Volume 5, Issue 6, June 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Effective Use of Various Crossover Operators in Hybrid Genetic Algorithm for Optimization of BDD Mapped Circuits

Rohit Kumar Sharma, Manu Bansal
ECED, Thapar University,
India

Abstract*— The ordering of the input variables plays significant role in minimizing the size of Binary Decision Diagram (BDD). The main idea of this article is based on the efficient use of various crossover operators in Hybrid genetic algorithm for optimizing the variable order in BDDs to reduce the number of nodes. In Hybridized Genetic Algorithm, Genetic Algorithm has embedded with Branch and Bound Algorithm to intelligently search for the best solution of an optimization problem. The proposed method is applied on the LGSynth93 Benchmark circuits to reduce the node count and optimizing the variable order. The number of nodes reduced up to 64.96% for a benchmark circuit after applying this technique.

Keywords*— BDDs, Genetic Algorithm, Crossover Operators, Branch and Bound, LGSynth93.

I. INTRODUCTION

Binary Decision Diagrams are data structure for representation and manipulation of Boolean functions [1]. The size of the BDD is hugely depends upon the order of variables. In BDD's calculations, the major problem deals with calculating the fine order of variables. Since in VLSI Design, area of the logic circuit should be minimize as much as possible to reduce the chip size. There are many approaches such as Static, Dynamic and Heuristic techniques exist for obtaining optimum order of variables in BDD. Basic genetic algorithm is an efficient technique to find solutions of such problems. Crossover operators used in genetic algorithm also play significant role in finding good result. Therefore, crossover operator should be chosen wisely for different problems. In the recent years, the hybrid techniques have evolved at a good pace in research field because of their better efficiency in providing good results. This paper deals with efficient use of basic three crossover operators (Order, cyclic and partially mapped operator) for optimizing the variable order of number of benchmark combinational logic circuits from LGsynth93 Benchmark suite which are used to represent in BDDs. Branch and Bound technique incorporates with basic genetic algorithm in the proposed method. The node reordering is taken care by standard BDD package Buddy-2.4.

This paper is organized as follows. Section 2 defines the BDDs, Section 3 discusses the basic genetic algorithm and then Section 4 provides the details of proposed method and Section 5 shows the simulation results. Simulation results shows the better results when compare with other optimization techniques.

II. BINARY DECISION DIAGRAMS

BDD is basically a data structure which is used to represent a Boolean function in the form of directed acyclic graph [2]. In VLSI design, the major applications of BDDs are in CAD software to synthesize and verify the logic circuits. A BDD consists of decision nodes and two terminal nodes. Each decision node performs a function; labelled by a Boolean variable and has two child nodes called low child (represented by dotted edges) and high child [1].

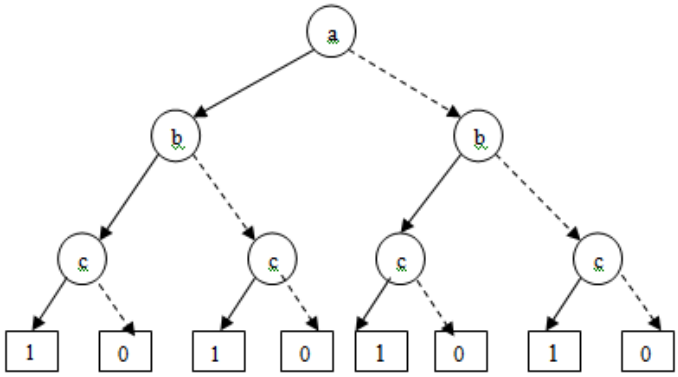


Fig 1: BDD for F(a,b,c) = (a+b).c

The two terminal nodes of BDD are 0-terminal and 1-terminal nodes. Each node can be represented as Shannon decomposition thus, node F is represented as : F = xiG + xi'H, Where, G is the positive cofactor of F with respect to xi (G = Fxi), and H is the negative cofactor of F with respect to xi (H = Fxi'), G is also known as the THEN node and H is also known as the ELSE node. A simple BDD of a random function F(a,b,c) = (a+b).c is shown in figure 1. If different variables placed in the same order on all paths to the root then that BDD becomes 'Ordered' (OBDD) [2]. The node count of OBDD can be reduced by changing the order of variables thus the size of the BDD is reduced by choosing proper variable order .The node counts further reduced by merging equivalent, isomorphic nodes and removing redundant nodes. An example is shown in figure 2, different variable order give different number of nodes.

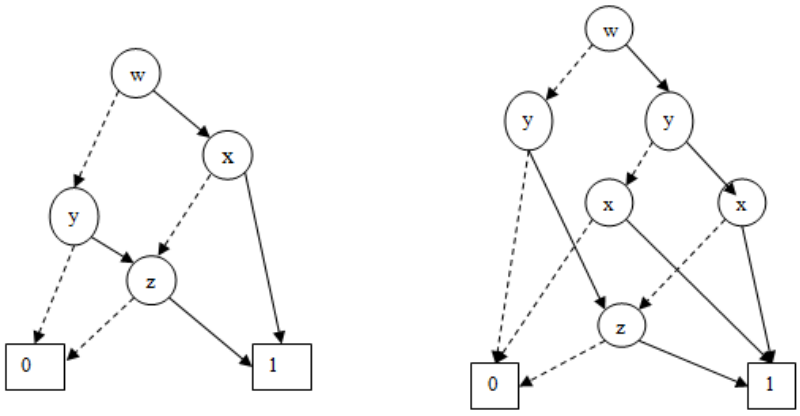


Fig 2: BDD of F(w,x,y,z) = wx+yz (a) with variable order w, x, y, z (b) with variable order w, y, x, z

.

III. BASIC GENETIC ALGORITHM

Genetic Algorithms are a family of excellent optimization techniques based on principle of natural selection and human genetics [9]. Genetic Algorithm is based on Darwinian Theory of survival of the fittest. The objective of GA is to find an optimal solution to a problem. GAs work by evolving a population of individuals over a number of generations .A fitness value is assigned to each individual depending on the application. For each generation, individuals are selected from existing population for reproduction, the individuals are crossed to generate new individuals, and the new individuals are mutated to introduce new characteristics. The solution with best fitness value will be the most optimum solution. The formulation of Genetic Algorithm for any problem involves the careful and efficient encoding of the solutions to form chromosomes, crossover and mutation operators and a cost function measuring the fitness of the chromosomes in a population. GAs use two basic processes from evolution: inheritance, or passing of features from one generation to other, and competition, or the survival of the fittest, which results in weeding out bad features from individuals in the population. This process is continued for a large number of iterations to obtain a best-fit (near-optimum) solution. The basic operators used in GA are selection, crossover and mutation.

Selection- GA begins with a set of solutions, represented by chromosomes, called the population. The selection of parents is done according to their fitness function and more fit solutions are given more chance to reproduce [12].

Crossover- It is performed between two selected individuals, called parents, by exchanging parts of their features (i.e. encodings) to form two new individual, called offspring [12].

Mutation- It brings variety into population by selecting a chromosome randomly from the population and modifies the chromosome at a point depending upon the value of a generated random number [12].

The basic Genetic Algorithm steps [8] shown below:

/*Algorithm GA */

1. Formulate initial population
2. Randomly initialize population
3. Repeat
4. Evaluate objective function
5. Find fitness function
6. Apply genetic operators
7. Reproduction
8. Crossover
9. Mutation
10. Until stopping criteria

IV. BDD MINIMIZATION USING PROPOSED METHOD

Espresso tool is used to convert the PLA format of benchmark circuit in to node equations. Then, these node equations are used in program to get the minimum number of nodes and optimizing the variable order. The flowchart of the whole process is shown in figure 3.

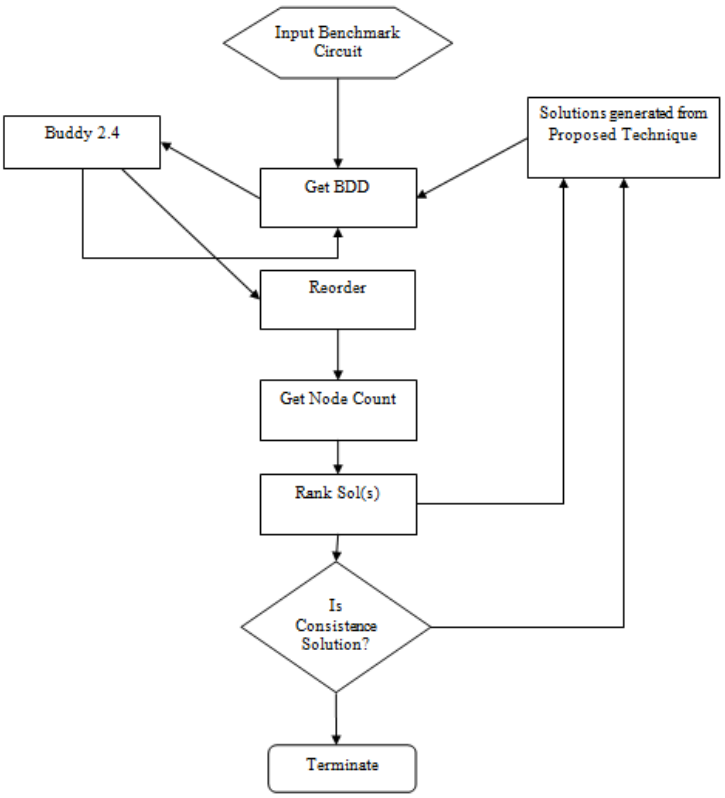


Fig 3: Flowchart of whole Process

At first, we get a BDD from the node equations. Then package *Buddy*-2.4 reorder the nodes, we get the new solutions. Then proposed algorithm is applied on these solutions to get better results. In proposed technique we start with selecting random solutions called population. Individual fitness of each solution is calculated. Fitness function depends upon the problems. Fitness function is different for different problems. Then Branch and Bound technique is applied on the initial population. Branch and Bound Algorithm discards fruitless solutions by using upper and lower bounds on fitness value.

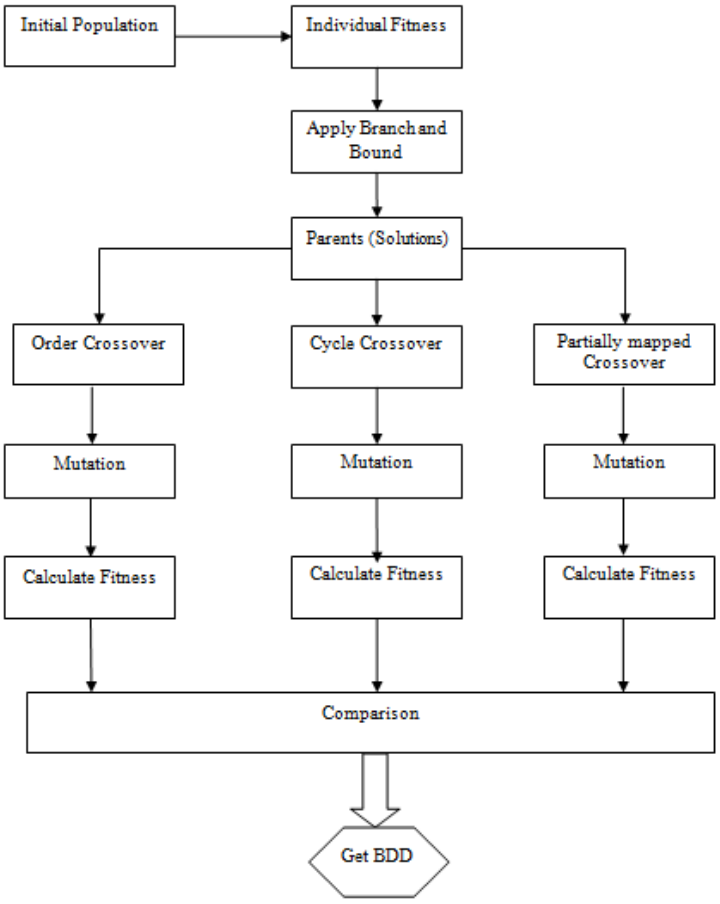
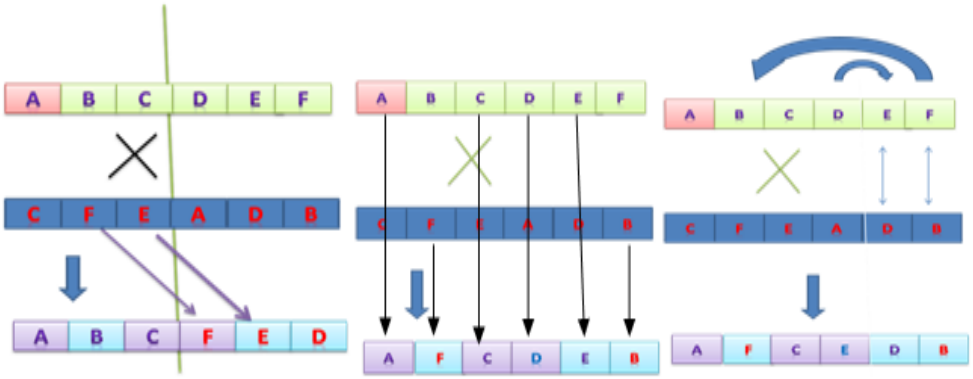


Fig 4: Flowchart of proposed technique

Then three different Crossover operators are applied on the same chromosomes separately. These crossover operators are: **Order crossover**: This operator conveys a sub-placement from the first parent without any changes, and then, to resolve conflicts, compresses the second parent by eliminating the cells conveyed by the first parent and shifting the rest of the cells to the left without changing their order. It then copies this compressed second part into the remaining part of the offspring array [12].

Cycle Crossover: In the offspring generated by the cycle crossover, every cell is in the same location as in one parent or the other. The basic idea behind cycle crossover is to avoid cell conflicts by finding non-overlapping sets of cells to pass from the two parents [12].

Partially Mapped: A cell in the segment of the first parent and a cell in the same location in the second parent define which cells in the first parent have to be exchanged to generate the offspring [12].



(a) (b) (c)

Fig 5: (a) Order crossover, (b) Cycle crossover and (c) Partially mapped crossover.

After performing mutation, fitness value of the solutions are calculated and then compared with each other. The crossover operator which will give better fitness value, the solution provided by that cross over operator is selected as parent of next generation. Finally, the algorithm terminates when there is no improvement in solution over a fixed number of generations.

V. SIMULATION RESULTS

Table 1: Comparison of results with other existing Algorithms

LGSynth93 Benchmark Circuits	#i	#o	Initial node count	WIN2 node count	WIN2ite node count	WIN3 node count	SIFT node count	Simple GA node count	Proposed Method node count	Up to % reduction w.r.t initial node count
5xp1	7	10	88	87	82	82	82	68	63	28.41
b12	15	9	91	86	86	65	65	67	59	35.16
bw	5	28	118	113	112	106	106	106	94	20.33
clip	9	5	254	178	108	105	105	96	89	64.96
con1	7	2	18	18	18	18	18	15	15	16.66
misex1	8	7	47	43	42	41	41	37	36	23.40
Z5xp1	7	10	69	68	68	68	68	68	61	11.59
sqrt8	8	4	42	40	39	37	37	35	33	21.42
inc	7	9	73	75	69	68	68	62	56	23.28

Where, #i and #o represents the number of inputs and outputs of the circuit respectively.

In this section, the result of simulation for various benchmark circuits from LGsynth93 Benchmark suite has been presented. The proposed method is implemented with C++ codes and simulated using BUDDY 2.4 package on Ubuntu 14.04. The results of proposed method for BDD ordering and node minimization are compared along with the results of available approaches for BDD minimization and ordering on the set of Boolean functions. The simulation results for various logic circuits have been displayed in Table 1. The proposed approach provides the optimum variable order with minimum number of nodes.

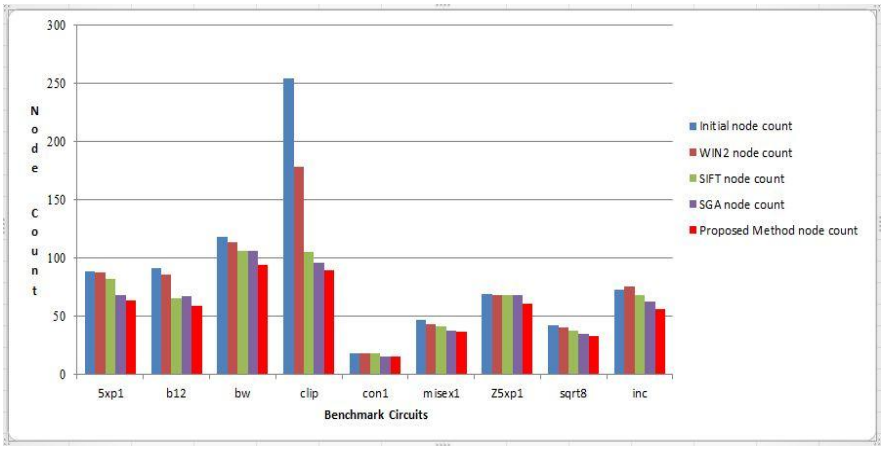


Fig 6: Comparison of node counts using column graph

VI. CONCLUSIONS

Presented here the technique for BDD optimization namely, Hybrid Genetic algorithm based optimization. Exhaustive experimentation has been done with LGSynth93 benchmark circuits to see the effectiveness of the proposed technique for area optimization. When we compared the result of this experiment with other established techniques we found that the proposed technique gives better results in most of the cases.

REFERENCES

[1] Sheldon B. Akers, "Binary Decision Diagrams," *IEEE Transactions on Computers*, vol. C-27, no. 6, June (1978)

[2] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Transactions on Computers*, vol. 35, August (1986)

[3] F. Towhidi, A.H. Lashkari, R.S. Hosseini, "Binary Decision Diagram (BDD)", *International Conference on Future Computer and Communication*, pg496-499,2009

[4] William N.N. Hung, Xiaoyu Song, El Mostapha Aboulhamid, and Michael A. Driscoll, "BDD Minimization by Scatter Search*," IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. 21, no. 8, August (2006)

[5] N. Zhuang, M.S.T. Benten, and P.Y.K Cheung, "Improved Variable Ordering of BDDs with Novel Genetic Algorithm*," IEEE International Symposium on Circuits and Systems*, vol. 3, 12-15 May (1996)

[6] Saurabh Chaudhary, and Anirban Dutta, "Algorithmic Optimization of BDDs and Performance Evaluation for Multi-level Logic Circuits with Area and Power Trade-offs," *IEEE International Conference on Electronics*, Circuits and Systems, July (2011)

[7] Misagh Takapoo, and M.B Ghaznavi-Ghoushchi, "IDGBDD: The novel use of ID3 to improve Genetic algorithm in BDD reordering," *International Conference on Electrical Engineering / Electronic Computer Telecommunication and Information Technology*,19-21 May (2010)

[8] Merz P and Freisleben B, "A genetic local search approach to the quadratic assignment problem," *Proceedings of the 7th International Conference on Genetic Algorithms*, (1997)

[9] Tom V Mathew, "Genetic Algorithm," Report submitted at IIT Bombay.

[10] P. W. C. Prasad, A. Assi, A. Harb and V. C. Prasad, "Binary Decision Diagrams: An Improved Variable Ordering using Graph Representation of Boolean Functions," *International Journal of Computer Science*, Vol. 1, no. 1, pp. 1-7, 2006.

[11] R. Ebendt, F. Gorschwin and R. Drechsler, "Advanced BDD Minimization", *Springer*, New York, 2005.

[12] Pinaki Mazumder, Elizabeth M. Rudnick, Genetic Algorithms for VLSI Design, Layout & Test Automation.

