# Recognition of Fanout-free Functions

Speaker: Yi-Ling Liou
Author: Tsung-Lin Lee
Chun-Yao Wang

Dept. of Computer Science
National Tsing Hua University, Taiwan

# Outline

- Introduction
- Background
- Our Improvements
- Experimental Results
- Conclusions

# Introduction

- A Boolean function is called a fanout-free function if it has a form in which each variable appears exactly once

- $f_7 = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$
  $= (x_1x_2x_3 + x_6)(x_4 + x_5)$

# Introduction (cont′d)

- Fanout-free function is a tree-like structure
- ATPG
- Technology mapping algorithm, DAGON
- Testability evaluation

# Introduction (cont′d)

- J. P. Hayes proposed an algorithm to recognize fanout-free functions
- We propose some efficient ways to accelerate the algorithm

# Outline

- Introduction
- Background
- Our Improvements
- Experimental Results
- Conclusions

# Unate

*Definition* :

$f(X)$ is positive unate in $x_i \in X$

if $f(x_i = 1) = 1$ whenever $f(x_i = 0) = 1$ ;

$f(X)$ is negative unate in $x_i \in X$

if $f(x_i = 0) = 1$ whenever $f(x_i = 1) = 1$

$$f = x_1 + \overline{x}_1\overline{x}_2$$
$$= x_1x_2 + x_1\overline{x_2} + \overline{x}_1\overline{x}_2$$
$$= x_1 + \overline{x_2}$$

| $x_1$ | $x_2$ | $f$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

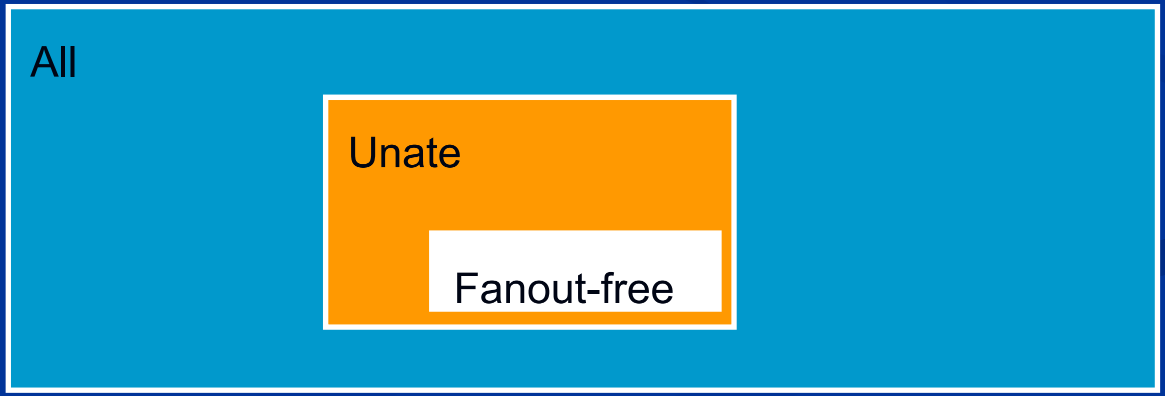
$x_1$ is positive unate; $x_2$ is negative unate

# Unate Function

- Definition:
  - f(X) is a unate function if all variables are unate
- A unate function could be a fanout-free function

# Fanout-free Function

- fanout-free $\rightarrow$ unate
  - $f = x_1x_2 + x_1x_3 = x_1(x_2 + x_3)$
- unate $\not\rightarrow$ fanout-free
  - $f = x_1x_2 + x_2x_3 + x_3x_1$



# Preliminary

- We discuss unate functions only
  - Not unate → Not fanout-free
- Without loss of generality, we assume variables are positive unate
  - $F(x_1,x_2)=x_1+x'_2$
  - $G(x_1,x'_2)=x_1+x_2$

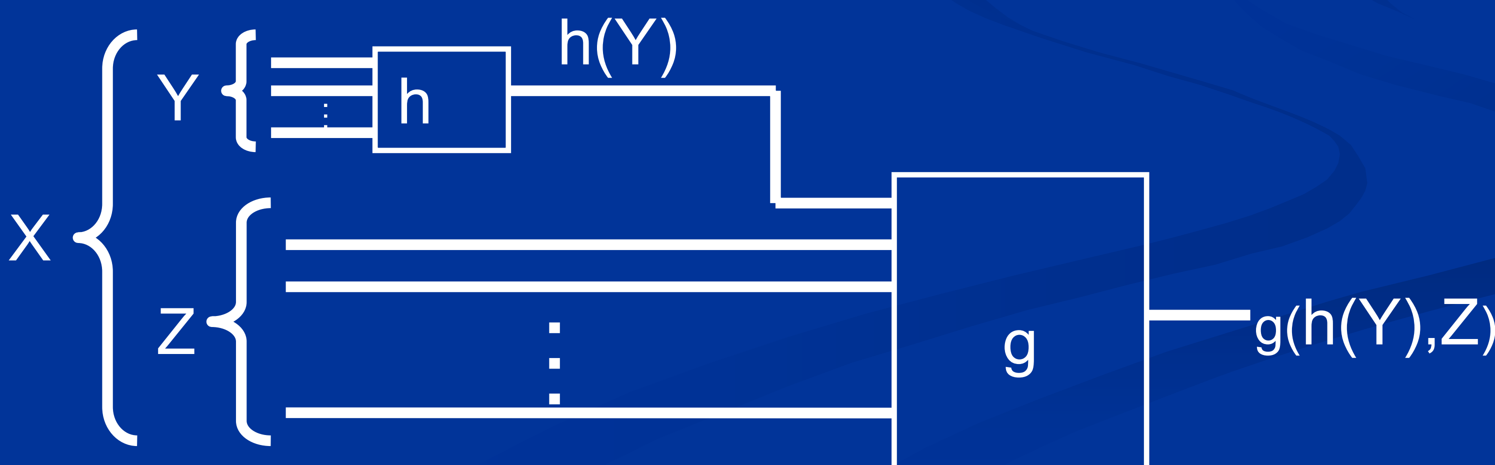
# Disjunctive Decomposition

- Simple disjunctive decomposition extracts a single-output sub-function
- Input variable set is disjunctive from the others

$$Y \cap Z = \phi$$

$$Y \cup Z = X$$

$$f(X) = g(h(Y), Z)$$

# Adjacency Relation

- Definition:
  - $x_i \neq x_j$ , $x_i$ is adjacent to $x_j$ if $f(x_i=a) = f(x_j=a)$ for some constant a. It is denoted by $=_a$

$$f_7(x_1,x_2,x_3,x_4,x_5,x_6) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$\boxed{\begin{aligned} f_7(x_1 = 0) &= x_4x_6 + x_5x_6 \\ f_7(x_2 = 0) &= x_4x_6 + x_5x_6 \\ f_7(x_3 = 0) &= x_4x_6 + x_5x_6 \end{aligned}}$$

$$f_7(x_4 = 0) = x_1x_2x_3x_5 + x_5x_6$$

$$f_7(x_5 = 0) = x_1x_2x_3x_4 + x_4x_6$$

$$f_7(x_6 = 0) = x_1x_2x_3x_4 + x_1x_2x_3x_5$$

$$f_7(x_1 = 1) = x_2x_3x_4 + x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$f_7(x_2 = 1) = x_1x_3x_4 + x_1x_3x_5 + x_4x_6 + x_5x_6$$

$$f_7(x_3 = 1) = x_1x_2x_4 + x_1x_2x_5 + x_4x_6 + x_5x_6$$

$$\boxed{\begin{aligned} f_7(x_4 = 1) &= x_1x_2x_3 + x_6 \\ f_7(x_5 = 1) &= x_1x_2x_3 + x_6 \end{aligned}}$$

$$f_7(x_6 = 1) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4 + x_5$$

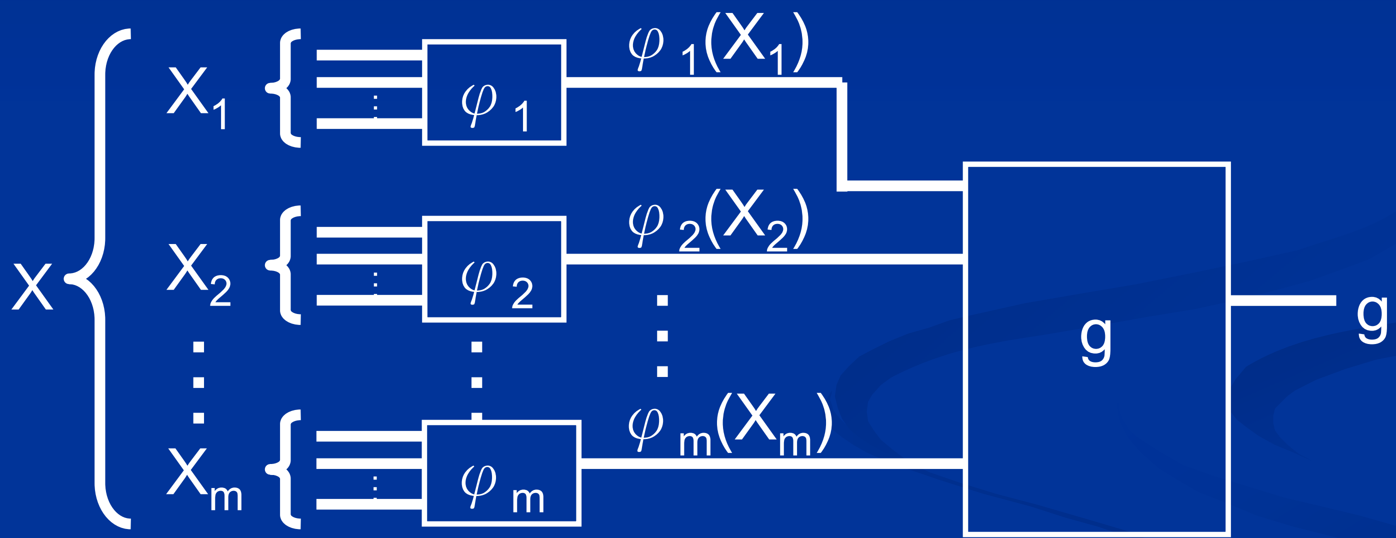
$\boxed{x_1 =_0 x_2 =_0 x_3}$ $\boxed{x_4 =_1 x_5}$ $\boxed{x_6}$

# Adjacency Relation (cont'd)

- Adjacency is reflexive, symmetric, and transitive
  - Reflexive: $x_i =_a x_i$
  - Symmetric: $x_i =_a x_j \rightarrow x_j =_a x_i$
  - Transitive: $x_i =_a x_j$ and $x_j =_a x_k \rightarrow x_i =_a x_k$
- Adjacency is an equivalence relation
  - E.R. can partition elements into disjoint sets
- (Math: Adjacency Relation is irreflexive and symmetric)

# Adjacency and Decomposition

- Let the variables of g(X) be partitioned into blocks by AR

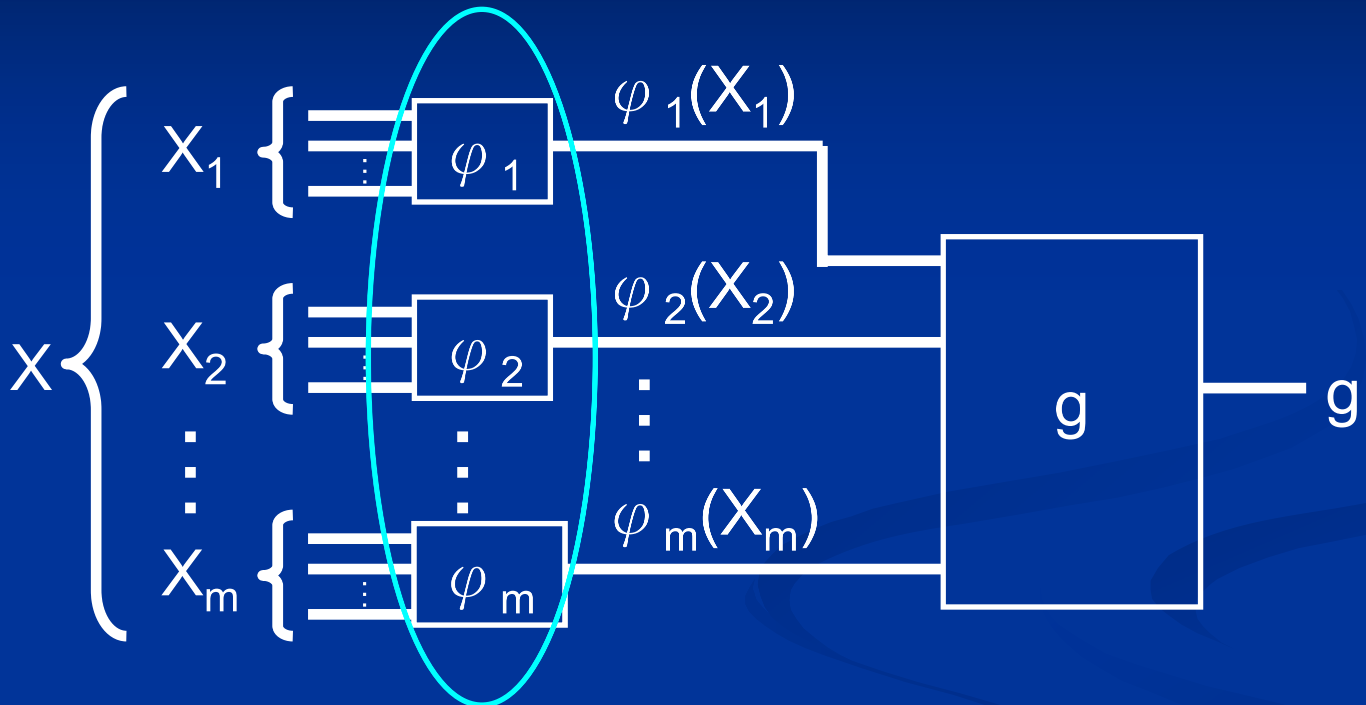



$$g(X) = g(\varphi_1(X_1), \varphi_2(X_2), \cdots, \varphi_m(X_m)),$$

$$X_i \cap X_j = \phi \qquad \forall i \neq j,$$

$$X_1 \cup X_2 \cup \cdots \cup X_m = X$$

# Adjacency and Decomposition (cont′d)




- These are elementary gates (AND or OR)
- A corresponding elementary function is composed of a subset of variables

# Adjacency and Decomposition (cont′d)

- $=_0$: AND gate
- $=_1$: OR gate

$$f_7(x_1,x_2,x_3,x_4,x_5,x_6) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$x_1 =_0 x_2 =_0 x_3 \quad : \quad \varphi_1 = x_1x_2x_3$$

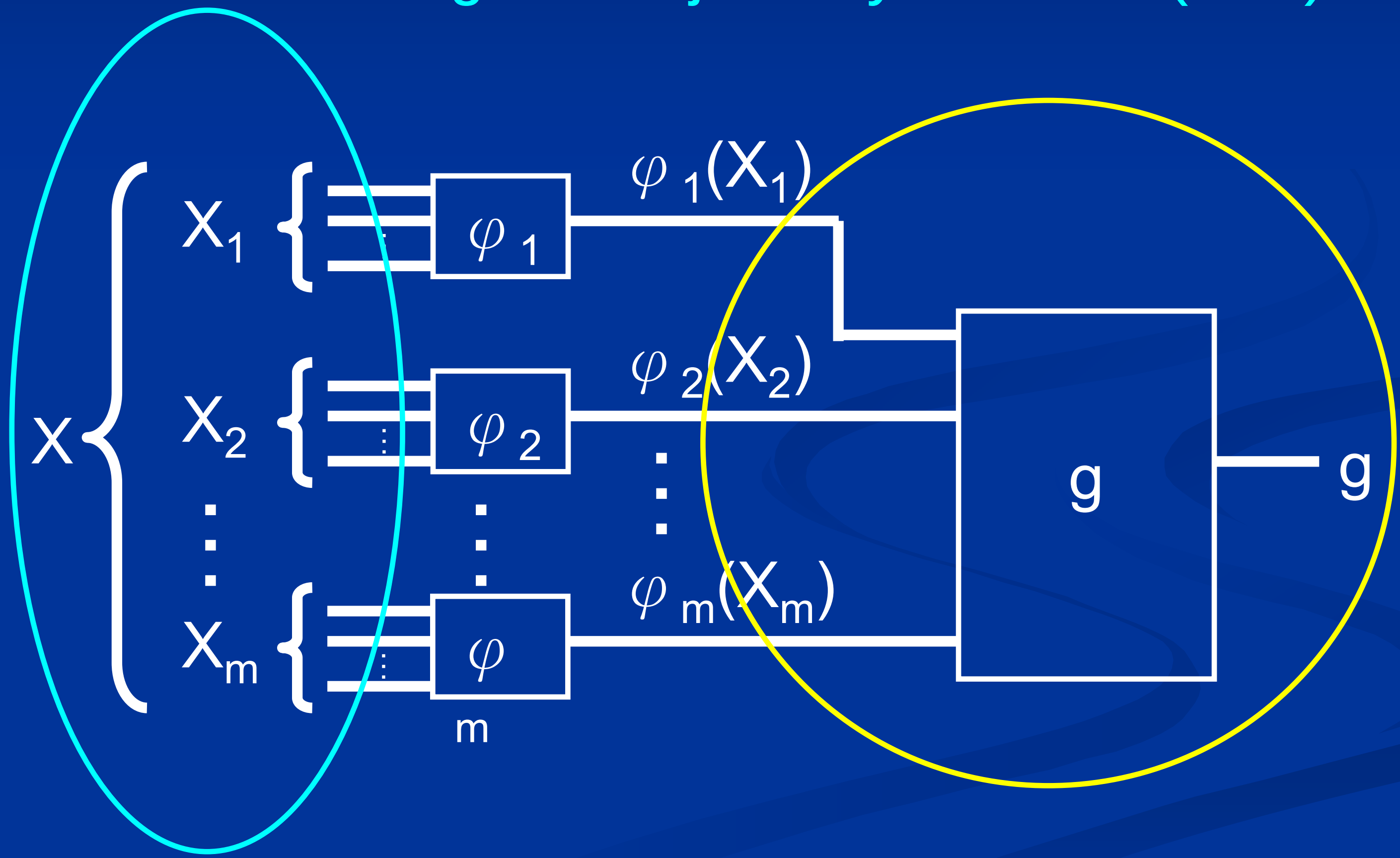
$$x_4 =_1 x_5 \quad : \quad \varphi_2 = x_4 + x_5$$

$$x_6 \quad : \quad \varphi_3 = x_6$$

$$f_7(x_1,x_2,x_3,x_4,x_5,x_6) = F(\varphi_1,\varphi_2,\varphi_3)$$

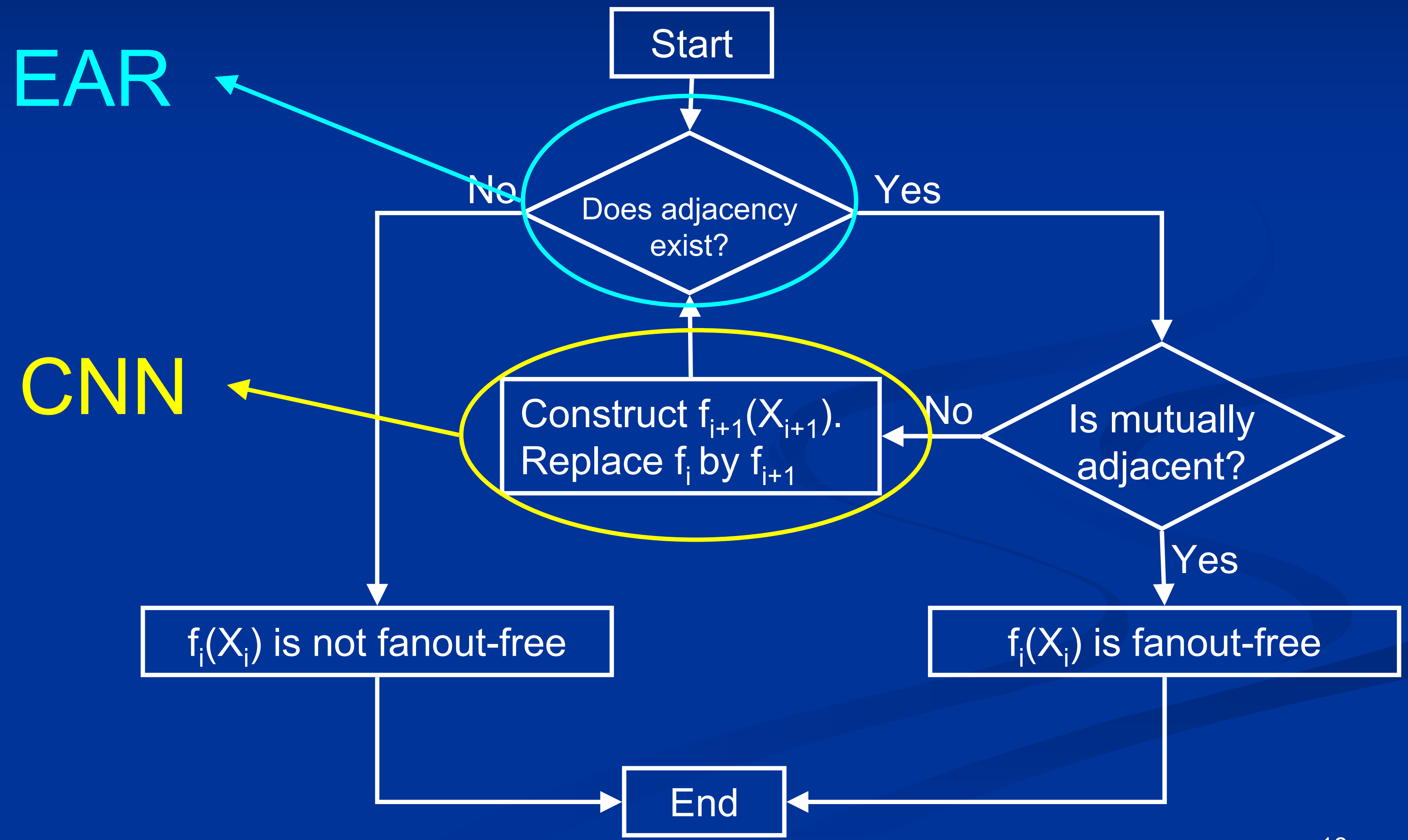
# Concept

1. Examining the Adjacency Relation (EAR)




2. Constructing the New Network (CNN)

# The Procedure of J. P. Hayes Algorithm



# Example

$$f_7(X) = f_7(x_1, x_2, x_3, x_4, x_5, x_6) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$$

$f_7$ is a unate function

$$\boxed{\begin{aligned} f_7(x_1 = 0) &= x_4x_6 + x_5x_6 \\ f_7(x_2 = 0) &= x_4x_6 + x_5x_6 \\ f_7(x_3 = 0) &= x_4x_6 + x_5x_6 \end{aligned}}$$

$$f_7(x_4 = 0) = x_1x_2x_3x_5 + x_5x_6$$

$$f_7(x_5 = 0) = x_1x_2x_3x_4 + x_4x_6$$

$$f_7(x_6 = 0) = x_1x_2x_3x_4 + x_1x_2x_3x_5$$

$$f_7(x_1 = 1) = x_2x_3x_4 + x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$f_7(x_2 = 1) = x_1x_3x_4 + x_1x_3x_5 + x_4x_6 + x_5x_6$$

$$f_7(x_3 = 1) = x_1x_2x_4 + x_1x_2x_5 + x_4x_6 + x_5x_6$$

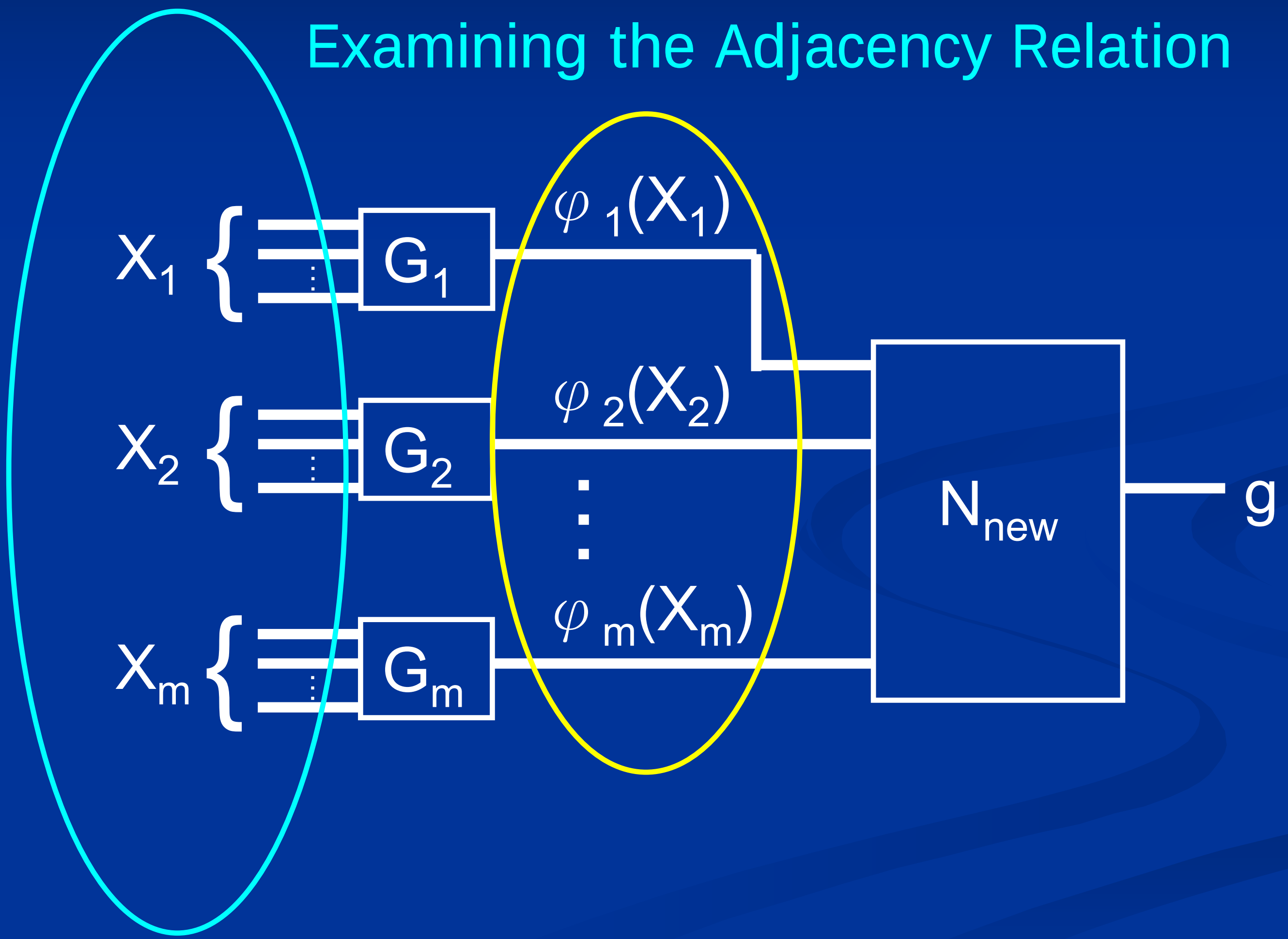
$$\boxed{\begin{aligned} f_7(x_4 = 1) &= x_1x_2x_3 + x_6 \\ f_7(x_5 = 1) &= x_1x_2x_3 + x_6 \end{aligned}}$$

$$f_7(x_6 = 1) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4 + x_5$$

Thus $X$ is partitioned into three blocks:

$\boxed{X_1 = \{x_1, x_2, x_3\}}$ $\boxed{X_2 = \{x_4, x_5\}}$ $\boxed{X_3 = \{x_6\}}$

# Example (cont'd)



# Example (cont'd)

$$\varphi_1 = x_1 x_2 x_3,\ \ \varphi_2 = x_4 + x_5,\ \ \varphi_3 = x_6$$

Thus $f_7(X) = f_{7.1}(\varphi_1, \varphi_2, \varphi_3)$

$$f_7(X) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$$

| $\varphi_1$ | $\varphi_2$ | $\varphi_3$ | $f_{7.1}$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |

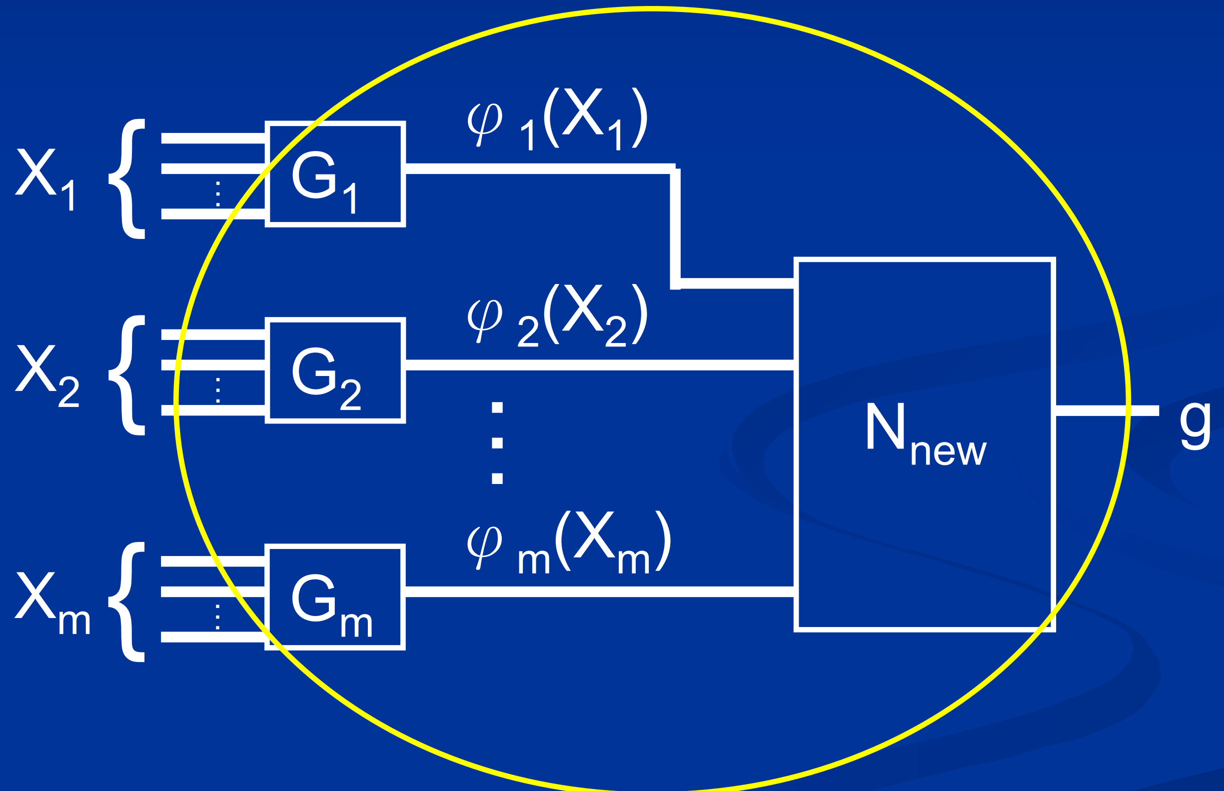
| $\varphi_1$ | $\varphi_2$ | $\varphi_3$ | $f_{7.1}$ |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

$$f_{7.1}(\varphi_1, \varphi_2, \varphi_3) = \overline{\varphi}_1\varphi_2\varphi_3 + \varphi_1\varphi_2\overline{\varphi}_3 + \varphi_1\varphi_2\varphi_3$$

$$= \varphi_1\varphi_2 + \varphi_2\varphi_3$$

# Example (cont'd)

Constructing the New Network



# Outline

- Introduction
- Background Materials
- Our Improvements
- Experimental Results
- Conclusions

# Bottlenecks

- Examining the Adjacency Relation
  - About $n^2$ E.C. should be done ($O(n^2 2^n)$)

$$
\begin{aligned}
f_7(x_1=0) &= x_4x_6 + x_5x_6 & f_7(x_1=1) &= x_2x_3x_4 + x_2x_3x_5 + x_4x_6 + x_5x_6 \\
f_7(x_2=0) &= x_4x_6 + x_5x_6 & f_7(x_2=1) &= x_1x_3x_4 + x_1x_3x_5 + x_4x_6 + x_5x_6 \\
f_7(x_3=0) &= x_4x_6 + x_5x_6 & f_7(x_3=1) &= x_1x_2x_4 + x_1x_2x_5 + x_4x_6 + x_5x_6 \\
f_7(x_4=0) &= x_1x_2x_3x_5 + x_5x_6 & f_7(x_4=1) &= x_1x_2x_3 + x_6 \\
f_7(x_5=0) &= x_1x_2x_3x_4 + x_4x_6 & f_7(x_5=1) &= x_1x_2x_3 + x_6 \\
f_7(x_6=0) &= x_1x_2x_3x_4 + x_1x_2x_3x_5 & f_7(x_6=1) &= x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4 + x_5
\end{aligned}
$$

- Constructing the New Network
  - Time complexity is $O(2^m)$

| $\varphi_1$ | $\varphi_2$ | $\varphi_3$ | $f_{7.1}$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |

| $\varphi_1$ | $\varphi_2$ | $\varphi_3$ | $f_{7.1}$ |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Do EAR without E.C.

Do CNN in O(1)

# Disappearance Property

- Adjacency $\longrightarrow$ Disappearance

$$x_i =_a x_j \Leftrightarrow f(x_i = a) = f(x_j = a)$$

$$\Rightarrow \begin{array}{l} 1.\ x_i \text{ is disappeared in } f(x_j = a) \\ 2.\ x_j \text{ is disappeared in } f(x_i = a) \end{array}$$

Example:

$$f_7 = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$f_7(x_4 = 1) = x_1x_2x_3 + x_6 = f_7(x_5 = 1)$$

# Disappearance Property (cont′d)

- Disappearance → Adjacency

$x_i$ is disappeared in $f(x_j = a)$ and $x_j$ is disappeared in $f(x_i = a)$

$\Rightarrow f(x_i = a) = f(x_j = a) \Leftrightarrow x_i =_a x_j$

# Corollary

Adjacency ⇔ Disappearance

- Disappearance property also holds in multiple variables
  - Adjacency is transitive
    - $x_i =_a x_j$ and $x_j =_a x_k$

      $\rightarrow x_i =_a x_k$

      $\rightarrow x_k$ disappears in $f(x_j=a)$

      $x_i$ disappears in $f(x_k=a)$

# Effect of Disappearance Property

Adjacency ⇔ Disappearance

- Examine the Adjacency Relation by checking appearances of variables
- Construct the New Network by substituting variables

# Improvement on EAR

$$f_7(X) = f_7(x_1, x_2, x_3, x_4, x_5, x_6) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$\boxed{f_7(0_1) = x_4x_6 + x_5x_6} \qquad f_7(1_1) = x_2x_3x_4 + x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$f_7(0_2) = x_4x_6 + x_5x_6 \qquad f_7(1_2) = x_1x_3x_4 + x_1x_3x_5 + x_4x_6 + x_5x_6$$

$$f_7(0_3) = x_4x_6 + x_5x_6 \qquad f_7(1_3) = x_1x_2x_4 + x_1x_2x_5 + x_4x_6 + x_5x_6$$

$$f_7(0_4) = x_1x_2x_3x_5 + x_5x_6 \qquad f_7(1_4) = x_1x_2x_3 + x_6$$

$$f_7(0_5) = x_1x_2x_3x_4 + x_4x_6 \qquad f_7(1_5) = x_1x_2x_3 + x_6$$

$$f_7(0_6) = x_1x_2x_3x_4 + x_1x_2x_3x_5 \qquad f_7(1_6) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4 + x_5$$

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ |
|---|---|---|---|---|---|---|
| $f_7(0_1)$ | 0 | 0 | 0 | 1 | 1 | 1 |
| $f_7(0_2)$ | 0 | 0 | 0 | 1 | 1 | 1 |
| $f_7(0_3)$ | 0 | 0 | 0 | 1 | 1 | 1 |
| $f_7(0_4)$ | 1 | 1 | 1 | 0 | 1 | 1 |
| $f_7(0_5)$ | 1 | 1 | 1 | 1 | 0 | 1 |
| $f_7(0_6)$ | 1 | 1 | 1 | 1 | 1 | 0 |

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ |
|---|---|---|---|---|---|---|
| $f_7(1_1)$ | 0 | 1 | 1 | 1 | 1 | 1 |
| $f_7(1_2)$ | 1 | 0 | 1 | 1 | 1 | 1 |
| $f_7(1_3)$ | 1 | 1 | 0 | 1 | 1 | 1 |
| $f_7(1_4)$ | 1 | 1 | 1 | 0 | 0 | 1 |
| $f_7(1_5)$ | 1 | 1 | 1 | 0 | 0 | 1 |
| $f_7(1_6)$ | 1 | 1 | 1 | 1 | 1 | 0 |

# Improvement on EAR (cont'd)

$$
\begin{array}{r|cccccc}
 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \\
\rightarrow f_7(0_1) & 0 & \boxed{0} & \boxed{0} & 1 & 1 & 1 \\
\rightarrow f_7(0_2) & \boxed{0} & 0 & 0 & 1 & 1 & 1 \\
\rightarrow f_7(0_3) & \boxed{0} & 0 & 0 & 1 & 1 & 1 \\
f_7(0_4) & 1 & 1 & 1 & 0 & 1 & 1 \\
f_7(0_5) & 1 & 1 & 1 & 1 & 0 & 1 \\
f_7(0_6) & 1 & 1 & 1 & 1 & 1 & 0
\end{array}
$$

Checking whether $x_2 =_a x_3$ is unnecessary

$x_1 =_a x_2 =_a x_3$

# Improvement on CNN

- In the process of proving "Disappearance Property", we found

1. If $x_1 =_0 x_2 =_0 \cdots =_0 x_r$ $\quad (\varphi = x_1 x_2 \cdots x_r)$

$$\Rightarrow f(x_1, x_2, \cdots, x_r, A, Z) = x_1 x_2 \cdots x_r A + Z$$

$$\Rightarrow f(x_1 = \varphi, x_2 = 1, \cdots, x_r = 1, A, Z) = \varphi 1 \cdots 1 A + Z$$

2. If $x_1 =_1 x_2 =_1 \cdots =_1 x_r$ $\quad (\varphi = x_1 + x_2 + \cdots + x_r)$

$$\Rightarrow f(x_1, x_2, \cdots, x_r, A, Z) = (x_1 + x_2 + \cdots + x_r) A + Z$$

$$\Rightarrow f(x_1 = \varphi, x_2 = 0, \cdots, x_r = 0, A, Z) = (\varphi + 0 + \cdots + 0) A + Z$$

# Improvement on CNN

$$f_7(X) = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_4x_6 + x_5x_6$$

$$x_1 =_0 x_2 =_0 x_3, \quad x_4 =_1 x_5$$

$$\varphi_1 = x_1x_2x_3, \quad \varphi_2 = x_4 + x_5, \quad \varphi_3 = x_6$$

$$\begin{aligned}
&f_{7.1}(\varphi_1, \varphi_2, \varphi_3) \\
&= f_7(x_1 = \varphi_1, x_2 = 1, x_3 = 1, x_4 = \varphi_2, x_5 = 0, x_6 = \varphi_3) \\
&= \varphi_1 1 1 \varphi_2 + \varphi_1 1 1 0 + \varphi_2\varphi_3 + 0\varphi_3 \\
&= \varphi_1\varphi_2 + \varphi_2\varphi_3
\end{aligned}$$

**We only need constant time!**

# Outline

- Introduction
- Background
- Our Improvements
- Experimental Results
- Conclusions

# Experimental Results

- Implement our algorithm and re-implement JPH proposed in [4]
  - Sun Blade 2500 machine
- IROF was proposed in [3]
- All three algorithms recognize identical fanout-free functions

# Experimental Results (cont′d)

| | lit(SOP) | #Vars | JPH [4] | IROF [3] | Ours |
|---|---|---|---|---|---|
| L2_B10 | 10240 | 20 | 4 | 0.30 | 0.11 |
| L4_B3 | 3072 | 24 | 6 | 0.10 | 0.08 |
| L4_B6 | 7290 | 24 | 277 | 0.21 | 0.18 |
| L6_B4 | 672 | 20 | 3 | 0.02 | 0.02 |
| L6_B8A | 132 | 52 | >1hr | 0.02 | 0.29 |
| L6_B8B | 24192 | 52 | >1hr | 0.74 | 2.08 |
| L8_B5 | 3380 | 29 | >1hr | 0.09 | 0.11 |
| L10_B3 | 2160 | 30 | >1hr | 0.07 | 0.16 |
| L14_B3 | 6720 | 42 | >1hr | 0.20 | 0.72 |

# Outline

- Introduction
- Background
- Our Improvements
- Experimental Results
- Conclusions

# Conclusions

- Disappearance property accelerates the J. P. Hayes' procedure of recognizing fanout-free functions
- Our method also produces a partially fanout-free function on recognizing a fanout function, while IROF gets nothing

Thank you !!