Partitioning VLSI circuits on the basis of Genetic Algorithms and Comparative Analysis Of KL And SA Partitioning Algorithms

Shivam Singh[1],Shivam Srivastava[2],Shalini Panjwani[3],Saurabh Rawat[4]
(1,2,3)- BTech(ECE) & (4)- Graphic Era University

Abstract— Circuit partitioning is the one of the fundamental problems in VLSI design. It appears in several stages in VLSI design, such as logic design and physical design. Circuit partitioning is generally formulated as the graph partitioning problem**.** Circuit partitioning problem is a well known NP hard problem. The potential of Genetic Algorithm has been used to solve many computationally intensive problems (NP hard problems) because existing conventional methods are unable to perform the required breakthrough in terms of complexity, time and cost. The presented work deals with the problem of partitioning of a circuit using Genetic Algorithm. The program inputs the adjacency matrix, generates graph of the circuit and partitions the circuit based on crossover operator. The program produces a set of vertices that are highly connected to each other but highly disconnected from the other partitions.In VLSI circuit partitioning the problem of obtaining minimum cut is of prime importance.To enhance other criteria like power,delay and area in addition to miminum cit is included.For this problem, a heuristic proposed by Kernighan and Lin is the most well-known and widely used one in practical applications. with the passage of time and advancement of the technological methods. In this paper, the comparison of two partitioning techniques is done based on cut-set. One, KL and second algorithm is Simulated Annealing algorithm. However, due to recent advances of semiconductor technologies, a VLSI chip may contain millions of transistors, and hence the size **of** the problem of circuit partitioning also becomes very large. Good partitioning techniques can positively influence the performance and cost of a VLSI product.

Index Terms— Partioning, KL Algorithm, SA Algorithm, Design Flow.

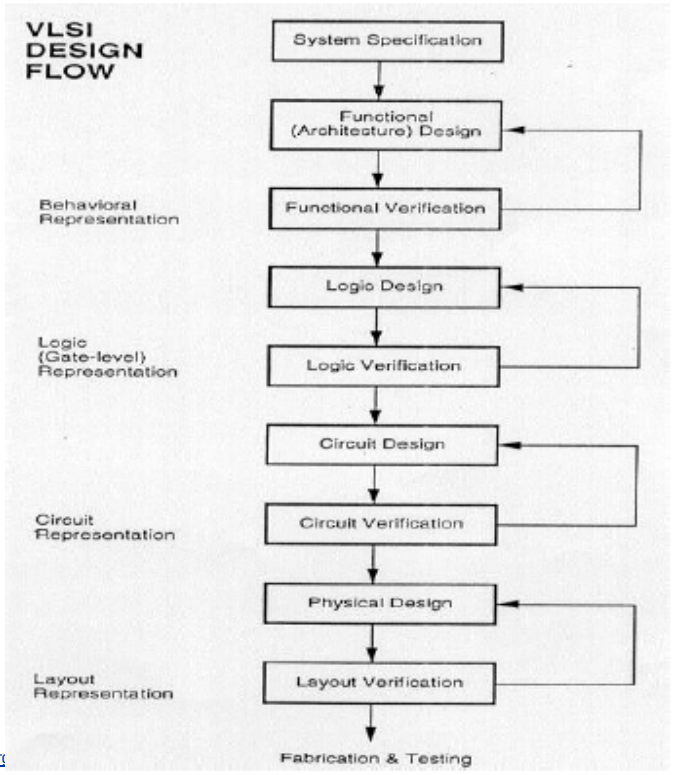
1.INTRODUCTION

Partitioning is a technique to divide a circuit or system into a collection of smaller parts (components).[1] It is a design task to break a large system into pieces to be implemented on separate interacting components and on the other hand it serves as an algorithmic method to solve difficult and complex combinatorial optimization problems as in logic or layout synthesis. The main reason that partitioning has become a central and sometimes critical design task today is the enormous increase of system complexity in the past and the expected further advances of microelectronic system design and fabrication. The main goal of this paper is to design a class of iterative algorithms for VLSI multiobjective partitioning such that circuit delay,power dissipation and interconnect cut set are minimized under the balanced constraint. The main search algorithms used are Genetic Algorithm(GA),Tabu Search(TS).These algorithms are used to find a solution that minimize these costs while satisfying balance constraints.

2. VLSI PARTITIONING

VLSI design is a complex process and is therefore it is broken down into no of intermediate steps. Moreover partitioning is considered to be an NP Complete problem which means polynomial time algorithm exists for solving the problem.

VLSI circuit partitioning is a vital part of physical design stage. The essence of circuit partitioning is to divide the circuit into a number of sub-circuits with minimum interconnections between them. This can be accomplished by recursively partitioning a circuit into two parts until we reach desired level of complexity. Thus two way partitioning is basic problem in circuit partitioning, which can be described as [2].This paper proposes a different approach to solve circuit partitioning problem. We made use of evolutionary algorithm & clustering approach.



2.1 .REVIEW OF PARTITIONING ALGORITHMS

In 1970, Kernighan and Lin [6] proposed a simple graph-based heuristic that iteratively improves an initial partitioning by using a *semi-greedy* graph search technique, since then numerous combinatorial optimization techniques have been applied to solve the graph and circuit partitioning problems like *constructive algorithms, iterative improvement algorithms*. Then *Fiduccia* and *Mattheyses*[7] proposed a more efficient method for implementing Kernighan and Lin‟s algorithm leading to a fast linear time algorithm for partitioning (the F-M algorithm). *Krishnamurthy* proposed Look-Ahead (LA) mechanism. *S. Dutt* and *W. Deng* proposed a new probability-based augmentation of the graph partitioning algorithm, called the probabilistic gain computation approach (*PROP*). L. A. Sanchis modified Krishnamurthy‟s algorithm to perform multiway partitioning. *Wei and Cheng*, introduced the *ratio-cut-algorithm* Among the various well-know stochastic optimization methods, the *simulated annealing algorithm* has been widely used for solving numerous VLSI layout optimization problems. *Yih and Mazumder* [8] have presented a *neural network model* for circuit partitioning, using iterative improvement techniques. *Genetic Algorithm (GA)* was invented by Prof. John Holland [9] at the University of Michigan in 1975. Later, Prof. *David Goldberg* [5] at the University of Illinois made this topic popular.

KL Algorithm The K-L (Kernighan-Lin) algorithm was first suggested in 1970 for bisecting graphs in relation to VLSI layout. It is an iterative algorithm. Starting from a load balanced initial bisection, it first calculates for each vertex the gain in the reduction of edge-cut that may result if that vertex is moved from one partition of the graph to the other. At each inner iteration, it moves the unlocked vertex which has the highest gain, from the partition in surplus (that is, the partition with more vertices) to the partition in deficit. This vertex is then locked and the gains updated. The procedure is repeated even if the highest gain may be negative, until all of the vertices are locked. The last few moves that had negative gains are then undone and the bisection is reverted to the one with the smallest edge-cut so far in this iteration. This completes the outer one iteration of the K-L algorithm and the iterative procedure is restarted. Should an outer iteration fail to result in any reductions in the edge-cut or load imbalance, the algorithm is terminated. The initial bisection is generated randomly and for large graphs, the final result is very dependent on the initial choice. The K-L algorithm is a local optimization algorithm, with a limited capability for getting out of local minima by way of allowing moves with negative gain.

Simulated Annealing :

Simulated annealing (SA) is a generic probabilistic, heuristic algorithm for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. It is often used when the search space is discrete (e.g., all tours that visit a given set of cities). For certain problems, simulated annealing may be more efficient than exhaustive enumeration — provided that the goal is merely to find an acceptably good solution in a fixed amount of time, rather than the best possible solution. The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The heat causes the atoms to become unstuck from their initial positions (a local minimum of the internal energy) and wander randomly through states of higher energy; the slow cooling gives them more chances of finding configurations with lower internal energy than the initial one. In order to apply the SA method to a specific problem, one must specify the following parameters: the state space, the energy (goal) function E(), the candidate generator procedure neighbour(), the acceptance probability function P(), and the annealing schedule temperature() AND initial temperature <init temp>. These choices can have a significant impact on the method's effectiveness. Unfortunately, there are no choices of these parameters that will be good for all problems, and there is no general way to find the best choices for a given problem.

Fiduccia Mattheyeses(FM)

It is the modification of KL group method

For circuit partitioning. The original KL heuristic is an improvement procedure where the current partition is improved by swapping pair of nodes belonging to the two subset of the current partition.

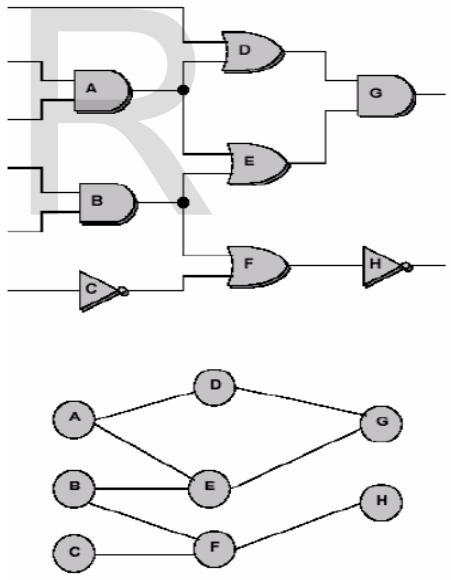


Figure 2: Converting Circuits to Graphs

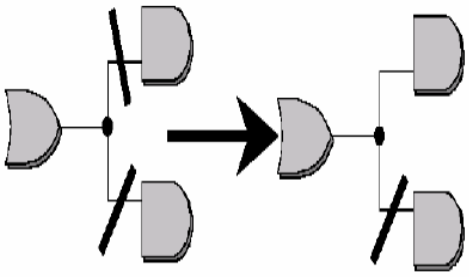


Figure 3: Figure showing difference in F-M & K-L approach

Our Ob-

jective: Any Partitioning problem can be expressed more naturally in graph theoretic terms. A hyper-graph G = (V, E) representing a partition problem can be described as follows. Let V = {V1, V2, …., Vn} be a set of vertices and E = {e1,e2,…,em} be a set of hyper-edges, then

□□Each vertex represents a component.

□□There is a hyper-edge joining the vertices whenever the component corresponding to these vertices are to be connected Thus each hyper-edge is the subset of vertex set ei □V, i = 1, 2, …, m

In other words, we can say that each net is represented by a hyper-edge. **The partitioning problem is** to partition set of vertices (or components) V into V1, V2, …, Vk such that

Vi □Vj = □i □j

□j=1Vi = V

We aim at minimizing the following objectives.

1. Cutsize

The cardinality of the set containing the net cut by the cluster is called cutsize.

2. Delay

The delay of the circuit is optimized by optimizing the delay between the input cell to the output cell,input cell to the internal cell and vice versa and internal cell to the output cell.By doing this the delay of the total system will be minimized.

3. Power dissipation

The power should be evenly disrtributed throughout the system hence it should be verified that total power of each module sis almost same.

4. Network Area

Area is an important constraint .it should be kept small.

3. Results and Discussions:

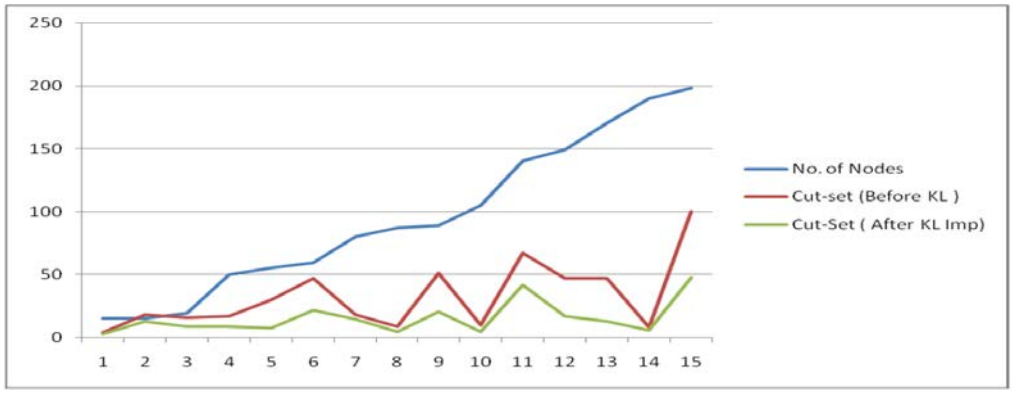
From many of the standard VLSI Netlist circuits, 15 net-list files are chosen at random and studied by applying the KL and SA partitioning algorithms. The different standard VLSI Net-list files are chosen, whose nodes, i.e. circuit elements vary from 10 to 200, also with different nets. MATLAB program is made, that implements the KL partitioning algorithm. the interfacing matrix created by the MATLAB by reading the Netlist circuit file is given as an input to this program and results are obtained in the form of two partitions and cut-set. The following table is drawn based on cut-set before and after implementing the KL algorithm.

KL algorithm implementation results on 15 selected VLSI net-list files

Depending upon the circuit configuration and the no of nets in the given VLSI NET circuit, the results or optimization is better in the circuits with the larger number of nodes. Also, it takes more time and iterations to execute the circuits with the larger number of nodes.

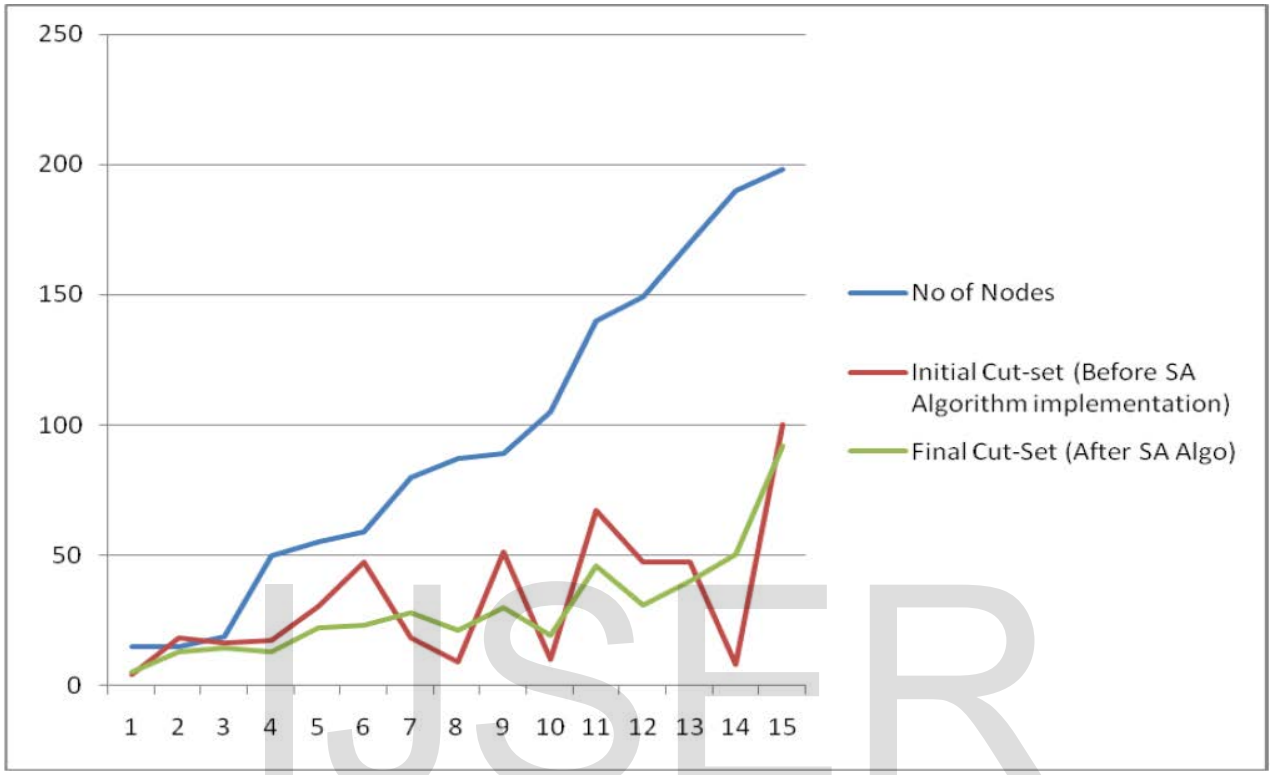
Sr. No.	File Name	No of Nodes	No. of Nets	Initial Cut-set (Before KL Algorithm implementation)	No of iterations	Final Cut-Set (After KL Algo)
1.	spp_N15_E16_R1_1121.netD	15	16	4	9	3
2.	spp_N15_E45_R2_1653.netD	15	45	18	9	13
3.	spp_N19_E25_R2_765.netD	19	25	16	11	9
4.	spp_N50_E53_R3_926.netD	50	53	17	26	9
5.	spp_N55_E54_R4_516.netD	55	54	30	29	8
6.	spp_N59_E79_R6_306.netD	59	79	47	31	22
7.	spp_N80_E90_R5_339.netD	80	90	18	41	15
8.	spp_N87_E91_R5_382.netD	87	91	09	45	05
9.	spp_N89_E131_R5_353.netD	89	131	51	46	21
10.	spp_N105_E109_R5_376.netD	105	109	10	54	5
11.	spp_N140_E162_R11_123.netD	140	162	67	71	42
12.	spp_N149_E186_R10_164.netD	149	186	47	76	17
13.	spp_N170_E184_R6_132.netD	170	184	47	86	13
14.	spp_N190_E194_R11_165.netD	190	194	08	96	6
15.	spp_N198_E338_R12_101.netD	198	338	100	100	48

Showing the cut-set before and after KL Algorithm implementation



Sr. No.	File Name	No of Nodes	No. of Nets	Initial Cut-set (Before SA Algo-rithm im-plementa-tion)	Final Cut-Set (After SA Algo)
1.	spp_N15_E16_R1_1121.netD	15	16	4	05
2.	spp_N15_E45_R2_1653.netD	15	45	18	13
3.	spp_N19_E25_R2_765.netD	19	25	16	14
4.	spp_N50_E53_R3_926.netD	50	53	17	13
5.	spp_N55_E54_R4_516.netD	55	54	30	22
6.	spp_N59_E79_R6_306.netD	59	79	47	23
7.	spp_N80_E90_R5_339.netD	80	90	18	28
8.	spp_N87_E91_R5_382.netD	87	91	09	21
9.	spp_N89_E131_R5_353.netD	89	131	51	30
10.	spp_N105_E109_R5_376.netD	105	109	10	19
11.	spp_N140_E162_R11_123.netD	140	162	67	46
12.	spp_N149_E186_R10_164.netD	149	186	47	31
13.	spp_N170_E184_R6_132.netD	170	184	47	40
14.	spp_N190_E194_R11_165.netD	190	194	08	50
15.	spp_N198_E338_R12_101.netD	198	338	100	92

As observed from the results, SA algorithm is showing better results only in certain cases or circuits. Otherwise, cut-set becomes larger than the previous cut-set taken. This is because the parameters taken are constant for all the 15 VLSI NET circuits. But, the results may be different and may be better if different parameters are selected for the different circuit configurations. As for example, if the tolerance level and no of nodes to swap parameters are increased for the large nodes circuit, it may give the better results.

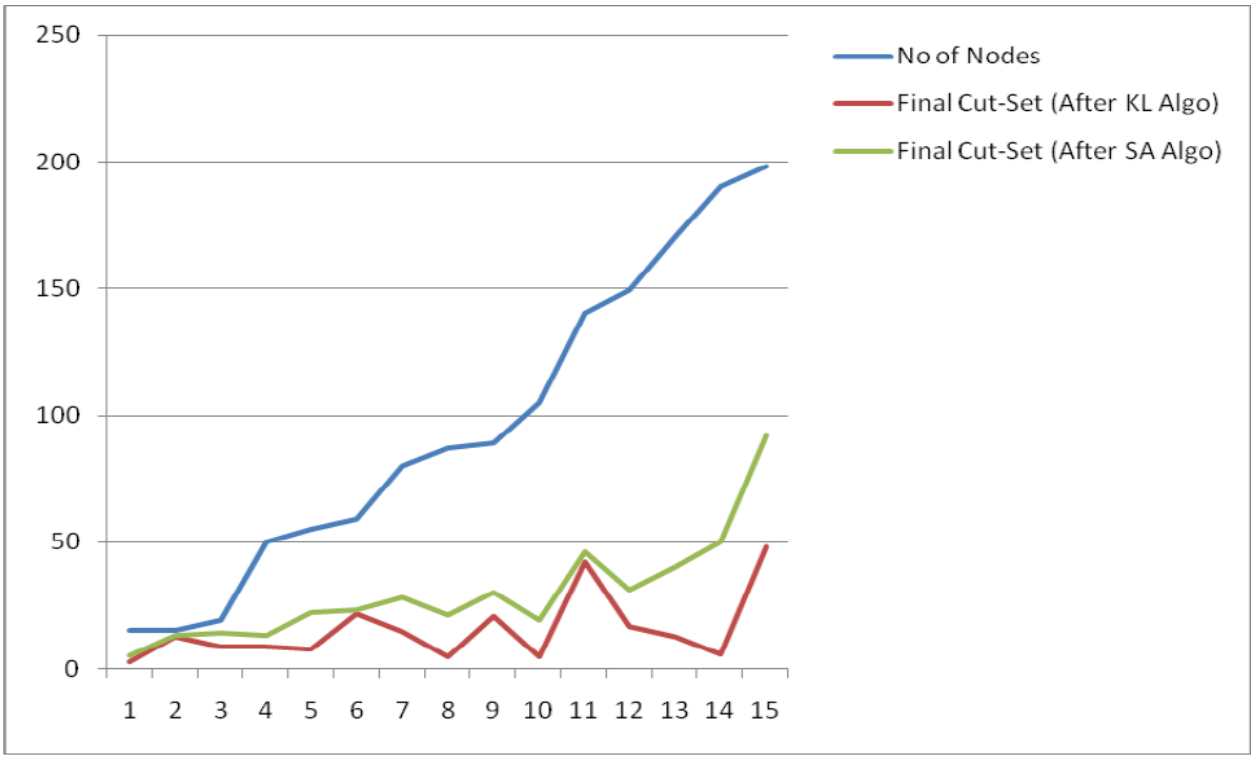


Showing the cut-set before and after SA Algorithm implementation

Comparison of KL and SA Partitioning Algorithm

In this section, the results of KL and SA Algorithm are compared. KL and SA algorithm results are tabulated and graph is drawn for the both Algorithms and same VLSI Net-list files

Sr. No.	File Name	No of Nodes	No. of Nets	Final Cut-Set (After KL Algo)	Final Cut-Set (After SA Algo)
1.	spp_N15_E16_R1_1121.netD	15	16	3	05
2.	spp_N15_E45_R2_1653.netD	15	45	13	13
3.	spp_N19_E25_R2_765.netD	19	25	9	14
4.	spp_N50_E53_R3_926.netD	50	53	9	13
5.	spp_N55_E54_R4_516.netD	55	54	8	22
6.	spp_N59_E79_R6_306.netD	59	79	22	23
7.	spp_N80_E90_R5_339.netD	80	90	15	28
8.	spp_N87_E91_R5_382.netD	87	91	05	21
9.	spp_N89_E131_R5_353.netD	89	131	21	30
10.	spp_N105_E109_R5_376.netD	105	109	5	19
11.	spp_N140_E162_R11_123.netD	140	162	42	46
12.	spp_N149_E186_R10_164.netD	149	186	17	31
13.	spp_N170_E184_R6_132.netD	170	184	13	40
14.	spp_N190_E194_R11_165.netD	190	194	6	50
15.	spp_N198_E338_R12_101.netD	198	338	48	92



4. Conclusion And Future Aspects :

We have studied about various algorithms for partitioning VLSI circuits.The results obtained of these algorithms differ on the basis of the parameters used.The presented partitioning techniques can be extended to perform the optimization of other steps for
VLSI physical design. Due to the increasing requirements on partitioning tools further developments and improvements are very desirable. It has beenobserved in the past that even minor algorithmic modifications canbe very effective.From the application viewpoint, highly constrainedperformance-driven partitioning is attractive for research andaccurate but efficient delay calculation remains an important issue.
With the rapidly increasing computing power in mind enumerativemethods will become more attractive.On the higher levels ofabstraction, applying logic synthesismethods, e.g. logic replication and retiming, seems to have greatoptimization potential. The estimation of system properties needsattention such that designers can examine potential partitioning solutions quickly at the highest abstraction level possible.Benchmarking of partitioning approaches (as of other classes of design problems) urgently needs to be improved. Last but not least, synergy effects could result from coordinating the partitioning activities on different levels of abstraction and for different applications at the major conferences.

REFERENCES

[1.] Sung-Mo kang and Yusuf Leblebici, " CMOS Digital Integrated Circuits", Analysis and design, 3rd edition, TMH, 2003.

[2.] M. Morris Mano, "Digital Design", 2nd edition, Prentice Hall of India, 2000.

[3.] Bernhard M. Riess, Konrad Doll, and Frank M. Johannes (1994) —Partitioning every large circuit using analytical placement techniques.|| In *Design Automation Conference (DAC)*, pages 646-651. ACM/IEEE.

[4.] Bernhard M. Riess, Heiko A. Giselbrecht, and Bemd Wurth, (1995) —A new k-way partitioning approach for multiple types of FPGAs.|| In *Asia and South Pacific Design Automation Conference (ASP-DAC)*, IFIP/ACM/IEEE.

[5.] Goldberg D.E., "Genetic Algorithms in Search, Optimization and Machine learning", Pearson Education, 2004.

[6.] Kernighan B. W. and S. Lin, "An efficient heuristic procedure for partitioning graphs," Bell Systems Technical Journal, vol. 49, pp. 291—307, 1970.

[7.] Fiduccia C. M. and R. M. Mattheyses, "A linear-time heuristic for improving network partitions," Proc. Design Automation Conf., pp. 175—181, 1982.

[8.] Yih J S. and P. Mazumder, "A neural network design for circuit partitioning," IEEE Trans. Computer-Aided Design, vol. 9, no. 10, Oct. 1990.

[9.] Holland J. H., Adaptation in Natural and Artificial Systems, Ann Arbor, M University of Michigan Press, 1975.