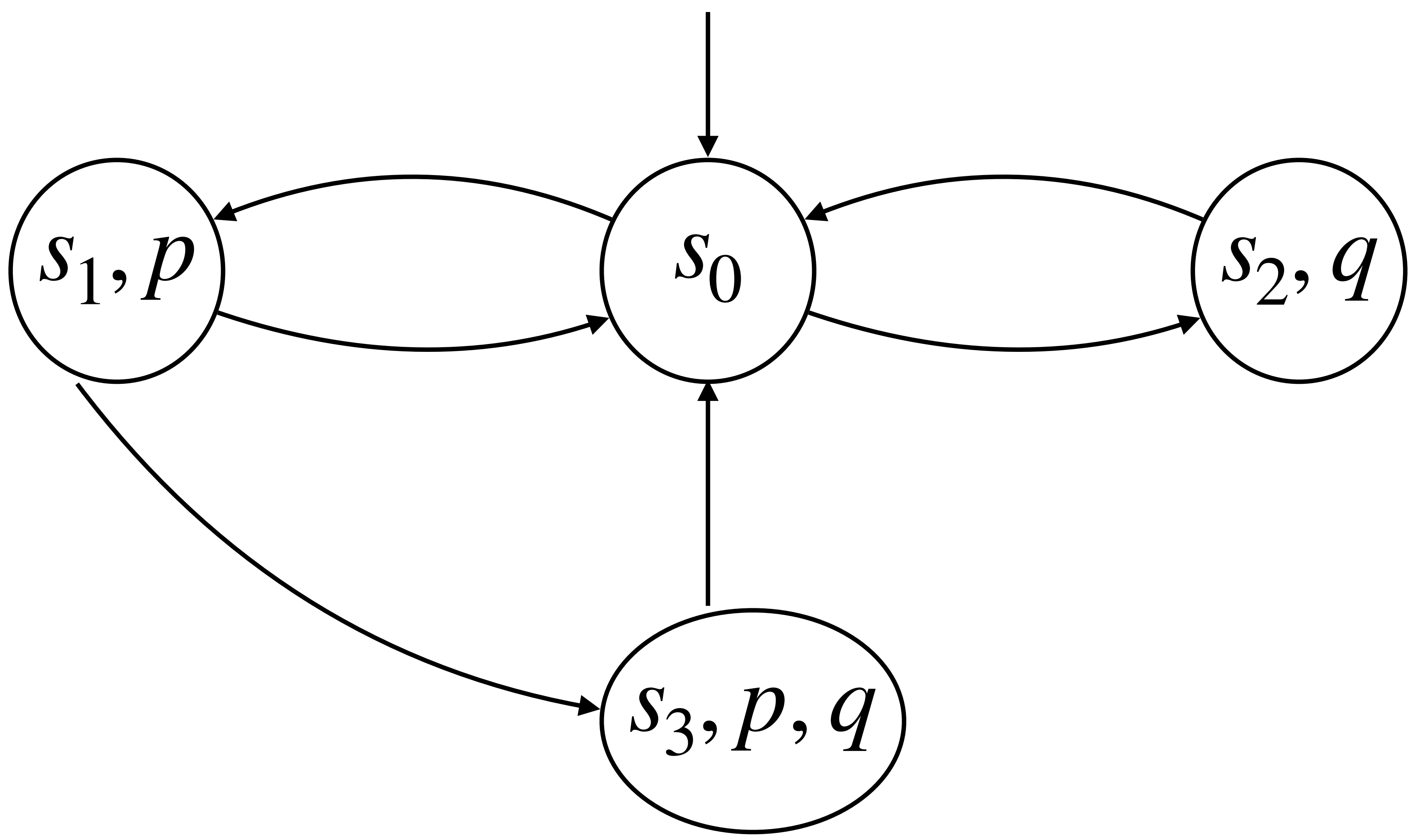
1. If R is accessed by P, then property p is True.
2. If R is accessed by Q, then property q is True.



Does $\forall \Box \neg (p \wedge q)$

K = 2

Bounded Model Checking with SAT (BMC)



Does $\forall \Box \neg (p \wedge q)$

K = 2

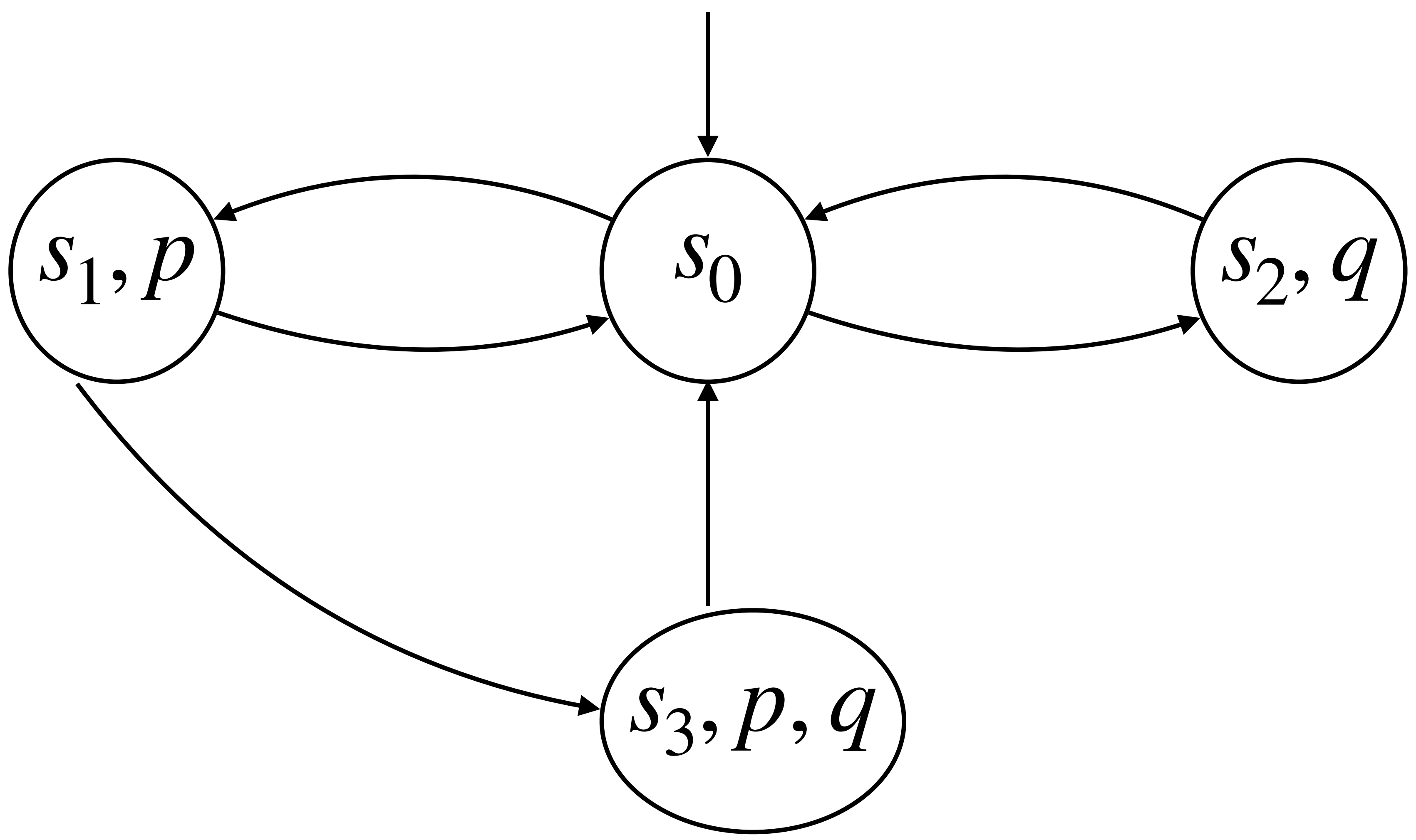
$$M_k = (\neg p_0 \wedge \neg q_0) \wedge ((\neg p_0 \wedge \neg q_0 \wedge p_1 \wedge \neg q_1') \vee (\neg p_0 \wedge \neg q_0 \wedge \neg p_1' \wedge q_1'))$$

K = 1

$$\begin{aligned} M_k = {} & (\neg p_0 \wedge \neg q_0) \wedge ((\neg p_0 \wedge \neg q_0 \wedge p_1 \wedge \neg q_1') \vee (\neg p_0 \wedge \neg q_0 \wedge \neg p_1' \wedge q_1')) \\ & \wedge (((p_1 \wedge \neg q_1' \wedge p_2' \wedge q_2') \vee (p_1 \wedge \neg q_1' \wedge \neg p_2' \wedge \neg q_2')) \vee (p_1' \wedge q_1' \wedge \neg p_2' \wedge \neg q_2'))) \end{aligned}$$

K = 2

Bounded Model Checking with SAT (BMC)



Does $\forall \Box \neg (p \wedge q)$

K = 2

$$M_k = (\neg p_0 \wedge \neg q_0) \wedge ((\neg p_0 \wedge \neg q_0 \wedge p_1 \wedge \neg q_1') \vee (\neg p_0 \wedge \neg q_0 \wedge \neg p_1' \wedge q_1'))$$

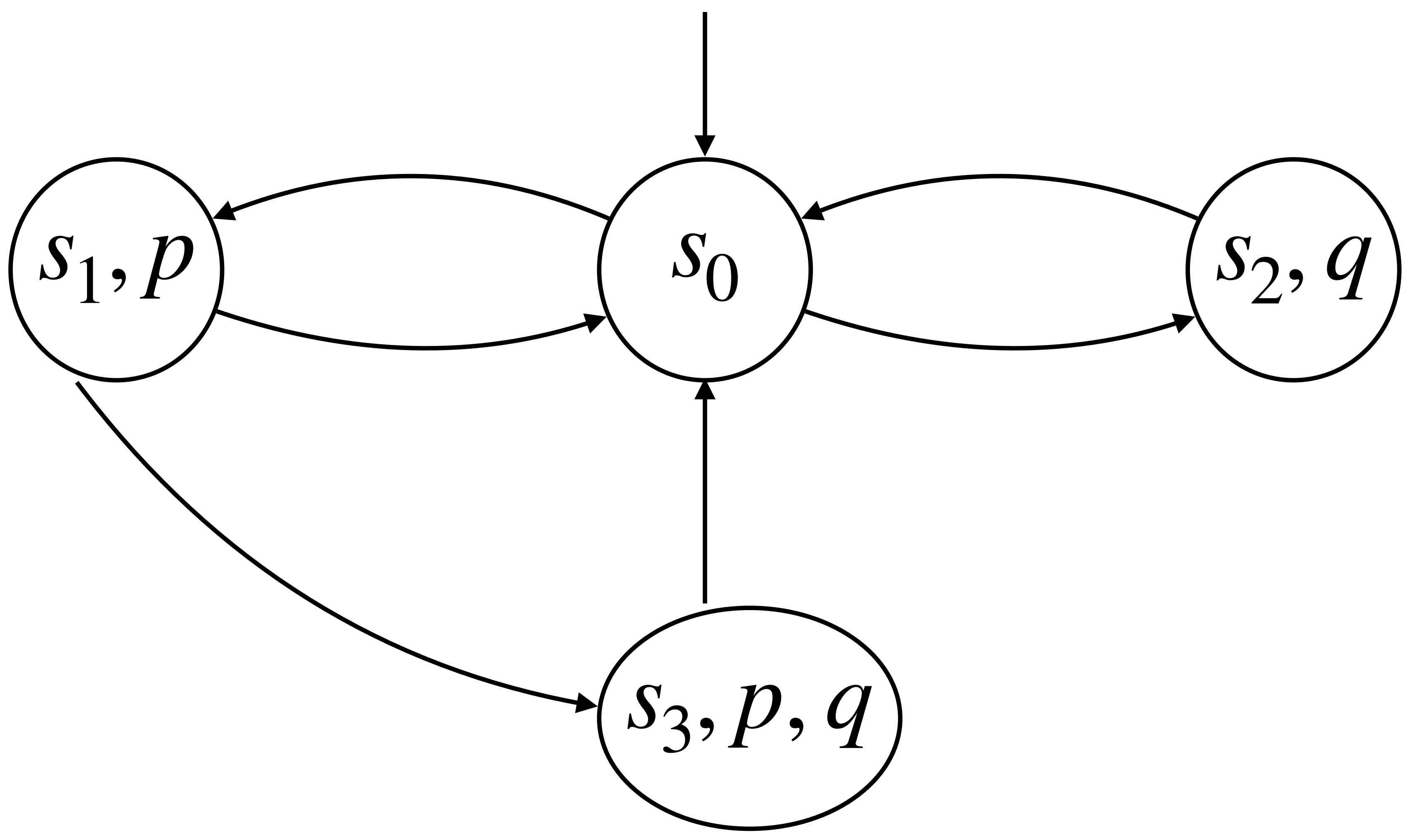
$$\wedge (((p_1 \wedge \neg q_1' \wedge p_2' \wedge q_2') \vee (p_1 \wedge \neg q_1' \wedge \neg p_2' \wedge \neg q_2')) \vee (p_1' \wedge q_1' \wedge \neg p_2' \wedge \neg q_2')))$$

K = 2

$\neg F = \exists \Diamond (p \wedge q)$ $\quad \neg F_k = p_2' \wedge q_2'$

$SAT\{M_k \wedge \neg F_k\}$

Bounded Model Checking with SAT (BMC)



Does $\forall \Box \neg(p \wedge q)$

K = 2

$$M_k = (\neg p_0 \wedge \neg q_0) \wedge ((\neg p_0 \wedge \neg q_0 \wedge p_1' \wedge \neg q_1') \vee (\neg p_0 \wedge \neg q_0 \wedge \neg p_1' \wedge q_1'))$$
$$\wedge (((p_1' \wedge \neg q_1' \wedge p_2' \wedge q_2') \vee (p_1' \wedge \neg q_1' \wedge \neg p_2' \wedge \neg q_2')) \vee (p_1' \wedge q_1' \wedge \neg p_2' \wedge \neg q_2')))$$

K = 2

$$\neg F_k = p_2' \wedge q_2'$$

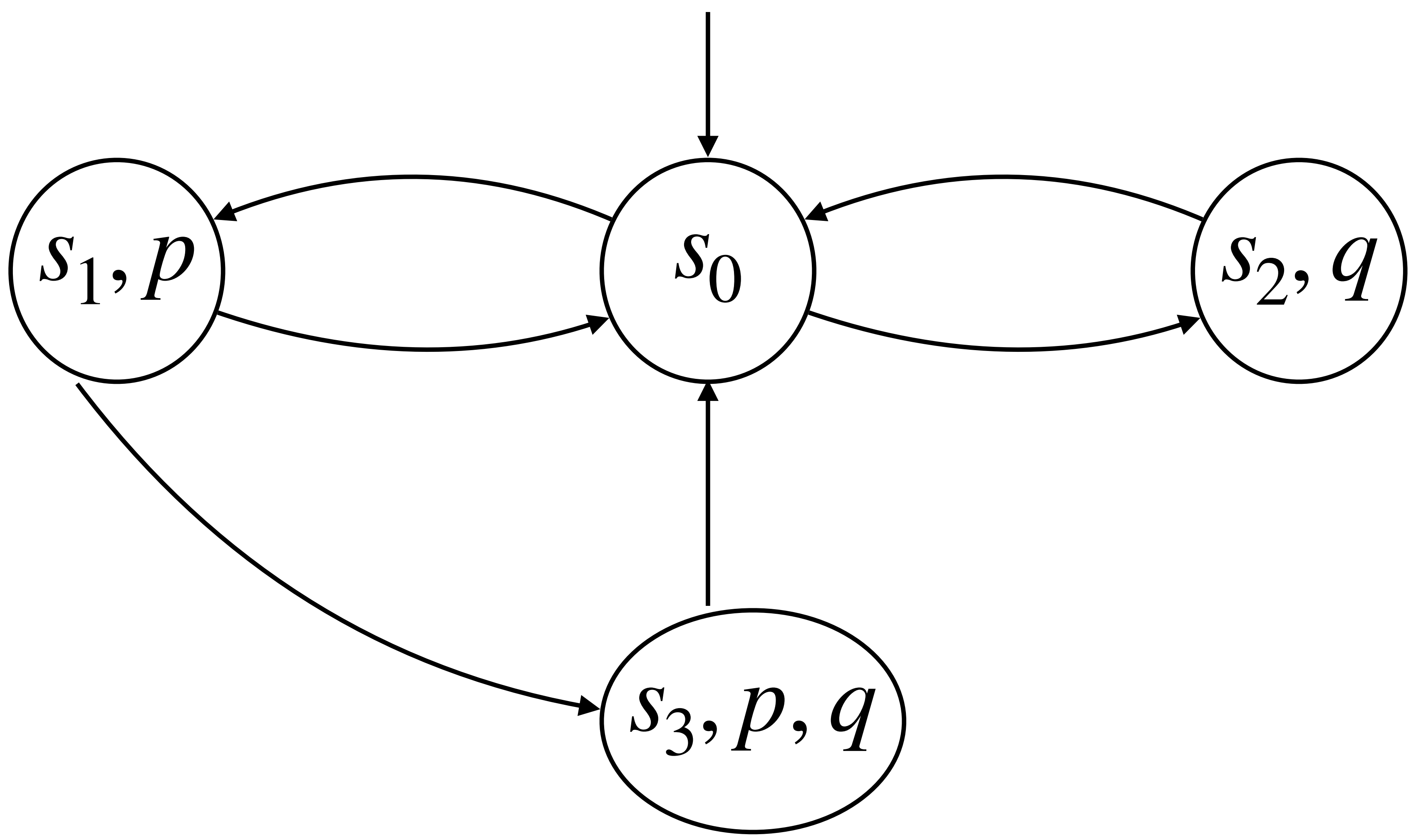
$SAT\{M_k \wedge \neg F_k\}$

$$\sigma = \langle p_0 = 0, q_0 = 0, p_1' = 1, q_1' = 0, p_2' = 1, q_2' = 1 \rangle$$

$M_k \nvDash F_k$

s_o, s_1, s_3

Bounded Model Checking with SAT (BMC)



Does $\forall \square \neg(p \wedge q)$

K = 1

$$M_k = (\neg p_0 \wedge \neg q_0) \wedge ((\neg p_0 \wedge \neg q_0 \wedge p_1 \wedge \neg q_1') \vee (\neg p_0 \wedge \neg q_0 \wedge \neg p_1' \wedge q_1'))$$

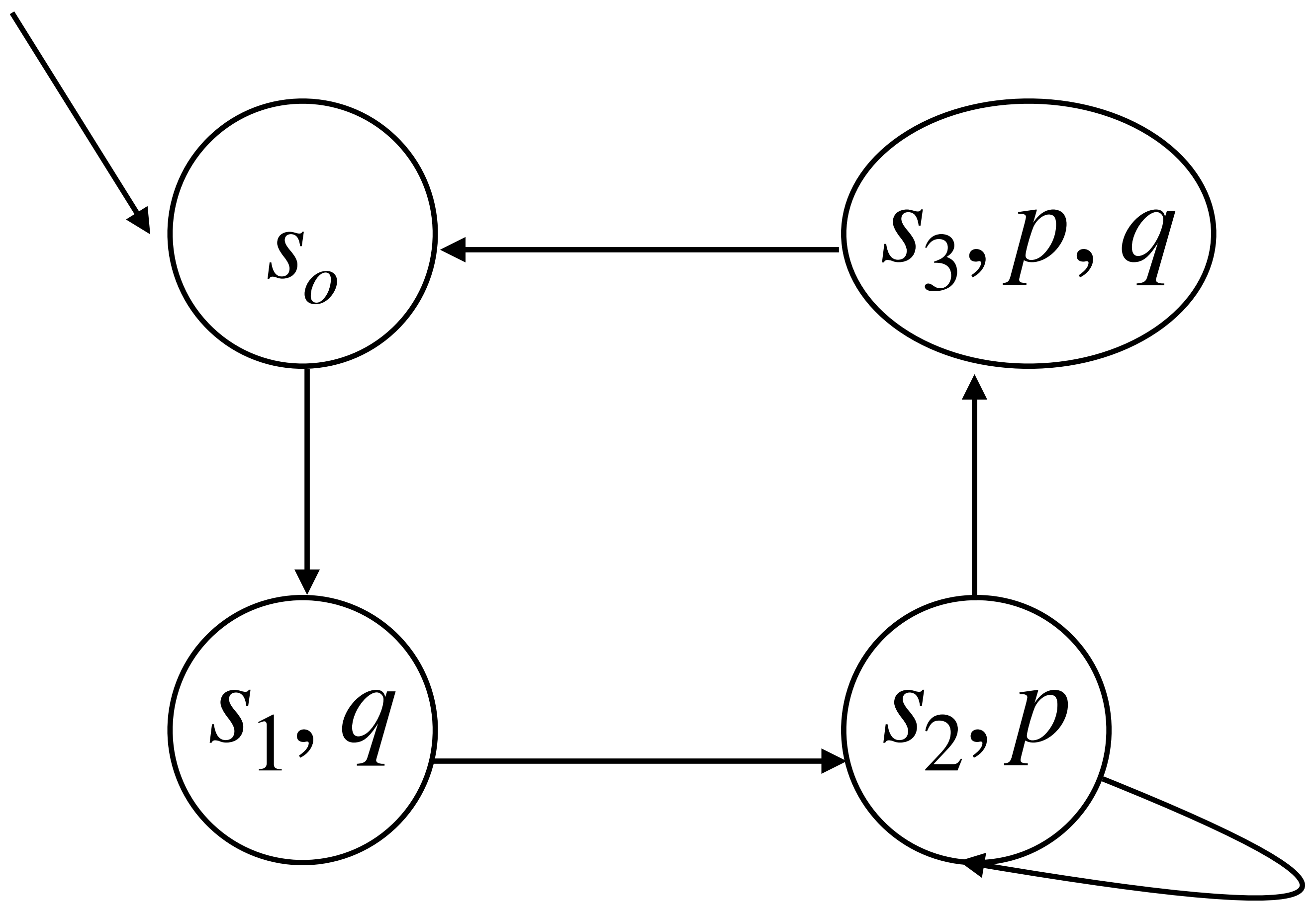
K = 1

$$\neg F_k = p_1' \wedge q_1'$$

$SAT\{M_k \wedge \neg F_k\}$ $\longrightarrow$ UNSAT, $M_{k=1} \vDash F_{k=1}$

Bounded Model Checking with SAT (BMC)

Two-bit counter

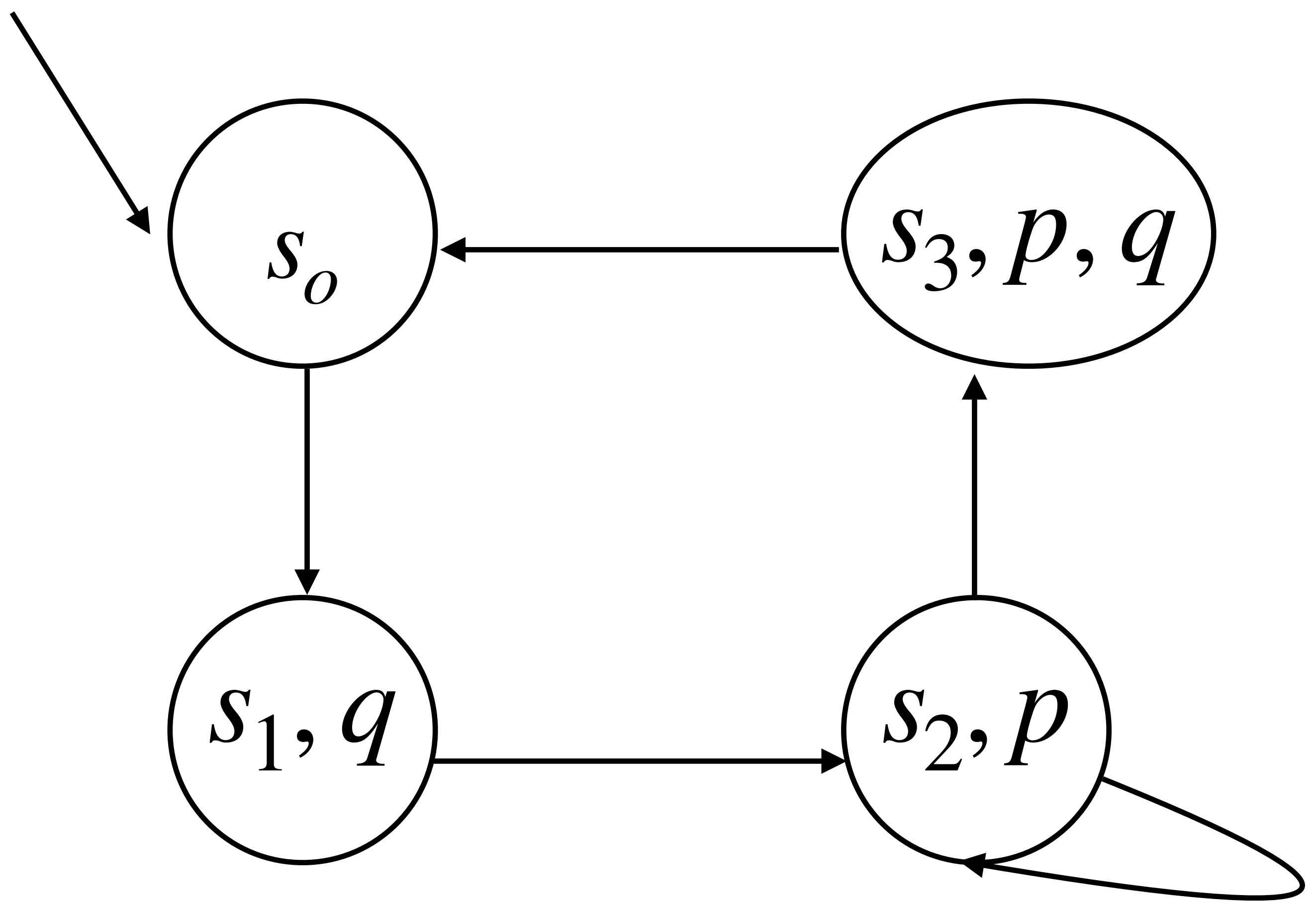


$$F = \forall \Diamond (p \wedge q) \quad \neg F = \exists \Box \neg p \vee \neg q$$

K = 3

Bounded Model Checking with SAT (BMC)

Two-bit counter



$$F = \forall \Diamond (p \wedge q) \quad \neg F = \exists \Box \neg p \vee \neg q$$

K = 3

$$M_k = (\neg p_o \wedge \neg q_o) \wedge (\neg p_o \wedge \neg q_o \wedge \neg p_1' \wedge q_1') \wedge (\neg p_1' \wedge q_1' \wedge p_2' \wedge \neg q_2') \;\; \wedge ((p_2' \wedge \neg q_2' \wedge p_3' \wedge \neg q_3') \vee (p_2' \wedge \neg q_2' \wedge p_3' \wedge q_3'))$$

$$\neg F_k = (\neg p_o \vee \neg q_o) \wedge (\neg p_1' \vee \neg q_1') \wedge (\neg p_2' \vee \neg q_2') \wedge (\neg p_3' \vee \neg q_3')$$

$M_k \wedge \neg F_k$ $\qquad$ $SAT\{M_k \wedge \neg F_k\}$

$$\sigma = \langle p_0 = 0, q_0 = 0, p_1' = 0, q_1' = 1, p_2' = 1, q_2' = 0, p_3' = 1, q_3' = 0 \rangle$$

$M_k \nvDash F_k$ $\qquad$ s_o, s_1, s_2, s_2