Lecture 10 Tree I

Application model a structure XML
index structure BST, AVL Tree ...
data compression, compilers.

Def: Tree is a connected undirected graph with no simple circuit.
no loop or multiple edge

Rooted Tree vertex — root. nodes
parent / child ancestor / descendants
leaf / internal vertex substree

M-ary tree / full m-ary tree every vertex has most m children.
$m=2$ binary tree

Theorem ① n vertices $\rightarrow$ n edges
② full m-ary tree with i internal vertices
$\rightarrow n = m \cdot i + 1$ vertices

level $\rightarrow$ length of path from root to vertex
(h)height $\rightarrow$ maximum level
balance $\rightarrow$ all leave at level h or $h-1$.

Theorem ③ m-ary tree of height h $\rightarrow$ m^h leaves
↳ m-ary tree has l leaves $\rightarrow$ height $\geq \lceil \log_m l \rceil$
↳ m-ary tree full & balance $\rightarrow$ height $= \lceil \log_m l \rceil$

Ordered Tree

- Tree traversal
  - Preorder parent, right child, left child.
  - Postorder right child, left child, parent.
  - Inorder right child, parent, left child.

* Arithmetic expression
  - infix - need parentheses
  - prefix - preorder traversal
  - postfix - postorder traversal
* logic expression

Binary Search Tree

$x.left.key \le x.key \le x.right.key$

T root — T.root   key value — x.key
left child — x.left   right child — x.right

Operation   search $O(h)$ insert/delete $O(h)$
minimum/maximum/predecessor/successor $O(h)$

Search algorithm

Search(x, k)
1. if x = NIL or k = x.key
2.   return x
3. if k < x.key
4.   Search(x.left, k)
5. else
6.   Search(x.right, k)

Minimum algorithm

Minimum(x)
1. while x.left ≠ NIL
2.   x ← x.left
3. return x

Insertion algorithm

Insert(T, z)
1. y ← NIL   x ← T.root
2. while x ≠ NIL
3.   y ← x
4.   if z.key < x.key
5.   x ← x.left
6.   else
7.   x ← x.right.
8. z.p ← y
9. if y = NIL
10.   T.root ← z
11. else if z.key < y.key
12.   y.left ← z
13. else
14.   y.right ← z

Deletion Algorithm

Transplant(T, u, v)
    if u.p = NIL
        T.root ← v.
    else if u = u.p.left
        u.p.left ← v
    else
        u.p.right ← v
    if v ≠ NIL
        v.p ← u.p

Delete(T, z)
    if z.left = NIL
        Transplant(T, z, z.right)
    else if z.right = NIL
        Transplant(T, z, z.left)
    else
        y ← Minimum(z.right)
        Delete(T, y)
        replace z by y.

other type of data → string. (in alphabetic order)

Lecture 12 Boolean Algebra & Circuits

expression: constant value
variable
formed by
sub-expression

	Logic	Boolean algebra
Values	false (F)	0
	true (T)	1

operator
- complement $\bar{0} = 1$ $\bar{1} = 0$
- boolean sum $0+0=0$ $0+1=1$ $1+0=1$ $1+1=1$
- boolean product $0 \cdot 0 = 0$ $0 \cdot 1 = 0$ $1 \cdot 0 = 0$ $1 \cdot 1 = 1$

precedence $+ \rightarrow \cdot \rightarrow \bar{\ } \rightarrow (\ )$

Convesion b/t logical expression & Boolean algebra.

$(T \wedge F) \vee \neg(T \vee F) \equiv F \iff 1 \cdot 0 + (\overline{1+0}) = 0$

prove identity using truth table

Identity
- zero property $x \cdot \bar{x} = 0$
- unit property $x + \bar{x} = 1$
- double complement $\bar{\bar{x}} = x$
- idempotent $x + x = x$ $x \cdot x = x$
- identity $x + 0 = x$ $x \cdot 1 = x$
- Domination $x + 1 = 1$ $x \cdot 0 = 0$
- Commutative $x+y=y+x$ $x \cdot y = y \cdot x$
- associative $x+(y+z)=(x+y)+z$ $x \cdot (y \cdot z) = (x \cdot y) \cdot z$
- distribution $x \cdot (y+z) = x \cdot y + y \cdot z$ $x + (y \cdot z) = (x+z) \cdot (y+z)$
- De Morgan $\overline{(x+y)} = \bar{x} \cdot \bar{y}$ $\overline{(x \cdot y)} = \bar{x} + \bar{y}$
- Absorption $x + x \cdot y = x \cdot y$ $x \cdot (x+y) = x$

Duality interchange $+$ & $\cdot$
interchange 0 & 1

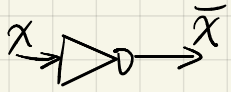
Boolean Function $F(x,y)$ defined using Boolean expression.

number of different boolean function: 2^n combination for n variable
& 2^{2^n} different function.

sum of product expansion

Logic gate.

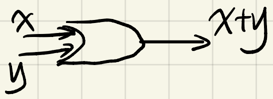
NOT gate



$F(x)=\bar{x}$

x	$F(x)$
1	0
0	1

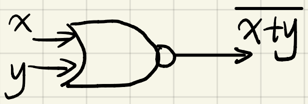
OR gate



$F(x,y)=x+y$

x	y	$F(x,y)$
1	1	1
1	0	1
0	1	1
0	0	0

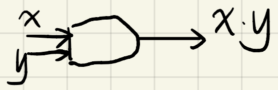
NOR gate



$F(x,y)=\overline{x+y}$

x	y	$F(x,y)$
1	1	0
1	0	0
0	1	0
0	0	1

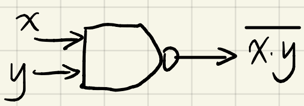
AND gate



$F(x,y)=x\cdot y$

x	y	$F(x,y)$
1	1	1
1	0	0
0	1	0
0	0	0

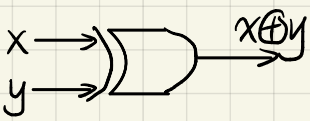
NAND gate



$F(x,y)=\overline{x\cdot y}$

x	y	$F(x,y)$
1	1	0
1	0	1
0	1	1
0	0	1

XOR gate

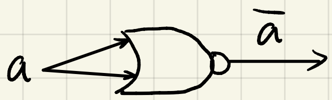


$F(x,y)=x\oplus y$
$=x\cdot\bar{y}+\bar{x}\cdot y$

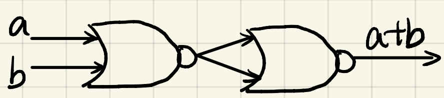
x	y	$F(x,y)$
1	1	0
1	0	1
0	1	1
0	0	0

Combination circuit

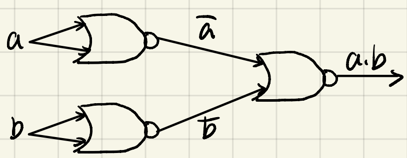
NOT: $\bar{a}=\overline{a+a}$



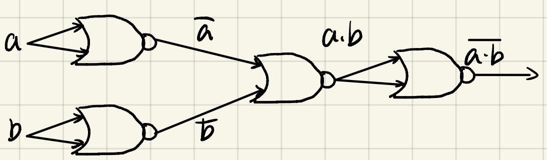
OR: $a+b=\overline{\overline{a+b}}$



AND $a\cdot b=\overline{\bar{a}+\bar{b}}$



NAND $\overline{a\cdot b}=\bar{a}+\bar{b}$



XOR $a\cdot\bar{b}+\bar{a}\cdot b=\overline{\bar{a}+b}+\overline{a+\bar{b}}$

minimization of circuit

- How to simplify $F(x,y) = \bar{x}\cdot\bar{y} + x\cdot\bar{y} + xy$

K-map

	y	$\bar{y}$
x	1	1
$\bar{x}$		1

x $\quad$ $\bar{y}$

$\bar{y} + x$

$$F(x,y,z) = x\bar{y}\bar{z} + x\bar{y}z + \bar{x}yz + \bar{x}\bar{y}$$

$\bar{x}\bar{y} \rightarrow \bar{x}\bar{y}z + \bar{x}\bar{y}\bar{z}$

K-map

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x			1	1
$\bar{x}$	1		1	1

$\bar{x}z$ $\quad$ $\bar{y}$

$F(x,y,z) = \bar{y} + \bar{x}z$

$$F(x,y,z) = xyz + x\bar{z} + x\bar{y}z + \bar{x}\bar{y}z + \bar{x}yz$$

$x\bar{z} \rightarrow xy\bar{z} + x\bar{y}\bar{z}$

$F(x,y,z) = x + z$

K-map

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x	1	1	1	1
$\bar{x}$	1			1

x $\quad$ z

$$F(w,x,y,z) = \bar{x}y\bar{z} + \bar{x}\bar{y}\bar{z} + wxz + \bar{w}xz$$

K-map

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
wx	1			1
$w\bar{x}$		1	1	
$\bar{w}\bar{x}$		1	1	
$\bar{w}x$	1			1

$\rightarrow \bar{x}\bar{z}$

xz

$F(w,x,y,z) = \bar{x}\bar{z} + xz$