International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 1, Issue 3 (November 2012), PP.39-45
www.irjes.com

Comparative Analysis Of KL And SA Partitioning Algorithms Implemented On VLSI Circuit Partitioning

S. Gurjot Singh[1] Prof. J.P.S. Raina.[2] Prof. Supreet Singh[2]
[1]*AP,RIMT-MAEC MandiGobindgarh*
[2]*BBSBEC, Fatehgarh Sahib*

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.*** *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.*

The main objective to Partition a circuit into parts is that every component is within a prescribed range and the # of connections among the components is minimized.

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 the 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.

Simulated Annealing algorithm works in a different way. It is a heuristic algorithm which tries to find the for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. 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.

MATLAB software is used for programming and implementation of the algorithms. First, we have chosen standard net-list files to apply the algorithms. Then, we apply KL algorithm and then, SA Algorithm and draw the graphs based on minimum cut-set.

I. Introduction

Partitioning is a technique to divide a circuit or system into a collection of smaller parts (components). It is on the one hand 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 size of VLSI designs has increased to systems of hundreds of millions of transistors. The complexity of the circuit has become so high that it is very difficult to design and simulate the whole system without decomposing it into sets of smaller subsystems. This divide and conquer strategy relies on partitioning to manipulate the whole system into hierarchical tree structure.

II. K-L ALGORITHM Implementation:

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.

III. Simulated Annealing Algorithm Implementation:

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.

Our Objective:

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 $V = \{V_1, V_2, \ldots, V_n\}$ be a set of vertices and $E = \{e1,e2,\ldots,e_m\}$ 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

$e_i \subseteq V$, 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 $V_1, V_2, \ldots, V_k$ such that

$V_i \cap V_j = \phi \quad i \neq j$

$\cup_{i=1} V_i = V$

Our Objective of the partitioning problem is

- to partition the circuit in such a way that cut size is minimized.

Cut Size = $$\sum_{i=1}^{k}\sum_{j=1}^{k} c_{ij}\,(i \neq j)\,.$$

Input: A weighted graph G = (*V*, *E*) with

- Vertex set *V*. (|*V*| = 2*n*)
- Edge Set *E*. (|*E*| = *e*)
- Cost *c* (*A*, *B*) for each edge {*A*, *B*} in *E*.

Output: 2 partitions *X* & *Y* such that

❖ Total cost of edges "crossing" the partition is minimized such that each partition has *n* vertices with some tolerances.

IV. 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.

Table 5.1 KL algorithm implementation results on 15 selected VLSI net-list files

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

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.

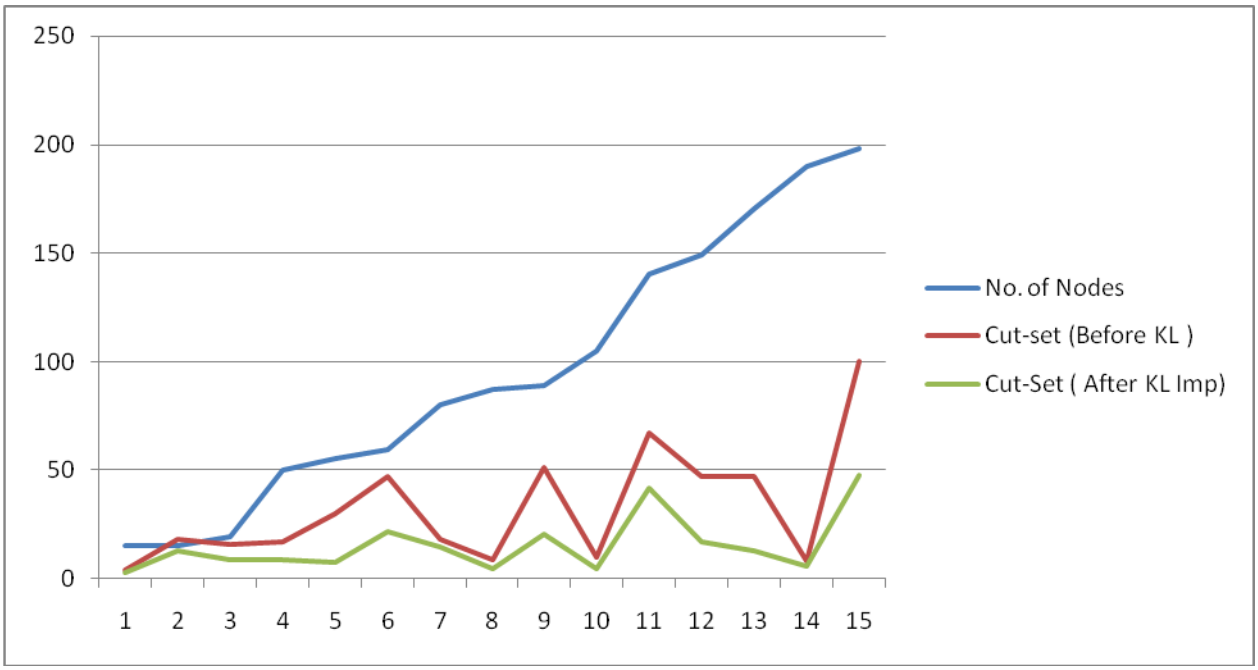

Figure 5.3 showing the cut-set before and after KL Algorithm implementation

5.3 SA Algorithm Implementation Results

In this section, Simulated Annealing Algorithm is applied to the 15 randomly selected VLSI Netlist standard circuits. Since SA algorithm is heuristic algorithm, therefore, its parameters need to be defined before implementation and partitions may also change because of random nature of the applied algorithm.

The SA Algorithm is applied by fixing the following parameters for all the NET- files.

initial_temperature=1000; % Initial temperature to start with

cooling_rate=0.3;

threshold=700; % defines the number of iterations

no_of_nodes_to_swap=8;

t=2; % tolerance

Table 5.2 SA algorithm implementation results on 15 selected VLSI net-list files

Sr. No.	File Name	No of Nodes	No. of Nets	Initial Cut-set (Before SA Algorithm implementation)	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.

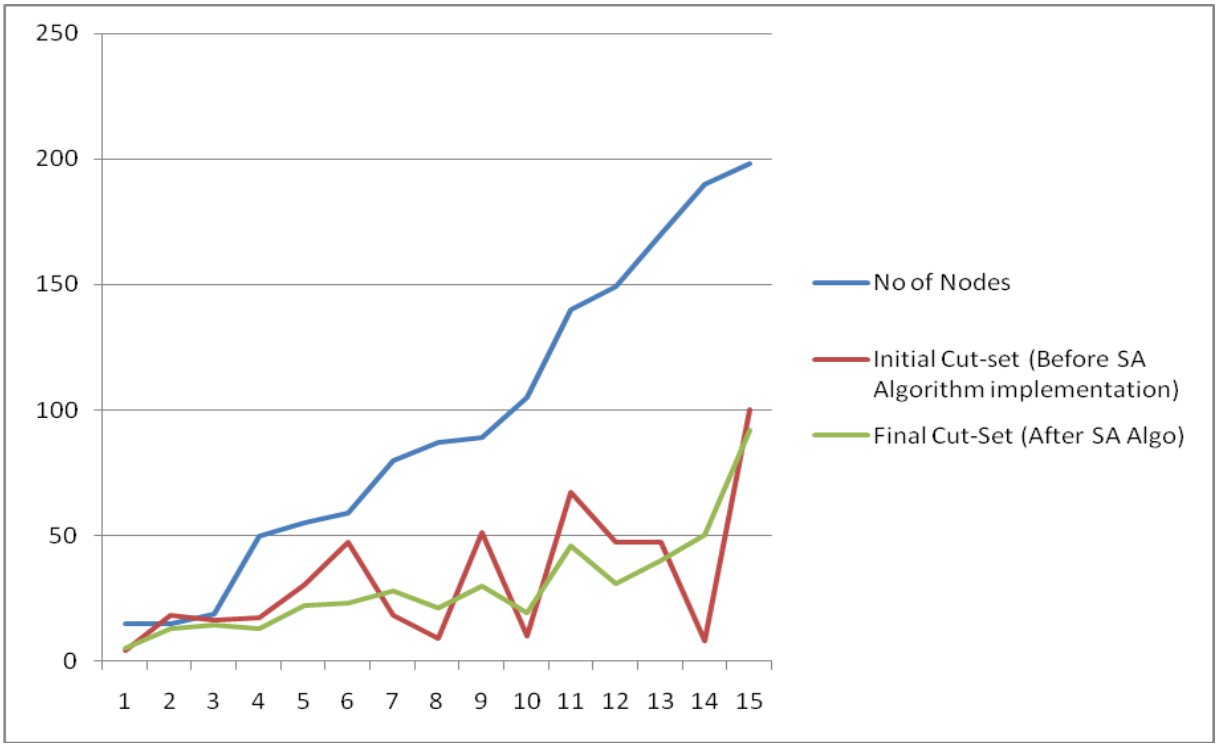

Figure 5.4 showing the cut-set before and after SA Algorithm implementation

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.

Table 5.3 showing comparison of KL and SA Algorithm implementation results

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

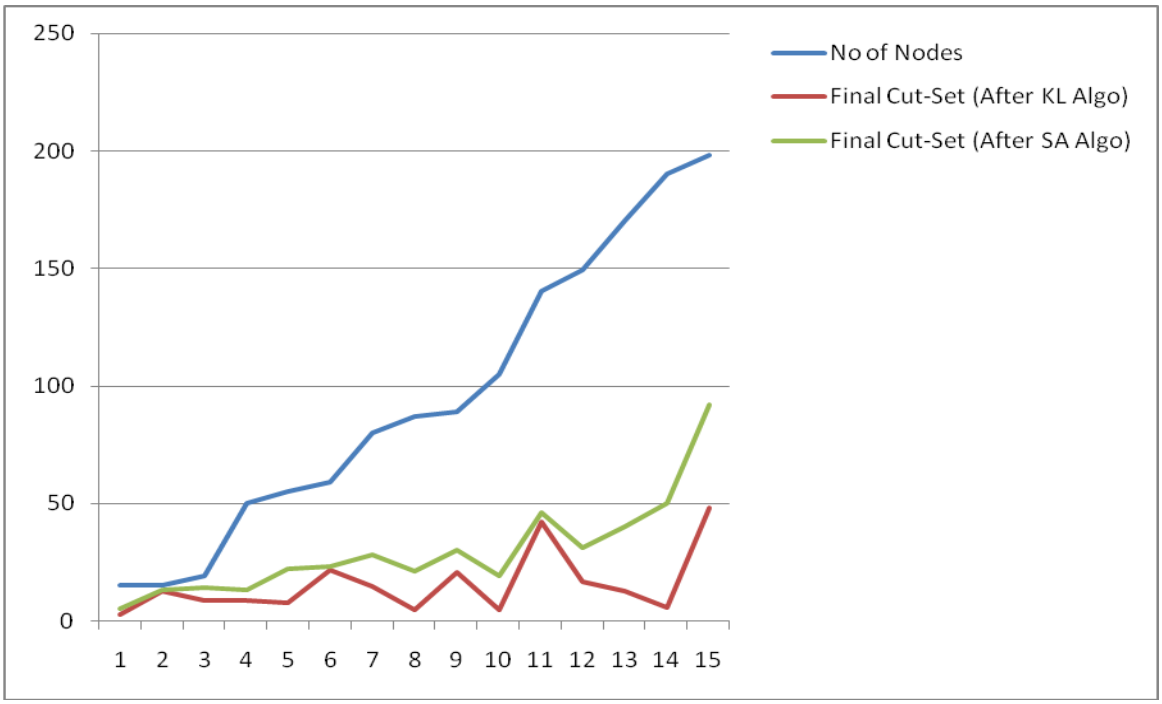

Figure 5.5 showing the results of comparison of KL and SA Algorithm

References

[1.] Kernighan and Lin, (Feb, 1970) "An efficient heuristic procedure for partitioning graphs," *The Bell System Technical Journal*, vol. 49, no. 2, 291-297 pg.

[2.] C. M. Fiduccia and R. M. Mattheyses (1982) – "A linear time heuristic for improving network partitions" In *Proceedings of the* 19th *Design Automation Conference*, pages 175-181.

[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.] Hirendu Vaishnav and Massoud Pedram. (1995) "Delay optimal partitioning targeting low power VLSI circuits." In *International Conference on Computer Aided Design (ICCAD)*, pages 638- 643. IEEE IACM.

[6.] http://www.facweb.iitkgp.ernet.in/~isg/CAD/SLIDES/07-partitioning.pdf

[7.] Kennedy, J. Eberhart, R. (1995), "Particle Swarm Optimization". In: Proceedings IEEE International Conference on Neural Networks, vol. IV, Perth, Australia, pp. 1942-1948.

[8.] Dirk Behrens, Klaus Harbich, Erich Barke,(1996) "Hierarchical Partitioning" Proceeding of IEEE International Conference ICCAD.

[9.] T. Bui, C. Heigham, C. Jones, and T. Leighton (1989) "Improving the performance of the Kernighan-Lin and Simulated Annealing graph bisection algorithms." In *Proceedings of the* 26th *Design Automation Conference*, pages 775-778.

[10.] A. G. Hoffmann (1991) "Towards optimizing global Min-Cut partitioning." In *Proceedings of the* 2nd *European Design Automation Conference*, pages 167-171. IEEE.

[11.] Kirkpatrick, S., Gelatt, C. and Vechi, M., (May 1983) "Optimization by Simulated Annealing", Science, 220 (4598), 671-680.

[12.] Krishnamurthy, B., (May 1984) "An Improved Min-Cut Algorithm for Partitioning VLSI " Networks", IEEE Trans. Computers, C-33(5), 438-446.

[13.] Crama, Y., and M. Schyns, (1999), "Simulated annealing for complex portfolio selection problems."

[14.] Hu Xiaohui, A., Eberhart, R., (2002). "Multi objective optimization using dynamic neighborhood Particle Swarm Optimization." In: Proceedings of the 2002 Congress on Evolutionary Computation, vol. 2, May 12-17, pp. 1677-1681