

# Experimental Verification of the Entropy Based Method for Minimization of Switching Functions on Pseudo Ternary Decision Trees

S. N. Yanushkevich,<sup>\*</sup> V. P. Shmerko,<sup>†</sup> R. S. Stanković,<sup>¶</sup>  
P. Dziurzanski,<sup>‡</sup> D. V. Popel<sup>‡</sup>

<sup>\*†‡</sup> Department of Computer Aided Design Systems, Faculty of Computer Science & Information Systems,  
Technical University of Szczecin, POLAND, januszki@wi.t.univ.szczecin.pl

<sup>¶</sup> Department of Computer Science, Faculty of Electronic, University of Niš, 18 000 Niš, YUGOSLAVIA, stanko@503c1.elfak.ni.ac.yu

<sup>‡</sup> Department of Computer Science, State University of Informatics & Radioelectronics, Minsk, BELARUS, popel@bseu.minsk.by

*Abstract— We present a new entropy based method for minimization of sum-of-product (SOP) expressions of switching functions. Unlike the recent results which utilize binary Decision Trees (DTs), we study the minimization procedure as a heuristic search based on information measures on the Free pseudo ternary DTs. The main contribution of this paper is an experimental justification of the entropy based method that yields the extreme improvements compared to some recent results on application of information theory methods for SOP minimization of switching functions. In most cases, we obtained that the quality of minimization with our minimizer InfoMin-2 is the same as that produced by ESPRESSO package, but is often performed faster.*

*Keywords— Switching functions, Entropy, Minimization, Decision tree*

## I. INTRODUCTION

The entropy based strategies are effective when the tasks to be solved formalized as the conversion of decision tables into DTs. Decision tables are used in CAD to specify which action must be taken for any condition in some exhaustive set of conditions. There are some approaches to design an optimal DT from a given decision table [5].

<sup>\*</sup> Research supported in parts from the St. Francis Xavier University (Dept. of Computer Science), Antigonish, Nova Scotia (CANADA) and in parts from the Committee of Scientific Research (POLAND) is acknowledged

<sup>†</sup> The author gratefully acknowledges the facilities provided as James Chair Professor from the St. Francis Xavier University, Antigonish, Nova Scotia (CANADA), support in parts from the Committee of Scientific Research (POLAND) is acknowledged

<sup>‡</sup> Support from the Found of Fundamental Researches of Republic of Belarus and Technical University of Szczecin (POLAND) is acknowledged

Among them, we note the branch-and-bound technique and dynamic programming methods that are the exhaustive search methods. Decomposition allows to reduce the search space, however, the methods do not guarantee the optimal solution.

Authors of [6] were the first who have applied the information theoretic criterion to convert decision tables with don't cares into near-optimal DTs. The truth table of a logic function is considered as a special case of a decision table with variables replacing the tests in the decision table. The technique to convert the decision table into a DT and its optimization has been applied for logic function minimization in [7], [8]. The search for the best implicants to appear in the final solution is realized by using DT, so that Shannon expansion formula  $\bar{x} \cdot f|_{x=0} \oplus x \cdot f|_{x=1}$  is applied at the each node in the DT.

In our recent papers [3] and [16] we gave a justification for application of the information measure for switching and multiple-valued functions. We showed that in the simplest understanding, the information theory methods give the same results as the probabilistic approach.

In our best knowledge, until now, the entropy based approach was studied only for binary DTs. In this paper we consider SOP minimization through ternary DTs and present our new program InfoMin-2 that uses the pseudo ternary DTs instead DTs with two-output Shannon nodes used in our previous program InfoMin-1 [15]. The nodes of such a DT have three outputs like Sasao's ternary trees [12], [13], [14], but the third edge assigns don't care value

with respect to Morreale's operator [10], [11] for generating the prime-irredundant cubes.

## II. BASICS

The decomposition of switching functions known as classical Shannon expansion has been introduced in 1938<sup>1</sup>. In 1948, Shannon suggested the information entropy as a measure to represent the value of information in numerical values.<sup>2</sup> We recall here some basic statements from the Shannon's information theory taken from some classical books, see, for example [1].

Let  $A = \{a_1, a_2, \dots, a_n\}$  is a complete set of events with the probabilities distribution  $\{p(a_1), p(a_2), \dots, p(a_n)\}$ . The *entropy* of the finite field  $A$ , is given by  $H(A) = -\sum_{i=1}^n p(a_i) \cdot \log p(a_i)$ , where logarithm is for the base 2. The entropy can never be negative, i.e.,  $\log p(a_i) \leq 0$ , and thus  $H(A) \geq 0$ . The entropy is zero if and only if  $A$  contains one event only.

*Example 1:* Let us consider a switching function that takes the value 1 with the probability  $p_1$  and the value 0 with the probability  $p_0$ . Then, (i) the entropy is minimal, i.e.,  $H(A) = 0$  *bit/pattern* when  $p_0 = 0$  or  $p_0 = 1$ , and (ii) the entropy reaches its maximum  $H(A) = 1$  *bit/pattern* when  $p_0 = p_1 = 0,5$ . ■

$x_1 x_2 x_3$	$f$	$x_1 x_2 x_3$	$f$
000	1	100	1
001	0	101	1
010	1	110	1
011	1	111	0

TABLE I

TRUTH TABLE OF A 3-VARIABLES SWITCHING FUNCTION.

*Example 2:* Let us calculate the entropy for the completely specified function given in Table I. We obtain  $H(f) = -0,25 \cdot \log 0,25 - 0,75 \cdot \log 0,75 = 0,8113$  *bit/pattern*,  $H(x_1) = H(x_2) = H(x_3) = -0,5 \cdot \log 0,5 - 0,5 \cdot \log 0,5 = 1$  *bit/pattern*, i.e., uncertainty in the variables of this function is the same. ■

Let  $A$  and  $B$  be finite fields of events with the probabilities distribution  $\{p(a_i)\}, i = 1, 2, \dots, n$ , and  $\{p(b_j)\}, j = 1, 2, \dots, m$ , respectively. The *conditional entropy* of the finite field  $A$  with respect to  $B$  is defined by  $H(A|B) = -\sum_{i=1}^n \sum_{j=1}^m p(a_i, b_j) \cdot \log \frac{p(a_i, b_j)}{p(b_j)}$ .

*Example 3:* The conditional entropy of the function  $f$  given in Table I with respect to the variable  $x_1$  is  $H(f|x_1) = -0,125 \cdot \log \frac{0,125}{0,5} - 0,125 \cdot \log \frac{0,125}{0,5} - 0,375 \cdot \log \frac{0,375}{0,5} - 0,375 \cdot \log \frac{0,375}{0,5} = 0,8113$  *bit/pattern*. By the analogy,  $H(f|x_2) = -0,125 \cdot \log \frac{0,125}{0,5} - 0,125 \cdot \log \frac{0,125}{0,5} - 0,375 \cdot \log \frac{0,375}{0,5} - 0,375 \cdot \log \frac{0,375}{0,5} = 0,8113$  *bit/pattern*, and  $H(f|x_3) = -0 \cdot \log \frac{0}{0,5} - 0,25 \cdot \log \frac{0,25}{0,5} - 0,5 \cdot \log \frac{0,5}{0,5} - 0,25 \cdot \log \frac{0,25}{0,5} = 0,5$  *bit/pattern*. So, the variable  $x_3$  conveys more information than the variables  $x_1$  and  $x_2$ . ■

<sup>1</sup> *Trans. AIEE, Vol. 27, 1938, 713-723*

<sup>2</sup> *Bell Syst. Tech. J., Vol.27, 1948, 379-423, 623-656.*

In the following section, we briefly review information measures for AND/OR trees and their extension to ternary DT (pseudo ternary DT) design.

## III. DECISION TREES

In our recent study of the entropy based method for minimization of switching function we used the binary DTs [3]. In this paper, we report results of experiments with pseudo ternary DTs.

### A. BINARY DTs

Binary DTs consists of nodes with two outgoing edges. Different assignments of a given function  $f$  to a DT produces different DTs. For example, AND/OR DTs are defined by using the Shannon decomposition ( $S$ -decomposition)  $f = \bar{x} \cdot f_0 \vee x \cdot f_1$ , where  $f_0 = f|_{x=0}$  and  $f_1 = f|_{x=1}$ . The correspondence between a binary AND/OR DT and a switching function  $f$  is defined as follows: (i) for a terminal node  $v$ ,  $f_v = 1$  if  $v = 1$ , and  $f_v = 0$  if  $v = 0$ , (ii) for a nonterminal node  $f_v = \bar{x} \cdot f_0 \vee x \cdot f_1$ .

In the considerations in this paper, we use the following basic concepts and definitions related to binary DTs.

A path in the DT consists of the edges connecting a non-terminal node with a terminal node. A complete path connects the root node with a terminal node. DTs are graphical representations of functional expressions for the represented functions. Each complete path in a DT corresponds to a term in the functional expression for the represented function  $f$ .

Decision diagrams (DDs) are derived by the reduction of DTs. Different orderings of variables in a DT, produce DDs with different number of nodes. Therefore, the variable ordering is a very important problem in DDs representations. For that reason the following concepts are introduced.

A DT is an *ordered DT* if the variables assigned to the nodes in the each path appear in the same order.

*Free DTs* are a generalization of ordered DTs derived by permitting permutation of variables in a subtree irrespective to the order of variables in the other subtree rooted at the considered node.

The considerations in this paper are an elaboration and generalization of the entropy based method for SOP minimization of a switching function  $f$  through AND/OR DTs [3], [7], [9], [16]. In these considerations, the following basic concepts are used.

*Definition 1:* The information model of the  $S$ -node corresponding to the Shannon expansion of a function  $f$  with respect to the variable  $x$ , is represented by the relation

$$H^S(f|x) = H(f|_{x=0}) + H(f|_{x=1}) \quad (1)$$

A criterion to choose a variable  $x$  in  $S$ -decomposition is that the conditional entropy of the function  $f$  with respect to the given variable  $x$  has to be minimum.

In what follows, we use the information theory notations of  $S$ -decomposition (1) to define an entropy based method for minimization of switching function on pseudo ternary DT.

## B. TERNARY AND PSEUDO-TERNARY DTS

Ternary DTS (TDTs) are a generalization of the binary DTS derived by permitting nodes with three outgoing edges. In TDTs derived as a generalization of the Binary Decision Trees (BDTs), it is assumed that the first two outgoing edges point to the cofactors of  $f$  determined as in the Shannon decomposition for  $f$ . The third outgoing edge points to a subfunction  $f_0 \# f_1$ , where  $\#$  denotes a binary operation. By choosing different operations, different TDTs are defined. Examples are AND-TDTs, OR-TDTs, EXOR-TDTs, and Kleene-TDTs, defined by using the logic operations AND, OR, EXOR, and the Kleene operation, respectively. In these TDTs, the correspondence between a TDT and a switching function  $f$  is defined as follows: (i) for terminal node  $v$ ,  $f_v = 1$  if  $v = 1$  and  $f_v = 0$  if  $v = 0$ , (ii) for nonterminal node  $f_v = \bar{x} \cdot f_0 \vee x \cdot f_1 \vee 1 \cdot f_2$ , where  $f_0 = f|_{x=0}$ ,  $f_1 = f|_{x=1}$  and  $f_2 = f_0 \# f_1$ , the symbol  $\#$  denotes any of the operations AND, OR, EXOR, or Kleene operator [13], [17].

*Example 4:* Let us build an EXOR ternary DT based on entropy based technique for the switching function given by Table I. The first edge of the root node corresponds to subfunction  $f_0$ , the second - to  $f_1$ , and the third branch corresponds the subfunction  $f_2 = f_0 \oplus f_1$ . Let us start with variable  $x_3$ , because it follows from Example 3 that the order of decomposed variables is  $\{x_3, x_2, x_1\}$ . The first edge corresponds to  $x_3 = 0$ , i.e.  $f_0 = \{000, 010, 100, 110\}$  (Figure 1a). By analogy, the second edge corresponds to  $x_3 = 1$ ,  $f_1 = \{001, 011, 101, 111\}$ . The third edge is formed as  $f_2 = f_0 \oplus f_1$ . The final result of building the DT yields the following AND/EXOR form:  $f = \bar{x}_3 \oplus x_1 x_3 \oplus x_2 x_3$ . ■

Basic concepts in the binary DTS representations can be directly extended to the TDTs. For example, a *Free ternary DT* is a generalization of a ternary DT derived by allowing permutation of variables in a subtree independently on the other subtree related to a non-terminal node.

For the further consideration in this paper, we define a pseudo TDTs as follows.

A *Pseudo ternary DT* is a ternary DT with the third outgoing edge of the node point to the subfunction in form of *Morreale's* operator defined in Table II [10]. In this table, the part (a) shows how to obtain the subfunctions for the first  $f'_0$  and the second edge  $f'_1$ . The part (b) shows how to remove the covered assignments to obtain  $f''_0$  and  $f''_1$ . The part (c) shows how to obtain the subfunction of the third edge  $f_d$  for a node.

The correspondence between AND/OR pseudo ternary DT and a switching function is defined as follows: (i) for terminal node  $v$ ,  $f_v = 1$  if  $v = 1$  and  $f_v = 0$  if  $v = 0$ , (ii) for nonterminal node  $f_v = \bar{x} \cdot f_0 \vee x \cdot f_1 \vee f_d$ , where  $f_d$  is computed as shown in Table IIc.

Further study in this paper is based on the improved *Morreale's* algorithm [10] for SOP minimization of switching function. We use the techniques of calculation based on the information estimations for the improved *Morreale's* algorithm. The results of iterations of the algorithm are denoted by  $f_0, f_1, f'_0, f''_0, f'_1, f''_1$ . In each node, we

choose one variable for the  $S$ -expansion, then we determine the cubes which do not depend of this variable. In the next step, we eliminate these cubes from the subfunctions  $f_0$  and  $f_1$ . The results of these operations give  $f'_0$  for the left edge and  $f'_1$  for the middle edge. Let  $g_0$  and  $g_1$  denote covers of subfunctions for the left and middle edges, respectively. Then we remove the covered cubes from  $f_0$  and  $f_1$ , to produce  $f''_0$  and  $f''_1$ . The subfunction  $f_d$  is obtained as the product of  $f''_0$  and  $f''_1$  and consists of the cubes which do not depend on the variable used the considered node.

*Example 5:* Minimize the switching function given in Table I, by using the entropy based (improved) *Morreale's* algorithm (Figure 1b).

1° To build the nodes corresponding to the  $S$ -expansion, we have to choose a variable with the smallest conditional entropy  $\min\{H(f|x)\}$ . In our example, for the first node we obtain the following results:  $H(f|x_1) = 0,8113 \text{ bit/pattern}$ ,  $H(f|x_2) = 0,8113 \text{ bit/pattern}$  and  $H(f|x_3) = 0,5 \text{ bit/pattern}$ . So, the variable  $x_3$  is selected.

2° Create two outgoing edges: the first,  $f_0$ , corresponds to all the assignments of the values to variables for which  $x_3 = 0$ , the second,  $f_1$ , corresponds to the assignments where  $x_3 = 1$ . So, it yields  $f_0 = \{000, 010, 100, 110\}$ ,  $f_1 = \{001, 011, 101, 111\}$ .

3° Create the third edge  $f_d$ . Change the values of functions  $f_0$  and  $f_1$ , into  $f'_0$  and  $f'_1$ , respectively, as it is defined in Table IIa. The results of this step are the two subfunctions without cubes which do not depend on the variable selected in the step 1°.

4° Steps 1°-3° are repeated for the next nodes, until a leaf node is achieved. If the leaf node takes the value 1, change the values of the functions  $f'_0$  or  $f'_1$  into  $f''_0$  or  $f''_1$  respectively, by deleting the covered assignments as shown in Table IIa. In our case, the second level of the DT contains two leaf nodes - the first one with the value 1 ( $g_0$ ) and the next one with the value 0 ( $g_1$ ). The first terminal node corresponds to  $\bar{x}_3$ , so it covers all the assignments in  $f_0 = \{000, 010, 100, 110\}$ .

5° For each node whose outgoing edges are ended by the leaf nodes, create the edge  $f_d$ . It includes all the assignments not yet covered by the assignments for edges  $f_0$  and  $f_1$ . In our case, these are the assignments covered by the variable  $\bar{x}_3$ , i.e.  $\{001, 011, 101, 111\}$ . Here, we omit the variable which corresponds to the third edge, as shown in Table IIc. ■

## IV. ENTROPY BASED SOP MINIMIZATION OF SWITCHING FUNCTIONS ON PSEUDO TERNARY DECISION TREES

In this section, we introduce an algorithm for SOP minimization of switching functions based on the information estimations on pseudo ternary DTS.

### A. ALGORITHM

The entropy based algorithm to derive the SOP expression for given function  $f$  through the pseudo ternary DT for  $f$  consists of the following steps:

$f'_0, f'_1:$	$f_1, f_0$	$f_0, f_1$	$f''_0, f''_1:$	$g_i$	$f_i$	$f_d:$	$f'_1, f''_0$	$f''_1, f'_0$
	$0$	$0$		$0$	$0$		$0$	$0$
	$1$	$0$		$1$	$0$		$1$	$1$
	$d$	$0$		$1$	$-$		$d$	$d$

(a)
(b)
(c)

TABLE II  
MORREALE'S OPERATORS.

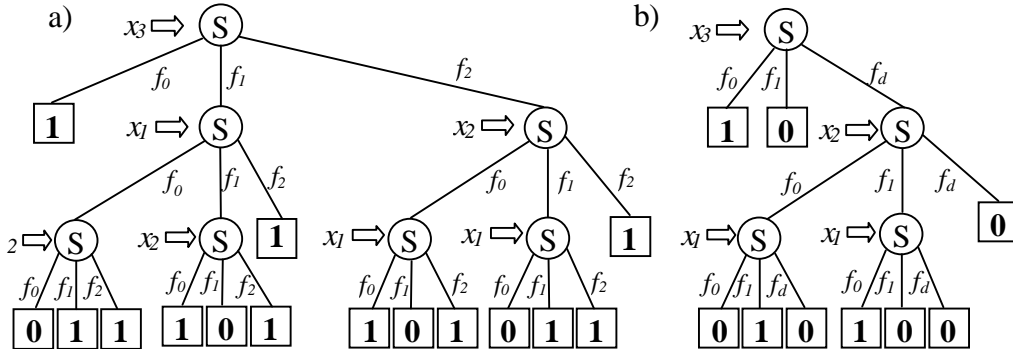


Fig. 1. Example of entropy based DT design for a free ternary DT with final result  $f = \bar{x}_3 \oplus x_1 x_3 \oplus x_2 x_3$  (a) and free pseudo ternary DT with resulting function  $f = \bar{x}_3 \vee \bar{x}_1 x_2 \vee x_1 \bar{x}_2$  (b) for the given in Table I switching function of 3-variables

- If all the values of the function are either a constant or don't care, return this constant value, else select the variable  $x_j$ ,  $j \in \{1, \dots, n\}$  with the smallest conditional entropy to build the current node in the DT,
- Perform the Shannon's decomposition for the the selected variable  $x_j$ :  $f_0 = f|_{x_j=0}$  and  $f_1 = f|_{x_j=1}$ ,
- Compute subfunctions  $f'_0$  and  $f'_1$  as shown in Table IIa for subfunctions  $f_0$  and  $f_1$ ,
- Run the algorithm for the results of the previous operation and write the results to two variables:  $f_{sop_0}$  and  $f_{sop_1}$ ,
- Calculate the subfunctions  $f''_0$  and  $f''_1$  (Table IIb) for two subfunctions  $f_{sop_0}$  and  $f_{sop_1}$  and their covers,
- Compute subfunction  $f_d$  (Table IIc),
- Run the algorithm for the subfunction  $f_d$ , and write the result as  $f_{sop_d}$ ,
- Determine the final expression  $(\bar{x}_i \cdot f_{sop_0}) \vee (x_i \cdot f_{sop_1}) \vee f_{sop_d}$ .

A pseudo-code of this algorithm is shown in Figure 2, for minimization of both single output functions and multiple output functions.

*Example 6: (Comments to the algorithm) Input:* truth table of a 3-variables function given in Table IIIa. *Output:* SOP expression.

- Check, whether all the specified values of the function are constant. In our example we have no such a situation.
- Calculate the conditional entropy  $H^S(f|x_i)$  by using (1) for each variable  $x_i$ ,  $i = 1, 2, 3$ , to chose a variable for the optimal path. It follows from the Example 3 that the variable  $x_3$  is chosen first.
- Form the subfunctions  $f_0$  and  $f_1$  for the Shannon's decomposition of  $f$ , i.e.  $f_0 = f|_{x_3=0}$  and  $f_1 = f|_{x_3=1}$ . Then compute subfunctions  $f'_0$  and  $f'_1$  with respect to the rules

given in Table IIa. The result is represented in Table IIIb.

- Perform the algorithm for subtrees with the roots  $f = f_0$  and  $f = f_1$ . Let  $g_0$  be the cover of  $f_0$  and  $g_1$  be the cover of  $f_1$ . To specify subfunctions  $f''_0$  and  $f''_1$  we have to use the rules from Table IIb. At the moment we have enough information to determine the third edge,  $f_d$ , of this DT node (Table IIc). The result is shown in Table IIIc.
- Perform the algorithm for  $f = f_d$ . The conditional entropies are  $H^S(f|x_1) = H^S(f|x_2) = 1$ , so we have to select any of these variable to add to the optimal path and create a node. Let us choose  $x_1$ . Then we obtain the following assignments to the edge (Table IIId), so that  $f_{d0}$  is shown in Table IIIe and  $f_{d1}$  - in Table IIIf.

By following the algorithm in this way, we obtain the optimal path  $\{H(f|x_3), H(f|x_2), H(f|x_1)\}$  and the minimal SOP expression  $f = \bar{x}_3 \vee \bar{x}_1 x_2 \vee x_1 \bar{x}_2$ . ■

*Example 7:* We demonstrate in Figure 3 the entropy based method for a free pseudo ternary DT for the completely specified switching function given in Table I. ■

## V. EXPERIMENTS

We implemented our InfoMIN-2 program package in C++ for OS LINUX REDHAT with Pentium II 333MHz processor. To evaluate the efficiency of the program, we compare it in LGSynth91 benchmarks<sup>3</sup> with the exact solutions of the ESPRESSO package and also with InfoMin-1 that implements the entropy based minimizer based on the binary DTs<sup>4</sup>. The results are shown in

<sup>3</sup>[http://www.cbl.ncsu.edu/pub/Benchmark\\_dirs/LGSynth91/twolexamples](http://www.cbl.ncsu.edu/pub/Benchmark_dirs/LGSynth91/twolexamples)

<sup>4</sup>We reported this program in V. Shmerko, S. Jaroszewicz, D. Simovici and V. Cheushev, "For Remarks on Minimization of Strongly

```

/* (input  $f(x_1, \dots, x_n): \{0, 1\}^n \rightarrow \{0, 1, d\}$  */
/* (output)  $fsop$ : SOP expression
eDT (U)
{
  if( $\forall x \in \{0, 1\}^n : f(x) \neq 1$ ) { $fsop \leftarrow 0$ }
  else if( $\forall x \in \{0, 1\}^n : f(x) \neq 0$ ) { $fsop \leftarrow 1$ }
  else
  {
    for( $\forall x_i \in x_1, x_2, \dots, x_n$ )
      calculate the entropy
       $H^S(f|x_i) \leftarrow Smodel(f, x_i)$ 
      choose the variable  $x_j$  for adding to the optimal path
      and free DT node construction
       $x_j: H^S(f|x_j) \leq H^S(f|x_i), \forall i \in 1, \dots, n$ 
       $f_0 \leftarrow f|_{x_j=0}$ ; /*the subfunction
      on  $x_j = 0$  */
       $f_1 \leftarrow f|_{x_j=1}$ ; /*the subfunction
      on  $x_j = 1$  */
       $f'_0 \leftarrow Calculatef'_0(f_0, f_1)$  /*Table IIa*/
       $f'_1 \leftarrow Calculatef'_1(f_0, f_1)$  /*Table IIa*/
       $fsop_0 \leftarrow eDT(f'_0)$ 
       $fsop_1 \leftarrow eDT(f'_1)$ 
      Let  $g_0$  be the cover of  $fsop_0$  and  $g_1$ 
      be the cover of  $fsop_1$ .
       $f''_0 \leftarrow Calculatef''_0(f_0, g_0)$  /*Table IIb*/
       $f''_1 \leftarrow Calculatef''_1(f_1, g_1)$  /*Table IIb*/
       $f_d \leftarrow Calculatef_d(f'_0, f'_1)$  /*Table IIc*/
       $fsop_d \leftarrow eDT(U_D)$ 
       $fsop \leftarrow (\bar{x}_i \cdot fsop_0) \vee (x_i \cdot fsop_1) \vee fsop_d$ 
    }
  }
return( $fsop$ );
}

```

(a)

```

/* (input  $f(x_1, \dots, x_n): \{0, 1\}^n \rightarrow \{0, 1, d\}^m$  */
/* (output)  $fsop$ : SOP expression for a system of switching functions */
eDT (U)
{
  for( $\forall k \in 0, 1, \dots, m-1$ )
  {
    if( $\forall x \in \{0, 1\}^n : f_k(x) = d$ ) { $f_k sop \leftarrow d$ }
    else if( $\forall x \in \{0, 1\}^n : f_k(x) \neq 1$ ) { $f_k sop \leftarrow 0$ }
    else if( $\forall x \in \{0, 1\}^n : f_k(x) \neq 0$ ) { $f_k sop \leftarrow 1$ }
    else break;
    if ( $k=m$ ) return  $fsop$ ;
  }
  for( $\forall x_i \in \{x_1, x_2, \dots, x_n\}$ )
  /* calculate the sum of the entropy for each function */
   $H^S(f|x_i) \leftarrow \sum_{k=0}^{m-1} Smodel(f_k, x_i)$ 
  /* choose the variable  $x_j$  for adding to the optimal path
  and free DT node construction */
   $x_j: H^S(f|x_j) \leq H^S(f|x_i), \forall i \in 1, \dots, n$ 
   $f_{0,k} \leftarrow f^k|_{x_j=0}; k = 0, \dots, m-1$  /*the subfunction
  on  $x_j=0$ */
   $f_{1,k} \leftarrow f^k|_{x_j=1}; k = 0, \dots, m-1$  /*the subfunction
  on  $x_j=1$ */
   $f'_{0,k} \leftarrow Calculatef'_{0,k}(f_{0,k}, f_{1,k}); k = 0, \dots, m-1$  /*Table IIa*/
   $f'_{1,k} \leftarrow Calculatef'_{1,k}(f_{0,k}, f_{1,k}); k = 0, \dots, m-1$  /*Table IIa*/
   $fsop_{0,k} \leftarrow eDT(f'_{0,k})$ 
   $fsop_{1,k} \leftarrow eDT(f'_{1,k})$ 
  /* Let  $g_0$  be the cover of  $fsop_{0,k}$  and  $g_1$ 
  be the cover of  $fsop_{1,k}$  */
   $f''_{0,k} \leftarrow Calculatef''_{0,k}(f_{0,k}, g_{0,k}); k = 0, \dots, m-1$  /*Table IIb*/
   $f''_{1,k} \leftarrow Calculatef''_{1,k}(f_{1,k}, g_{1,k}); k = 0, \dots, m-1$  /*Table IIb*/
   $f_{d,k} \leftarrow Calculatef_d(f''_{0,k}, f''_{1,k}); k = 0, \dots, m-1$  /*Table IIc*/
   $fsop_d \leftarrow eDT(U_D)$ 
   $fsop \leftarrow (\bar{x}_i \cdot fsop_0) \vee (x_i \cdot fsop_1) \vee fsop_d$ 
}
return( $fsop$ );
}

```

(b)

Note: Calculations  $Calculatef'_0$ ,  $Calculatef'_1$ ,  $Calculatef''_0$ ,  $Calculatef''_1$  and  $Calculatef_d$  based on rules given in Table II.

Fig. 2. The sketch of the entropy based algorithms to derive SOP expression via conversion truth table into free pseudo ternary DT for single (a) and a system (b) of switching functions

		Initial function				
		$x_1x_2x_3$	$f$	$x_1x_2x_3$	$f$	
(a)		000	1	100	1	
		001	0	101	1	
		010	1	110	1	
		011	1	111	0	

		1st node, 1st and 2nd edges					
		$x_1$	$x_2$	$f_0$	$f_1$	$f'_0$	$f'_1$
(b)		0	0	1	0	1	0
		0	1	1	1	$d$	$d$
		1	0	1	1	$d$	$d$
		1	1	1	0	1	0

		First node, the third edges								
		$x_1$	$x_2$	$f_0$	$g_0$	$f''_0$	$f_1$	$g_1$	$f''_1$	$f_d$
(c)		0	0	1	1	$d$	0	0	0	0
		0	1	1	1	$d$	1	0	1	1
		1	0	1	1	$d$	1	0	1	1
		1	1	1	1	$d$	0	0	0	0

		The second node									
		$x_2$	$f_0$	$f'_0$	$g_0$	$f''_0$	$f_1$	$f'_1$	$g_1$	$f''_1$	$f_d$
(d)		0	0	0	0	0	1	1	1	$d$	0
		1	1	1	1	$d$	0	0	0	0	0

		The third node								
		$f_0$	$f'_0$	$g_0$	$f''_0$	$f_1$	$f'_1$	$g_1$	$f''_1$	$f_d$
(e)		0	0	0	0	1	1	1	$d$	0

		The fourth node								
		$f_0$	$f'_0$	$g_0$	$f''_0$	$f_1$	$f'_1$	$g_1$	$f''_1$	$f_d$
(f)		1	1	1	$d$	0	0	0	0	0

TABLE III

ENTROPY BASED MINIMIZATION (EXAMPLE 6)

Tables IV, V, VI and VII, where  $P$  denotes the number of products,  $L$  denotes the number of literals, and  $t$  denotes the time in seconds.

### A. Small 1-output circuits .

In our first series of experiments summarized in Table IV, we studied the minimization of small 1-output functions. For example, the exact optimal SOP expression for **bw01** contains 6 products with 25 literals. InfoMin-1, which utilizes the approach based on the binary DT, produces 8 products with 39 literals. The solution produced by InfoMin-2 is equal to the optimum.

In the case of the benchmark function **majority**, we observed that the exact method gives 5 products and 13 literals. Both InfoMin-1 and Infomin-2 produce solutions with more literals (20 and 17, respectively), but the second method is closer to the optimum.

**Observation.** The entropy based method for minimization of small one-output functions on pseudo ternary DT (InfoMin-2 package) gives the same results as ESPRESSO method in terms of both products and literals, but is four times faster.

### B. Small multi-outputs functions.

Unspecified Logic Functions”, *The 7th Int. Workshop on Post-Binary Ultra-Large Scale Integration - ULSI'98*, Fukuoka, Japan, 1998

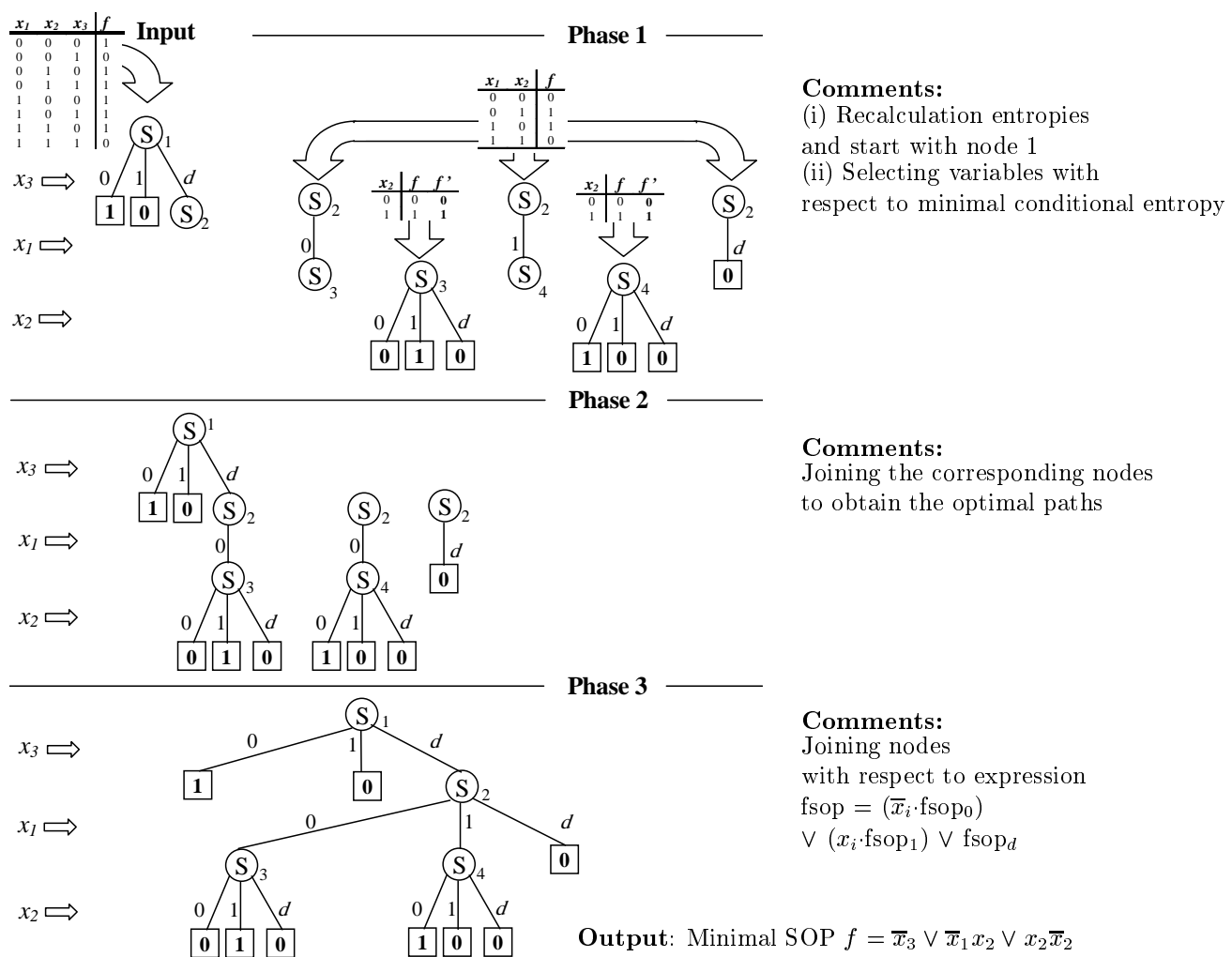


Fig. 3. Entropy based improved Morreale's algorithm for minimization completely specified switching function of three variables on free pseudo ternary DT

	IN	ESPRESSO			InfoMin-1			InfoMin-2		
		<i>P</i>	<i>L</i>	<i>t</i>	<i>P</i>	<i>L</i>	<i>t</i>	<i>P</i>	<i>L</i>	<i>t</i>
f21	4	3	9	0.00	3	11	0.00	3	9	0.00
f24	4	3	9	0.00	3	11	0.00	3	9	0.00
misex24	4	3	10	0.00	3	11	0.00	3	10	0.00
bw01	5	6	25	0.00	8	39	0.00	6	25	0.00
bw17	5	3	6	0.00	3	9	0.00	3	8	0.00
bw9	5	3	11	0.00	4	17	0.00	3	11	0.00
majority	5	5	13	0.00	5	20	0.00	5	17	0.00
misex27	5	6	19	0.00	6	23	0.00	6	19	0.00
sam	5	6	25	0.00	8	39	0.00	6	25	0.00
xor5	5	16	80	0.00	16	80	0.00	16	80	0.00
misex21	6	11	42	0.00	14	72	0.00	11	42	0.01
misex54	6	11	42	0.00	13	65	0.00	11	42	0.01
5x01	7	7	27	0.01	7	36	0.00	7	27	0.02
con11	7	4	11	0.00	10	46	0.00	4	11	0.01
con12	7	5	12	0.01	6	20	0.00	5	14	0.01
z41	7	15	56	0.00	15	90	0.00	15	64	0.01
z42	7	28	136	0.00	32	200	0.00	28	144	0.01
<b>Total</b>	<b>94</b>	<b>135</b>	<b>533</b>	<b>0.02</b>	<b>156</b>	<b>789</b>	<b>0.00</b>	<b>135</b>	<b>557</b>	<b>0.08</b>

TABLE IV  
SOP MINIMIZATION BY ESPRESSO, INFOMIN-1, AND INFOMIN-2 FOR SMALL 1-OUTPUT BENCHMARKS.

In the second series of experiments, summarized in Table V, we studied the proposed minimization strategy on small functions with multiple outputs. For example, for **adr2** with 4 inputs and 3 outputs, the

exact optimal SOP expression contains 11 products and 32 literals. InfoMin-1 produces 15 products and 60 literals. InfoMin-2 yields an expression with 11 products and 32 literals, which is the optimum.

The result for **sao2** shows that the exact method gives 58 products and 420 literals. Both InfoMin-1 and InfoMin-2 produce worse solutions requiring 70 products with 631 literals and 61 products with 433 literals, respectively.

**Observation.** *InfoMin-1 produces solutions with at about 50% more products and at about 100% more literals compared with ESPRESSO. InfoMin-2, in contradiction to InfoMin-1, reduces the number of products for at about 30% and the number of literals for at about 50%.*

### C. Totally symmetric one-output functions.

In this series of experiments, summarized in Table VI, we used functions that was defined as totally symmetric functions [2], [4]. For example, the 1st, 2nd and 3rd output of **rd53** are the totally symmetric functions.

**Observation.** *InfoMin-1 produces solutions with at about 20% more products and at about 10% more literals than ESPRESSO. InfoMin-2, in comparison with InfoMin-1, reduces the number of products for 5% and the number of literals for 20%. For a difficult function as **9sym**, the program InfoMin-2 yields at about 80% worse result (in number of literals) than ESPRESSO.*

### D. Weakly specified functions (Table VII).

In this series of experiments, summarized in Table VII, we used randomly generated switching function with 100-400 variables that was specified by 200-600 sets only. We have compared the results with the results produced by ESPRESSO-EXPAND package.

**Observation.** *The program InfoMin-2 produces the results at about 90% faster in comparison with ESPRESSO. The number of products and the number of literals are 30% and 100% smaller, respectively, in comparison with InfoMin-1 that uses free binary DT.*

But as we can see, the number of products and literals produced by ESPRESSO is three times fewer on the average.

## VI. CLOSING REMARKS

We propose for the first time to convert truth table of a switching function to the ternary and pseudo ternary DT.

The main contribution of this paper is the entropy based method for minimization of switching functions on pseudo ternary DT. The main feature of proposed method is that the SOP minimization is realized on free pseudo ternary DT. We compared our algorithm with the previously developed program InfoMin-1 based on the entropy measures on binary DTs and showed that our InfoMin-2 program allows to obtain in many cases the extremely better results. Our experimental study shows that

(i) In the case of small 1-output and multi-outputs circuits we achieve the results close to ESPRESSO.

(ii) For the class of symmetric functions we achieved practically the same results as ESPRESSO, except for **9sym**.

(iii) For weakly specified switching functions the number of literals is at about 10 times fewer in comparison with the entropy based method on binary DT. On the other hand, ESPRESSO results are better for few percents, but the runtime is 3 times worse.

In summary, InfoMin-2 produces better results than InfoMin-1. In some cases, the results produced with InfoMin-2 are equal or comparable with these derived with ESPRESSO. However, InfoMin-2 is quite much faster than ESPRESSO.

## REFERENCES

- [1] R. B. Ash. *Information Theory*. John Wiley and Sons, 1967.
- [2] J. Butler, G. Dueck, G. Holowinski, V. Shmerko, and S. Yanushkevich. On recognition of symmetries for switching functions in Reed-Muller form. In *Proc. 5th Int. IAPR Conference Pattern Recognition and Information Processing - PRIP'99, Minsk, Republic of Belarus*, pages 215–234, 1999.
- [3] V. Cheushev, V. Shmerko, D. Simovici, and S. Yanushkevich. Functional entropy and decision trees. In *Proc. IEEE Int. Symp. on Multiple-Valued Logic, Japan*, pages 357–362, 1998.
- [4] G. Dueck, J. Butler, S. Yanushkevich, and V. Shmerko. Optimal polarity for Reed-Muller and arithmetic expansion of symmetric logic functions. Part 1: Optimal fixed polarity Reed-Muller representation of symmetric Boolean functions. Technical Report TR98-1, *St. Francis Xavier University, Dept. of Mathematics, Statistics and Computer Science, Antigonish, CANADA*, 1998.
- [5] R. M. Goodman and P. Smyth. Decision tree design from a communication theory standpoint. *IEEE Trans. on Information Theory*, IT-34(5):979 – 994, 1988.
- [6] C. R. P. Hartmann, P. K. Varshney, K. G. Mehrotra, and C. L. Gerberich. Application of information theory to the construction of efficient decision trees. *IEEE Trans. on Information Theory*, IT-28(5):565–577, 1982.
- [7] A. M. Kabakcioglu, P. K. Varshney, and C. R. P. Hartman. Application of information theory to switching function minimization. *IEE Proceedings, Pt E*, 137:389–393, 1990.
- [8] A. Lloris, J. F. Gomez, and R. Roman. Using decision trees for the minimization of multiple-valued functions. *Int. J. Electronics*, 75(6):1035–1041, 1993.
- [9] A. Lloris-Ruiz, J. F. Gomez-Lopera, and R. Roman-Roldan. Entropic minimization of multiple-valued functions. In *Proc. IEEE Int. Symp. on Multiple-Valued Logic*, pages 24–28, 1993.
- [10] S. Minato. *Binary Decision Diagrams and Applications for VLSI CAD*. Cluwer Academic Publishers, 1996.
- [11] E. Morreale. Recursive operators for prime implicant and irredundant normal form determination. *IEEE Trans. on Computer*, C-19(6):504–509, 1970.
- [12] (Eds.) T. Sasao and M. Fujita. *Representations of Discrete Functions*. Kluwer Academic Publishers, 1995.
- [13] T. Sasao. *Ternary Decision Diagrams and their Applications*, chapter 12, pages 269–292. In [12], 1995.
- [14] T. Sasao. *Switching Theory for Logic Synthesis*. Kluwer Academic Publishers, Norwell, MA, 1999.
- [15] V. P. Shmerko, D. V. Popel, R. S. Stankovic, V. A. Cheushev, and S. N. Yanushkevich. Information theoretical approach to minimization of AND/EXOR expressions of switching functions. In *this Proceedings*.
- [16] D. Simovici, V. Shmerko, V. Cheushev, and S. Yanushkevich. Information estimations of logic functions. In *Proc. Int. Conf. on Applications of Computer Systems, Szczecin, Poland*, pages 287–300, 1997.
- [17] R. Stankovic. *Spectral Transform Decision Diagrams in Simple Questions and Simple Answers*. IP NAUKA, Belgrad, Yugoslavia, 1998. ISBN 86-7621-065-9.

	IN	OUT	ESPRESSO			InfoMin-1			InfoMin-2		
			P	L	t	P	L	t	P	L	t
adr2	4	3	11	32	0,00	15	60	0,00	11	32	0,00
rd53	5	3	31	140	0,00	31	155	0,00	31	140	0,00
squar5	5	8	25	88	0,01	30	150	0,01	27	92	0,11
con1	7	2	9	23	0,01	19	88	0,00	9	23	0,00
rd73	7	3	127	756	0,06	127	889	0,01	127	756	0,05
dist	8	5	120	710	0,18	164	1235	0,03	133	764	0,22
f51m	8	8	28	129	0,01	62	404	0,04	28	129	0,53
log8mod	8	5	38	172	0,02	47	274	0,02	42	184	0,07
misex1	8	7	12	51	0,00	13	63	0,01	13	53	0,10
rd84	8	4	255	1774	0,31	255	2040	0,03	255	1774	0,18
root	8	5	57	300	0,08	72	490	0,02	65	333	0,13
clip	9	5	117	614	0,51	446	3990	0,08	136	691	0,42
sao2	10	4	58	420	0,09	70	631	0,09	61	433	0,62
<b>Total</b>	<b>95</b>	<b>62</b>	<b>888</b>	<b>5209</b>	<b>1,28</b>	<b>1351</b>	<b>10469</b>	<b>0,34</b>	<b>938</b>	<b>5404</b>	<b>2,43</b>

TABLE V

SOP MINIMIZATION OF SMALL MULTI-OUTPUTS FUNCTIONS WITH ESPRESSO, INFOMIN-1, INFOMIN-2.

	IN	ESPRESSO			InfoMin-1			InfoMin-2		
		P	L	t	P	L	t	P	L	t
rd531	5	5	20	0,00	5	24	0,00	5	20	0,00
rd532	5	16	80	0,00	16	80	0,00	16	80	0,00
rd533	5	10	40	0,01	14	64	0,00	14	56	0,00
bw11	5	2	10	0,00	2	10	0,00	2	10	0,00
rd731	7	42	252	0,01	48	320	0,01	48	288	0,01
rd732	7	64	448	0,00	64	448	0,01	64	448	0,01
rd733	7	35	140	0,00	35	224	0,01	35	140	0,01
z4m14	7	4	12	0,00	16	80	0,00	4	12	0,01
rd841	8	84	588	0,04	92	708	0,02	92	644	0,04
rd842	8	128	1024	0,01	128	1024	0,02	128	1024	0,03
rd843	8	70	350	0,14	73	528	0,01	73	365	0,04
9sym	9	84	504	7,79	148	1170	0,04	148	888	0,12
sym10	10	65	244	0,04	77	612	0,01	65	244	0,17
m1812	15	7	28	0,00	7	37	0,69	7	28	4,02
m1813	15	7	28	0,00	9	54	0,75	7	28	4,57
m1814	15	4	8	0,00	4	12	0,62	4	8	4,09
m1815	15	4	8	0,00	6	21	0,67	4	8	4,09
m1816	15	5	5	0,00	5	15	0,48	5	5	3,82
<b>Total</b>	<b>166</b>	<b>636</b>	<b>3789</b>	<b>8,04</b>	<b>749</b>	<b>5431</b>	<b>3,34</b>	<b>721</b>	<b>4296</b>	<b>21,03</b>

TABLE VI

MINIMIZATION OF TOTALLY SYMMETRIC ONE-OUTPUT CIRCUITS, WITH 5-15 INPUTS WITH ESPRESSO, INFOMIN-1, AND INFOMIN-2.

IN	SPEC	ESPRESSO			InfoMin-1			InfoMin-2		
		P	L	t	P	L	t	P	L	t
100	100	6	25	13,68	25	525	0,37	16	71	1,41
100	200	13	68	38,94	54	1328	0,96	41	224	3,97
100	300	16	89	70,16	75	1835	1,34	51	312	7,96
100	400	22	131	230,78	113	2668	1,78	75	475	12,77
100	500	27	164	227,26	137	3455	2,06	87	571	19,52
200	100	6	25	31,72	29	932	1,74	18	77	7,38
200	200	11	57	100,83	53	2049	4,02	35	187	17,77
200	300	15	82	245,18	84	3157	6,17	54	314	27,43
200	400	20	120	484,15	102	3999	7,07	69	429	41,11
200	500	25	154	886,76	141	5412	10,27	82	522	55,75
300	100	6	25	81,16	24	1175	4,44	15	62	22,64
300	200	10	49	278,71	53	2826	10,92	30	154	50,64
300	300	15	79	520,28	76	3955	14,76	51	297	88,71
300	400	19	109	1047,23	86	4719	18,31	82	521	146,53
300	500	24	149	5368,39	129	7504	27,10	95	616	194,26
400	100	6	26	145,17	23	1301	8,34	16	68	48,52
400	200	8	38	382,47	50	3600	21,78	32	163	106,93
400	300	13	66	859,01	80	5845	33,80	56	323	179,51
400	400	19	112	7400,67	103	6985	41,36	71	440	304,53
<b>Total</b>		<b>281</b>	<b>1568</b>	<b>18412,55</b>	<b>1437</b>	<b>63270</b>	<b>7284,52</b>	<b>976</b>	<b>5826</b>	<b>1337,34</b>

Note: SPEC - number of specified values of the function with IN variables

TABLE VII

MINIMIZATION OF SOP EXPRESSIONS FOR WEAKLY SPECIFIED FUNCTIONS OF 100-400 VARIABLES WITH ESPRESSO, INFOMIN-1, AND INFOMIN-2.