Xiao-hua LI, Ji-zhong SHEN, 2017. An algorithm for identifying symmetric variables based on the order eigenvalue matrix. *Frontiers of Information Technology & Electronic Engineering,* **18**(10):1644-1653.
https://doi.org/10.1631/FITEE.1601052

An algorithm for identifying symmetric variables based on the order eigenvalue matrix

Key words: Boolean function; Symmetric variable; Boolean logic algebra system; Order eigenvalue matrix; Truth table

Corresponding author: Ji-zhong SHEN
E-mail: jzshen@zju.edu.cn

Motivation

- A new symmetry detection algorithm based on minterm expansion or the truth table. Experimental results show that the proposed algorithm is convenient and efficient.

- Disadvantages of existing methods:
  - Applicability of the number of logical variables usually $n \leq 6$
  - The identification processes are complicated
  - Only six types can been identified
  - Not applicable the Boolean function with don't-care terms

Features of our method

– Avoids the restriction by the number of logic variables of the graphical method, spectral coefficient methods, and AND-XOR expansion coefficient methods

– Solves the problem of completeness in the fast computation method.

– The new algorithm is an optimal detection method in terms of the applicability of the number of logic variables, the Boolean function including don’t-care terms, detection type, and complexity of the identification process.

Framework of our method (I)

The algorithms can be divided into two parts:

Table 6 Symmetric conditions of the logic variable

Sequence number	Condition of $[a_i]_{x_i x_j}$	Type of symmetry
1	$[a_i]_{00} \oplus [a_i]_{11} = [0,0,\ldots,0]$	$E(x_i \vert x_j)$
2	$[a_i]_{01} \oplus [a_i]_{10} = [0,0,\ldots,0]$	$N(x_i \vert x_j)$
3	$[a_i]_{01} \oplus [a_i]_{11} = [0,0,\ldots,0]$	$S(x_i \vert x_j)$
4	$[a_i]_{00} \oplus [a_i]_{10} = [0,0,\ldots,0]$	$S(x_i \vert \overline{x}_j)$
5	$[a_i]_{10} \oplus [a_i]_{11} = [0,0,\ldots,0]$	$S(x_j \vert x_i)$
6	$[a_i]_{00} \oplus [a_i]_{01} = [0,0,\ldots,0]$	$S(x_j \vert \overline{x}_i)$
7	$[a_i]_{00} \oplus [a_i]_{11} = [1,1,\ldots,1]$	$CE(x_i \vert x_j)$
8	$[a_i]_{01} \oplus [a_i]_{10} = [1,1,\ldots,1]$	$CN(x_i \vert x_j)$
9	$[a_i]_{01} \oplus [a_i]_{11} = [1,1,\ldots,1]$	$CS(x_i \vert x_j)$
10	$[a_i]_{00} \oplus [a_i]_{10} = [1,1,\ldots,1]$	$CS(x_i \vert \overline{x}_j)$
11	$[a_i]_{10} \oplus [a_i]_{11} = [1,1,\ldots,1]$	$CS(x_j \vert x_i)$
12	$[a_i]_{00} \oplus [a_i]_{01} = [1,1,\ldots,1]$	$CS(x_j \vert \overline{x}_i)$

Framework of our method (II)

The flow chart of the program implemented in C language:

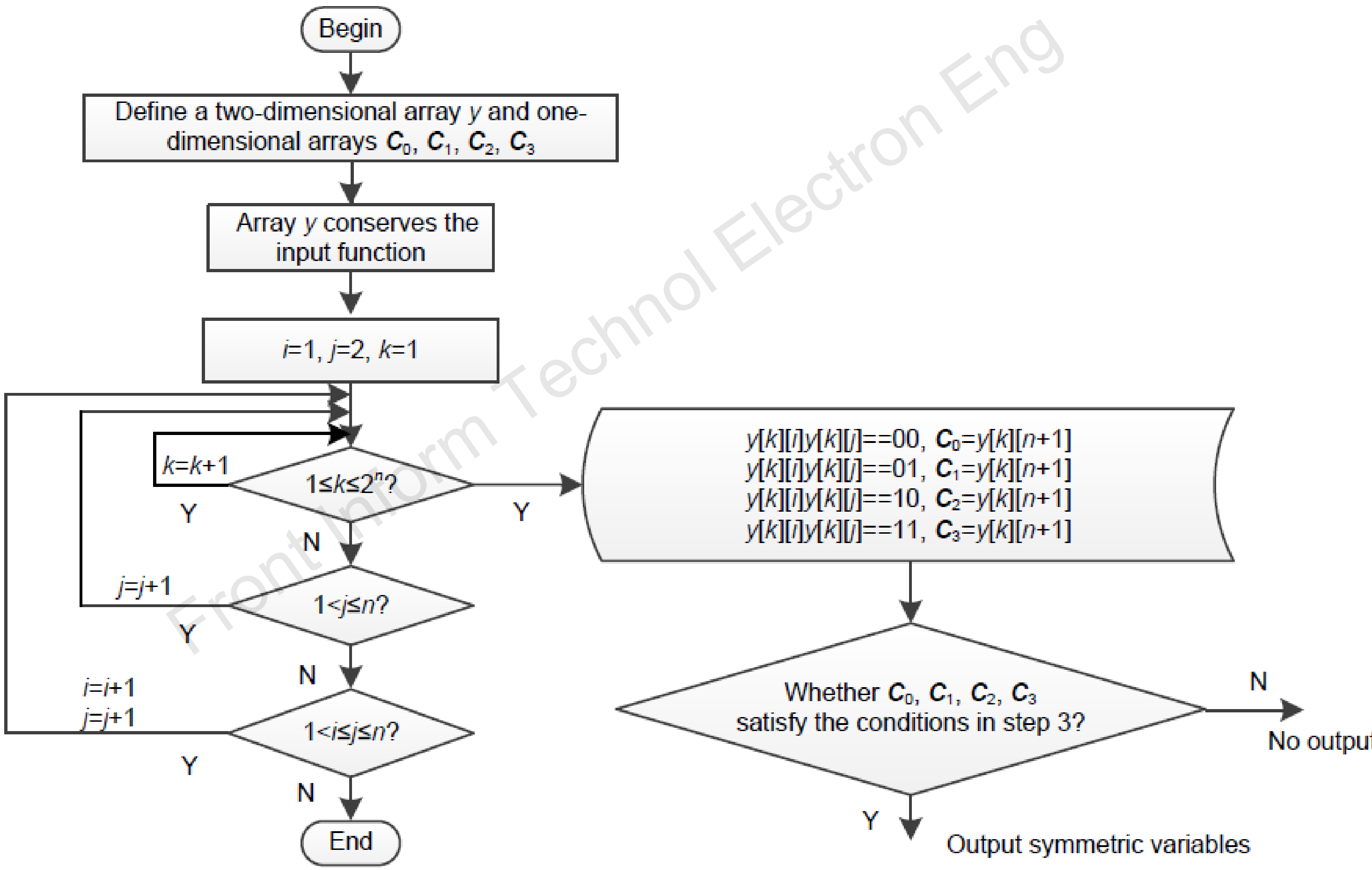

Fig. 1 Flowchart of the program

Major results

The new algorithm is an optimal detection method in terms of the applicability of the number of logic variables, the Boolean function including don't-care terms, detection type, and complexity of the identification process.

Table 11 Experimental results for the test function

Name	n	t	$E(x_i \| x_j)$	$N(x_i \| x_j)$	$S(x_i \| x_j)$	$S(x_i \| \overline{x}_j)$	$S(x_j \| x_i)$	$S(x_j \| \overline{x}_i)$	$CE(x_i \| x_j)$	$CN(x_i \| x_j)$	$CS(x_i \| x_j)$	$CS(x_i \| \overline{x}_j)$	$CS(x_j \| x_i)$	$CS(x_j \| \overline{x}_i)$
test1	3	1.12	0	1	0	1	0	0	0	2	1	2	0	2
test2	3	2.11	2	0	2	1	0	1	1	1	1	0	3	2
d	4	4.43	1	0	0	1	0	1	1	1	3	1	0	1
ae5	5	7.88	1	0	1	0	1	1	2	0	1	1	1	0
6tt	6	9.74	1	1	1	1	0	2	0	2	0	1	0	1
7fc	7	11.32	1	1	1	0	1	0	2	0	2	1	1	0
up8	8	15.77	2	1	0	1	1	0	2	1	2	1	1	0
prt	9	18.14	2	1	1	2	2	1	0	1	0	1	0	1
yy9	9	27.17	1	2	3	3	1	2	2	2	3	1	2	2
qov	10	29.16	1	1	2	1	1	0	1	1	1	1	2	1
ipt	11	31.54	1	0	1	2	1	1	1	1	1	0	1	1
sw1	12	33.08	1	2	0	2	1	1	2	1	2	2	2	2
shu	13	45.83	2	3	5	1	3	2	3	0	0	1	0	1
we2	14	58.77	3	2	3	4	1	6	1	1	4	1	1	3
iyr	15	69.88	4	2	3	5	6	2	1	1	5	6	1	1
qro	16	88.15	3	4	2	9	3	3	8	2	1	2	3	4
ll3	17	142.00	7	2	3	2	2	7	6	4	4	5	4	2
vc7	18	193.00	5	6	5	4	3	5	4	6	4	2	4	2

'name', 'n', and 't' express the test function, the number of logic variables in the test function, and the symmetry detection time (in seconds), respectively. Columns 4–15 give the statistical numbers of symmetric variables existing in the test function for the 12 types of symmetry. Test functions 'test1' and 'test2' are the same as those used in Examples 1 and 2