Technology Mapping for FPGAs with Complex Block Architectures by Fuzzy Logic Technique*

Jun-Yong Lee

Department of Computer Science
University of Minnesota
Minneapolis, Minnesota 55455
Tel: 612-626-7511
Fax: 612-625-0572
e-mail: jlee@cs.umn.edu

Eugene Shragowitz

Department of Computer Science
University of Minnesota
Minneapolis, Minnesota 55455
Tel: 612-625-3368
Fax: 612-625-0572
e-mail: shragowi@cs.umn.edu

Abstract— This paper describes a technology mapper for FPGAs with the complex structure of logic blocks. Most technology mappers developed so far are not effective for such complex logic block architectures as XILINX XC4000 series. The proposed mapper applies a constructive mapping algorithm and fuzzy logic rules to balance such criteria as area, timing, routability and others. Performance of the mapper is demonstrated on the set of MCNC benchmarks.

I. Introduction

FPGAs (Field Programmable Gate Arrays) are among the most popular types of ASICs. In SRAM-based FPGAs, which are the focus of this paper, any k-input Boolean function is implemented by using 2^k bits of look-up table.

Recently, many technology mapping algorithms were developed by universities and by industry. Among them are Chortle-crf [4], Chortle-d [3], Flowmap [9], mis_pga [7], VISMAP [6], XILINX's XNFMAP and PPR [11], and others.

The majority of technology mapping methods are designed to map a Boolean network into the XILINX XC3000 FPGA circuit structure, where each CLB has 2^5 bits of SRAM LUT capable of implementing Boolean expressions with up to 5-inputs. As the FPGAs has become more popular, the FPGA architecture has evolved. In the new SRAM-based FPGA architectures, such as XILINX XC4000 [2] or AT&T's ORCA [5], a configuration of a single CLB becomes more complex.

For example, some substantial limitations are imposed on the mapping procedure for XILINX XC4000 FPGAs. Two of three inputs to the upper-level look-up table (H-LUT) should be outputs of the lower-level look-up tables (F-LUT and G-LUT). The second important restriction is that only two outputs are allowed in one CLB, either directly or via flipflops. Restrictions mentioned above limit direct applicability and efficiency of the many previously developed algorithms to this new architecture.

In this paper the MOFL (Multi-criteria Optimization using Fuzzy Logic) technology mapping algorithm for complex CLB structures, particularly for the XILINX XC4000 family of FPGAs, is introduced. The input to the MOFL technology mapping procedure is a DAG of Boolean expressions. Prior to the technology mapping, the SIS logic optimizer [1] is applied to logic input.

The proposed algorithm utilizes memory resources of CLBs in order to obtain better timing performance without sacrificing area.

It is well known that timing is one of the main weakness of FPGAs. But often it is very important to obtain the best timing performance without substantial losses in area. This work addresses this aspect of the mapping procedure.

Most of the known mapping algorithms [4], [3], are based on decomposition of the input network, presented usually in a graph form, into a set of fanout-free trees or a set of small pieces of graphs. When the network is decomposed into a set of trees, a technology mapper loses the flexibility of mapping over the boundaries of trees. The proposed technology mapper does not decompose the DAG network into trees but directly uses the graph as a source structure for the technology mapping algorithm to retain flexibility in optimizing delays.

The MOFL technology mapping algorithm applies fuzzy logic for mapping of gates to CLBs. The fuzzy logic approach was successfully used for such CAD applications as placement for standard cell [13] and sea-of-gate design styles [12].

II. Fuzzy Logic Background

A classical (crisp) set is normally defined as a collection of elements or objects. For a given set $A \subseteq X$, each element $x \in X$ either belongs to the set or does not belong

*This work was supported in part by the National Science Foundation under the Grant MIP-9123945.

to the set, where X is a set of elements denoted generically by x.

But fuzzy sets are different from classical sets in that they allow partial or gradual degrees of membership. A *fuzzy set* $\tilde{A}$ over X is defined as a set of ordered pairs:

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) \,|\, x \in X\},$$

where $\mu_{\tilde{A}}(x)$ is a membership function which associates with every member x of X a number in the interval [0,1], representing the degree of membership of x in A.

To represent fuzzy sets, linguistic variables are generally used [8]. The values of linguistic variables are not numbers but words or sentences in a natural or artificial language. For example "age" is a linguistic variable whose values can be "young", "old", "very old" and so on. Fig. 2 represents an example of membership function "young" defined graphically for the linguistic variable "age".

Fuzzy decisions are made using plain language rules describing relations between linguistic variables and thier values defined by membership functions. Rules are stated in a simple IF/THEN format and input values can be combined using **and** or **or** operations. *Min* and *max* are most commonly used functions for fuzzy *intersection* (*and*) and *union* (*or*) operators, but other functions were also proposed [15] for more sophisticated applications. (See Sec. III.A)

III. Multi-criteria Optimization Using Fuzzy Logic

In many cases, solutions of engineering problems require an achievement of several goals simultaneously. In FPGA mapping, there are several objectives to be achieved: area, delay, routability and others. All these objectives can not be opmitized in the same time, because they may contradict to each other. Thus, the best compromise for multiple objectives is a solution which balances them.

The multi-criteria optimization problem can be presented as selection of $x \in X$ which optimizes the decision

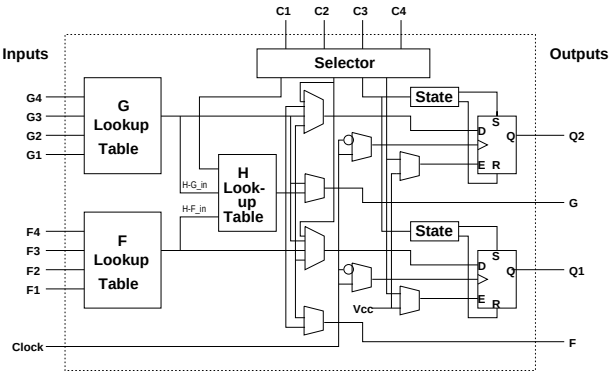

Fig. 1. The architecture of XC4000 FPGA CLB

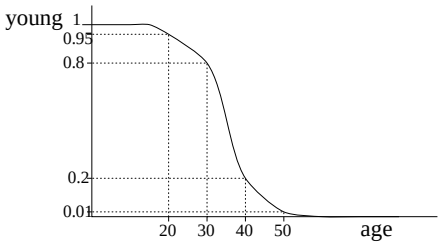

Fig. 2. Membership function "young" for the linguistic variable "age"

function $D(x)$

$$D(x) = f(C_1(x), C_2(x), \cdots, C_n(x))$$

where X is a solution space and C_i's are criteria defined over the set X.

In the proposed MOFL system, the multi-criteria optimization procedures are used for selection of nodes to be mapped into each CLB. The decision function $D(x)$ is implemented by a multilevel function of fuzzy logic operators defined by fuzzy rules. Fuzzy logic model is selected because classical models are poorly suited for a solution with multiple criteria and varying degrees of importance of different objectives during design process.

In the following subsections, fuzzy logic operators used in this work and the methodology of balancing multiple criteria of varying importance are described in detail.

A. Application of compensatory fuzzy And / Or operators

Several methodologies have been proposed for multi-criteria decision-making [16]. Most of them use single aggregation operators such as *min*, weighted sum, or γ-operator [10] to devise a decision function from multiple criteria.

The decision-making process can be represented in fuzzy logic form by a set of IF/THEN rules, which include combinations of *and* and *or* operators. Such representation allows to reflect complex interactions between goals in decision-making.

In many fuzzy logic applications, *min* and *max* operators have been used as fuzzy *intersection* (*and*) and *union* (*or*). But *min* and *max* are not the only possible operators to model *intersection* and *union*. The operators in fuzzy logic are divided into two categories: compensatory operators and non-compensatory operators.

For the non-compensatory operation, the weaker element of the operator cannot compensate for the stronger element. For instance, in *max* operation, which is a non-compensatory operation, the criteria with smaller value cannot influence the result of the *max* operation. *Min* is also a non-compensatory operation, while the *product* and *algebraic sum* are.

In multi-criteria optimization, it is important to make use of all the information for every criterion. This goal can be better served by using *intersection* and *union* operators defined by Dubois & Prade [15]. These operators provide compensatory results according to the value of parameter α.

The *intersection* and *union* of two fuzzy sets $\tilde{A}$ and $\tilde{B}$ are defined as

$$\mu_{\tilde{A}\cap\tilde{B}}(x) = \frac{\mu_{\tilde{A}}(x)\cdot\mu_{\tilde{B}}(x)}{\max\{\mu_{\tilde{A}}(x),\mu_{\tilde{B}}(x),\alpha\}}, \quad (1)$$

and

$$\begin{aligned}\mu_{\tilde{A}\cup\tilde{B}}(x) &= \frac{\mu_{\tilde{A}}(x)+\mu_{\tilde{B}}(x)-\mu_{\tilde{A}}(x)\cdot\mu_{\tilde{B}}(x)}{\max\{(1-\mu_{\tilde{A}}(x)),(1-\mu_{\tilde{B}}(x)),\alpha\}} \\ &- \frac{\min\{\mu_{\tilde{A}}(x),\mu_{\tilde{B}}(x),(1-\alpha)\}}{\max\{(1-\mu_{\tilde{A}}(x)),(1-\mu_{\tilde{B}}(x)),\alpha\}} \quad (2)\end{aligned}$$

where for $\alpha \in [0,1]$

Operators in this form present the case of dual logic, which enables to combine fuzzy logic and crisp logic. Fuzzy logic operations defined according to (Eq. 1) and (Eq. 2) are used in the decision-making process described in the following sections.

B. Ranking criteria using preference rules

The degree of importance associated with each criterion may vary during a design process. There are many methods to assign a numerical value to importance of criteria. This process is called 'ranking'.

The most popular method of ranking is a method of weighting coefficients. In this method, averaging operators are used as aggregators of criteria. But ranking effective for one operator may not work for other aggregation operators.

In the proposed system, this method of weighting exponents is applied. The *intersection* operator (Eq. 1) is composed of the $\min(\mu_{\tilde{A}},\mu_{\tilde{B}})$ and the algebraic product $\mu_{\tilde{A}}\cdot\mu_{\tilde{B}}$. The method of assigning larger exponents to the more important criteria works also properly for the algebraic product operation. The variable with the larger exponent receives a bigger influence on the product, as it is expected.

For the *union* operator (Eq. 2), the way of weighting exponents is different. The result of *union* operator, composed of *max* and algebraic sum, is more influenced by the larger value of the participating criteria. Therefore, a smaller exponent should be given as a weight to the more important criteria. This makes the membership grade of the resulting *union* in the solution set of decision function D more dependent on the important criteria.

IV. Technology Mapping Strategies

A. Overview of a Mapping Algorithm

The purpose of our algorithm is to map a given DAG of Boolean equations to an FPGA format in a way that satisfies multiple objectives, with preferences given to some of them. In the proposed mapping algorithm, a heuristic is used in a constructive manner.

From a given DAG, an algorithm selects the best node to start mapping to a CLB. In this selection process, a set of fuzzy logic criteria is used for choosing the best candidate.

Starting from the seed node, the mapping is continued to the nodes connected to the already mapped node in the CLB until no more node can be mapped into the CLB. Among the three LUTs of CLB, H-LUT is packed first with the seed node, and then F-LUT and G-LUT are packed.

The procedure `H_map` is used to map the chosen seed node to H-LUT. The procedure `H_map` continues mapping of nodes connected to the first node until the capacity of H-LUT is exhausted; then it selects nodes to start mapping to F-LUT and G-LUT in the current CLB. The outputs of the selected nodes, which are called H-F_in and H-G_in, are connected to H-LUT as depicted in Fig. 1. The procedure `F_map` is used to map nodes to both F-LUT and G-LUT, which are functionally identical.

The procedures `H_map` and `F_map` select nodes to continue mapping using fuzzy logic rules similar to those used in the selection of a seed node for a new CLB. Details of a fuzzy logic implementation will be discussed in the following sections.

After mapping is completed for one CLB, a new seed node is chosen for a new CLB, and the above sequence is repeated until all the nodes of the given DAG are mapped into a CLB network.

However, there are some cases when the above sequence cannot result in efficient utilization of the CLB capacity. For instance, when the subnet for the chosen node is small enough to fit into a single F-LUT, it is more efficient to map the subnet into a single F-LUT than to use the entire CLB of three LUTs.

In another case when the nodes connected to the chosen node have complex interconnections, the restrictions due to the CLB structure can make it difficult to continue mapping from the originally chosen seed node and may result in poor utilization of the CLB.

In these cases, the procedure `F_map` is called to map the seed node and some other nodes connected to it into a single F-LUT.

Therefore, once a seed node is chosen for mapping into a new CLB, it is examined to see whether the CLB could be fully utilized by mapping with the nodes connected to the seed node. If efficient utilization of CLB is not likely, the algorithm proceeds to mapping into a single F-LUT, leaving the other LUTs for future usage. Fig. 3 represents this mapping algorithm.

B. Criteria Used in Mapping Procedures

The following criteria are used to select a node to start packing a new CLB or to select the next node to continue

```
V ← all nodes in the network

While ( V ≠ φ ) {
  Choose a seed node n from V which optimizes D(x)
    /* D(x) is a multi-criteria decision function and x is an element
    in the solution space X */
  If a subnet from the node n utilizes the whole CLB,
   {
    Call H_map for node n
    /* procedure H_map packs H-LUT for node n */
    Call F_map for node H-F_in for F-LUT
     /* H-F_in is the next node to map obtained as a result of previous execution of H_map */
    Call F_map for node H-G_in for G-LUT
     /* H-G_in is the next node to map obtained as a result of previous execution of H_map */
    }
   else {
    Call F_map for node n
    /* procedure F_map packs F-LUT (or G-LUT) for node n */
    }
  Remove all mapped nodes from V
  }
```

Fig. 3. The pseudo-code of mapping algorithm

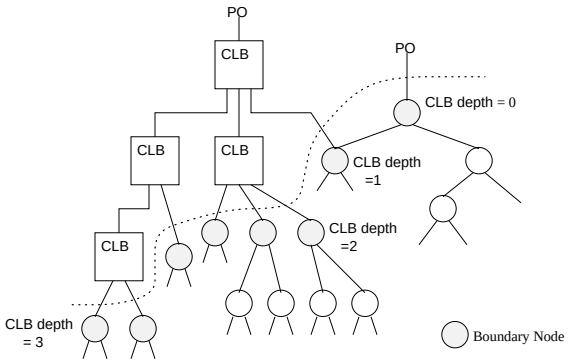

Fig. 4. The boundary nodes

packing H-LUT, F-LUT or G-LUT in already selected CLB. Definitions for many of these criteria were given in [14].

Maximal path length (C1) A path from a primary input (PI) to a primary output (PO) that corresponds to the maximal number of nodes in a DAG is called a critical path.

Level of node (C2) The level of the node helps to estimate maximal delay from PIs to the output of the node. A node with a higher level is likely to produce a larger delay. This criterion gives another measure for delay optimization, along with the maximal path length.

Location of node (C4) & CLB depth (C3) A boundary node is an unmapped primary output node or an unmapped node whose output is directed to the input of CLB. Fig. 4 provides examples of boundary nodes for partial mapping. Initially all primary outputs (POs) are boundary nodes.

Size of a subnet for a node (C5) Size of a subnet for a node affects delay from output of the node from PIs. A node with a larger subnet size requires a larger number of CLBs, causing larger timing delay and area.

Number of supports (C6) The number of PIs supporting the subnet of the node is a relative barometer to determine the complexity of a subnet. The more complex subnet is more likely to be mapped into the larger network of CLBs.

Number of fanouts (C7) The number of fanouts is another criterion when the area optimization is emphasized. In the proposed mapping algorithm, nodes are duplicated when they are needed for delay reduction. Nodes with larger fanouts are more likely to be duplicated and may cause an increase in the number of CLBs.

Number of fanins (C8) The number of fanins is usually used to measure a node size. In order to pack the CLBs efficiently, a larger node should be mapped into the CLB first. Otherwise, larger nodes should be partitioned into many smaller nodes, and more CLBs will be needed to cover all nodes.

C. Fuzzy Logic Rules for Selection of Seed Node

Criteria defined in the previous section are used to formulate rules applied to find a seed node for CLB mapping and to continue mapping for H-LUT, F-LUT or G-LUT. For each step of a process, a decision function $D(x)$ is applied to choose a node from DAG that maximizes the value of $D(x)$.

Fig. 5 lists these objectives and criteria used to define the decision function to find a seed node for a CLB. A value of each criterion is obtained from one or more fuzzy logic membership functions. When more than one membership function are used to compute the upper level criterion, the fuzzy logic rules are implemented by operations defined in the previous section.

The criteria for subobjectives from the same hierarchical level are merged by fuzzy logic operations, with weights assigned to each criterion. These subobjectives are then combined by another fuzzy logic operation and supplied to the upper level. The following fuzzy rules and preference rules are used to formulate the multi-criteria decision function $D_1(x)$, which is used to select a seed node for a new CLB. Those procedures are described in top-down manner from main objectives to basic criteria.

- **Fuzzy Rule 1 (FR 1):** If a node from the DAG provides a large decrease in timing delay **and** a small number of new CLBs, then a node is a good candidate to be mapped into CLBs.

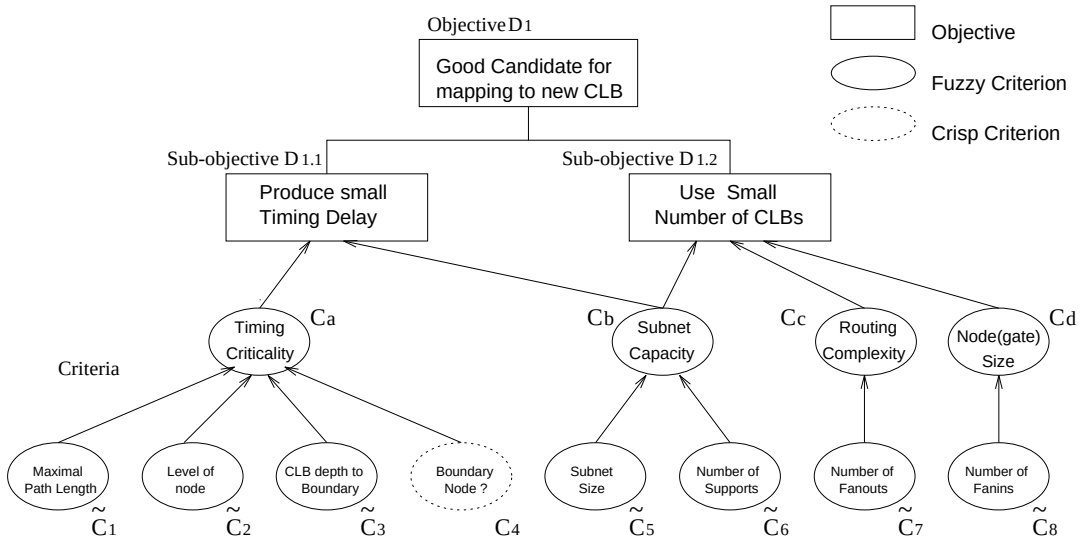

Fig. 5. The objectives and constrains used to choose a seed node for a CLB

Fuzzy Rule 1 is modified by the Preference Rule (PR 1) to obtain a formula for computing a value of decision function D.

- **Preference Rule for Fuzzy Rule 1 (PR 1):** The decrease of timing delay has mild preference over the number of CLBs.

These rules are stated to choose an $x \in X$ which maximizes the value of decision function $D_1(x)$

$$\begin{aligned} D_1(x) &= f(D_{1.1}(x), D_{1.2}(x)) \\ &= D_{1.1}(x)^{w_{11}} \cap D_{1.2}(x)^{w_{12}} \end{aligned}$$

where exponents w_{11} and w_{12} express preferences defined by PR 1. $D_{1.1}(x)$ and $D_{1.2}(x)$ are defined on the lower hierarchical levels as follows:

- **Fuzzy Rule 1.1 (FR 1.1):** If a node produces critical timing **and** a subnet for the node is large, then it is a good candidate to secure small timing delay in mapping.

- **Preference Rule for Fuzzy Rule 1.1 (PR 1.1):** In the early stage of mapping, timing criticality (C_a) has strong preference over subnet capacity (C_b).

These rules are stated to choose an $x \in X$ that maximizes the value of decision function $D_{1.1}(x)$

$$\begin{aligned} D_{1.1}(x) &= f(C_a(x), C_b(x)) \\ &= {C_a}^{w_{111}} \cap {C_b}^{w_{112}} \end{aligned}$$

The PR 1.1 defines the weights for subobjectives by exponents:

$$w_{111} = 1 + 2 \cdot \mu_{\tilde{k}}, \; w_{112} = \frac{1}{1 + 2 \cdot \mu_{\tilde{k}}} \; ;$$

$\mu_{\tilde{k}}$ is a membership function for linguistic variable *early stage*. Thus,

$$D_{1.1}(x) = {C_a}^{1+2\cdot\mu_{\tilde{k}}} \cap {C_b}^{\frac{1}{1+2\cdot\mu_{\tilde{k}}}}$$

- **Fuzzy Rule 1.2 (FR 1.2):** If a subnet of a node is large **and** its routing is complex **and** gate size is large, then it is a good candidate to secure a small number of CLBs in mapping.

Fuzzy Rule 1.2 is also modified by a preference rule in the similar way. The subobjectives $D_{1.1}$ and $D_{1.2}$ are functions of several criteria, and each criterion is produced from one or more lower level criteria by fuzzy logic operations according to the fuzzy rule. For example:

- **Rule for criterion C_a:** If (the maximal path length for a node is long **and** level of the node is large) **or** (the node is a boundary node **and** the CLB-depth to the node is large), then the node has critical timing.

The criterion function C_a is

$$C_a(x) = \mu_{(\bar{C_1} \cap \bar{C_2}) \cup (\bar{C_3} \cap C_4)}(x)$$

where $\mu_{\bar{C_1}}$, $\mu_{\bar{C_2}}$, and $\mu_{\bar{C_3}}$ are membership functions for the linguistic variables *long maximal path length*, *high level* and *large CLB depth* respectively. $\mu_{\bar{C_4}}$ is a crisp (classical) logic function of either 1 or 0, depending on whether the node is a boundary node or not. Hence,

$$C_a(x) = \begin{cases} \mu_{(\bar{C_1} \cap \bar{C_2}) \cup \bar{C_3}}(x) & \text{if } x \text{ is a boundary node} \\ \mu_{(\bar{C_1} \cap \bar{C_2})}(x) & \text{otherwise} \end{cases}$$

The mapping procedures used for H-LUT and F-LUT (G-LUT) are slightly different from one used for selection of a node for a new CLB in the usage of criteria. But the main principles are the same. Descriptions of these subsystems are omitted due to the space limitations.

V. Experimental results

The proposed MOFL technology mapper was implemented with approximately 7000 lines of C code. In order to demonstrate performance of the mapper, 25 benchmark circuits were selected from the set of MCNC test cases. All test cases were chosen from the combinational multilevel examples.

The test cases were first optimized by the SIS logic optimizer [1] and then were processed by the MOFL technology mapper and the XILINX's technology mapper. Performance of a technology mapper can be effectively evaluated after placement and routing. For these reasons, the results of mapping were submitted to the XILINX's layout system.

XILINX's FPGA development system, called XACT, includes various subsystems such as FPGA design editor, technology mapper, placement and routing system, timing analyzer and others. Among them, the Partition, Placement, and Route (PPR) program performs technology mapping and layout for the XILINX XC4000 series FPGAs. It was observed that the updated version of the XILINX mapper emphasised routability and timing performance rather than area. It does not pack CLBs as tightly as possible to improve routability and timing.

Results presented in Table 1, obtained by the MOFL and completed by the XILINX layout system, correspond to the strategy of balancing delay and area with mild preference given to delay. The number of CLBs in Table 1 is the actual number of CLBs used to obtain the timing performance listed in the same table. For the MOFL mapper, the number of CLBs could be further reduced if area optimization is emphasized.

For the 25 test cases, the MOFL technology mapper used on average 2.3% more CLBs than XILINX. In contrast, the number of maximal CLB levels was reduced by 27.7%, which resulted in 11.6% reduction of the maximal delay of circuits after layout. The difference between ratios of reduction in numbers of levels and delay is mainly due to input and output circuitry, which is independent of the technology mapping procedure. Run time for the MOFL mapper does not exceed the run time of the XILINX mapper.

TABLE I
Experimental Results

Circuit Name	MOFL			Xilinx PPR 5.0		
	No. of CLBs	max. Levels	Delay (ns)	No. of CLBs	max. Levels	Delay (ns)
9symml	60	5	61.8	63	8	82.5
alu2	114	13	154.3	112	20	179.5
alu4	188	16	158.4	181	18	197.0
apex7	67	4	62.2	58	6	77.0
b9	31	4	51.6	26	5	54.0
c1355	118	13	118.3	124	12	112.4
c1908	129	14	147.1	138	17	156.6
c432	61	12	137.6	48	15	150.2
c499	127	10	109.4	156	13	119.2
c8	28	3	49.7	27	6	64.3
c880	109	8	98.8	110	12	113.4
count	31	7	83.0	30	12	104.4
example2	98	5	64.5	98	6	69.1
frg1	34	5	62.1	27	7	72.3
i9	104	7	88.7	78	8	93.0
lal	24	4	51.4	25	6	56.5
pcle	13	3	47.7	15	6	63.1
pcler8	23	4	62.9	22	8	81.9
sct	18	4	52.1	21	6	55.9
t481	194	8	103.2	186	13	136.7
term1	57	5	70.2	51	7	66.6
vda	148	7	85.8	167	9	107.0
x1	95	3	49.3	80	5	62.5
x2	12	3	41.6	14	3	39.2
z4ml	7	3	45.8	9	4	43.1
ratio	1.023	0.723	0.884	1	1	1

References

[1] R. K. Brayton, G. D. Hachtel and A. Sangiovanni-Vincentelli, "MIS, a multiple-level logic optimization system," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp.1062-1081, Nov. 1987.

[2] *Xilinx Programmable Logic Data Book*, Xilinx Inc., San Jose 1994.

[3] R. J. Francis, J. Rose, and Z. Vranesic, "Technology mapping for lookup table-based FPGAs for performance," *Proc. ICCAD'91*, pp. 568-571, Nov. 1991.

[4] R. J. Francis, J. Rose, and Z. Vranesic, "Chortle-crf:Fast technology mapping for lookup table-based FPGAs," *Proc. 28th ACM/IEEE DAC*, pp. 613-619, 1991.

[5] D. Hill, B. Britton, B. Oswald, N.-S. Woo, S. Singh, T. Poon, B. Krambeck, "A new architecture for high-performance FPGAs," *Int. Workshop on FPGA*, Sep. 1992.

[6] N.-S. Woo, "A heuristic method for FPGA technology mapping based on the edge visibility," *Proc. 28th ACM/IEEE Design Automation Conference*, pp. 248-251, 1991.

[7] R. Murgai, N. Shenoy, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Improved logic synthesis algorithms for table look up architectures," *Proc. ICCAD'91*, pp. 564-567, Nov. 1991.

[8] L. A. Zadeh, "Fuzzy sets," *Information and Control 8*, pp. 338-353, 1965.

[9] J. Cong and Y. Ding "On area/depth trade-off in LUT-based FPGA technology mapping," *Proc. DAC*, pp. 213-218, Jun. 1993.

[10] H. J. Zimmermann, " Fuzzy set theory and its applications," 2nd ed. Kluwer Academic Publishers, 1991.

[11] *XACT Reference Guide* , Xilinx Inc., San Jose 1994.

[12] R. Lin and E. Shragowitz, "Fuzzy logic approach to placement problem," *Proc. 29th ACM/IEEE Design Automation Conference*, pp. 153-158, 1992.

[13] E. Kang, R. Lin and E. Shragowitz, "Fuzzy logic approach to VLSI placement," *IEEE Trans. VLSI Systems*, vol. 2 , pp. 489-501 , Dec. 1994.

[14] J.-Y. Lee and E. Shragowitz, "Performance driven technology mapper for FPGAs with complex logic block structures," *Proc. IEEE International Symposium on Circuits and Systems*, vol. 2 pp. 1219-1222, Apr. 1995.

[15] D. Dubois and H. Prade, "New results about properties and semantics of fuzzy set-theoretic operators," *Fuzzy sets-theory and applications to policy analysis and information systems* pp. 59-75, Plenum Press, 1980.

[16] R. Yager, "Multiple objective decision-making using fuzzy sets," *Int. Journal of Man-Machine Studies*, vol. 9 pp. 375-382, 1977.