IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 2, May 2011
ISSN (Online): 1694-0814
www.IJCSI.org

Hybrid Fuzzy Direct Cover Algorithm for Synthesis of Multiple-Valued Logic Functions

Mostafa Abd-El-Barr

Information Science Department, Kuwait University, Safat 13060, P.O. Box 5969, Kuwait

Abstract

Direct Cover (DC) based techniques for synthesis of multiple-valued logic (MVL) functions have been reported in the literature. In this paper, we propose Fuzzy-based DC algorithm for synthesis of MVL functions. The proposed algorithm uses the principles of Fuzzy Logic in selecting the set of minterms and the appropriate set of implicants needed to synthesize a given MVL function. The proposed Fuzzified-Direct-Cover (FZDC) heuristic is tested in comparison with five other DC-based algorithms reported in the literature. The benchmark used in our comparison consists of 50000 2-variable 4-valued randomly generated functions. The basis for comparison is the chip area consumed in terms of the average number of implicants needed to synthesize a given MVL function. It is shown that on average the FZDC heuristic requires less number of implicants to synthesize a given MVL function as compared to those required by any of the other five DC-based heuristics.

Keywords: *Multi-Valued Logic (MVL) synthesis, Heuristic algorithms, Direct Cover (DC) algorithms, Weighted direct cover algorithm (WDC), Ordered DC algorithm (ODC), Fuzzified-Direct-Cover algorithm (FZDC).*

1. Introduction

Signal processing using multiple-valued logic (MVL) is carried out using more than two logic levels [11][14][17][19]. Successful hardware realization of MVL circuits has been reported in the literature. Examples of MVL circuits reported in the literature using binary CMOS (Complementary Metal Oxide Semiconductor) circuits [45]-[47] include arithmetic circuits [24], [39], [48], and [49], memory (ROM, RAM, and Flash) [38] and [51]-[52], and machine learning [4][7].

Deterministic synthesis of MVL functions is more complex than its binary counterpart. Exact minimization of MVL functions is prohibitively expensive and the use of heuristic algorithms in MVL synthesis is eminent [33], [50]. A number of heuristic algorithms for producing near minimal sum-of-products realization of MVL functions have been introduced in the literature [9] [31] [34][36]. These algorithms can be categorized as direct cover-based [2][8][10][15], algebraic-based [12][16][18][26][29], decomposition-based [1][4]-[7] [12] [25] [35], decision diagram-based [21], [22], [27], [28], [32], and [37], and iterative-based [29]. Among these, the direct cover-based techniques have shown promising results and therefore have received increasing attention by MVL systems designers.

In the context of MVL functional synthesis, the following definitions are relevant to present.

Definition 1: An n-variable r-valued function, $f(X)$, is a mapping $f : R^n \rightarrow R$, where $R = \{0,1,\cdots,r-1\}$ is a set of r logic values with $r \geq 2$ and $X = \{x_1, x_2, \cdots, x_n\}$ is a set of r-valued n variables.

Definition 2: *A window literal* of a MVL variable *x* is defined as:

$$ {}^{a}x^{b} = \begin{cases} (r-1) & if\,(a \leq x \leq b) \\ 0 & \text{otherwise} \end{cases} $$

where $a, b \in R$ and $a \leq b$.

Definition 3: A *tsum* (truncated sum) operator is defined as

$$ tsum(a_1, a_2, \ldots, a_n) = a_1 \oplus a_2 \oplus \ldots \oplus a_n = \begin{cases} a_1 + a_2 + \ldots + a_n & if \;\; \sum_{i=1}^{n} a_i < r-1 \\ r-1 & otherwise \end{cases} $$

Where $a_i \in R$ and $\oplus$ represents the truncated sum operation.

Definition 4: The maximum (MAX) of two MVL variables is defined as:

$$ MAX\;(x_1, x_2) = \begin{cases} x_1 & \text{if} \quad x_1 \geq x_2 \\ x_2 & \text{otherwise} \end{cases} $$

Definition 5: The minimum (MIN) of two MVL variables is defined as:

$$MIN\ (x_1, x_2) = \begin{cases} x_1 & \text{if} \quad x_1 \le x_2 \\ x_2 & \text{otherwise} \end{cases}$$

A functionally complete set of operators is the set capable of realizing all possible functions. A number of functionally complete sets have been used in synthesizing MVL functions. In terms of the set of MVL operators discussed above, the set consisting of {*Literal, MIN, TSUM, Constant*} is used in this paper.

Definition 6: A *product term* (PT), $P(x_1, x_2, ..., x_n)$, is defined as the minimum of a set of *window literals* such that

$$P(x_1, x_2, ..., x_n) = c \bullet {}^{a_1,b_1}x_1, {}^{a_2,b_2}x_2, ..., {}^{a_n,b_n}x_n = \min(c, {}^{a_1,b_1}x_1, {}^{a_2,b_2}x_2, ..., {}^{a_n,b_n}x_n)$$

with $a_i, b_i \in R$, $a_i \le b_i$ and c (the value of the PT) $\in$ {1,2,…, r-1}.

Definition 7: An assignment of values to variables such that $x_1 = a_1, x_2 = a_2 = ..., x_n = a_n$, where $a_i \in$ {0, 1, …, r-1}, in an MVL function $f(x_1, x_2, ..., x_n)$ is called a *minterm*, iff: $f(x_1, x_2, ..., x_n) \ne 0$.

A *minterm* is a special case of a *PT* for which $a_1 = b_1, a_2 = b_2 = ..., a_n = b_n$. If the value of a *minterm* is r, then it is considered as *don't care* and is represented as d.

Definition 8: An *implicant* of a function $f(x_1, x_2, ..., x_n)$, is a PT, $I(x_1, x_2, ..., x_n)$, such that $f(x_1, x_2, ..., x_n) \ge I(x_1, x_2, ..., x_n)$ for all assignments of x_i's.

Fig. 1 shows an example of a 2-varaible 4-valued function. Some of the minterms in the figure are $1\bullet{}^{0}X_1{}^{0}\bullet{}^{3}X_2{}^{3}$, $2\bullet{}^{0}X_1{}^{0}\bullet{}^{1}X_2{}^{1}$ and $3\bullet{}^{0}X_1{}^{0}\bullet{}^{2}X_2{}^{2}$ while $2\bullet{}^{0}X_1{}^{1}\bullet{}^{1}X_2{}^{1}$and $2\bullet{}^{0}X_1{}^{1}\bullet{}^{1}X_2{}^{2}$ are examples for *implicants*.

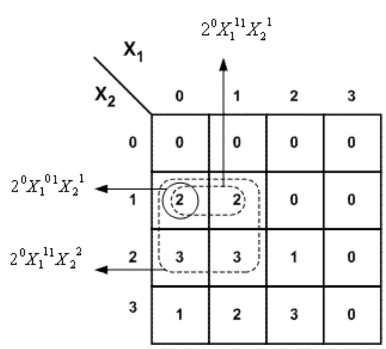

Fig. 1. A Tabular Representation of $f(X_1, X_2)$.

The paper is organized as follows. In Section 2, we briefly present related published work. In Section 3 we introduce the hybrid Fuzzy based Direct Cover heuristic (FZDC). The results obtained using the proposed and the related DC-based heuristics using a benchmarks consisting of 50000 2-variable 4-valued randomly generated functions are presented in Section 4. In Section 5, we show a case study for the application of both the introduced and the related heuristics. The presented case study is that for a 1-bit 4-valued adder block. Concluding remarks are drawn in Section 6.

This document is set in 10-point Times New Roman. If absolutely necessary, we suggest the use of condensed line spacing rather than smaller point sizes. Some technical formatting software print mathematical formulas in italic type, with subscripts and superscripts in a slightly smaller font size. This is acceptable.

2. Related Work

In this section, we briefly cover the related work published in the literature.

2.1 The Direct Cover Algorithm

The Direct Cover (DC) techniques for synthesis of MVL functions consist of the following main steps:

1. Choose a minterm (see Definition 7),
2. Identify a suitable implicant (see Definition 8) that covers the chosen minterm,
3. Obtain a reduced function by subtracting the identified implicant from the (remaining part of the) function, and
4. Repeat steps 1 to 3 until no more minterms remain uncovered.

The method used in the selection of minterms and implicants is crucial in obtaining reduced number of product terms to cover a given function. The Direct Cover approaches reported in the literature differ in the way minterms are selected and the way according to which implicants are identified. Different minterm selection metrics have been reported in the literature. These include using the Isolation Weight (IW) [3] as a measure of the degree to which other minterms cluster around a given minterm, the Isolation Factor (IF) [10] which measures the number of directions in which a given minterm can be combined with a nonzero number of other minterms, and the Clustering Factor (CFN) [8] which measures the number of minterms that are connected to a given minterm. Similarly, a number of metrics were used in selecting the appropriate implicant to cover a given minterm. These include the Largest Number of Minterms Reduced to Zero (LRZ) [3] [15] which counts the number of 0 (or don't care) that resulted from removing the selected implicant from the given MVL function, the Relative Break Count (RBC) [10] which a measure of the degree to which a function is simplified if a specific implicant is chosen, and the Neighborhood Relative Count (NRC) [8] which measures the degree of the strength to which a given implicant couples with its neighbors.

It should be noted that there is no general agreement on which set of criteria is the best to use for a given MVL function. An analysis of a limited subset of the different suggested criteria has been reported in [23]. Considering the different minterm/implicant selection criteria, it is possible to combine all metrics to create a new selection criterion. One possible way to achieve this is by using the weighted sum approach. However, the different scale and behavior of each criterion makes it difficult to determine the perfect weights for each objective and the perfect way of aggregating these into a single function. We have reported in the literature two approaches to deal with such issue. These are the weighted DC-based (WDC) and the Ordered DC-based (ODC) heuristic algorithms [53]. These are briefly covered below.

2.2 The Weighted Direct Cover (WDC) [53]

In the WDC, the term *weight pattern* is used to specify the weight for each selection criterion. Assuming 4-valued logic, we assume that the weight should be integer in the set {0, 1, 2, 3}. Consider the minterm selection process. In this case, a weight pattern of (112) means that the following weights are used: w_{CF}=1, w_{CFN}=1, and w_{IW}=2. The number of different weight patterns for minterms (CF, CFN and IW) combined with the different weight patterns for implicants (LRZ, RBC, and NRC) is calculated as $4^6 = 4096$. We have shown in [53] that it is possible to reduce the number of weight combinations to $(24)^2 = 576$. It has also been shown in [53] that the best weighted combination of criteria is as follows:

(a) *Minterm Selection:*

$w_{CF} = 0, w_{CFN} = 1, w_{IW} = 2$, weight pattern (012)

(b) Implicant Selection:

$w_{RBC} = 1, w_{NRC} = 0, W_{LRZ} = 0$, weight patter (100).

Using the patterns the patterns shown above, a combined weight patter is written shortly as 012-100.

Example 1: Consider the SUM output function of the 1-bit 4-valued adder building block. This building block receives two 4-valued inputs x_i and y_i and produces two outputs S_i and C_i. The SUM output S_i is given by the table shown in Fig. 2. We have used the WDC algorithm to synthesis the SUM function. The total number of implicants needed to synthesize the SUM function is eight. This is shown below using the MVL operators introduced in Section 1.

$$S_i(x_i,y_i) = 1\bullet x_i^{23}\bullet y_i^{33} \oplus 1\bullet x_i^{33}\bullet y_i^{23} \oplus 1\bullet x_i^{02}\bullet y_i^{11} \oplus 1\bullet x_i^{11}\bullet y_i^{02} \oplus 2\bullet x_i^{01}\bullet y_i^{22} \oplus 2\bullet x_i^{22}\bullet y_i^{01} \oplus 3\bullet x_i^{00}\bullet y_i^{33} \oplus 3\bullet x_i^{33}\bullet y_i^{00}$$

A benchmark consisting of 50000 randomly generated 4-valued 2-variable functions has been used to test the performance of the WDC. It was found that the (best) average number of product terms required to cover a given 4-valued 2-variable function is 7.24914. More details about the obtained results are included in Section 4, in which we hold a comparison between the DC-based heuristics.

x_i	y_i 0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Fig. 2. The SUM output S_i of a 1-bit 4-valued adder block

2.2 The Ordered Direct Cover (ODC) [53]

We have observed that different ordering of the Minterm and the Implicant criteria can result in different number of product terms for a given MVL function. Hence, in the ODC the ordering of the selection criteria for both minterm and implicant selection is taken into consideration while synthesizing a given MVL function. It should be noted that the minterm selection criteria are: Smallest CF, Smallest CFN, and Smallest IW. The implicant selection criteria are: Smallest RBC, Smallest NRC, and Largest LRZ. A number of different orderings of these criteria have been experimented with using the same benchmark consisting of 50000 randomly generated 4-valued 2-variable functions. It was found that the following criteria ordering: minterm selection: Smallest IW; implicant selection: Smallest RBC followed by Largest LRZ.

Example 2: We have applied the ODC in synthesizing the SUM output function of the 1-bit 4-valued adder building block (as explained in Example 1 above). The total number of implicants needed to synthesize the SUM function is eight (the same as in the case of the WDC). This is shown below using the MVL operators introduced in Section 1.

$$S_i(x_i,y_i) = 1\bullet x_i^{33}\bullet y_i^{23} \oplus 1\bullet x_i^{23}\bullet y_i^{33} \oplus 1\bullet x_i^{11}\bullet y_i^{02} \oplus 1\bullet x_i^{02}\bullet y_i^{11} \oplus 2\bullet x_i^{22}\bullet y_i^{01} \oplus 2\bullet x_i^{01}\bullet y_i^{22} \oplus 3\bullet x_i^{33}\bullet y_i^{00} \oplus 3\bullet x_i^{00}\bullet y_i^{33}$$

The same benchmark consisting of 50000 randomly generated 4-valued 2-variable functions which has been used in the case of WDC has been used to test the performance of the ODC. It was found that the (best) average number of product terms required to cover a given 4-valued 2-variable function is 7.20234. This is somewhat better than the results obtained using the WDC.

More details about the obtained results are included in Section 4, in which we hold a comparison between the DC-based heuristics.

3. Fuzzy based Direct Cover Algorithm

In DC-based algorithms, the implicants that should be included in the final synthesis of the given function are the ones that best cover the chosen set of minterms and such that all minterms are covered. The selection of a minterm is based on the quality of its CF, CFN, IW or a combination thereof. We observe that such condition can be translated into a *Fuzzy Logic Rule* of the form:

IF *a given minterm has a good CF criterion* ***OR*** *a good CFN criterion* ***OR*** *a good IW criterion,* ***THEN*** *it is a potential minterm to select.*

If a designer wants to emphasize a given criterion he/she will have to use a set of *Preference Rules* in order to give more significance to that particular criterion. In order to formulate the (PRs), preference terms need to be defined. These terms will be associated with the main linguistic terms. In the fuzzy rules, the linguistic terms ‘‘low’’, ‘‘high’’, and ‘‘good’’ have been used. A number of approaches to find preference terms and preference rules based on *Membership Functions* has been reported in the literature [20][40][41][43][44]. These approaches map a PR relation *P* to a fuzzy membership function μ_P in the range [0, 1] with ‘1(0)’ means the criterion is definitely preferred (not preferred).

There are two basic types of fuzzy operators. The operators for the intersection, interpreted as the logical “AND", and the operators for the union, interpreted as the logical “OR" of fuzzy sets. The intersection operators are known as triangular norms (t-norms), and union operator as triangular co-norms (t-co-norms or s-norms). In multi criteria decision problem, such as ours, this translates to two extremes which lead to the formation of overall functions. One extreme is the case requiring all criteria to be satisfied, which leads to the pure-AND-ing operation. On the other hand, when it is required to satisfy any of the criteria, the pure OR-ing operation is used. The formulation of multi criterion decision functions neither requires the pure “AND-ing" of t-norm nor the pure “OR-ing" of *s-norm*. The reason for this is the complete lack of compensation of *t-norm* for any partial fulfillment and complete submission of s-norm to fulfillment of any criteria.The use of ordered weighted averaging OWA operator has been used by a number of researchers in the solution of multi-objective problems [42]. The OWA category of operators allows easy adjustment of the degree of ‘‘ANDing’’ and ‘‘ORing’’ embedded in the aggregation. The ‘‘OR-like’’ and ‘‘AND-like’’ OWA for two fuzzy sets *A* and *B* are expressed in the following equations.

$$\mu_{A\cup B}(x) = \beta \times Max\ (\mu_A, \mu_B) + (1-\beta)\times \frac{1}{2}(\mu_A + \mu_B)$$

$$\mu_{A\cap B}(x) = \beta \times Min\ (\mu_A, \mu_B) + (1-\beta)\times \frac{1}{2}(\mu_A + \mu_B)$$

In the above equations μ represents the membership value in a fuzzy function; $\beta\in [0, 1]$ is a constant parameter, which represents the degree to which the OWA operator resembles the pure ‘‘OR’’ or pure ‘‘AND’’, respectively. Our proposed Fuzzy-based Direct Cover (FZDC) algorithm employs fuzzy rules (along with preferences) to select the best set of minterm and the most appropriate implicant covering each such that the whole function is covered. The goodness of a minterm (implicant) is examined using the abovementioned fuzzy rules and preferences. Looking at these rules, it is easy to deduce that we can use the ‘OR-‘like operator to aggregate all decision criteria. Tables 1 shows the mathematical formulae we introduced for each membership function in the minterm selection. Table 2 shows the same for implicant selection.

Table 1: Membership functions in Min terms selection.

Minterm Selection		
Technique	Criterion	Our Formulated Membership Function
DM [10]	CF	$\mu_{CF} = \begin{cases} 1 & if\ CF = 0 \\ \frac{Max_{CF} - CF}{Max_{CF}} & if\ 0 < CF < Max_{CF} \\ 0 & if\ other\ wise \end{cases}$
ND [8]	CFN	$\mu_{CFN} = \begin{cases} 1 & if\ CFN = 0 \\ \frac{Max_{CFN} - CFN}{Max_{CFN}} & if\ 0 < CFN < Max_{CFN} \\ 0 & otherwise \end{cases}$
BS [3]	IW	$\mu_{IW} = \begin{cases} 1 & if\ IW = Min_{IW} \\ \frac{Max_{IW} - IW}{Max_{IW} - Min_{IW}} & if\ Min_{IW} < IW < Max_{IW} \\ 0 & otherwise \end{cases}$

Table 2: Membership functions in implicant selection.

Implicant Selection		
Technique	Criterion	Our Formulated Membership Function
DM [10]	RBC	$\mu_{RBC} = \begin{cases} 1 & if\ RBC = Min_{RBC} \\ \frac{MAX_{RBC} - RBC}{Max_{RBC} - Min_{RBC}} & if\ MIN_{RBC} < RBC < Max_{RBC} \\ 0 & otherwise \end{cases}$
ND [8]	NRC	$\mu_{NRC} = \begin{cases} 1 & if\ NRC = Min_{NRC} \\ \frac{MAX_{NRC} - NRC}{Max_{NRC} - Min_{NRC}} & if\ MIN_{NRC} < NRC < Max_{NRC} \\ 0 & otherwise \end{cases}$
BS [3]	LRZ	$\mu_{LRZ} = \begin{cases} 1 & if\ LRZ = Max_{LRZ} \\ \frac{LRZ}{Max_{LRZ}} & if\ 0 < LRZ < Max_{LRZ} \\ 0 & otherwise \end{cases}$

Efficiency of the proposed fuzzy selection process is influenced by the value of β in OWA operator (see the μ equations above). In order to assess such influence, we have experimented with the effect of β on the results obtained using the FZDC algorithm. Table 3 summarizes such effect in terms of the average number of product terms needed to synthesize a given function from the 50000 benchmark functions.

Table 3: Effect of β

β	0.2	0.5	0.8	0.9
# PT	7.19526	7.1945	7.19358	7.1945

In addition to the impact of β there is also the impact of the fuzzy preference rules. Since there are three criteria for each of the minterm and implicant selection, there will be additional 6 parameters that need to be fine tuned to get the best performance of the algorithm. In order to obtain the best result using the proposed fuzzy-based selection criteria, the following set of experiments are conducted:

1. Experiments with different fuzzy operators
2. Experiments with different parameter values in the fuzzy operators

For the first experiment, the following fuzzy operators are used:

1. Max operator
2. Max operator with fuzzy preference rules
3. The Ordered Weighted Averaging (OWA) operator
4. OWA operator with fuzzy preference rules
5. Weighted Averaging (using fuzzy preference as the weight)

The fuzzy preference rules can be obtained by looking at the results of WDC and ODC algorithms. Using the information obtained from our work on these two approaches, we adopted some general rules:

(A) According to the WDC:

1. minterm selection process
   PM1a. IW is preferred as compared to other criteria
   PM1b. Either IW or CFN is preferred as compared to CF
   PM1c. If CF is preferred more than IW, then the preference of CFN should be greater than or equal to that of CF.
2. implicant selection process
   PL1a. RBC is strongly preferred as compared to other criteria
   PL1b. LRZ is slightly preferred as compared to NRC

(B) According to the ODC:

1. minterm selection process
   PM2a. IW is strongly preferred as compared to other criteria
   PM2b. CF is slightly preferred as compared to CFN
2. implicant selection process
   PL2a. RBC is strongly preferred as compared to other criteria
   PL2b. LRZ is slightly preferred as compared to NRC

As can be seen, the preference rules for implicant's selection are consistent in both WDC and ODC. However, this is not the case with minterm's selection. Although we can also deduce a general rule that IW is the strongly preferred criterion from PM1a and PM2b, there is some inconsistency in PM1b, PM1c and PM2b. However, since results of ODC are superior to those of WDC, we will use the preference rules deduced from ODC experiments. Table 4 shows the fuzzy preference used for the first experiment. It should be noted that we have used β = 0.5 for the OWA operator in the obtaining the results reported in Table 4.

Table 4: Fuzzy preference for minterm and implicant criteria

Minterm		**Implicant**	
Criteria	**Fuzzy Preference**	**Criteria**	**Fuzzy Preference**
IW	0.9	RBC	0.9
CF	0.2	LRZ	0.2
CFN	0.1	NRC	0.1

Using the abovementioned fuzzy preferences, we tested the proposed fuzzy selection criteria against the 50000 randomly generated 4-valued 2-variable benchmark MVL functions. Five different fuzzy operators are used for this purpose. Table 5 shows the results of the experiment.

Table 5: Performance of different fuzzy operators in FZDC.

Operator	**# PT** No CMV	**# PT** With CMV
Max	8.55136	8.03014
Max with pref.	7.30344	7.21964
OWA	7.38226	7.28898
OWA with pref.	7.27186	7.19450
Weighted average	7.30646	7.19784

It should be noted that we list the results obtained in two cases: not considering minterm values in any order (No CMV) and taking minterm values in ascending orders; lower to higher values (With CMV). The results show that taking the minterm value into consideration while selecting the next minterm to cover produces on average better results in terms of the average number of implicants needed to cover a given MVL function. This can be attributed to the use of the tsum operator as a connecting operator for the obtained implicants. This result is consistent with the observation made in [10].

Example 3: We have applied the FZDC in synthesizing the SUM output function of the 1-bit 4-valued adder building block (as explained in Example 2 above). The total number of implicants needed to synthesize the SUM function is eight (the same as in the case of the ODC).

This is shown below using the MVL operators introduced in Section 1.

$$S_i(x_i,y_i) = 1.\bullet x_i^{23}\bullet y_i^{33} \oplus 1.\bullet x_i^{33}\bullet y_i^{23} \oplus 1.\bullet x_i^{02}\bullet y_i^{11} \oplus 1.\bullet x_i^{11}\bullet y_i^{02} \oplus 2\bullet x_i^{01}\bullet y_i^{22} \oplus 2.\bullet x_i^{22}\bullet y_i^{01} \oplus 3.\bullet x_i^{00}\bullet y_i^{33} \oplus 3.\bullet x_i^{33}\bullet y_i^{00}$$

In addition, we have also applied the three reported DC algorithms, i.e. ARM, BS, and MD in synthesizing the SUM output function of the 1-bit 4-valued adder block. The results obtained are shown below.

Example 4: The results obtained in the case of ARM [15]

$$S_i\ (x_i, y_i) = 1\bullet {}^3X_1{}^3\bullet {}^2X_2{}^2 \oplus 1\bullet {}^1X_1{}^2\bullet {}^0X_2{}^0 \oplus 1\bullet {}^0X_1{}^1\bullet {}^1X_2{}^1 \oplus 1\bullet {}^2X_1{}^2\bullet {}^3X_2{}^3 \oplus 2\bullet {}^0X_1{}^0\bullet {}^2X_2{}^2 \oplus 1\bullet {}^2X_1{}^3\bullet {}^0X_2{}^0 \oplus 2\bullet {}^3X_1{}^3\bullet {}^3X_2{}^3 \oplus 1\bullet {}^1X_1{}^1\bullet {}^1X_2{}^2 \oplus 2\bullet {}^3X_1{}^3\bullet {}^0X_2{}^0 \oplus 3\bullet {}^2X_1{}^2\bullet {}^1X_2{}^1 \oplus 2\bullet {}^1X_1{}^1\bullet {}^2X_2{}^2 \oplus 3\bullet {}^0X_1{}^0\bullet {}^3X_2{}^3$$

Example 5: The results obtained in the case of BS [3]

$$S_i\ (x_i, y_i) = 1\bullet {}^3X_1{}^3\bullet {}^2X_2{}^3 \oplus 1\bullet {}^2X_1{}^3\bullet {}^3X_2{}^3 \oplus 1\bullet {}^0X_1{}^0\bullet {}^1X_2{}^3 \oplus 1\bullet {}^0X_1{}^0\bullet {}^2X_2{}^2 \oplus 1\bullet {}^1X_1{}^3\bullet {}^0X_2{}^0 \oplus 1\bullet {}^2X_1{}^2\bullet {}^0X_2{}^0 \oplus 2\bullet {}^1X_1{}^1\bullet {}^1X_2{}^2 \oplus 3\bullet {}^2X_1{}^2\bullet {}^1X_2{}^1 \oplus 2\bullet {}^3X_1{}^3\bullet {}^0X_2{}^0 \oplus 2\bullet {}^0X_1{}^0\bullet {}^3X_2{}^3 \oplus 1\bullet {}^1X_1{}^1\bullet {}^2X_2{}^2$$

Example 6: The results obtained in the case of DM [10]

$$S_i\ (x_i, y_i) = 1\bullet {}^2X_1{}^3\bullet {}^3X_2{}^3 \oplus 1\bullet {}^3X_1{}^3\bullet {}^2X_2{}^3 \oplus 1\bullet {}^0X_1{}^2\bullet {}^1X_2{}^1 \oplus 1\bullet {}^1X_1{}^1\bullet {}^0X_2{}^2 \oplus 2\bullet {}^0X_1{}^1\bullet {}^2X_2{}^2 \oplus 2\bullet {}^2X_1{}^2\bullet {}^0X_2{}^1 \oplus 3\bullet {}^0X_1{}^0\bullet {}^3X_2{}^3 \oplus 3\bullet {}^3X_1{}^3\bullet {}^0X_2{}^0$$

As can be seen synthesis of the SUM output function of the 1-bit 4-valued adder using the ARM [15] requires 11 PTs as compared to 8 PTs if the FZDC is used (a saving of 50%) while synthesis of the same function using the BS [3] requires 11 PTs as compared to 8 PTs using the FZDC (a saving of 37.5%). It is only in the case of DM [10] that the same number of PTs will be needed as in the case of the FZDC. A number of other experiments were conducted in order to further assess the performance of the FZDC. These are explained below.

4. Comparison

In order to assess the performance of the proposed FZDC algorithm with respect to the performance of existing DC-based techniques, we have tested the five DC-based algorithms (ARM [15], BS [3], DM [10], WDC [53], and ODC [53]) as well as the proposed FZDC algorithm using the set of benchmark consisting of 50000 4-valued 2-variable randomly generated functions. The criterion used for assessing these algorithms is the average number of implicants required to cover a given function. The results obtained are shown in Table 6.

As can be seen from Table 6 the proposed FZDC algorithm performs on average better than the other five DC-based algorithms. For more insight into the obtained results, we report in Table 7 the constituents of the 50000 4-valued 2-variable functions benchmark classified according to the number of minterms in each function generated. The average number of Product Terms (PTs) required to synthesis the functions in each category are listed in Table 7.

Table 6: Overall Comparison among different algorithms

Algorithm	Average Number of Product Terms needed
ARM [15]	7.89012
BS [3]	7.93882
DM [10]	7.24786
WDC [53]	7.24914
ODC [53]	7.20234
FZDC	7.19422

Table 7: Average # PTs using MVL Functions having Different Number of Minterms

#Minterms/ #Fuctions	ARM [15]	BS [3]	DM [10]	ODC [53]	WDC [53]	FZDC
16/500	7.594	7.562	7.002	7.086	7.01	6.946
15/2679	8.295	8.307	7.51	7.481	7.501	7.423
14/6589	8.355	8.405	7.569	7.516	7.559	7.500
13/10585	8.275	8.352	7.541	7.491	7.545	7.484
12/11230	8.049	8.098	7.382	7.33	7.383	7.320
11/9003	7.707	7.757	7.129	7.086	7.134	7.087
10/5434	7.323	7.366	6.831	6.787	6.837	6.794
9/2575	6.871	6.879	6.473	6.436	6.479	6.444
8/1038	6.309	6.322	6.023	5.978	6.022	5.981
7/277	5.726	5.751	5.527	5.484	5.523	5.505
6/75	5.133	5.147	4.973	4.96	4.987	4.960
5/13	4	4	4	3.923	4	4
4/1	4	4	4	4	4	4
3/1	3	3	3	3	3	3

The results shown in Table 7 reveal that the proposed FZDC algorithm performs better than the other DC-based algorithms in each category of functions within the 50000 benchmark.

In order to analyze further the results obtained, we have computed the maximum percentage improvement achieved by the FZDC algorithm with respect to any of the other DC-based algorithm. Table 8 provides such information. From Table 8 we can see that the maximum percentage of improvement achieved over all DC-based techniques is 12.0667 and this is achieved in 6589 functions out of the 50000 (about 13.2% of the benchmark functions).

Furthermore, we have categorized in Table 9 the improvement achieved by the proposed FZDC.

The results shown in Table 9 reveal that for about 62% (31083 functions) of the 50000 benchmark functions improvement greater than or equal to 10% have been achieved using FZDC. For about 37% (18550 functions) of the 50000 benchmark functions improvement between 5% and 10% has been achieved. This indicated that for

99.9 % (49985 functions) of the 50000 benchmark functions an improvement greater than or equal to 5% have been possible with the proposed FZDC algorithm. It is only in 15 functions out of the 50000 (about 0.003%) there has been no improvement.

Table 8: Maximum improvement achieved by FZDC

#Functions	ARM [15]	BS [3]	DM [10]	FZDC	Max.% Improvement
500	7.594	7.562	7.002	6.946	9.329
2679	8.295	8.307	7.51	7.423	11.909
6589	8.355	8.405	7.569	7.500	12.067
10585	8.275	8.352	7.541	7.484	11.598
11230	8.049	8.098	7.382	7.320	10.628
9003	7.707	7.757	7.129	7.087	9.454
5434	7.323	7.366	6.831	6.794	8.419
2575	6.871	6.879	6.473	6.444	6.750
1038	6.309	6.322	6.023	5.981	5.701
277	5.726	5.751	5.527	5.505	4.469
75	5.133	5.147	4.973	4.960	3.770
13	4	4	4	4	0
1	4	4	4	4	0
1	3	3	3	3	0

Table 9: The percentage of improvement achieved by FZDC

Functions with	**Number of functions**	**% age**
Improve. ≥ 12%	6589	≅ 13.2%
10% ≤ improve. <12%	(2679+10585+11230) = 24494	≅ 49%
< 10% improve. ≤ 5%	(500+9003+5434+2575+1038) = 18550	≅ 37.1%
< 5% improve.	(277+75)= 352	≅ 0.7%
no improvement	15	≅ 0.03%

5. Conclusions

In this paper, a Fuzzy-DC-based algorithm (FZDC) for synthesis of MVL functions has been proposed. The algorithm uses the principles of Fuzzy logic in order to make the selection of minterms and the appropriate implicants needed to cover them in synthesizing a given MVL function. The performance of the proposed FZDC algorithm in terms of the average number of product terms (implicants) required to synthesize a given MVL function from a set of benchmark consisting of 50000 4-bvalued 2-variable functions has been assessed. In making such assessment we have compared the proposed FZDC algorithm with five other existing DC-based algorithms. The results obtained using the introduced FZDC algorithm were compared to those obtained using existing DC-based techniques. It has been shown that the proposed algorithm performs better than the other existing DC-based algorithms. In particular it has been found that the proposed algorithm achieved a maximum of about 12% improvement in the number of implicants required in synthesizing 6589 functions (about13.2% of the benchmark functions). The FZDC achieved improvement of greater than or equal to 10% in 31283 functions (about 62% of the benchmark functions). It has also been shown that in 49633 functions (about 99% of the benchmark functions) an improvement of greater than or equal to 5% is possible using the proposed FZDC algorithm. We have also shown the applicability of the FZDC algorithm in synthesizing the SUM output of a 1-bit 4-valued adder circuit. In this later case, it was found that the FZDC algorithm produces results that are as good as those produced by our previously proposed ODC & WDC [53] and the algorithm reported in [10].

Acknowledgment

The author would like to acknowledge the contribution made by Mr. Bambang Sarif in an earlier draft of the paper.

References

[1] Mishchenko, B. Steinbach, M. Perkowski, ”Bi-decomposition of multi-valued relations”. Proc. International Workshop on Logic and Synthesis 2001, pp. 35-40.
[2] Mishchenko and R. Brayton, ”Simplification of non-deterministic multi-valued networks”, Proc. International Conference on Computer Aided Design 2002, pp.557-562.
[3] Besslich P. W. "Heuristic Minimization of MVL functions: A Direct Cover Approach." IEEE Transactions on computer, Feb. 1986, pp.134-144.
[4] Zupan, B., Machine Learning Based on Functional Decomposition. PhD thesis, University of Ljubljana, Slovenia, 1997.
[5] Files, C., R. Drechsler, and M. Perkowski. Functional decomposition of MVL functions using multi-valued decision diagram. International Symposium on Multi-Valued Logic (ISMVL), May 1997, pages 27–32.
[6] Files, C. and M.A Perkowski, “New multi-valued functional decomposition algorithms based on MDDs”. IEEE Trans. CAD. Vol. 19(9), Sept. 2000, pp. 1081-1086.
[7] Files, C. and M. A Perkowski, “Multi-valued functional decomposition as a machine learning method”, Proc. ISMVL '98, pp. 173 -178.
[8] Yang, C. and Y.-M. Wang. A neighborhood decoupling algorithm for truncated sum minimization. Proceedings of the 20th ISMVL, 1990, May 1990, pp. 153-160.
[9] Yildirim, C., J. T. Butler and C. Yang, "Multiple-valued PLA minimization by concurrent multiple and mixed simulated annealing," Proceedings of the 23rd ISMVL, May 1993, pp. 17-23.
[10] Dueck, G. and Miller, D. M., "A Direct Cover MVL Minimization Using the Truncated Sum." Proceeding of the 17th ISMVL, May 1987, pp. 221-227.
[11] Olmsted, D., “The History and Principles of Multi-valued Logic”, 1998.
[12] Miller, D., Decomposition in many valued design. PhD thesis, University of Manitoba, May 1976.
[13] Dubrova, E., Y. Jiang, R. Brayton, “Minimization of multi-valued functions in Post algebra”, Proc. International Workshop on Logic and Synthesis 2001, pp. 132-137.

[14] Post, E., "Introduction to a general theory of elementary propositions", American Journal of Mathematics, Vol. 43, 1920, pp. 163-185.
[15] Promper, G. and J. A. Armstrong. Representation of multiple valued functions using the direct cover method. IEEE transactions on Computers, September 1981, pp. 674-679.
[16] Dueck, G. and G. H. J. van Rees, "On the Maximum Number of Implicants Needed to Cover a Multiple-Valued Logic Function Using Window Literals", Proceedings of the 21st ISMVL, May 1991, pp. 280-286.
[17] Epstein, G., G. Frieder and D. C. Rine, "The development of Multiple-valued logic as related to computer science", Computer, Vol. 7 (9), 1974, pp. 20-32.
[18] Goa, J.-H. and R. Brayton. Optimization of multi-valued multi-level networks. Proceedings of the International Symposium on Multiple-Valued Logic, Tahoe City, June 2002.
[19] Lukasiewicz, J. english translation: On three valued-logic, in L. Borkowski (ed), Select Works, North-Holland, Amsterdam, Vol. 5, 1920, pp. 169-171.
[20] Zadeh, L., "Fuzzy Sets, Information and Control, 8(3): 338-353, June 1965.
[21] Miller, D., "Spectral transformation of multiple-valued decision diagrams", Proc. 24th ISMVL, May 1994, pp. 89-96.
[22] Miller, D., Drechsler, R., "Implementing a multiple-valued decision diagram package", Proc. 28th ISMVL, May 1998, pp. 52-57.
[23] Abd-El-Barr, M. and Louai Al-Awami, "Analysis of Direct Cover Algorithms for Minimization of MVL Functions", Dec. 2003, pp. 308- 312.
[24] Kameyama, M., S. Kawahito, T. Higuchi, " A Multiplier chip with Multiple-Valued Bidirectional Current-Mode Logic Circuits", IEEE Computer, April, 1988, pp. 43-56.
[25] Perkowski, M., M. Marek-Sadowska, L. Jozwiak, T. Luba, S. Grygiel, M. Nowicka, R. Malvi, Z. Wang, J. S. Zhang, "Decomposition of multiple-valued relations". Proc. 27th ISMVL, May 1997, pp. 13-18.
[26] Cheung, P. and D. M. Purvis. A computer-oriented heuristic minimization algorithm for multiple-output multivaued switching functions. Proceeding of the 4th ISMVL, May 1974, pp. 112-120.
[27] Drechsler, R., Dragan Jankovi´c Radomir S. Stankovi´c, Generic Implementation of DD Packages in MVL, EUROMICRO Conference, 1999. Proceedings. 25th, Vol. 1, 1999, pp. 352-359.
[28] Drechsler, R. Mitch Thornton David Wessels, MDD-Based Synthesis of Multi-Valued Logic Networks, Proceedings 30th ISMVL, May 2000, pp. 41-46.
[29] Hong, R., S. J. and D. L. Ostapko. Mini: A heuristic approach for logic minimization. IBM J. Res. Develop., 1974, 18:443-458.
[30] Brayton, R. and S. P. Khatri. Multi-valued logic synthesis. International Conference on VLSI Design, Goa, India, Jan 1999.
[31] Rudell, R. and A. Sangiovanni-Vincentelli, "Multiple-valued minimization for PLA optimization". IEEE Trans. CAD, vol. 6(5), Sep. 1987, pp. 727-750.
[32] Stankovi´c, R.S., "Functional decision diagrams for multiple-valued functions", Proc. 25th ISMVL, May 1995, pp. 284-289.
[33] Tirumalai, P. and Butler, J. "Analysis of Minimization Algorithms for Multiple-Valued Programmable Logic Arrays." Proceedings 28th ISMVL, May 1988, pp. 226-236.
[34] Tirumalai, P. Multiple-valued programmable logic arrays. Ph.D. Thesis, Northwestern University, Evanston, USA, 1989.
[35] Luba, T., Decomposition of multiple-valued functions. Proceedings 25th ISMVL, May 1995, pp. 256–261.
[36] Sasao, T., "On the Optimal design of Multiple-Valued PLA's," IEEE Trans. Comput., C-38, 4, 1989, pp. 582-592.
[37] Sasao, T. and J.T. Butler. Planar multiple-valued decision diagrams. Proceedings 25th ISMVL, May 1995, pp. 28–35.
[38] Naiff, K., D. A. Rich and K. G. Smalley, "A Four-State ROM Using Multilevel Process Technology", IEEE Journal of Solid-State Circuits, Vol. 19, No. 2, April 1984, pp. 174–179.
[39] Razavi, H., S. E. Bou-Ghazale, "Design of a Fast CMOS Ternary Adder", Proceedings 17th ISMVL, May 1997, pp. 20–23.
[40] Zadeh, L., Fuzzy Sets, Information and Control 8 (1965) 338-353.
[41] Shragowitz, E., J. Lee, E. Kang, Application of Fuzzy Logic in Computer-aided VLSI Design, IEEE Transactions on Fuzzy Systems 6(1) (1998) 163-172.
[42] Yager, R., On Ordered Weighted Averaging Aggregation Operators in Multi-criteria Decision Making, IEEE Transactions on Systems, Man, and Cybernetics 18(1), 1988, pp. 183-190.
[43] Kacprzyk, J., M. Fedrizzi, H. Nurmi, Group decision making and consensus under fuzzy preferences and fuzzy majority, Fuzzy Sets and Systems 49, 1992, pp. 21-31.
[44] Marimin, M. Umano, I. Hatono, H. Tamura, Linguistic labels for expressing fuzzy preference relations in fuzzy group decision making, IEEE Transactions on Systems, Man, Cybernetics, Part B 28 (2) (1998) 205-218.
[45] Dubrova, E., "Multiple-Valued Logic in VLSI", Multiple-Valued Logic: An International Journal, 2002, pp. 1-17.
[46] Current, K., "Multiple-Valued Logic Circuits", Computer Engineering Handbook, 2nd Edition, CRC Press 2008, Vojin Oklobzija (Editor), Digital Design and Fabrication, Chapter 8, pp. 1-25.
[47] Khan, M., "Synthesis of quaternary reversible/quantum comparators", Journal of System Architectures, vol. 54, no. 10, October 2008, pp. 977-982.
[48] Kawahito, S., Kameyama, M., Higuchi, T., and Yamada, H., A 32×32-bit multiplier using multiple-valued MOS current-mode circuits", IEEE Journal of Solid-State Circuits, vol. 23(1), Feb. 1988, pp. 124-132.
[49] Kim, J., and Ahn, S., "High-speed CMOS de-multiplexer with redundant multi-valued logic", International Journal of Electronics, 94, 2007, pp. 915-924.
[50] Tirumalai, P. and Butler, J., "Minimization Algorithms for Multiple-Valued Programmable Logic Arrays", IEEE Transactions of Computers, vol. 40, no. 2, Feb. 1991, pp. 167-177.
[51] Intel Strata TM Flash. Available at http://www.intel.com/design/flash/isf/overview.pdf

[52] Okuda, T. and Murotani, T., "A four-level storage 4GB DRAM", IEEE Journal of Solid State Circuits, vol. 32, no. 11, 1997, pp. 1743-1747.

[53] Abd-El-Barr, M. and Bambang Sarif, "Weighted and Ordered Direct Cover Algorithms for Minimization of MVL Functions", Proceedings of the IEEE 37th International Symposium on Multiple-Valued Logic (ISMVL07), Oslo, Norway, May 2007, pp. 48-52.

Mostafa Abd-El-Barr (Senior Member of IEEE) has received his B.Sc. and M.SC. degrees in Electrical Engineering (Computer Track), Cairo University, Egypt in 1973 and 1979, respectively. He received his PhD degree in Computer Engineering, University of Toronto, Ontario, Canada in 1986. In July 1986 he joined the faculty of the Department of Computer Science, University of Saskatchewan (U. of S.), Saskatoon, Canada, where he was promoted to the rank of full professor in July 1993. He was also an associate member of the Department of Electrical Engineering, U. of S. In September 1996, he joined the Department of Computer Engineering at King Fahd University of Petroleum and Minerals (KFUPM), Saudi Arabia as a full professor. In September 2003 he was appointed as the founding chairman of the Information Science Department, Kuwait University (KU). In September 2005 he was appointed as the Vice Dean for Academic Affairs, CFW, KU. He is currently an Adjunct Professor with the Department of Electrical and Computer Engineering, University of Victoria, BC, Canada. His research interests include Design and Analysis of Reliable & Fault-Tolerant Computer Systems, Computer Networks Optimization, Parallel Processing/Algorithms, Information and Computer Systems Security, Beyond-Binary Logic System Design & Analysis, VLSI System Design & Algorithms, and Digital Systems Testing. He is the author and/or co-author of more than 150 scientific papers published in journals and conference proceedings. He is a certified professional engineer in the Province of Ontario, Canada.