SUBOPTIMAL DECOMPOSITION OF BOOLEAN FUNCTIONS

Taghavi Afshord S., Pottosin Y.
United Institute of Informatics Problems, National Academy of Sciences of Belarus
Minsk, Republic of Belarus
E-mail: {taghavi, pott}@newman.bas-net.by

Decomposition of Boolean functions is an effective technique to obtain the most compact circuits. The ternary matrix cover approach is a powerful method in two-block disjoint decomposition. Before constructing the desired superposition, this method needs to encode a table. To find a better solution, it is important to use a particular approach for encoding, because its result has a direct influence on the obtained functions. We present an efficient algorithm to encode this table. It uses the approach connected with the assembling Boolean hyper cube method. The benefits and impacts of the suggested technique are discussed.

Introduction

To decompose a system of completely specified Boolean functions, first of all, one should search for an appropriate partition [1]. The suitable partition is supposed that already has been prepared. The goal of the current paper is minimizing the total number of terms in disjunctive normal forms (DNFs) of the obtained superposition. This minimization will decrease the size of a PLA (Programmable Logic Array) which is practically important [2].

Various methods for decomposition of Boolean functions are based on representations of functions. A Boolean function can be given in the form of the compact table [3], [4] that is a two-dimensional table as a Karnaugh map or decomposition chart, but may have the less size. The ternary matrix cover approach for decomposition of a system of Boolean functions is used [4]. Cover map and compact table are the key features of this approach. To obtain the systems of Boolean functions as a result of decomposition, one must encode the columns and rows of the compact table according to the obtained covers of the ternary matrices. They should be encoded by values of the subsets of arguments separated in a certain way.

In addition, each column should be encoded by binary codes in optional manner. But the result of this encoding and consequently its method has the direct influence on the obtained superposition. A novel method for encoding of the columns is suggested that leads to lowering the number of terms of DNFs. The encoding process in the suggested method is described as a constructing an n-dimensional Boolean hypercube. It is like assembling a simple mechanical structure. Here n is related to the number of distinguished columns of the compact table. The formal definition of the problem is 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 Z_1 and Z_2 of the set $X = \{x_1, x_2, \ldots, x_n\}$ of the arguments, respectively such that $X = Z_1 \cup Z_2$ and $Z_1 \cap Z_2 = \emptyset$. 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 [2], [5].

I. Constructing a Boolean hypercube

The process of constructing the Boolean hypercube can be represented as the sequence of n steps. At the sth step, the set of $(s-1)$-dimensional hypercubes are considered. They join into pairs, and s-dimensional hypercube is obtained from each pair by adding edges properly. After n steps, an n-dimensional Boolean hypercube will be constructed. The n-component Boolean vectors are assigned to the vertices of the hypercube where the neighborhood relation between the vectors should be represented by the edges of the hypercube.

At the first step of this process, 1-dimensional hypercubes in the form of $2^n - 1$ nonadjacent edges are composed of 0-dimensional hypercubes which is represented by 2^n isolated vertices. At the last, nth step, an n-dimensional hypercube is assembled from two $(n-1)$-dimensional ones by adding $2^n - 1$ edges.

Let us consider the formation of s-dimensional hypercubes on sth step more in details. The form of representation of hypercubes is very important here. Any k-dimensional hypercube is represented by a sequence S of 2^k vertex indices which is taken from the set $\{1, 2, \ldots, 2^n\}$. The edges are specified implicitly; two vertices are connected with an edge if and only if their places in S correspond to the places of neighbor codes in the Gray codes sequence of the same length as the length of S.

Before the performance of any step, the number of hypercubes is always even. More exactly, for sth step it is equal to 2^{n-s}, $0 \leq s < n$. The current situation is that some set C_s of s-dimensional hypercubes ($C_s = \emptyset$ before the performance of the step) and some set C_{s-1} of $(s-1)$-dimensional hypercubes exists. All pairs of hypercubes from C_{s-1} are looked through and one of them is chosen. The suitable edges are added to form s-dimensional hypercube from this pair that is introduced to C_s, and then the pair is removed from C_{s-1}. The performance of this step is accomplished when $C_{s-1} = \emptyset$.

The criterion for coupling two hypercubes is as follows. Let w_{ij} be a function taking its values from the set of pairs of distinguished columns of the compact table. The values of compact table are binary codes. So, for each pair of distinguished columns, the number of different peer-to-peer bits is calculated. It is clear that the value of w_{ij} will be from the set of positive integers. For two $(s-1)$-dimensional hypercubes which have been represented by sequences S' and S'', the sum $\sum w_{ij}$ is calculated. The summing is performed over all pairs i, j of indices of the vertices that take the same places in the sequences. This sum varies with permutations of vertices of one of the sequences, say S''. Of course, only those permutations may be taken here into consideration, which preserve the adjacency relation among the vertices.

For all the proper permutations, $\sum w_{ij}$ is calculated. Then according to its minimum value, the equivalent pair of hypercubes is chosen. They are joined into an s-dimensional hypercube by edges between vertices that are in the related places of S' and S''. The sequence that represents the composed hypercube is formed by concatenation of the sequences S' and S''. The sequence S'' may be changed its order according to the selected permutation before the concatenation. The number of feasible permutations for each hypercube in sth step is $(s-1)! \cdot 2^{s-1}$.

II. The process of encoding

The final process to encode the compact table is encoding of the constructed hypercube. At first, the basic encoding is obtained by using Karnaugh map. Then, a supplementary improvement is made on the basic encoding if it is needed and it can be possible. It means, sometimes it is possible to achieve more efficient encoding with additional efforts. In case the current hypercube contains some virtual vertices, it would be done.

So, it is searched for a feasible version of the encoding vector which assigns the maximum possible 1's for the codes of the virtual vertices. A feasible version is related to such permutations keeping the vicinity relations. The operations of searching for the feasible version are similar to the hypercube construction. Unfortunately, it is not possible always to find the optimal version, especially when the number of virtual vertices is more. But the basic encoding is still improved.

III. To obtain the desired superposition

Now, to obtain the solution of the task, the systems of functions $\boldsymbol{y} = \varphi(\boldsymbol{w}, \boldsymbol{z}_2)$ and $\boldsymbol{w} = g(\boldsymbol{z}_1)$ should be constructed [3], [4]. For that, the functions connected with the blocks of the cover maps of Z_1 and Z_2 must be obtained. Then the DNFs of the functions connected with the blocks of Z_1 and Z_2 will be calculated. Finally a minimization with the well-known Espresso logic minimizer is performed on the obtained superposition.

The method of Boolean hypercube encoding is used to establish a better superposition. Here, a better solution is related to the size of PLA, and it is specified by the number of rows of the matrices representing decomposed systems. The smaller number of the rows of each matrix implies a better solution of the task.

As we tested the suggested method for the several benchmarks, in the most cases, the numbers of the rows of the matrices obtained by the hypercube encoding method were less than the numbers of rows of the matrices obtained by a trivial encoding method. This reduction is expected to be more when increasing the number of the arguments of DNFs of a given system. Indeed, the comparable work has been done on the problem of an optimal state assignment of a finite state machine (FSM) [6]. Those results are confirmed our method as well.

Conclusion

The ternary matrix cover approach is an efficient technique for the problem of decomposition of systems of Boolean functions. The encoding of the compact table columns has a direct influence on the quality and cost of the designing of the digital devices. So, its optimization will cause a significant improvement on the obtained solution. In this paper, we suggested the assembling of Boolean hypercube for the encoding of the columns. This encoding improves the desired superposition and reduces the size of PLA which is important in the practical applications.

1. Muthukumar, V. An efficient variable partitioning approach for functional decomposition of circuits / V. Muthukumar, R. J. Bignall, H. Selvaraj // Journal of Systems Architecture. – 2007. – Vol. 53, No. 1. – P. 53–67.
2. Hassoun, S. Logic Synthesis and Verification. The Springer International Series in Engineering and Computer Science / S. Hassoun, T. Sasao. – Kluwer Academic Publishers, 2001. – 454 p.
3. Pottosin, Yu. V. Tabular Methods for Decomposition of Systems of Completely Specified Boolean Functions / Yu. V. Pottosin, E. A. Shestakov. – Minsk: Byelorusskaya Nauka, 2006. – 327 p. (In Russian)
4. Taghavi Afshord, S. On decomposing systems of Boolean functions via ternary matrix cover approach / S. Taghavi Afshord, Yu. V. Pottosin // International Journal of Advanced Science and Technology. – 2013. – Vol. 55, No. 6. – P. 33–42.
5. Zakrevskij, A. Optimization in Boolean Space / A. Zakrevskij, Yu. Pottosin, L. Cheremisinova. – Tallinn, Estonia: TUT Press, 2009. – 242 p.
6. Pottosin, Yu. V. State assignment of a finite state machine for a low power implementing circuit / Yu. V. Pottosin, S. A. Pottosina // Digital Technologies 2011. 8^{th} International Conference, November 10-11, 2011, Zilina, Slovak Republic. – Zilina: University of Zilina, 2011. – P. 113-116.