OSTIS-2013
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822:514

DECOMPOSIBALITY AND SEARCH FOR ALL SOLUTIONS OF A SYSTEM OF BOOLEAN FUNCTIONS

Taghavi S.A.* and Pottosin Y.V.*

* *United Institute of Informatics Problems of the NAS of Belarus, Minsk, Republic of Belarus*

{taghavi,pott}@newman.bas-net.by

Abstract: The problem of series two-block decomposition of completely specified Boolean functions is considered. Analysis and investigation of such systems are very important in logical design context. Recently, a method for solving this problem was suggested based on using the ternary matrix cover approach. Using this method a computer program developed. This paper is focused on decomposability of a system of Boolean functions. In decomposable systems, the number of solutions and time elapsed to achieve them was investigated.

Keywords: Boolean functions, decomposition, cover map, compact table

1. Introduction

The problem of decomposition of Boolean functions is one of the most important problems of logical design that makes it an object of great attention by many researchers in this field. The survey [Perkowski, 1995] shows a considerable number of papers are already published on this topic. It is important to find a successful solution for this problem because it has a direct influence on the quality and cost of digital devices designed. We consider the problem of decomposition of a system of Boolean functions in the following statement. A system of completely specified Boolean functions $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ is given where $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$. The superposition $\boldsymbol{y} = \varphi(\boldsymbol{w}, \boldsymbol{z}_2)$, $\boldsymbol{w} = g(\boldsymbol{z}_1)$ where $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are vector variables whose components are Boolean variables in the subsets Z_1 and Z_2 respectively that form a partition of the set $X = \{x_1, x_2, \ldots, x_n\}$ of arguments. At that, the number of components of the vector variable $\boldsymbol{w}$ must be less than that of $\boldsymbol{z}_1$. Such a kind of decomposition is called two-block disjoint decomposition by [Zakrevskij, 2009]. The subsets Z_1 and Z_2 are called bound and free sets respectively. Only a few papers deal with the search for the partition $\{Z_1, Z_2\}$, at which this problem has a solution**.** Among the papers considering this question, we can point out [Bibilo, 2009], [Jóźwiak, 2000], [Perkowski, 1995] and [Zakrevskij, 2007].

Searching for a solution of this kind is NP-complete problem because it has been proved that this problem equivalent with well-known set covering problem (SCP). While to be aware of decomposability of the given system of Boolean functions, finding only one such a pair is satisfying, due to analyzing of the task and search for the best solution we were motivated to find all possible solutions. For that we used the ternary matrix cover approach [Pottosin, 2010]. Using a compact table one can find rather easily the existence of a solution of the problem for a given system of functions, and if it does exist, the corresponding superposition can be easily found.

2. Definitions and setting the problem

Let a system of completely specified fnctions $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, where $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ and $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$, be given by matrices $\boldsymbol{U}$, $\boldsymbol{V}$ that are the matrix representation of the system of disjunctive normal forms (DNFs) of the given functions [Zakrevskij, 2009]. Matrix $\boldsymbol{U}$ is a ternary matrix of $l \times n$ dimension where l is the number of terms in the given DNFs. The columns of $\boldsymbol{U}$ are marked with the variables $x_1, x_2, \ldots, x_n$, and the rows represent the terms of the DNFs (the intervals of the space of the variables $x_1, x_2, \ldots, x_n$). The matrix $\boldsymbol{V}$ is a Boolean matrix. Its dimension is $l \times m$, and its columns are marked with the variables $y_1, y_2, \ldots, y_m$. The ones in this columns point out the terms in the given DNFs. A row $\boldsymbol{u}$ in $\boldsymbol{U}$ *absorbs* a Boolean vector $\boldsymbol{a}$ if $\boldsymbol{a}$ belongs to the interval represented by $\boldsymbol{u}$.

The task considered is set as follows. Given a system of completely specified Boolean functions $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, the superposition $\boldsymbol{y} = \varphi(\boldsymbol{w}, \boldsymbol{z}_2)$, $\boldsymbol{w} = g(\boldsymbol{z}_1)$ must be found where $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are vector variables whose components are Boolean variables in the subsets of the set $X = \{x_1, x_2, \ldots, x_n\}$, Z_1 and Z_2 respectively such that

$X = Z_1 \cup Z_2$ and $Z_1 \cap Z_2 = \varnothing$. At that, the number of components of the vector variable $\boldsymbol{w}$ must be less than that of $\boldsymbol{z}_1$. The main attention is paid to the search for subsets Z_1 and Z_2 such that the task would have a solution. It is clear that the subset Z_1 should have at least two members while the subset Z_2 can have only one.

3. INTRODUCING COVER MAP AND COMPACT TABLE

Any family π of different subsets (blocks) of a set L whose union is L, is called a *cover* of L. Let $L = \{1, 2, \dots, l\}$ be the set of numbers of rows of a ternary matrix $\boldsymbol{U}$. A cover π of L is called a *cover of the ternary matrix* $\boldsymbol{U}$ if for each value $\boldsymbol{x}^*$ of the vector variable $\boldsymbol{x}$ there exists a block in π containing all the numbers of those and only those rows of $\boldsymbol{U}$, which absorb $\boldsymbol{x}^*$. Block $\varnothing$ corresponds to the value $\boldsymbol{x}^*$, which is absorbed by no row of $\boldsymbol{U}$. Other subsets are not in π.

Let $t(x^*, U)$ be the set of numbers of those rows of U, which absorb x^*. For every block πj of π, we define the Boolean function $\pi j(x)$ having assumed that $\pi j(x^*) = 1$ for any $x^* \in \{0,1\}\pi$ if $t(x^*, U) = \pi j$, and $\pi j(x^*) = 0$ otherwise.

Let us define an operation $\vee(\pi i, V)$ over the rows of a binary matrix V, the result of which is the vector y^* ($y^* = \vee(\pi i, V)$) obtained by component-wise disjunction of rows V whose numbers are in the block πi. If $\pi i = \varnothing$, all the components of y^* are equal to 0. It is shown in [Pottosin, 2006] that $f(x^*) = y^* = \vee(\pi i, V)$ if $\pi i(x^*) = 1$.

There is a convenient way to construct the cover of a ternary matrix U when the number of arguments is not large. This technique uses the cover map that has the structure of the Karnaugh map. In any cell of a cover map of U corresponding to a vector x^*, there is the set $t(x^*, U)$, which is a block of the cover of U.

Let a pair of matrices, $\boldsymbol{U}$ and $\boldsymbol{V}$, give a system of completely specified Boolean functions $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, and let the matrix $\boldsymbol{U}_1$ be composed of the columns of $\boldsymbol{U}$, marked with the variables from the set $\boldsymbol{Z}_1$ and the matrix $\boldsymbol{U}_2$ from the columns marked with the variables from $\boldsymbol{Z}_2$. The covers of $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ are $\pi^1 = \{\pi^1_1, \pi^1_2, \dots, \pi^1_r\}$ and $\pi^2 = \{\pi^2_1, \pi^2_2, \dots, \pi^2_s\}$. Let us construct a table M. Assign the blocks $\pi^1_1, \pi^1_2, \dots, \pi^1_r$ and the Boolean functions $\pi^1_1(\boldsymbol{z}_1), \pi^1_2(\boldsymbol{z}_1), \dots, \pi^1_r(\boldsymbol{z}_1)$ to the columns of M, and $\pi^2_1, \pi^2_2, \dots, \pi^2_s$ and $\pi^2_1(\boldsymbol{z}_2), \pi^2_2(\boldsymbol{z}_2), \dots, \pi^2_s(\boldsymbol{z}_2)$ to the rows of M. At the intersection of the i-th column, $1 \le i \le r$ and the j-th row, $1 \le j \le s$, of M, we put the value $\boldsymbol{y}^* = \vee(\pi^1_i \cap \pi^2_j, \boldsymbol{V})$. The table M is called the *compact table*. It gives the system of Boolean functions $y = f(x)$ in the following way: the value of the vector Boolean function $f(x^*)$ is $\vee(\pi 1i \cap \pi 2j, V)$ at any set argument values x^*, for which $\pi 1i(z1) \wedge \pi 2j(z2) = 1$.

Having the compact table for a system of functions $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, it is easy to construct the desired systems $\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{w}, \boldsymbol{z}_2)$ and $\boldsymbol{w} = \boldsymbol{g}(\boldsymbol{z}_1)$. The columns of the compact table are encoded with binary codes; equal columns may have the same codes. The length of the code is equal to $\lceil \log_2 r' \rceil$ where r' is the number of different columns of the table and $\lceil a \rceil$ is the least integer, which is not less than a. So, the system of functions $\boldsymbol{w} = \boldsymbol{g}(\boldsymbol{z}_1)$ is defined. The value of the vector variable $\boldsymbol{w}$ at any set of values of the vector variable $\boldsymbol{z}_1$ turning the function $\pi^1_i(\boldsymbol{z}_1)$ into 1 is the code of the i-th column, $1 \le i \le r$. Naturally, there is no solution to this task at the given partition $\{Z_1, Z_2\}$ of the set X of arguments if the length of the code is not less than the length of $\boldsymbol{z}_1$. Otherwise, the compact table whose columns are assigned with the values of the variable $\boldsymbol{w}$ can be considered as a form of representation of the other desired system of functions $\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{w}, \boldsymbol{z}_2)$. The value of $\boldsymbol{y}$ at the value of $\boldsymbol{w}$ assigned to the i-th column, $1 \le i \le r$, and at any value of $\boldsymbol{z}_2$ turning $\pi^2_j(\boldsymbol{z}_2)$ into 1, $1 \le j \le s$, is the vector that is at the intersection of the i-th column and the j-th row [Pottosin, 2010].

Example 1. Let a system of completely specified functions $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ was given by the following pair of matrices:

$$\boldsymbol{U} = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & - \\ 0 & 1 & 0 & 0 & - \\ 0 & 1 & - & 0 & 1 \\ 0 & - & 0 & 0 & - \\ 0 & 0 & - & 0 & 1 \\ 1 & 1 & 0 & 1 & - \\ 1 & 1 & - & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} 1\\2\\3\\4\\5\\6\\7 \end{matrix}, \quad \boldsymbol{V} = \begin{array}{c} \begin{matrix} y_1 & y_2 \end{matrix} \\ \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{array} \begin{matrix} 1\\2\\3\\4\\5\\6\\7 \end{matrix}.$$

For the partition of the set of arguments into subsets $Z_1 = \{x_1, x_2, x_3\}$ and $Z_2 = \{x_4, x_5\}$, we have the following matrices:

$$\boldsymbol{U}_1 = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & - \\ 0 & - & 0 \\ 0 & 0 & - \\ 1 & 1 & 0 \\ 1 & 1 & - \end{bmatrix} \end{array} \begin{matrix} 1\\2\\3\\4\\5\\6\\7 \end{matrix}, \quad \boldsymbol{U}_2 = \begin{array}{c} \begin{matrix} x_4 & x_5 \end{matrix} \\ \begin{bmatrix} 1 & - \\ 0 & - \\ 0 & 1 \\ 0 & - \\ 0 & 1 \\ 1 & - \\ 1 & 1 \end{bmatrix} \end{array} \begin{matrix} 1\\2\\3\\4\\5\\6\\7 \end{matrix}.$$

To find the length of $\boldsymbol{w}$ in the superposition $\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{w}, \boldsymbol{z}_2)$, $\boldsymbol{w} = \boldsymbol{g}(\boldsymbol{z}_1)$ where $\boldsymbol{z}_1 = (x_1, x_2, x_3)$ and $\boldsymbol{z}_2 = (x_4, x_5)$, we construct the covers of the ternary matrices $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$: $\pi^1 = \{\varnothing, \{3\}, \{5\}, \{7\}, \{6, 7\}, \{1, 4, 5\}, \{2, 3, 4\}\}$ and $\pi^2 = \{\{1, 6\}, \{2, 4\}, \{1, 6, 7\}, \{2, 3, 4, 5\}\}$ (In continue we will discuss how to obtained this covers). The corresponding compact table is represented in Table 1 that has seven different columns.

Table 1 – The compact table for the system of functions in Example 1 with $\boldsymbol{z}_1 = (x_1, x_2, x_3)$ and $\boldsymbol{z}_2 = (x_4, x_5)$

	$\varnothing$	3	5	7	6,7	1,4,5	2,3,4
1,6	00	00	00	00	01	10	00
2,4	00	00	00	00	01	01	11
1,6,7	00	00	00	01	01	10	00
2,3,4,5	00	10	01	00	00	11	11

Clearly, this task has no solution at the given subsets Z_1 and Z_2, because to encode the columns of the compact table with the values of $\boldsymbol{w}$, three variables are needed that is not less than the length of $\boldsymbol{z}_1$.

4. SEARCH FOR APPROPRIATE PARTITION

To search for an appropriate partition of the set of arguments we use ternary matrix covers and compact tables induced by them. Let a few free variables be to find that constitute the set Z_2 (then the set of bound variables would be $Z_1 = X \setminus Z_2$). To do this, we use the operation of dividing a ternary matrix cover by the cover of a column of the matrix. Let us determine the operation to divide the cover π of a ternary matrix $\boldsymbol{U}$ by the cover π^i of its i-th column as:

$$\pi / \pi^i = \pi^1 \times \pi^2 \times \ldots \times \pi^{i-1} \times \pi^{i+1} \times \ldots \times \pi^n$$

This operation can be easily fulfilled using the *cover map*, which, as well as Karnaugh map, has the lines of symmetry related to the variables of the Boolean space represented by this map [Zakrevskij, 2007]. To transform the cover map of a ternary matrix $\boldsymbol{U}$ into that of the matrix obtained from $\boldsymbol{U}$ by deleting the i-th column, one should superpose pair-wise the entries that are symmetric with regard to the lines relative to x_i, and put the unions of the superposed entries into the obtained entries. The obtained cover map would represent the desired cover [Pottosin, 2010].

Example 2. Figure 1 shows the cover map of the ternary matrix $\boldsymbol{U}$ from Example 1.

x_5, x_4, x_3, x_2, x_1

4	∅	∅	1	1	∅	5	4,5
2,4	∅	∅	∅	∅	∅	3	2,3,4
∅	∅	∅	6	6,7	7	∅	∅
∅	∅	∅	∅	∅	∅	∅	∅

Figure 1 – The cover map of matrix $\boldsymbol{U}$ from Example 1

x_5, x_3, x_2, x_1

1,4	∅	5	1,4,5
2,4	∅	3	2,3,4
6	∅	7	6,7
∅	∅	∅	∅

Figure 2 – The cover map obtained by dividing π by the cover of the column x_4.

x_3, x_2, x_1

1,4,5	5
2,3,4	3
6,7	7
∅	∅

Figure 3 – The cover map obtained by dividing π by the covers of the column x_4 and x_5

The cover of $\boldsymbol{U}$ is π = {∅, {1}, {3}, {4}, {5}, {6}, {7}, {2,4}, {4,5}, {6,7}, {2,3,4}}. As it can be seen from Figure 2 the division of π by the cover of the column x_4 will be {∅, {3}, {5}, {6}, {7}, {1,4}, {2,4}, {6,7}, {1,4,5}, {2,3,4}}. Having transformed this map by the described way with regard to x_5, we obtain {∅, {3}, {5}, {7}, {6,7}, {1,4,5}, {2,3,4}} as a result of dividing π by the covers of the columns x_4 and x_5 (see Figure 3).

The method used for the search for an appropriate partition consists in fulfilling the lexicographical enumeration and testing by the above way every variant of the set Z_2 if it would provide a solution of the task.

Example 3. Let the system of completely specified Boolean functions from Example 1 be given. Consider this variant that $Z_2 = \{x_2, x_4\}$, $Z_1 = \{x_1, x_3, x_5\}$. For this variant with the cover map in Figure 1, we obtain the cover map shown in Figure 4 and then we obtain Figure 5 from Figure 4.

x_5, x_4, x_3, x_1

2,4	∅	∅	1	1	∅	3,5	2,3,4,5
∅	∅	∅	6	6,7	7	∅	∅

Figure 4 – The cover map obtained by dividing π by the cover of the column x_2

x_5, x_3, x_1

1,2,4	∅	3,5	1,2,3,4,5
6	∅	7	6,7

Figure 5 – The cover map obtained by dividing π by the covers of the column x_2 and x_4

The compact table for the covers π^1 = {∅, {6}, {7}, {3,5}, {6,7}, {1,2,4}, {1,2,3,4,5}} and π^2 = {{1}, {4,5}, {6,7}, {2,3,4}} is represented by Table 2 that have four different columns. To encode these columns, two variables are sufficient. The codes of the columns are shown at the bottom of Table 2.

Table 2 – The compact table for the partition from Example 3

	∅	6	7	3,5	6,7	1,2,4	1,2,3,4,5
1	00	00	00	00	00	10	10
4,5	00	00	00	01	00	01	01
6,7	00	01	01	00	01	00	00
2,3,4	00	00	00	10	00	11	11
	00	01	01	10	01	11	11

To construct the system of functions $\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{w}, \boldsymbol{z}_2)$ and $\boldsymbol{w} = \boldsymbol{g}(\boldsymbol{z}_1)$ that are the solution of the task, the functions connected with the blocks of the covers obtained must be constructed. The DNFs of the functions connected with the blocks of π^1 can be obtained from the cover map in Figure 5: $\pi^1{}_1(\boldsymbol{z}_1) = x_3\,\bar{x}_5$, $\pi^1{}_2(\boldsymbol{z}_1) = x_1\,\bar{x}_3\,\bar{x}_5$, $\pi^1{}_3(\boldsymbol{z}_1) = x_1\,x_3\,x_5$, $\pi^1{}_4(\boldsymbol{z}_1) = \bar{x}_1\,x_3\,x_5$, $\pi^1{}_5(\boldsymbol{z}_1) = x_1\,\bar{x}_3\,x_5$, $\pi^1{}_6(\boldsymbol{z}_1) = \bar{x}_1\,\bar{x}_3\,\bar{x}_5$, $\pi^1{}_7(\boldsymbol{z}_1) = \bar{x}_1\,\bar{x}_3\,x_5$.

Similarly, the DNFs $\pi^2{}_1(\mathbf{z}_2) = \bar{x}_2\, x_4$, $\pi^2{}_2(\mathbf{z}_2) = \bar{x}_2\, \bar{x}_4$, $\pi^2{}_3(\mathbf{z}_2) = x_2\, x_4$, $\pi^2{}_4(\mathbf{z}_2) = x_2\, \bar{x}_4$ are obtained. As a result of simple minimization we obtain the following matrices representing the desired superposition $\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{w}, \mathbf{z}_2)$, $\boldsymbol{w} = \boldsymbol{g}\,(\mathbf{z}_1)$:

$$\begin{array}{cccc} w_1 & w_2 & x_2 & x_4 \end{array} \quad \begin{array}{cc} y_1 & y_2 \end{array} \qquad \begin{array}{ccc} x_1 & x_3 & x_5 \end{array} \quad \begin{array}{cc} w_1 & w_2 \end{array}$$

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & - & 1 & 0 \\ 1 & - & 0 & 0 \\ 1 & 1 & - & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}; \qquad \begin{bmatrix} 0 & 0 & - \\ - & 0 & 0 \\ 1 & - & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

5. IMPLEMENTATION AND RESULTS

We designed and developed special computer program on MSVC++ to find all solutions of systems of Boolean functions. Our program based on ternary matrix cover approach and the general scheme of our algorithm summarized in Figure 6. The experiments run on a Pentium 2.26GHz CPU with 3 GByte of main memory. As a benchmark, we generate many systems of completely specified Boolean functions using a prepared library explained in [Romanov, 1997], [Romanov, 2001] and [Romanov, 2005]. We considered three parameters for these systems; number of rows of matrix *U* that indicate number of conjunctions, number of columns of matrix *U* or number of arguments and number of columns of matrix *V* or number of functions. For every system and after generating matrices *U* and *V* as SOP (Some Of Product), first of all we expand matrix *U* to obtain corresponding matrix without don't cares. The rows which have don’t cares will replace with several suitable rows and consequently number of DNFs of system will increase exponentially.

Then we begin to provide cover map; for that we used gray code encoding system. On contrary to our example in section 4 that cover map is a two dimensional table, due to simplicity to store in computer memory and referring for it and also in the future calculations of compact table, we implemented it as a one dimensional array. An example of our approach with three variables is represented in Figure 5. The order of replacement of the variables on the array is important and this can be extended for any number of variables.

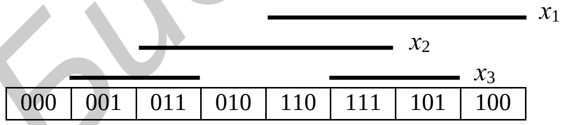

Figure 5 – The cover map model for three variables using gray code encoding system

In the array we store the values explained in section 4 and gray codes in the array of Figure 5 are symbolically shown to represent the correctness of the approach, but we also save the gray codes in the other list. In fact, the method of storing information in the mentioned array is as follows. Each row in matrix *U*, numbered with integers started from one. We compare the value of each row with gray codes list until the equal value to be founded. Then we add the row number of compared row in corresponding element of the array. We continue this way until all rows to be compared and the row numbers to be added to the array elements. At the final, we sweep the array and put the empty set to the elements with no value added.

```
-Algorithm for one system of Boolean
              functions
➢ Con ← ConValue
➢ Arg ← ArgValue
➢ Fun ← FunValue
➢ Generate SOP(Con,Arg,Fun)
  (i.e Matrices U and V)
➢ Expand Matrix U
  (i.e removing all don't cares in
  Matrix U with replacing the
  according rows with new ones)
➢ Compute Cover Map
  (Generate Gray Codes with Length 2^n
  and fill out the Cover Map Array
  According to the Algorithm Rules)
➢ C ← 0 (To Count Number of Solutions)
➢ for k ← 2 to n-1
     »Combination Generator(n,k)
     »for each combination of (n k)
          Check Current Partition
           1-Divide Cover Map Over Z1
           2-Divide Cover Map over Z2
           3-Compute Compact Table
           4-Compute Number of
             Different Columns (r) of
             Compact Table
           5-Encode the Columns of the
             Compact Table
           6-if log(r)≤ 2^(k-1) then
               Solution Founded
               (Produce the Solution
               i.e Matrices φ, w, y and x)
      »if Solution Founded then
          Add this partition to the set
          of Solutions and C ← C + 1
➢ if C=0 then
    Declare the system is not decomposable
➢ else
  Print C (which is the number of all
  solutions)
```

Figure 6 – Implemented algorithm for determining decomposability of system of Boolean Functions and find total number of solutions of system

To find all solutions of the task anyone should enquire into all possible partitions which constructing Z_1 and Z_2. The relatively simple method to address the appropriate partition can be done by lexicographical enumeration. After computing cover map of the current system of Boolean functions, in each stage we used Donald E. Knuth algorithm [Knuth, 2011] to generate all combinations of the arguments and of course for each partition we check whether it is a solution of the task or not. To obtain all *k*-element subsets of an *n*-element set, this algorithm is one of the fastest ones. Each *k*-element subsets is used to construct Z_1 elements and rest of the arguments will be the elements of the Z_2.

If a partition as a solution found, the program will keep it and will calculate four matrices; Matrix $\boldsymbol{\Phi}$, Matrix $\boldsymbol{Y}$, Matrix $\boldsymbol{X}$ and Matrix $\boldsymbol{W}$. These matrices are the solution of the task. In fact the current system of Boolean functions convert to two new systems with less arguments; Matrices $\boldsymbol{\Phi}$ and $\boldsymbol{Y}$ as $\boldsymbol{U}$ and $\boldsymbol{V}$ respectively, for the first system and also Matrices $\boldsymbol{X}$ and $\boldsymbol{W}$ as $\boldsymbol{U}$ and $\boldsymbol{V}$ respectively, for the second system. Although we obtain these matrices but they haven't influence in our results in this paper. We'll utilize them in the future works.

This manner is repeated for all partitions and if appropriate partition wasn't found, the program will declare the current system of Boolean functions isn't decomposable; otherwise the program will print number of solutions to the current system.

Now, we report experimental results for our approach in decomposition of Boolean functions, described in the previous sections. Due to space and time limitations, the results are shown refer only to the decompositions of systems with few arguments and too few functions. The results summarized in Table 3.

Table 3 – Experimental results

Con	Arg	Fun	NTP	NS	PS	ET
8	5	2	25	21	84	<1
10	6	2	56	49	87	4
10	6	3	56	13	23	5
10	6	4	56	18	32	4
20	7	3	119	25	21	21
15	8	3	246	64	26	131
15	8	4	246	40	16	134
20	8	5	246	8	3	104
40	8	10	246	8	3	201
20	9	4	501	48	10	497
30	9	5	501	10	2	688
30	10	3	1012	58	6	3634
30	10	5	1012	55	5	3197
40	10	4	1012	98	10	3100
30	12	6	4082	673	16	20413

The results show that more than 95% of generated systems are decomposable and all of them have several solutions when the system is decomposable. The first three columns in Table 3 represent the number of conjunctions (Con), number of arguments (Arg) and number of functions (Fun) respectively and these informs the parameters of a generated system of Boolean functions. The number of total partitions (NTP) counted when $2 \le |Z_1| \le n-1$. So it implies that the total partitions will be $\sum_{k=2}^{n-1}\binom{n}{k}$ which it is equal to $2^n-(n+2)$. Number of Solutions (NS) is part of results which it is found after the program was executed and the percentage of the Solutions (PS) is percent of NS to NTP. The last column represents elapsed time (ET) which is the running time of the program for each system of Boolean functions during obtaining all solutions and calculated in seconds.

CONCLUSION AND FUTURE WORK

We developed a computer program as an application to determine decomposability of system of Boolean functions via ternary matrix cover approach. The ternary matrix cover and the representation of a system of Boolean functions in the form of compact table are simple to realize and we implemented several systems with different parameters. Experimental results were interesting and show that usually a system has more solutions when it is decomposable. In the most cases the number of solutions will be high when the number of functions is small.

As a future work, optimization in encoding of compact table is proposed, because it has direct influence on quality of obtained solutions. Also it is useful to find a best partition among the solutions from the syntheses point of view which is useful in practical scene.

REFERENCES

[Pottosin, 2010] Pottosin, Yu.V., Shestakov, E., "Choice of Free Arguments in Decomposition of Boolean Functions Using the Ternary Matrix Cover Approach", *In Proceeding of the 6th International Conference on Neural Networks and Artificial Intelligence ICNNAI*, Brest, Belarus; BSTU, 2010, pp. 123-127.

[Bibilo, 2009] Bibilo, P.N., "Decomposition of Boolean Functions Based on Solving Logical Equations", *Byelaruskaya Navuka*, Minsk, Belarus, 2009, (In Russian).

[Jóźwiak, 2000] Jóźwiak, L. and Chojnacki A., "An Effective and Efficient Method for Functional Decomposition of Boolean Functions Based on Information Relationship Measures", *In Proceeding of 3rd Design and Diagnostics of Electronic Circuits and Systems Workshop, DDECS*. Bratislava: Institute of Informatics, Slovak Academy of Sciences, pp. 242-249, 2000.

[Perkowski, 1995] Perkowski, M.A., Grygiel S., "A Survey of Literature on Functional Decomposition. Version IV", *Technical report*, Department of Electrical Engineering, Portland State University, Portland, USA, 1995.

[Pottosin, 2006] Pottosin, Yu.V., Shestakov E.A., "Tabular Methods for Decomposition of Systems of Completely Specified Boolean Functions", *Byelorusskaya Nauka*, Minsk, Belarus, 2006, (In Russian).

[Zakrevskij, 2007] Zakrevskij, A.D, "Decomposition of Partial Boolean Functions: Testing for Decomposability According to a Given Partition", *Informatika Journal*, No. 1(13), pp. 16-21, 2007, (In Russian).

[Zakrevskij, 2009] Zakrevskij A., Pottosin Yu. V., Cheremisinova L., "Optimization in Boolean Space", *Tallinn: TUT Press*, 2009.

[Romanov, 1997] Romanov, V.I. and Vasilkova I.V., "Boolean Vectors and Matrices in C++", *Logic Design, Institute of Engineering Cybernetics of NASB*, Minsk, Belarus, pp. 150-158, 1997, (In Russian).

[Romanov, 2001] Romanov, V.I., "Tools development for logic designing", *Logic Design, Institute of Engineering Cybernetics of NASB*, Minsk, Belarus, pp. 151-170, 2001, (In Russian).

[Romanov, 2005] Romanov, V.I., "Tools for programming Boolean calculations", *Combinatorics for modern manufacturing, logistics and supply chains, Abstracts of the XVIII European Conference*, UIIP, NASB, Minsk, Belarus, C.57–58, 2005.

[Knuth, 2011] Knuth, D. E., "The Art of Computer Programming", Vol. 4A, Combinatorial Algorithms, part 1, Addison-Wesley Professional, ISBN-0201038048, 2011.

РАЗЛОЖИМОСТЬ И ПОИСК ВСЕХ РЕШЕНИЙ СИСТЕМЫ БУЛЕВЫХ ФУНКЦИЙ

Тагави С.А.*, Поттосин Ю.В.*

** Объединенный институт проблем информатики НАН Беларуси, г. Минск, Республика Беларусь*

{taghavi,pott}@newman.bas-net.by

Аннотация. Рассматривается задача последовательной двухблочной декомпозиции полностью определенных булевых функций. Анализ и исследование таких систем является весьма важным для логического проектирования. Ранее был предложен метод решения этой задачи, основанный на использовании покрытия троичной матрицы. Разработана компьютерная программа, использующая этот метод. Основное внимание данной работы сосредоточено на разложимости систем булевых функций. Для разложимых систем исследовалось число решений и время, затрачиваемое на их получение.

Ключевые слова: булева функция, декомпозиция, карта покрытия, компактная таблица.

Введение

Рассматривается задач а декомпозиции системы булевых функций в следующей постановке. Пусть система полностью определенных булевых функций $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, где $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ и $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$ задана матрицами $\boldsymbol{U}$ и $\boldsymbol{V}$, представляющими систему дизъюнктивных нормальных форм (ДНФ) заданных функций. Матрица $\boldsymbol{U}$ является троичной матрицей размерности $l \times n$, где l – число различных элементарных конъюнкций в заданной системе ДНФ. Столбцы матрицы $\boldsymbol{U}$ помечены переменными $x_1, x_2, \ldots, x_n$, а строки представляют упомянутые элементарные конъюнкции (интервалы пространства переменных $x_1, x_2, \ldots, x_n$). Матрица $\boldsymbol{V}$ является булевой матрицей размерности $l \times m$, и ее столбцы помечены переменными $y_1, y_2, \ldots, y_m$. Требуется найти суперпозицию $\boldsymbol{y} = \varphi(\boldsymbol{w}, \boldsymbol{z}_2)$, $\boldsymbol{w} = g(\boldsymbol{z}_1)$, где $\boldsymbol{z}_1$ и $\boldsymbol{z}_2$ – векторные переменные, компонентами которых являются булевы переменные из подмножеств Z_1 и Z_2 соответственно, образующих разбиение множества $X = \{x_1, x_2, \ldots, x_n\}$ аргументов такое, что $X = Z_1 \cup Z_2$ и $Z_1 \cap Z_2 = \varnothing$. При этом число компонент у векторной переменной $\boldsymbol{w}$ должно быть меньше чем у $\boldsymbol{z}_1$ [Zakrevskij, 2009].

Основная часть

Разработана компьютерная программа нахождения всех решений рассматриваемой задачи для заданной системы булевых функций. В качестве примеров генерировались многие системы полностью определенных булевых функций с помощью программ, описанных в работах [Romanov, 2001] и [Romanov, 2005]. Менялись три параметра этих систем: число элементарных конъюнкций, число аргументов и число функций. Затем строилась матрица покрытия, в которой использовался код Грея и которая представлялась как одномерная строка.

Для нахождения всех решений необходимо рассматривать все возможные разбиения, образованные множествами Z1 и Z2. Сравнительно простой метод обращения к подходящему разбиению использует лексикографический перебор. После получения матрицы покрытия для текущей системы булевых функций на каждом этапе генерируются все сочетания аргументов, а для каждого разбиения проверяется, имеется ли для него решение или нет. Если для разбиения существует решение, программа его сохраняет. Это повторяется для всех разбиений, и если подходящее разбиение не найдено, то программа объявляет текущую систему булевых функций неразложимой. Общая схема алгоритма представлена на рис. 6.

Результаты, приведенные в табл. 3, показывают, что сгенерированные системы имеют несколько решений, если они разложимы. Первые три столбца в табл. 3 представляют соответственно число элементарных конъюнкций (Con), число аргументов (Arg) и число функций (Fun). Общее число разбиений (NTP) подсчитано, когда $2 \leq |Z_1| \leq n - 1$. Этим числом будет $\sum_{k=2}^{n-1}\binom{n}{k}$, что равно $2^n - (n + 2)$. Частью результатов является число решений (NS), получаемое в результате выполнения программы, и доля NS в процентах (PS) по отношению к NTP. Последний столбец представляет время получения всех решений (ET) в секундах.

Заключение и дальнейшие работы

Разработана компьютерная программа определения разложимости систем булевых функций, использующая подход, связанный с покрытием троичной матрицы. Исследовано несколько систем с различными параметрами. Определенный интерес представляют экспериментальные результаты, показывающие, что если система разложима, то рассматриваемая задача имеет несколько решений. В большинстве случаев число решений велико при малом числе функций. В качестве дальнейшей работы предполагается исследовать оптимизацию кодирования компактной таблицы, поскольку это непосредственно влияет на качество получаемого решения. Полезно также в практическом плане с точки зрения синтеза логических схем находить наилучшее разбиение множества аргументов.