06 Oct, 2023:

⇒ LTL to GNBA Construction

LTL ϕ → Negation $\sim\phi$ → GNBA G_w → NBA A_w

$L_w(G) = \text{words}(\phi)$ $\quad L_w(A) = L_w(G)$

$M, s \models \phi$?

State m/c M → Compare → is any accepting run, then $M, s \not\models \phi$.

⇒ Vardi – Volpert theorem:

$\phi = a \, U \, (\sim a \wedge b)$

A run/trace that accept ϕ : $A_0, A_1, A_2, A_3, A_4 - - -$

$\{a\}\ \{a\}\ \{a,b\}\ \{b\}\ \emptyset - - - -$

$B_0\ B_1\ B_2\ B_3\ B_4 - - -$

words(ϕ) = { infinite trace that satisfies ϕ }

$L_w(G)$: Set of infinite words accepted by G.

words(ϕ) $\equiv L_w(G)$

GNBA: B_0 {a, $\sim b$, $\sim\psi$, ϕ}, B_1 {a, $\sim b$, $\sim\psi$, ϕ}, B_2 {a, b, $\sim\psi$, ϕ}, B_3 {$\sim a$, b, ψ, ϕ}

State labels in GNBA

$w \in L_w(G)$

$\phi = (a \wedge X a) \, U \, (a \wedge \sim X a)$ $\quad G = \{S, S_0, \Sigma, \delta, F\}$

⇒ ~~LTL~~: LTL($X, U, F, G \quad \wedge, \vee, \sim$) — Adequate set

LTL	GNBA
$\sim, \wedge, \vee$	in states
X	in the state transition
U	Via expansion rule + least fixed point.

$\phi_1 \, U \, \phi_2 \equiv \phi_2 \vee (\phi_1 \wedge X(\phi_1 \, U \, \phi_2))$

ϕ_2 → encoded in states; ϕ_1 → encoded in transition; $X(\phi_1 \, U \, \phi_2)$ → encoded in transition

Let ϕ is the LTL formula

Subf(ϕ) = {set of subformulas of ϕ}

$Cl(\phi)$ = Subf(ϕ) and their negations ∧

$Cl(\phi) = \{ a, Xa, (a \wedge Xa), (a \wedge \sim Xa), \phi, \sim a, \sim Xa, \sim(a \wedge Xa), \sim(a \wedge \sim Xa), \sim\phi \}$.

⇒ States : Certain subsets of $Cl(\phi)$ (i.e. elementary sets)

Possible subsets : $2^{Cl(\phi)}$

⇒ ~~Exponent~~ May have exponentially large GNBA.

Elementary set of $Cl(\phi)$:

$B \subseteq Cl(\phi)$ is elementary iff

① B is maximal consistent wrt propositional logic.

(i) $\psi \in B$, iff $\sim\psi \notin B$.

ii) $\psi_1 \wedge \psi_2 \in B$ iff $\psi_1 \in B$ and $\psi_2 \in B$.

② (iii) B is locally consistent w.r.t ∪

(i) if $\psi_1 \cup \psi_2 \in B$, and $\psi_2 \notin B$.

(ii) if $\psi_2 \in B$, then $\psi_1 \cup \psi_2 \in B$.

	a	Xa	ψ_1 $(a \wedge Xa)$	ψ_2 $(a \wedge \sim Xa)$	ϕ
B_0	0	0	0	0	0
B_1	0	1	0	0	0
→ B_2	1	0	0	1	1
→ B_3	1	1	1	0	0
→ B_4	1	1	1	0	1

ϕ is optional

Elementary set : $\{B_0, B_1, B_2, B_3, B_4\}$

Initial State : $S_0 = \{ B \in S \mid \phi \in B \}$

the set of elementary sets where ϕ holds

$$S_0 = \{ B_2, B_3 \}$$

Transition Relation :

Atomic proposition (AP) of ϕ : $\{a\}$.

For $B \in S$ and $A \in 2^{AP}$; $2^{AP} = \{ \{\theta\}, \{a\} \}$.

(i) $A \neq B \cap AP$, $\delta(B, A) = \theta$.

If A is not in the state level of B; there will be no transition ~~for~~ from B with A.

$\delta(B_0, \{a\}) = \theta$.

$\delta(B_0, \{\theta\}) =$ there will be transition.

ii) If $A \in B \cap AP$, then $\delta(B, A) =$ set of all $B' \in S$ s.t $B \xrightarrow{A} B'$

(a) $X_\psi \in B$, iff $\psi \in B'$.

(b) $\psi_1 \cup \psi_2 \in B$ iff $(\psi_2 \in B) \vee (\psi_1 \in B \wedge \psi_1 \cup \psi_2 \in B')$

~~$\delta(B_3, \{a\}) =$~~ ~~Since $X_a \subset B_3$~~

$\delta(B_3, \{a\}) = \{ B_2, B_3, B_4 \}$ Since $X_a \in B_3$

$\phi \in B_3$ iff $(\psi_2 \in B_3)$ $\vee$ $(\psi_1 \in B_3 \wedge \phi \in B')$

false. true.

consider only states where ϕ holds $\Downarrow$ $\{B_2, B_3\}$

$\delta(B_3, \{a\}) = \{ B_2, B_3 \}$ (Should satisfy both rules).

$\Rightarrow$ $\delta(B_0, \{a\}) = \theta$.

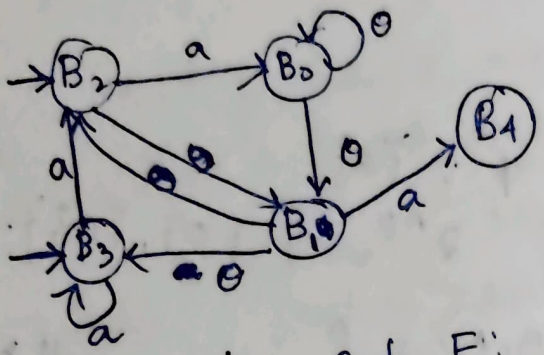
$\delta(B_1, \{a\}) = \theta$.

$\delta(B_0, \{\theta\}) = \{ B_0, B_1 \}$.

$\rightarrow \delta(B_2, \{a\}) = \{ B_0, B_1 \}$

Property : $\sim X_\psi \in B$ iff $\psi \notin B'$

$\phi \in B_2$ iff $(\underbrace{\psi_2 \in B_2}_{\text{holds}}) \vee (\underbrace{\dots}_{\text{not sup}})$



→ Accepting set F:

one accept state for each U in the formula.

$\therefore k = 1$;

∴ Mark all the states where U does not holds.

Acceptance set $F = \{ F_{\psi_1 U \psi_2} \mid \psi_1 U \psi_2 \in cl(\phi) \}$

where $F_{\psi_1 U \psi_2} = \{ B \in S \mid \psi_1 U \psi_2 \notin B \vee \psi_2 \in B \}$

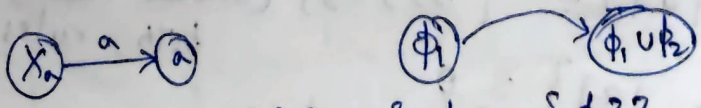
10 Oct, 2023:

LTL Model Checking Algorithm:

LTA formula to GNBA construction,

$$\phi = \underbrace{(a \wedge X a)}_{\phi_1} U \underbrace{(a \wedge \neg X a)}_{\phi_2}$$

$2^{|cl(\phi)|}$: # states in GNBA.



$2^{|AP|}$: { {a}, {b}, {a,b}, {∅} }

→ Acceptance Set:

For each 'U' operator, we will have one set of final states.

P U Q : one set ; (P U (Q U P)) : two sets, of final states.

$F_\phi = \{B_0, B_1, B_2, B_4\}$

$F_{\phi_1 U \phi_2} = \{B \in S \mid \phi_1 U \phi_2 \notin B \vee \phi_2 \in B\}$

→ $P U (q \cup r)$, ϕ_1

$q \cup r : \{B_1, B_2, B_3\}$

$P U \phi_1 : \{B_2, B_4, B_6\}$

$B_0\ B_1 B_9 B_2 \cdot B_1 B_9 B_2 - - -$

⇒ $M, s \models \phi$? $\sim\phi \rightarrow$ GNBA G_w s.t word $(\phi) \equiv L(G_{NA})$

NBA A_w

$M \otimes A_w$

If there is any accepting run, then $M, s \not\models \phi$

→ Product of M on A_w:

compose: $s_1 \xrightarrow{a} s_4$, $q_1 \xrightarrow{a} q_2$ ⇒ $s_1 q_1 \xrightarrow{a} s_4 q_2$

→ M_1: 1, 2 ; $M_1 \otimes M_2$ ⇒

M_2: a, b, c

Product automata: 1a, 2b, 1c, 2a, 1b, 2c

In general if $M_1, M_2, -- M_k$ with $n_1, n_2, -- n_k$ states, then product machine will have $n_1 \times n_2 \times -- \times n_k$ states.

⇒ State explosion problem.

In GNBA # states: $2^{|cl(\phi)|}$

Model M:

q_1 $\bar{a}\bar{b}$ → $\bar{a}b$ q_2

$a\bar{b}$ → ab q_4

$a\bar{b}$ q_3

ϕ : $\sim(a \cup b)$, $M, (q_3) \not\models \phi$?

start state

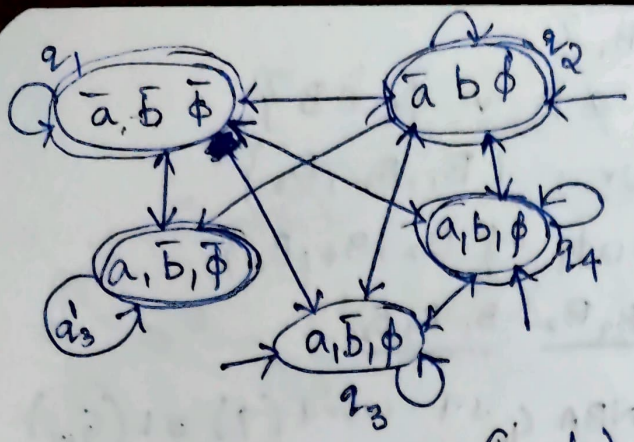
No

$q_3, q_2, q_2, - - -$ satisfies $(a \cup b)$

$\sim\phi$: $a \cup b$

$GNBA(\sim\phi) \equiv GNBA(a \cup b) \equiv NBA(a \cup b)$.

~~$\bar{a}\bar{b}\phi$~~



Not accepting run:

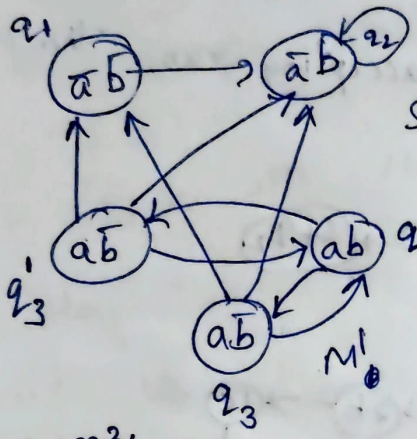
$q_3\ q_3\ q_3 - - -$

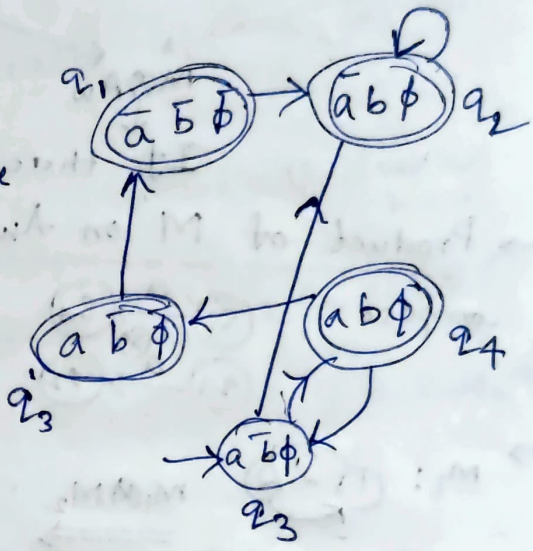
$a\bar{b}\ \ a\bar{b}\ \ a\bar{b}$

$q_3\ q_2\ q_2\ q_1 - - -$

$a\bar{b}\ \ \bar{a}b\ \ \bar{a}b\ \ \bar{a}b$

→ Accepting run





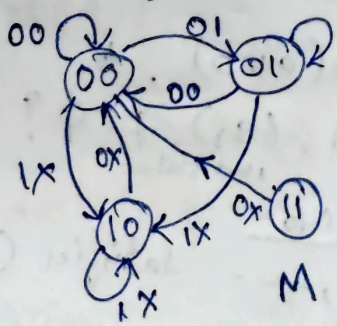
11 Oct, 2023.

⇒ Reduction of state explosion Problem in Model checking:

1. Cone-of-influence Reduction.

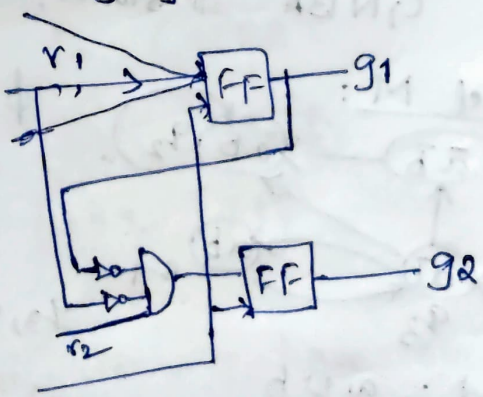
- Remove components of the design that donot influence the truth value of the given property.

$$G[r_1 \rightarrow Xg_1 \wedge XXg_1]$$

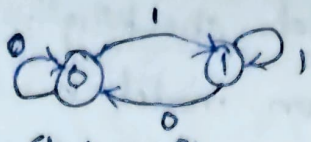


State : $g_1 g_2$

Input : $r_1 r_2$.



col of r1



State: g1

Input: r1

2. Pruning using assume constraints:

- Property specification language (SVA/PSL)
  - Assert (to describe property)
  - Assume (to describe any assumption under which the property need to be checked.)
- Assume, atleast one of the inputs is always true.

⇒ Remove all transitions with '00'.

3. Simulation Equivalence Relation:

- State machine can be reduced based on the property to be checked.

P: F(~g1 ∧ ~g2)

- Separate the states into two sets
  (i) where P satisfies and,
  (ii) where P does not satisfy.

⇒ If s1 is reachable from s2.

3a) Stuttering Equivalence ~~Propert~~ preserves:

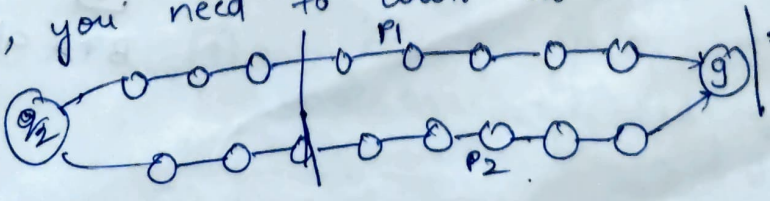
The truth value of untimed property.

F_g : # states from the start state to the states where g holds can be considered as one state.

3b) Bisimulation Equivalence preserves the truth value of timed property

$F_{[3,9]g}$: Btw 3rd clk to 9th clk 'g' must hold from current state/clk.

⇒ Here, you need to count the states.



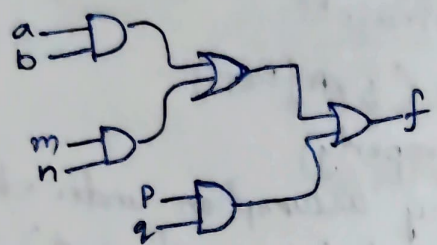
⇒ the states in P1 and P2 can be merged.

⇒ Symbolic Model checking :

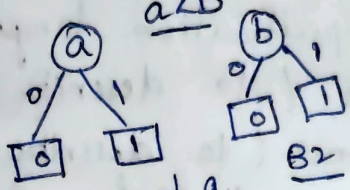
- State transition model is represented as ROBDD.

ROBDD from combinational ckt:

$f : (ab \vee mn) \vee pq$



a<b

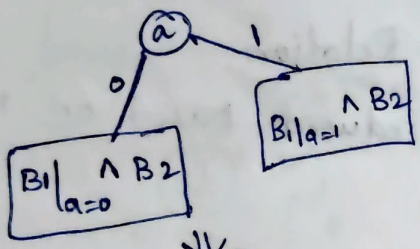


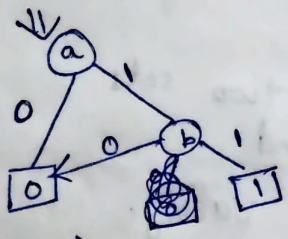
B1 : BDD of a

B2 BDD of b.

B1 AND B2

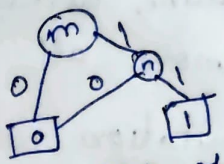
(same root node, take 'a' since high order)





B3 : ab

m<n.

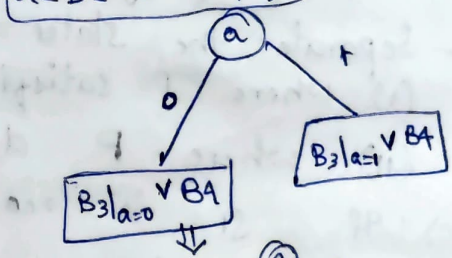


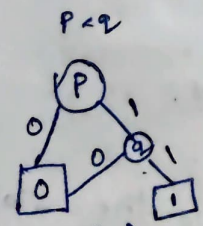
B4 : BDD of mn

B5: BDD of (ab v mn)

B3 v B4

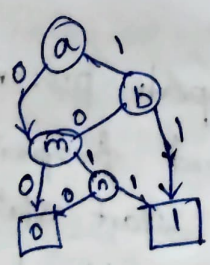
a<b<m<n<p<q



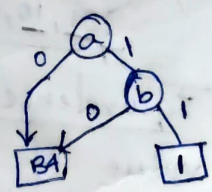


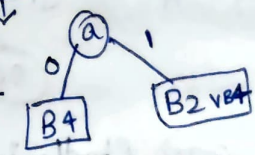
p<q

B6 : BDD of pq

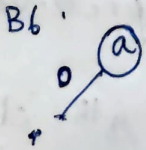


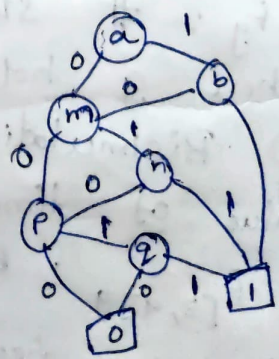
B5 : ab v mn





f : B5 v B6



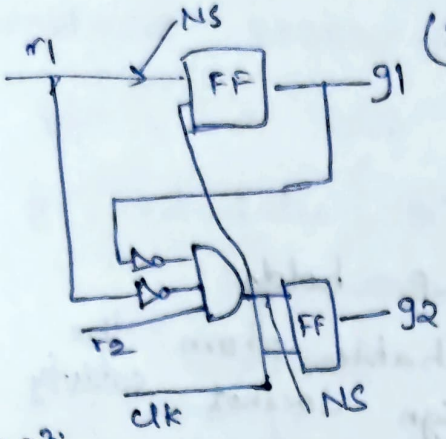


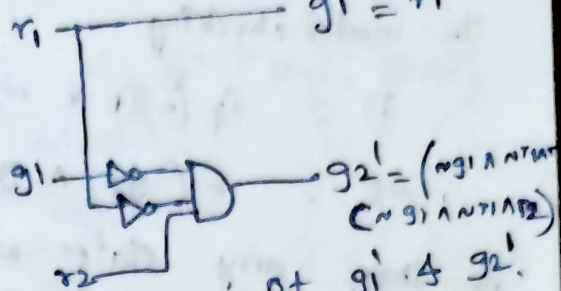
B7 : (ab v mn) v pq

ROBDD of sequential Circuit:

Next state = f(present state, Inputs)

$(g_1', g_2') = f(g_1 g_2, r_1 r_2)$.





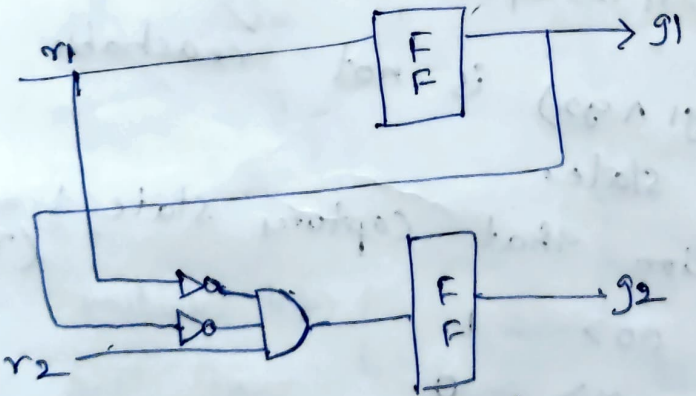
Construct ROBDD of g_1' & g_2'.

16 Oct, 23.

→ Symbolic Model checking - using BDD

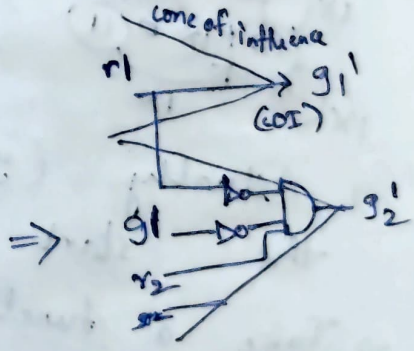
To model a seq Design using BDD.

- Drop the seq elements to extract the combinational logic.
- For each Flip Flop,
  - Treat its o/p as primary input (y).
  - Treat its input as primary o/p (y')
- Build BDD.

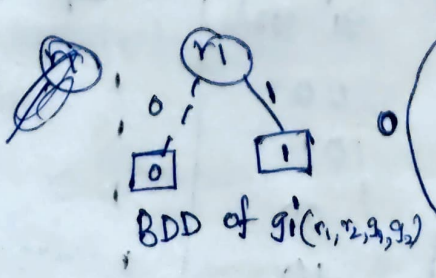


Present state
$< g_1 \; g_2 >$

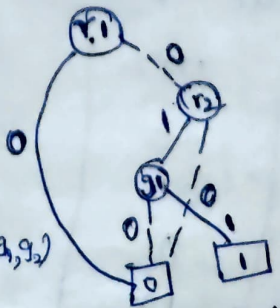
Inputs
$< r_1 \; r_2 >$



next state
$< g_1' \; g_2' >$



BDD of $g_1'(r_1, r_2, g_1, g_2)$



BDD of g_2'.

What will be next state bit g_1', g_2' based on present state (g_1, g_2) and inputs (r_1, r_2)

In model checking

$P : G(\sim g_1 \vee \sim g_2)$

$\sim P : F(g_1 \wedge g_2)$

Is there any state where $\sim P$ holds ?

⇒ If such state is reachable from the initial state then the design doesnot satisfy P.

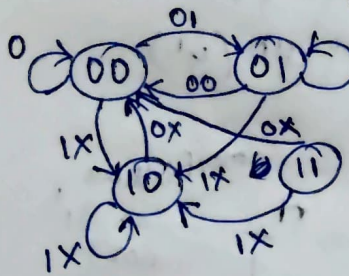
⇒ reachability analysis.

⇒ $S_0 \rightarrow S_1 \rightarrow S_2 - - - \rightarrow S_k$ ($\sim P$ holds.)

In B1 : when $r_1 = 1 \rightarrow g_1' = 1$.

In B2 : when $r_1 = 0$, $r_2 = 1$, $g_1 = 1 \Rightarrow g_2' = 1$

This doesnot prove that $g_1 \wedge g_2$ is reachable combinely.



<g1 g2> state
<r1 r2> inputs

⇒ Actually $(g_1 \wedge g_2)$ is not reachable from the start state.

⇒ Take a function that captures state transitions (T)

$\langle 00 \quad 00 \quad 00 \rangle = 1$ } valid transitions.

$\langle 00 \quad 00 \quad 01 \rangle = 0$.

Characteristic function table

$g_1 g_2$	$r_1 r_2$	$g_1' g_2'$	T
XX	00	00	1
0X	01	01	1
1X	01	00	1
XX	1X	10	1
	for all other values		0

$T = r_1' r_2' g_1' g_2' + g_1 r_1' r_2 g_1' g_2 + \dots + \dots$

SOP.

Construct ROBDD of the SOP as T.

⇒ B1 is a BDD of COI of g_1' (logic cone)

B1 calculates g_1' (next value)

logic cone ≡ next value.

XNOR

0	0	1
0	1	0
1	0	0
1	1	1

B3: BDD of g_1'.

B3: BDD of g_2'.

⇒ B1 XNOR B3 = B5

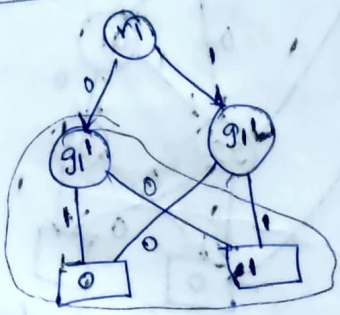
B2 XNOR B4 = B6

B5 AND B6 = B7 (T)

B7 is the BDD of T = (g_1 g_2 r_1 r_2 g_1' g_2')

0 X 0 1 0 1

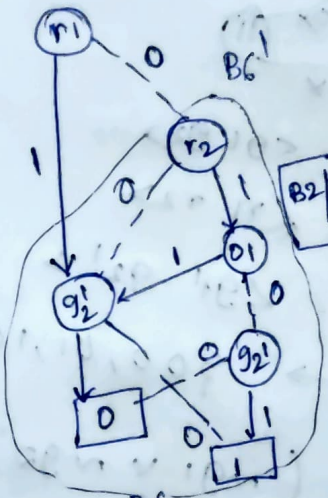
⇒ B1 XNOR B3



$B5'$

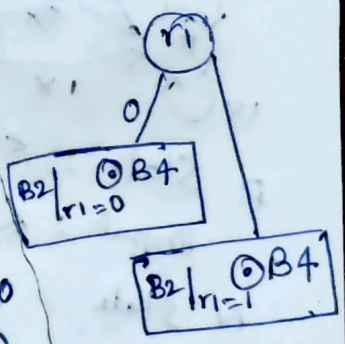
B5

B2 XNOR B4

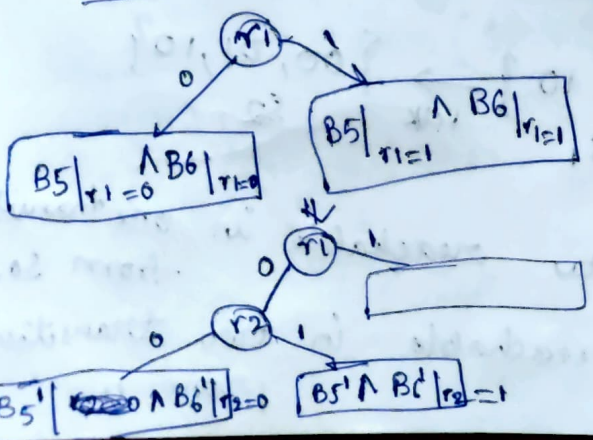


B6

B2 ⊙ B4

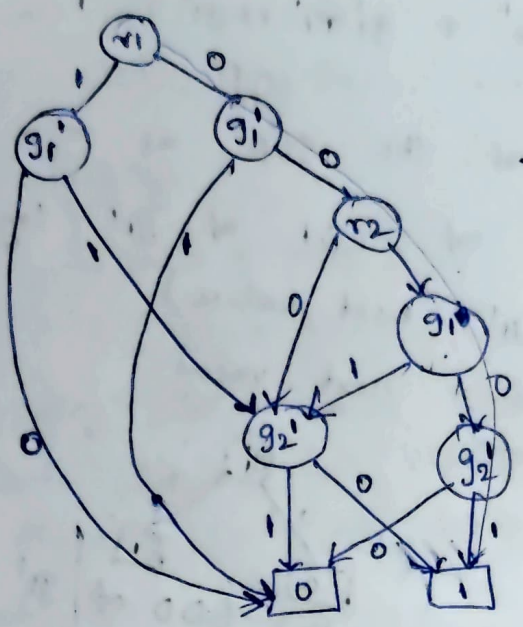


B5 AND B6:



$B5|_{r_1=0}$

Final:

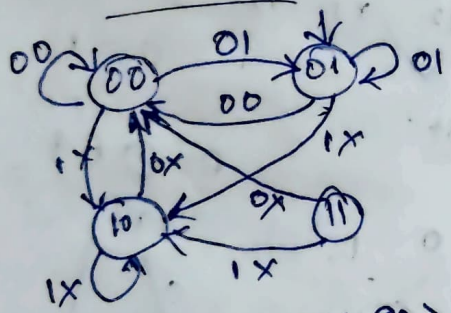


*All paths from root to the node 1 are the valid transitions

B 7

17 Oct, 2023:

Symbolic Reachability:

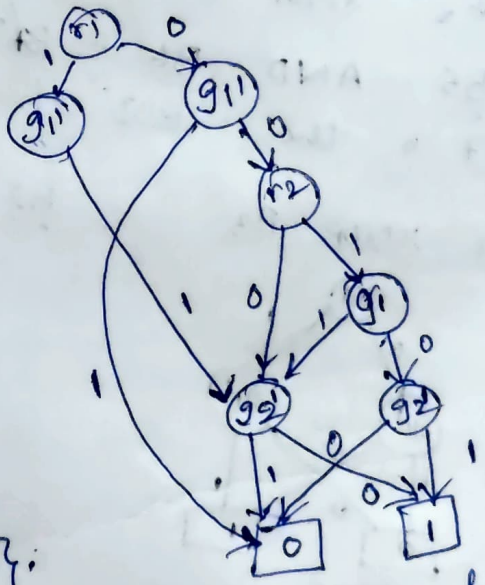


P State : <g1 g2>

Input : <r1 r2>

N State : <g1' g2'>

01 → {00, 01, 10}.



BDD of R.

⇒ P : G (¬g1 ∨ ¬g2)

⇓

¬P : F (g1 ∧ g2)

$\{\overset{g1\,g2}{01}\}_{S_0} \xrightarrow{1\,clk} \{00, 01, 10\}_{S_1} \xrightarrow{1\,clk} \{00, 01, 10\}_{S_2}$

S_0: initial state

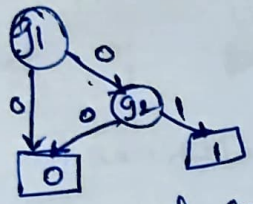
S_1: set of states reachable in one transition from S_0.

S_2: set of states reachable in two transition from S_0.

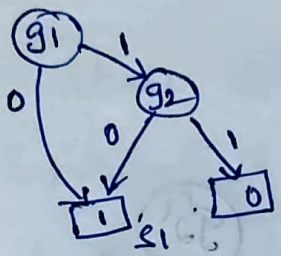
$S1 = S2$ (fixed point)

↳ $\sim P$ is not true in M

$\Rightarrow$ P is true in M



BDD of S_0.

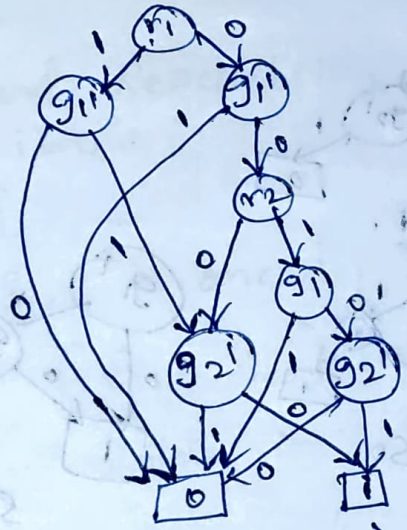


S_1

~~want~~ $S1, S2$ want to obtain S_1 from S_0 and R.

$\Rightarrow$ AND(S_0, R) $\equiv$ set of next state corresponding to S_0 (present state)

$\Rightarrow$ AND(S_0, R):

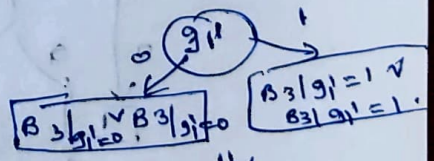


B_1 AND (S_0, R).

$\Rightarrow \exists_x B(x, y, z) \equiv B|_{x=0} \vee B|_{x=1}$

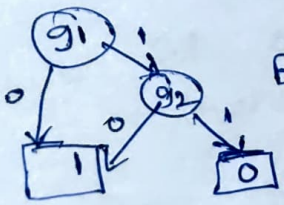
Elimination of variables from a BDD.

$\Rightarrow \exists r_1 \; \exists r_2 \; \exists g_1 \; \exists g_2 \; B_1$



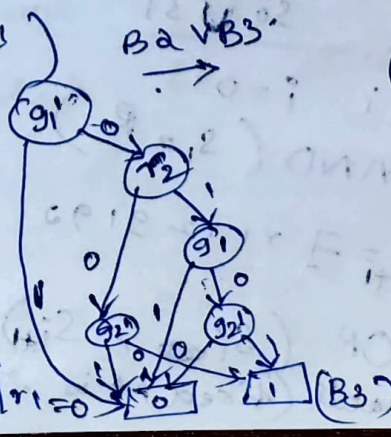
$S_1 = BDD(g_1', g_2')$

$B_2 \vee B_3$

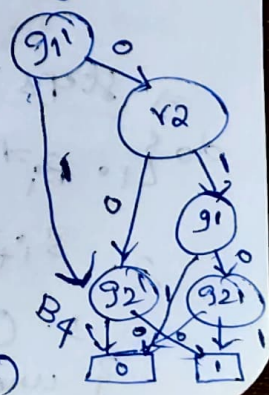


S_1

$B_1|_{r_1=1}$



$B_1|_{r_1=0}$ (B3)



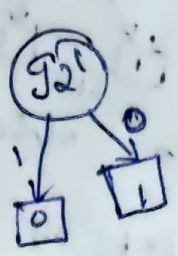
B_4

B_2

$\exists r_2$ ⟹

⟹ $B4|_{r_2=0}$

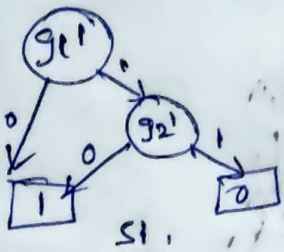
⟹



B5

⟹ $B4|_{r_2=1} = B6.$

$B6 \vee B5 \Rightarrow$



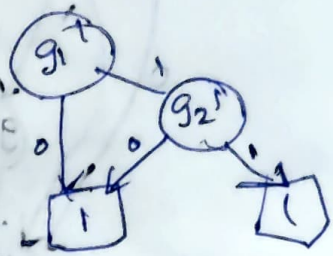
S1.

⟹ $OR(S0, S1)$

↓

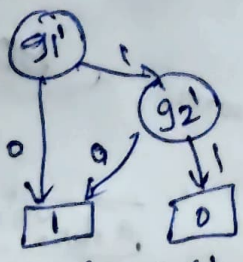
Set of states reachable in 0 or 1 cycles.

$S0_{in}$ i.



S1

$S0 \vee S1$



$S0 \vee S1$

⟹ steps i $i=0$

do { 1. $B_i = AND(S_i, R)$

2. $S_{i+1} = \exists r_1, r_2 \; g_1 g_2 \; B_i$

3. $OR(S_i, S_{i+1})$

} until (fixed pt is reached or a state with NP found).

18 Oct, 2023.

Reachability Analysis with BDD

Forward Reachability:

Set of present state S_i (BDD) (g_1, g_2).

Transition System R (BDD). $(\underbrace{g_1 g_2}_{\text{Present state}} \underbrace{r_1 r_2}_{\text{Input}} \underbrace{g_1' g_2'}_{\text{next state}})$

One state Fwd Reachability(S_i, R) {

$T\_NS_{i+1} = AND(S_i, R)$,

$NS_{i+1} = \exists r_1 \exists r_2 \exists g_1 \exists g_2 (T\_NS_{i+1})$

return NS_{i+1}

}.

⇒ Forward Reachability (S_0, R, P) // Check if P hold in S_0 in R.

{

$i = 0$.

do {

S_{i+1} = one_step_Fwd_Reachability (S_i, R).

if $(S_i = S_{i+1})$ ∿P satisfies in R; return.

if (∿P holds in S_{i+1}) then P does not hold in R; return;

i++;

}

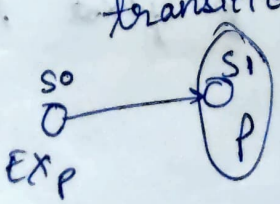
}

⇒ Backward Reachability:

EX_P

S_1 = Set of states where P holds.

S_0 = Set of states which has atleast one transition to S_1.



S_i = BDD of next state (g_1', g_2')

One_state_backward_Reachability (S_i, R)

{ ~~AND (S_i, R)~~

T_PS_{i+1} = AND (S_i, R);

(temporary)

$PS_{i+1} = \exists r_1 \exists r_2 \exists g_1' \exists g_2' \; T\_PS_{i+1}$;

return PS_{i+1};

}

→ Symbolic CTL model checking:

(A, E) (X, F, U, G) ($\wedge$, $\vee$, $\sim$, $\rightarrow$)

Adequate set (EX, EU, $\wedge$, $\sim$)

f: Boolean formula with state variables.

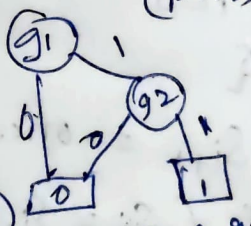
- construct a BDD of 'f'

- Set of states where f holds

$f \cup g$

$(p \wedge q)$ → $(g_1 \wedge g_2)$

$\sim f$: Swap the leaf node values.



$f = g_1 \wedge g_2$

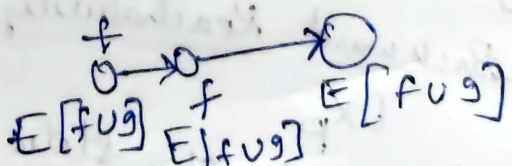
EX_f : 1. Construct the BDD of f (B_f)

2. One step Backward Reachability (B_f, R).

→ EU $E[f \cup g]$

1. Construct BDD B_f, B_g

$E[f \cup g] = g \vee [f \wedge EX[E[f \cup g]]]$



2. S_k = One_step_Backward_Pred (B_g, R).

3. PS = AND (S_k, B_f)

4. Repeat step 2, 3 until a fixed point is reached.

Symbolic LTL model checking:

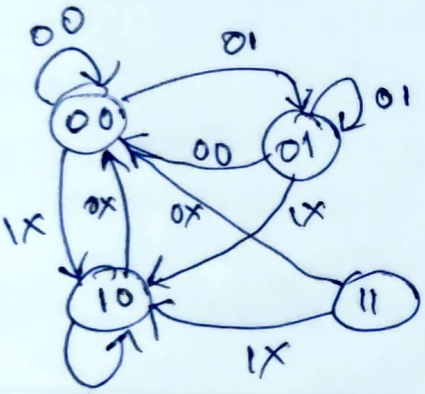
1. Construct NBA A_ϕ ⇒ Represent A_ϕ axis BDD
2. Compose A_ϕ with R (transition system).
3. Is there any accepting path in the product machine.

EG (True).

δ : Boolean exp over state variables.

what if a boolean expression over state variables and inputs (g_1 g_2, r_1 r_2).

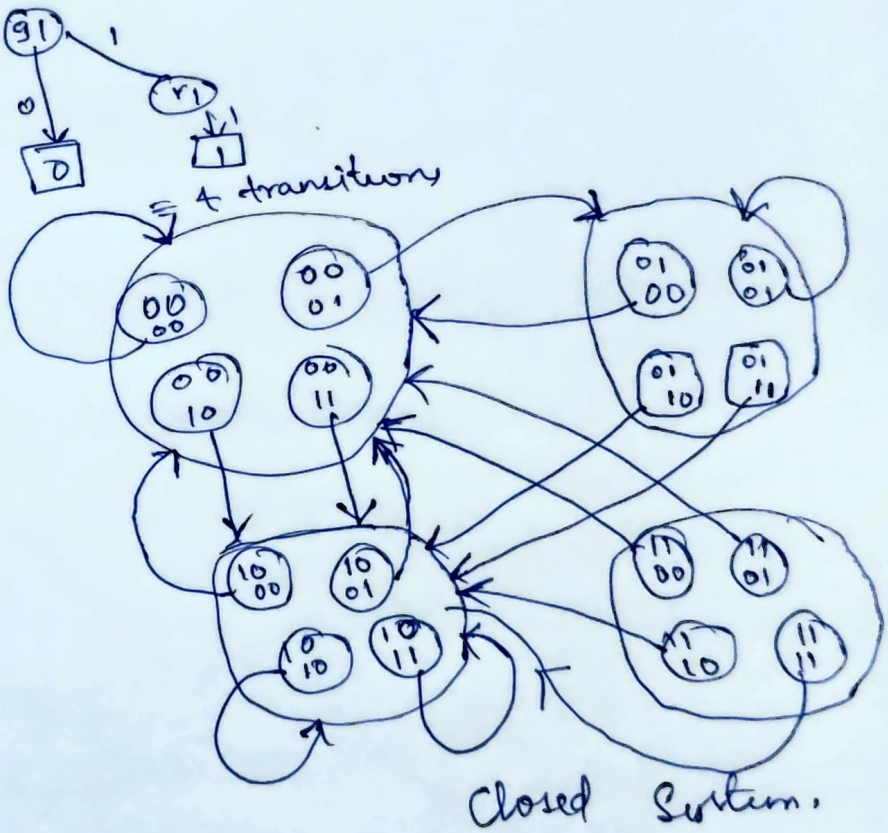
$EX_{(g_1 \wedge r_1)}$



open system.

State : g_1 g_2

input: r_1 r_2



Closed System.