CS344: Introduction to Artificial Intelligence

Pushpak Bhattacharyya
CSE Dept.,
IIT Bombay

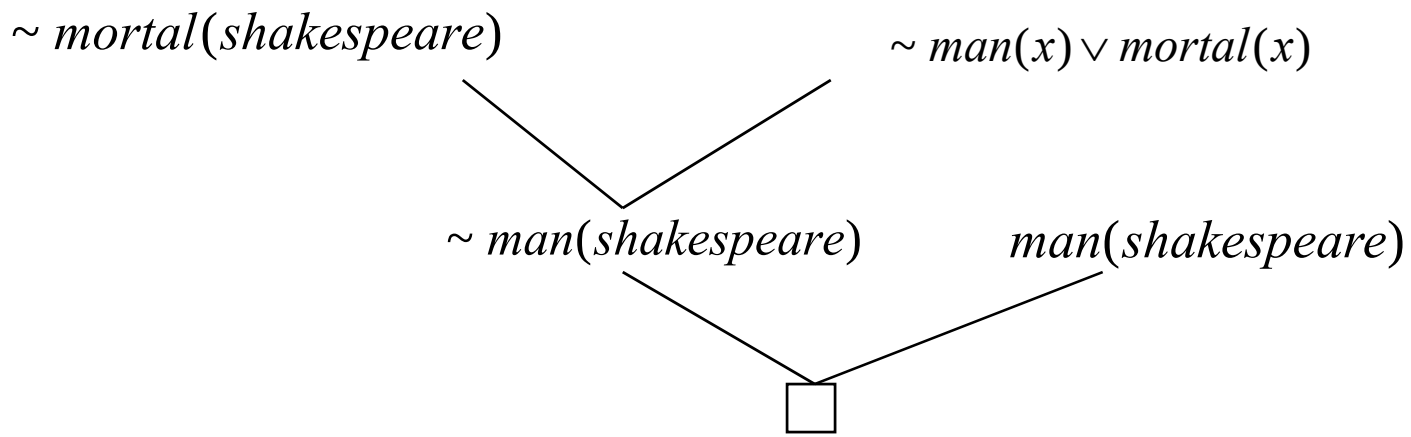
Lecture 10– Club and Circuit Examples

Resolution - Refutation

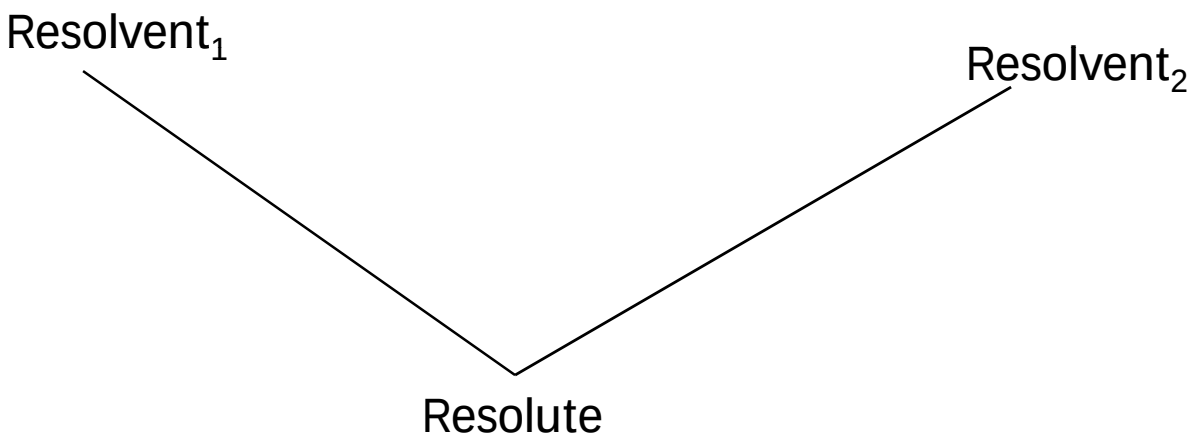
- $man(x) \rightarrow mortal(x)$
  - *Convert to clausal form*
  - $\sim man(x) \quad mortal(x)$
- Clauses in the knowledge base
  - $\sim man(x) \vee mortal(x)$
  - $man(shakespeare)$
  - $mortal(shakespeare)$

Resolution – Refutation contd

- *Negate the goal*
  - *~mortal(shakespeare)*
- Get a pair of resolvents



Resolution Tree



Search in resolution

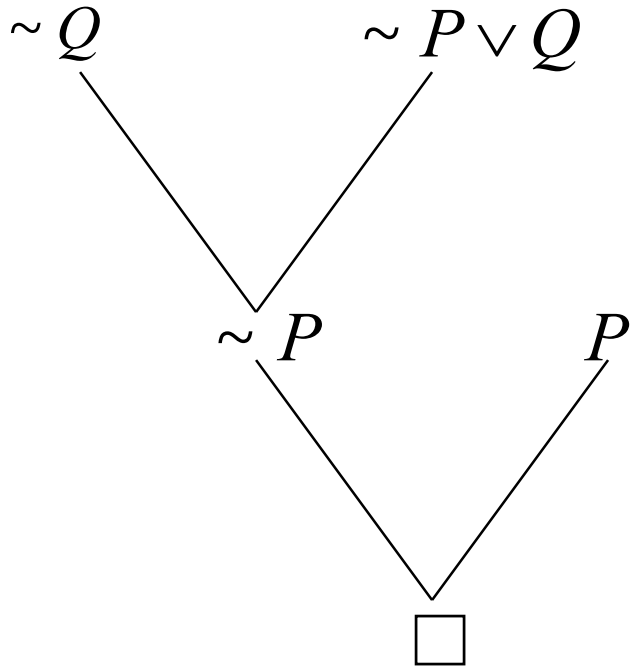
- Heuristics for Resolution Search
  - Goal Supported Strategy
    - Always start with the negated goal
  - Set of support strategy
    - Always one of the resolvents is the most recently produced resolute

Inferencing in Predicate Calculus

- Forward chaining
  - Given P, $P \rightarrow Q$, to infer Q
  - P, match $L.H.S$ of
  - Assert Q from $R.H.S$
- Backward chaining
  - Q, Match $R.H.S$ of $P \rightarrow Q$
  - assert P
  - Check if P exists
- Resolution – Refutation
  - Negate goal
  - Convert all pieces of knowledge into clausal form (disjunction of literals)
  - See if contradiction indicated by null clause $\square$ can be derived

1. P
2. $P \rightarrow Q$ converted to $\sim P \vee Q$
3. $\sim Q$

Draw the resolution tree (actually an inverted tree). Every node is a clausal form and branches are intermediate inference steps.



Terminology

- Pair of clauses being resolved is called the Resolvents. The resulting clause is called the Resolute.
- Choosing the correct pair of resolvents is a matter of search.

Himalayan Club example

- Introduction through an example *(Zohar Manna, 1974):*
  - Problem: A, B and C belong to the Himalayan club. Every member in the club is either a mountain climber or a skier or both. A likes whatever B dislikes and dislikes whatever B likes. A likes rain and snow. No mountain climber likes rain. Every skier likes snow. *Is there a member who is a mountain climber and not a skier?*
- Given knowledge has:
  - Facts
  - Rules

Example contd.

- Let *mc* denote mountain climber and *sk* denotes skier. Knowledge representation in the given problem is as follows:
  1. $member(A)$
  2. $member(B)$
  3. $member(C)$
  4. $\forall x[member(x) \rightarrow (mc(x) \vee sk(x))]$
  5. $\forall x[mc(x) \rightarrow \sim like(x,rain)]$
  6. $\forall x[sk(x) \rightarrow like(x, snow)]$
  7. $\forall x[like(B, x) \rightarrow \sim like(A, x)]$
  8. $\forall x[\sim like(B, x) \rightarrow like(A, x)]$
  9. $like(A, rain)$
  10. $like(A, snow)$
  11. $Question: \exists x[member(x) \wedge mc(x) \wedge \sim sk(x)]$
- We have to infer the 11th expression from the given 10.
- Done through Resolution Refutation.

Club example: Inferencing

1. $member(A)$
2. $member(B)$
3. $member(C)$
4. $\forall x[member(x) \rightarrow (mc(x) \vee sk(x))]$
   - Can be written as
   - $[member(x) \rightarrow (mc(x) \vee sk(x))]$
   - $\sim member(x) \vee mc(x) \vee sk(x)$
5. $\forall x[sk(x) \rightarrow lk(x, snow)]$
   - $\sim sk(x) \vee lk(x, snow)$
6. $\forall x[mc(x) \rightarrow \sim lk(x, rain)]$
   - $\sim mc(x) \vee \sim lk(x, rain)$
7. $\forall x[like(A, x) \rightarrow \sim lk(B, x)]$
   - $\sim like(A, x) \vee \sim lk(B, x)$

8. $\forall x[\sim lk(A,x) \rightarrow lk(B,x)]$

– $lk(A,x) \vee lk(B,x)$

9. $lk(A,rain)$

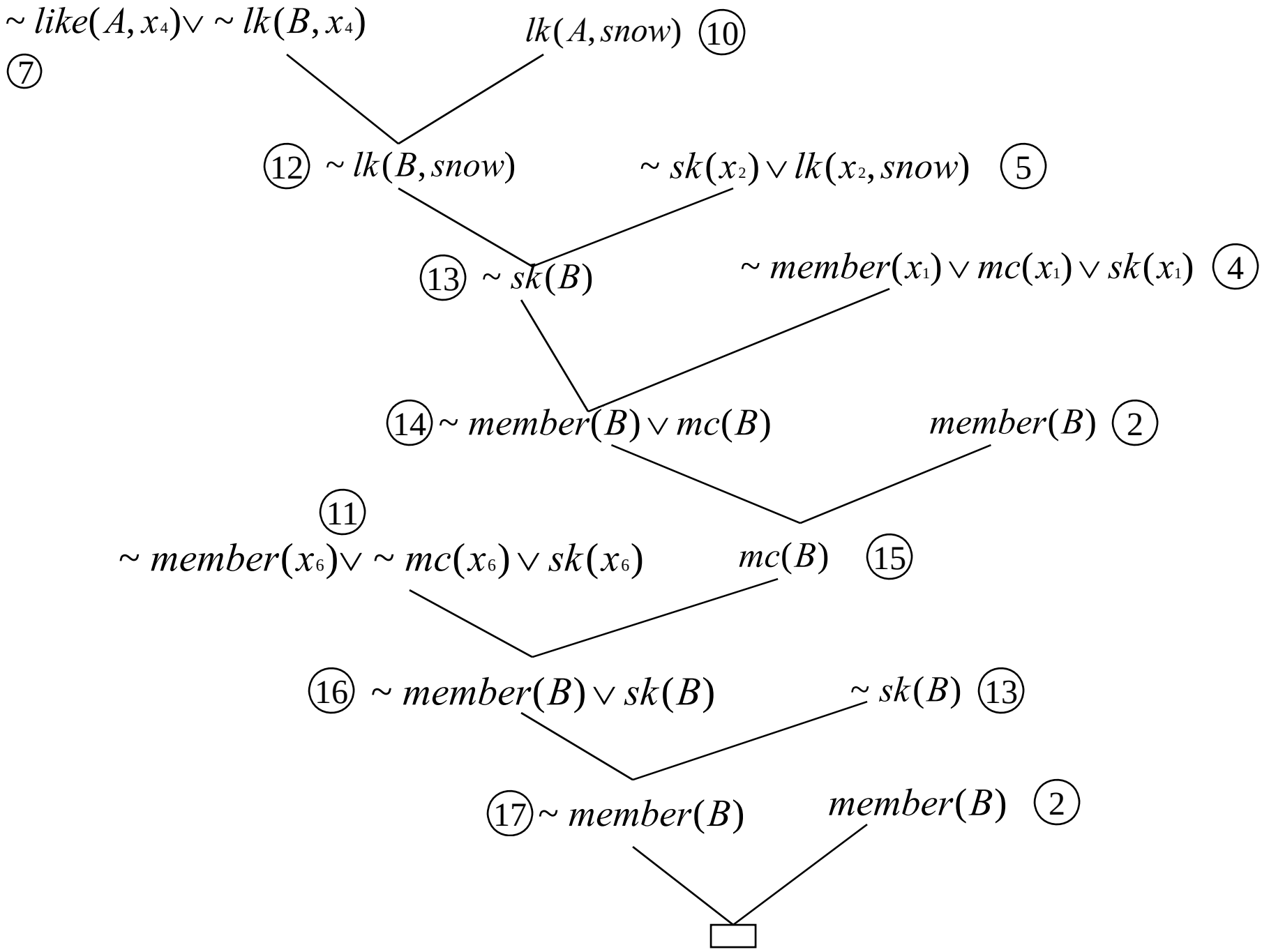
10. $lk(A,snow)$

11. $\exists x[member(x) \wedge mc(x) \wedge \sim sk(x)]$

– Negate– $\forall x[\sim member(x) \vee \sim mc(x) \vee sk(x)]$

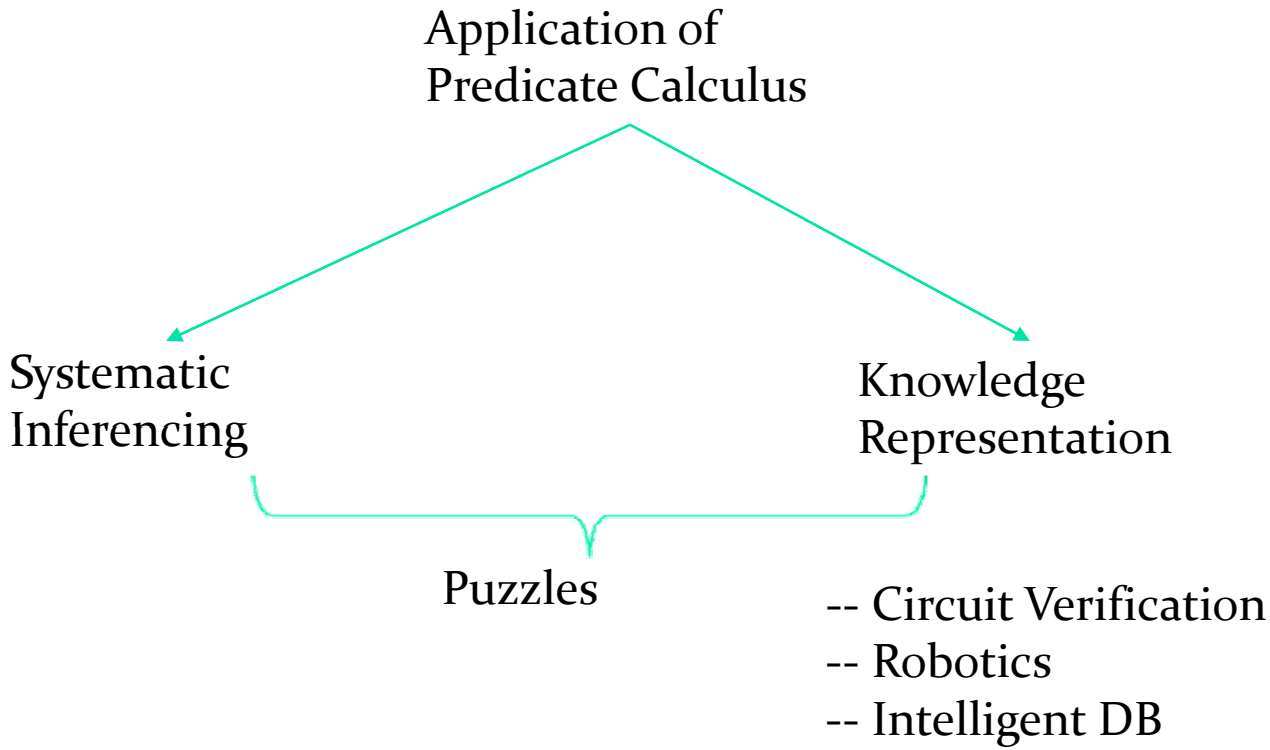
- Now standardize the variables apart which results in the following

1. $member(A)$
2. $member(B)$
3. $member(C)$
4. $\sim member(x_1) \vee mc(x_1) \vee sk(x_1)$
5. $\sim sk(x_2) \vee lk(x_2, snow)$
6. $\sim mc(x_3) \vee \sim lk(x_3, rain)$
7. $\sim like(A, x_4) \vee \sim lk(B, x_4)$
8. $lk(A, x_5) \vee lk(B, x_5)$
9. $lk(A, rain)$
10. $lk(A, snow)$
11. $\sim member(x_6) \vee \sim mc(x_6) \vee sk(x_6)$


~ like(A, x4) ∨ ~ lk(B, x4)
⑦
lk(A, snow) ⑩
⑫ ~ lk(B, snow)
~ sk(x2) ∨ lk(x2, snow) ⑤
⑬ ~ sk(B)
~ member(x1) ∨ mc(x1) ∨ sk(x1) ④
⑭ ~ member(B) ∨ mc(B)
member(B) ②
⑪
~ member(x6) ∨ ~ mc(x6) ∨ sk(x6)
mc(B) ⑮
⑯ ~ member(B) ∨ sk(B)
~ sk(B) ⑬
⑰ ~ member(B)
member(B) ②

Assignment

- Prove the inferencing in the Himalayan club example with different starting points, producing different resolution trees.
- Think of a Prolog implementation of the problem
- Prolog Reference (Prolog by Chockshin & Melish)


Application of
Predicate Calculus
Systematic
Inferencing
Knowledge
Representation
Puzzles
-- Circuit Verification
-- Robotics
-- Intelligent DB

Circuit Verification

- Does the circuit meet the specs?

- Are there faults?
- are they locatable?

Example : 2-bit full adder

C1	X2	X1	Y	C2
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

X_1, X_2: inputs; C_1: prev. carry; C_2: next carry; Y: output

K-Map

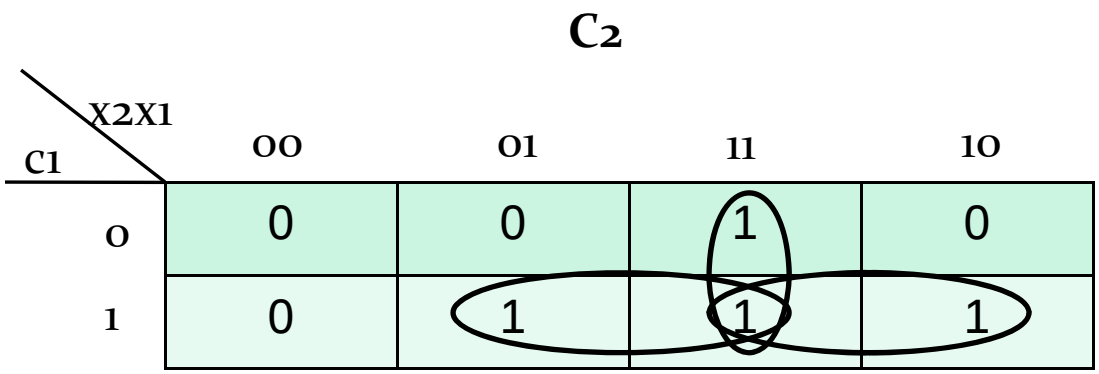
Y

C1 \ X2X1	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$Y = C1(\overline{X1 \oplus X2}) + \overline{C1}(X1 \oplus X2)$$

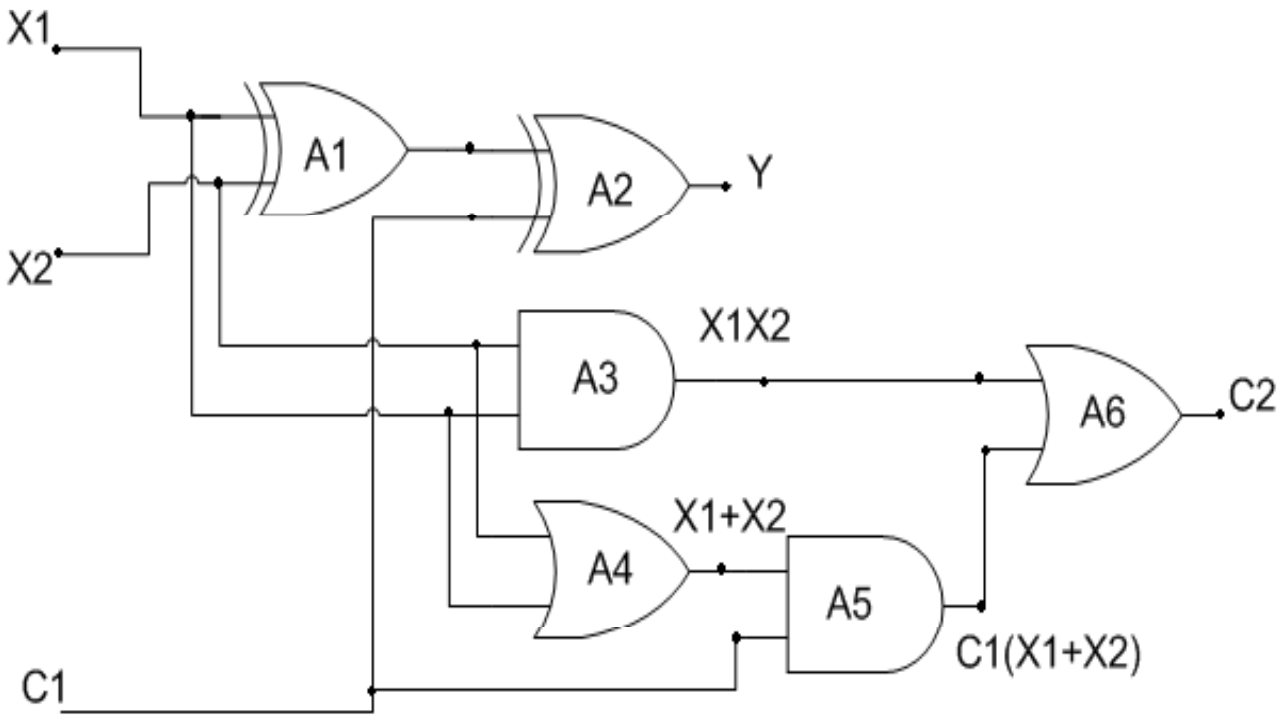
$$= (C1 \oplus (X1 \oplus X2))$$

K-Map (contd..)



$$C2 = X2X1 + C1(X1 + X2)$$

Circuit



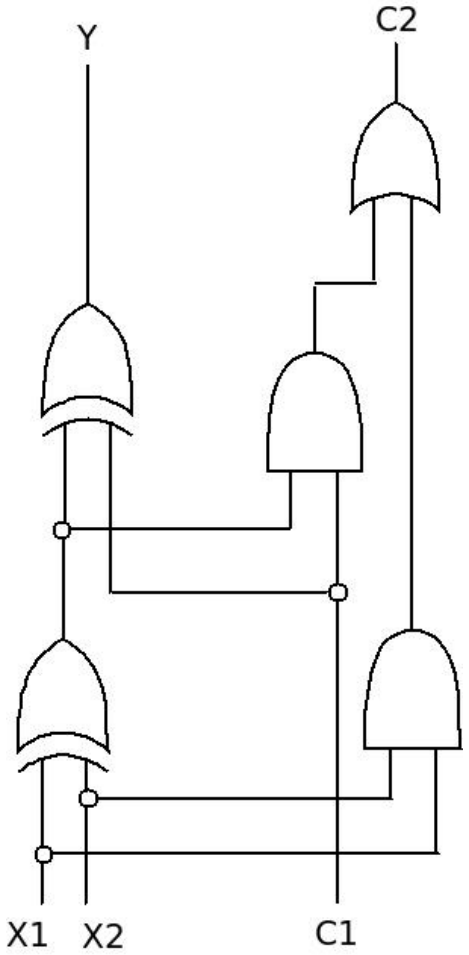
Verification

- First task (most difficult)
  - Building blocks : predicates
  - Circuit observation : Assertion on terminals

Predicates & Functions

Function–1	signal(t)	t is a terminal ; signal takes the value 0 or 1
Function–2	type(x)	x is a circuit element; type(x) takes the value AND, OR, NOT, XOR
Predicate – 3	connected(t1,t2)	t1 is an output terminal and t2 is an input terminal
Function-3	In(n,x)	n^{th} input of ckt element x
Function-4	Out(x)	Output of ckt element x

Alternate Full Adder Circuit



Functions

- type(X) : takes values AND, OR NOT and XOR, where X is a gate.
- in(n, X) : the value of signal at the n^{th} input of gate X.
- out(X) : output of gate X.
- signal(t) : state at terminal t = 1/0

Predicates

- connected(t1,t2): true, if terminal t1 and t2 are connected

General Properties

- Commutativity:

  $\forall t_1,t_2$ [connected(t_1,t_2) $\rightarrow$ connected(t_2,t_1)]

- By definition of connection:

  $\forall t_1,t_2$ [connected(t_1,t_2) $\rightarrow$ { signal(t_1) = signal(t_1)}]

Gate properties

1. OR definition:

$$\forall X\ [\{type(X) = OR\} \equiv \{(out(X) = 1) \equiv \exists y\ (in(y, X) = 1)\}]$$

2. AND definition:

$$\forall X\ [\{type(X) = AND\} \equiv \{(out(X) = 1) \equiv \forall y\ (in(y, X) = 1)\}]$$

Gate properties contd...

1. XOR definition:

$$\forall X\ [\{type(X) = XOR\} \equiv \{(out(X) = 1) \equiv (in(1, X) \neq in(2, X))\}]$$

2. NOT definition:

$$\forall X\ [\{type(X) = NOT\} \equiv \{out(X) \neq in(1, X)\} \wedge (no\_of\_input(X) = 1)]$$

Some necessary functions

a. no_of_input(x), takes values from N.

b. Count_ls(x), returns *#ls* in the input of X

$$\forall X\,[\{\text{type}(X) = \text{XOR}\} \equiv \{(\text{out}(X) = 1) \equiv \text{odd}((\text{count\_ls}(X))\}]$$

Circuit specific properties

- Connectivity:

  connected(x_1, in(1,A_1)),

  connected(x_1, in(2, A_1)),

  connected(out(A_1), in(1, A_2)) ,

  connected(c1, in(2, A_2)),

  connected(y, out(A_2)) ...

- Circuit elements:

  type(A_1) = XOR,

  type(A_2) = XOR,

  type(A_3) = AND ...