International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-2, July 2019

Pair Wise Swapping Based Hypergraph Partitioning Algorithms for VLSI Design

Mitali Sinha

Abstract: ***A VLSI integrated circuit is the most significant part of electronic systems such as personal computer or workstation, digital camera, cell phone or a portable computing device, and automobile. So development within the field of electronic space depends on the design planning of VLSI integrated circuit. Circuit partitioning is most important step in VLSI physical design process. Many heuristic partitioning algorithms are proposed for this problem. The first heuristic algorithm for hypergraph partitioning in the domain of VLSI is FM algorithm. In this paper, I have proposed three variations of FM algorithm by utilizing pair insightful swapping strategies. I have played out a relative investigation of FM and my proposed algorithms utilizing two datasets for example ISPD98 and ISPD99. Test results demonstrate that my proposed calculations outflank the FM algorithm.***

Index terms: Block, Graph, Hypergraph, netcut, netlist, VLSI.

I. INTRODUCTION

VLSI circuit comprises of different components put on a single chip which are associated through wires and a gathering of associated components can be spoken to as a block. Partitioning is a most essential step in the physical structure cycle in which a given VLSI circuit is partitioned into a lot of disjoint blocks of explicit sizes. Size of the wires within the components of two unique blocks is more than the size of wires of components inside a similar block; hence we need to limit the quantity of wires between the components of two distinct blocks to decrease the expense by utilizing a parceling algorithm. Input to the partitioning algorithm is a VLSI circuit and output is number of disjoint blocks with minimum number of inter-block wires. The Significant of a partitioning algorithm in any physical framework configuration is that it breaks down the entire complex framework into a lot of subsystems. Consequently every subsystem can be planned autonomously and all the while to speed of the structure procedure. Partitioning has numerous applications, for example, tremor examination, social database design, reproduction, testing and VLSI design.

A. Hypergraph partitioning problem.

Hypergraph is a theory of diagram wherein an edge relates any number of vertices and this edge is called as hyperedge. Numerically hypergraph can be spoken to as H(V, E) where V is the arrangement of vertices and E is the arrangement of hyperedges.

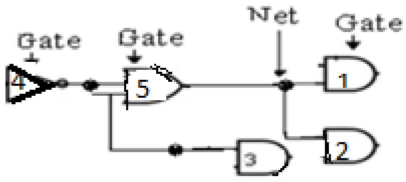
Revised Manuscript Received on 30 July 2019.

***** Correspondence Author

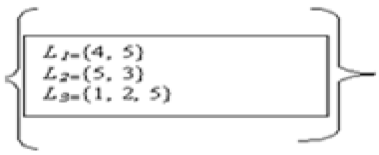
Er Mitali Sinha*, Department of computer science & Engineering, parala maharaja Engineering college Affiliated to Biju Patnaik uni-versity of Technology, Berhampur, Odisha, India

A circuit can be changed over to a hypergraph in which a vertex of hypergraph speaks to a component of the circuit and a hyperedge speaks to the arrangement of components which share same signal known as net. A lot of nets which speak to a circuit is known as netlist. Two components of a circuit are said to be neighbor if both are present within an at least one common net. Let's take an example to illustrate the partitioning problem. A circuit and its netlist portrayal are appeared in fig. 1(a), fig. 1(b) respectively. This netlist contains three nets named as L_1, L_2, and L_3. L_1 contains 4, 5 components and output of 4 is given as input to 5. Similarly L_2 and L_3 can be depicted. Netlist to hypergraph H(V,E) transformation is appeared in fig.1(c) in which V={1,2,3,4,5}and E={L_1,L_2,L_3}. The fundamental objective of the hypergraph partitioning is to put the parts of the circuit in their ideal blocks with the end goal that netcut or hyperedge cut can be limited. A net or hyperedge is said to be cut if its vertices are present in two or more blocks.

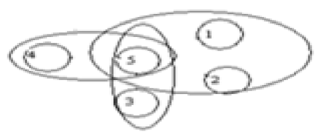
A hypergraph as appeared in fig.1(c) is given to a partitioning algorithm and gap it into two around equivalent sized blocks. The outcome of the partitioning algorithm can be 4, 5, 3 components are placed in one block and 1, 2 are placed in another block as appeared in fig.2. The netcut of this partition is one on the basis that the components of net L_3 are placed in both the blocks.



(a)



(b)



(c)

Fig.1 Input to a partitioning algorithm (a) Circuit (b) Netlist (c) Hypergraph

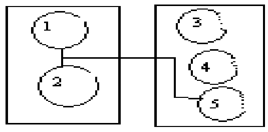


Fig. 2 Output of a partitioning Algorithm with two blocks

Hypergraph partitioning is a NP-hard [1] issue. Despite the fact that partitioning has broad applications in different fields, for example, information mining, work booking, picture handling, enhancing page blame and VLSI plan, various heuristic algorithms were created with polynomial time-complexity. Fiduccia and Mattheyses (FM) algorithm [4] is a first hypergraph partitioning algorithm which time complexity is polynomial in nature.

B. Literature review

So as to take care of the parceling issue in VLSI context, the first polynomial graph bi-partitioning algorithm was proposed in the year 1970 known as KL[2] algorithm. Graph is certifiably not an appropriate portrayal of a circuit [3] in light of the fact that it can't effectively change over a net to an edge or a lot of edges. The most right portrayal of a circuit in graph theorem is hypergraph. So in the year 1982 a hypergraph bi-partitioning algorithm was proposed known as FM algorithm [4]. The main advantage of this algorithm is its polynomial time-complexity which is directly proportional to the size of the circuit. But this algorithm is not worthy for a VLSI circuit. A VLSI circuit contains more than one million of transistors on a single chip. So a various variations of FM algorithm were developed [5,6,10,12,13]. C.J. Alpert and A.B. Kahng has completed an exhaustive review on netlist partitioning [14]. Another class of partitioning algorithm was developed [7,8,9,11,20]. These algorithms comprise of two stages. In the first phase clustering algorithm is applied on original hypergraph in order to reduce the size of hypergraph. Then partitioning algorithm is applied on the reduced hypergraph. Partitioning result of the reduced hypergraph is applied on original hypergraph. This partitioning result is refined using FM or variant of FM algorithm. A predominant expansion of FM algorithm is multi level FM (MLFM) algorithms [15,16,17,18,19] which gives better outcome both as far as solution quality and run time.The multi-level algorithm consists of three stages such as coarsening stages, initial partitioning of coarsest hypergraph, un-coarsening and refinement. In coarsening stage a sequence of coarser hypergraphs are produced until the size of the hypergraph is not greater than a pre-defined size. So a number of smaller hypergraphs are produced at different level of coarsening stages. Suppose H_0 is an original hypergraph, at the first level of coarsening phase H_1 is produced by applying clustering algorithm on H_0. The size of H_1 is smaller than H_0. At the next level of clustering phase H_2 hypergraph is produced from H_1 and its size is smaller than H_1.This process is continued until a pre-defined smallest size hypergraph is produced. The next stage in multi-level algorithm is initial partitioning stage, in which a random initial partitioning of coarsest hypergraph is performed which is refined by FM algorithm or by its variant. The last stage of multi-level algorithm is un-coarsening and refinement stage. In this stage, a partitioning of the coarser hypergraph is projected on its next level finer hypergraph and refined using FM or variant of FM algorithm. This process is continued until partitioning result is projected on original hypergraph and refined using FM or variant of FM algorithm.

C. My contribution

In this paper I have proposed a clever thought of pair wise swapping of components in hypergraph partitioning. At first a circuit is divided in to two roughly equal sized blocks with haphazardly assign the components of a circuit to blocks. Then the selection of components is in pair wise manner, each from one block to reduce the netcut using swapping technique. I have proposed three different types of swapping technique. My proposed algorithms are variations of FM algorithm which I call as Pair Wise swapping based Hypergraph Partitioning (PSHP). I have made a relative execution investigation of PSHPs with FM algorithm utilizing two informational indexes, for example, ISPD98 and ISPD99 benchmark circuits. My trial results demonstrate that PSHPs beat FM algorithm.

II. A PRIMITIVE HYPERGRAPH PARTITIONING ALGORITHM - FM ALGORITHM

The first fundamental hypergraph partitioning algorithm is FM algorithm [4]. It begins with an irregular beginning allotment of the hypergraph into two balanced blocks. Toward the start of the procedure every one of the vertices of hypergraph are made allowed to move from its own block to its complementary block and gain value of every vertex is determined. The gain value of a vertex depends on number of decrease in net from netcut when it is moved from its present block to its complimentary block. Vertices of two blocks are arranged using bucket sorting method according to their gain value. A component c with highest gain value of a bigger sized block is chosen to shift to its complementary block and remains locked all through the procedure. After c is shifted, the gain values of its neighbors are refreshed for next move and the netcut is recorded for that move. This is proceeded until all cells are locked. This whole procedure is known as a pass.
Toward the finish of a pass, the point where the ideal netcut was accomplished is chosen and the moves of all vertices after that point are dropped. The consequence of one pass of FM algorithm is given to next pass as input. This process is continued still no chance of improvement in netcut.

A. Pseudo-code of FM algorithm in a pass

1) An initial random partition is applied on the circuit in order to divide it into two balanced blocks. The size of a block is defined as number of components within that block.

2) The gain of each cell c is calculated on the basis of reduction of nets from netcut due to its move from its current block F to its complementary block T. The formula for gain value of each component c is FS(c) – TE(c). FS(c) is number of nets which contain only c in F block. TE(c) is number of nets which contain c in F block but no component in T block.

3) Components of two blocks are sorted using bucket sorting method.

4) The component c with highest gain value of a block with greater size is selected for shifting and it becomes locked in this pass. Gain values of c' neighboring cells are updated and reduction of netcut due to shifting is recorded.

5) step-4 is repeated until all the components are not locked.

6) After completion of a pass, the point at which reduction in netcut is optimal is selected and shifting of cells after that point is cancelled.

B. Limitation of FM algorithm

- In FM algorithm, selection of component for shifting is depended on the balancing constraint of block's size.
- Swapping of components provides better result than single shifting of a component [10, 21].

These two limitations are overcome in my proposed algorithms.

III. PROPOSED ALGORITHMS

A partitioning algorithm that swaps node pair gives a superior netcut enhancement than one that shifts single node at once [10, 21]. So I have applied pair wise swapping of nodes on hypergraph partitioning with three different techniques. Before going to discuss different pair wise technique I have presented here the gain or reduction in netcut due to pair wise swapping. Let's take u_i and v_j to swap and both are present in two different blocks. The gain due to swapping of u_i and v_j is denoted as Gain(PS) and calculated using the formula i.e $Gain(u_i) + Gain(v_j) - correct\text{-}term$.Gain($u_i$) is already discussed in section 2. The pseudo-code for correct-term is shown in fig.3.

A net which is cut is said to be critical if any one of its cell is shifted from its block to another, then that net is removed from netcut. As shown in fig.4 this net is cut and it is a critical net because its cells are present in both the blocks and if cell 5 is shifted, then this net is removed from netcut.

```
Correct-term
    Begin
        Correct term=0;
            For each criticalnet
                If((ui,vj)∈ criticalnet)
                    If(cardinality of criticalnet is 2)
                        Correct term=Correct term+2;
                    Else
                        Correct term=Correct term+1;
    End
```

Fig3. Pseudo-code for correct term

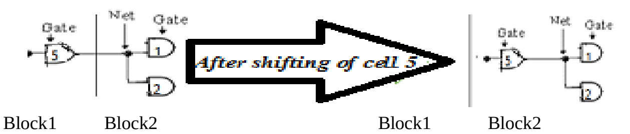


Fig.4 Critical net

A. Pair Wise swapping Based Hypergraph partitioning algorithms

I am now ready to represent the complete description of pair wise swapping algorithm with in a pass:

1. If a hypergraph contains odd number of vertices, then add a dome node. Dome node will have no connection with other vertices.
2. Now a random partition is applied on the hypergraph to divide the vertices into two equal sized blocks let's say A and B blocks. Initially gain due to pair wise swapping is zero, lets denote it as Gain(PS)=0 and netcut is equal to number of nets which components are present in both the block.
3. The gain of each vertex is calculated and it is based on reduction in netcut when it is shifted from its current block to other as described in FM algorithm.
4. Vertices of two blocks are arranged using bucket sorting method according to their gain value. Let's say $\{u_1,u_2,u_3 \ldots\ldots u_n\}$ vertices are sorted in block A and $\{v_1,v_2 \ldots\ldots\ldots\ldots v_n\}$ vertices are sorted in block B.
5. One vertex from each block is selected using a pair wise swapping technique. Let's say u_i from block-A and v_j from block-B are selected for swapping.Now Gain(PS)= Gain(PS) + Gain due to pair wise swapping and netcut=netcut - Gain due to pair wise swapping.
6. Now u_i and v_j are locked. Gain values of all unlocked neighboring vertices of u_i and $v_j$are updated and replaced in their respective buckets.
7. Step-5 and 6 are repeated until all the vertices are locked.
8. At the end of the pass the point at which Gain(PS) is maximum selected and all swaps after that point are cancelled.

I have discussed three types of pair wise swapping techniques below.

- In the first technique after vertices are sorted in both the blocks using bucket sorting which is mentioned above in step-4, a vertex with highest gain value i.e. u_1 is selected from block-A and its non-neighboring vertex with highest gain value is selected from block-B i.ev_k. The time complexity is required for finding the non-neighboring vertex of u_1 is O(k). This step is repeated n times to get n pairs if a hypergraph contains 2n vertices. So total time complexity of this step is O(n*k). Here k <= (maximum number of nets belong to a cell)*(maximum size of a hyperedge). For the future reference this technique is named as Pair-wise Swapping based Hypergraph Partitioing$_1$(PSHP$_1$).
- In the second technique, a vertex with highest gain value from each block is selected for swapping i.eu_1 from block-A and v_1 from Block-B are selected for swapping. The time complexity is required for selecting u_1 and v_1 is O(1). So for selections of n pairs, time complexity is O(n). For the future reference this technique is named as Pair-wise Swapping based Hypergraph Partitioning$_2$(PSHP$_2$).
- In the third technique, u_i and v_j from each block are selected so that u_i and v_j provides highest gain value than any other pair. The procedure for this technique is described below:

1)$\{u_1,u_2,u_3 \ldots\ldots u_n\}$ vertices are sorted in block-A and $\{v_1,v_2 \ldots\ldots\ldots\ldots v_n\}$ vertices are sorted in block B using gain value.

Retrieval Number: B3612078219/19©BEIESP
DOI: 10.35940/ijrte.B3621.078219
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

2) Suppose v_k is the non-neighboring vertex of u_1, with highest gain value. Now the best cell in the block-B is considered for swapping can be any one from $\{v_1, v_2 \ldots\ldots\ldots v_k\}$.

3) Suppose u_k is the non-neighboring vertex of v_1, with highest gain value. Now the best cell in the block-A is considered for swapping can be any one from $\{u_1, u_2, u_3 \ldots\ldots u_k\}$. The cells which are nominated as best can be arranged in matrix form. So the time complexity required to select the best pair from this matrix is $O(k^2)$.

4) This procedures are repeated n times to get n best pairs so total time complexity is $O(n*k^2)$ and k<<<<<<n.

For the future reference this technique is named as Pair-wise Swapping based Hypergraph Partitioning$_3$(PSHP$_3$).

IV. EXPERIMENTAL STUDY

FM and my proposed swapping techniques are tested using two extensive datasets such as ISPD98 and ISPD99 benchmark circuits.

A. Experimental Setup

The source code of the algorithms are applied in 'C++' language and home windows environment. I even have used a 32-bit compiler (Dev C++ Version four.Nine.9.2), 2 GBRAM and a processor which pace is two GHz. Input to the application is a document (dataset) and the output of the program is minimum netcut due to partition.

B. The Dataset

The ISPD98 benchmark circuit is the biggest dataset that's kept up through the Collaborative Benchmarking Laboratory. The ISPD98 benchmark circuit consists of eighteen different styles of records which includes IBM01 to IBM18. Each document comes with three exceptional codecs together with .Internet, .Are and .NetD. Another model of ISPD benchmark circuit is ISPD99. This benchmark circuit consists of 9 extraordinary sorts of file. These datasets are freely to be had inside the website online http://vlsicad.U.S.Edu/UCLAWeb.Html.

C. Performance evaluation on ISPD benchmark circuits

I have performed the experiments using two different datasets. For all the experiments I have defined Gain(μ) as follow

$$\text{Gain}(\mu) = \{(\text{Netcut}_{FM} - \text{Netcut}_{PSHP}) / \text{Netcut}_{FM}\}*100.$$

Performance of PSHP algorithms can be considered to be better if the Gain(μ) is positive. In the first experiment I have computed the netcut of FM and PSHP algorithms by considering ISPD98 as input datasets. In the second experiments I have computed the netcut of FM and PSHP algorithms by taking ISPD99 as input datasets. Gain(μ_1), Gain(μ_2),Gain(μ_3) are used for the comparison of PSHP$_1$and FM, PSHP$_2$and FM, PSHP$_3$ and FM respectively.

Experiment-1:ISPD98 as input dataset.

In this Experiment I actually have considered eighteen exclusive documents of ISPD98 benchmark circuit. I have computed the Netcut of FM and PSHPs algorithms in addition to the gain values as proven in table 1.

Experiment-2:ISPD99 as input dataset.

In this experiment I even have taken 9 one-of-a-kind files of ISPD99 benchmark circuit. I have computed the Netcut of FM and PSHPs algorithms in addition to the gain values as proven in table 2.

V. CONCLUSION

In this work I have proposed 3 styles of Pair-sensible Swapping Hypergraph Partitioning algorithms (PSHPs). I even have carried out an experimental look at to evaluate performance of my proposed PSHPs algorithms and FM algorithm through considering two enter datasets together with ISPD98 and ISPD99 benchmark circuits. From experimental end result it is concluded that my proposed algorithms PSHP$_1$ and PSHP$_3$ outperforms FM algorithm for netlists generated from above datasets. PSHP$_2$ provides better effects in netcut for all files as examine to FM except IBM01C.Thus PSHPs give higher result than FM. If PSHP is applied in partitioning phase of multi-level algorithm, it's going to give higher result than hmetis.

Table I: Netcut incurred by FM and PSHPs for ISPD98

File name	Initial Netcut	Net cut_{FM}	Net cut_{PSHP1}	Gain (μ_1)	Net cut_{PSHP2}	Gain(μ_2)	Net cut_{PSHP3}	Gain (μ_3)
IBM01	9151	1534	526	65.7	645	57.9	858	44.06
IBM02	13443	1595	529	66.8	529	66.8	529	66.8
IBM03	17422	4013	3022	24.6	2189	45.4	2885	28.1
IBM04	20643	4327	1016	76.5	1415	67.2	1016	76.5
IBM05	18895	6881	3060	55.5	2954	57	3402	50.5
IBM06	22798	5721	1541	73	1175	79.46	1475	74.2
IBM07	32044	7028	2493	64.5	2478	64.74	2516	64.2
IBM08	33499	9242	6014	34.9	5770	37.5	3321	64.06
IBM09	40173	10438	1828	82.48	1909	81.71	2809	73.08
IBM10	50647	10413	3904	62.5	2938	71.78	2454	76.43
IBM11	54221	12893	4086	68.3	5321	58.7	4086	68.3
IBM12	52102	14508	4246	70.7	3010	79.25	4312	70.27
IBM13	23076	5275	1265	76	1520	71.18	1159	78
IBM14	101990	22990	10270	55.32	19851	13.6	11257	51.04
IBM15	125878	29037	9678	66.67	9945	65.8	15149	47.82
IBM16	129985	37057	8797	76.26	9787	73.58	7268	80.4
IBM17	131364	42226	10252	75.72	4847	88.52	10062	76.2
IBM18	139169	36949	3055	91.73	3026	91.8	3055	91.73

REFERENCES

1. M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NPCompleteness,"San Francisco, CA: Freeman, 1979.
2. B.W Kernighan and S. Lin, "AnEfficient Heuristic procedure for partitioning Graphs",*The Bell system technical journal*, Vol.49 ,Feb. 1970, pp- 291-307.
3. B.W Kernighan and D.G schweikert, "A proper Model for Partitioning the electrical circuit" *proc. 9th Design Automation workshop*, Dallas, TX, june 1979, pp-57-62.
4. C.M. Fiduccia and R.M. Mattheyses, "A Linear Time Heuristic for improving Network Partition" *19th Design Automation Conference*, 1982, pp-175-181.
5. B. Krishnamurthy, "An Improved Min-cut Algorithm for Partitioning VLSI Network", *IEEE Transactions on computers*, Vol. C-33, May 1984, pp-438-446.
6. L. A. Sanchis, "Multi-way Network Partitioning", *IEEE Transaction on Computers*, Vol. 38, No. 1, January 1989, pp-62-81.
7. T. Bui', C. Heigham, C. Jones and T. Leighton, "Improving the performance of the Kernighan-Lin andSimulated Annealing Graph Bisection Algorithm", *26th ACM/IEEE Design Automation Conference*, 1989 pp-775-778.
8. L. Hagen and A.B. Kahng, "A new approach to effective circuit clustering", *Proc. of IEEE International Conference on Computer Aided Design*, 1992, pp- 422–427.

Retrieval Number: B3612078219/19©BEIESP
DOI: 10.35940/ijrte.B3621.078219
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Table II: Netcut incurred by FM and PSHP for ISPD99

File's Name	Initial Netcut	Net cut_{FM}	Net cut_{PSHP1}	Gain (μ_1)	Net cut_{PSHP2}	Gain (μ_2)	Net cut_{PSHP3}	Gain (μ_3)
IBM01A	9213	2148	599	72.11	599	72.11	364	83.1
IBM01B	4958	685	124	82	95	86.13	124	82
IBM01C	4878	800	451	43.6	852	-6.5	495	38.13
IBM01D	4985	1268	295	76.74	254	80	295	76.74
IBM06A	22889	5147	1315	74.45	1695	67.1	1328	74.2
IBM06B	9962	2565	948	63.41	554	78.4	948	63.41
IBM06C	14558	3044	977	68	938	69.2	963	68.4
IBM06D	8693	2360	871	63.09	870	63.1	829	65
IBM09A	40187	9966	2894	70.96	3147	68.42	3026	69.63
IBM09B	33104	8064	4040	50	3666	54.53	3830	52.51
IBM09C	36086	8022	4457	44.44	2311	71.2	4706	41.34
IBM09D	33809	8343	3890	53.37	5656	32.21	2674	67.95
IBM10A	50751	11989	2661	77.8	3329	72.23	2865	76.1
IBM10B	20021	4259	1299	69.5	1250	70.65	2551	40.1
IBM10C	32876	7819	2322	70.3	1159	85.18	2322	70.3
IBM10D	21601	4938	1394	71.77	3032	38.6	1394	71.77
IBM11A	54079	12833	4885	61.93	3371	73.73	4408	65.65
IBM11B	27379	5701	2404	57.83	3220	43.52	3006	47.27
IBM11C	29364	7613	2816	63	1654	78.27	2744	63.95
IBM11D	24600	5471	2948	46.12	2460	55.04	3219	41.16
IBM12A	51921	13339	3803	71.49	4105	69.23	3671	72.48
IBM12B	29498	6981	2623	62.43	2337	66.52	2755	60.53
IBM12C	25109	5533	2239	59.53	1909	65.49	2018	63.53
IBM12D	22915	6805	1992	70.73	1863	72.62	1841	72.94
IBM13A	66251	15645	3184	79.65	4355	72.16	3197	79.56
IBM13B	32990	6813	501	92.64	498	92.69	499	92.67
IBM13C	35728	4461	1220	72.65	1282	71.26	1427	68
IBM13D	31019	6766	1622	76.03	2076	69.32	2019	70.16
IBM16A	129950	36869	7633	79.3	6626	82.03	6844	81.44
IBM16B	72220	19083	3129	83.6	4172	78.14	2653	86.1
IBM16C	61793	18269	3047	83.32	4008	78.06	3047	83.32
IBM16D	48717	14341	4574	68.1	2216	84.54	3084	78.5
IBM17A	131753	43544	9609	77.93	5698	86.9	9609	77.93
IBM17B	74174	24782	3349	86.48	4468	82	2186	91.17
IBM17C	62567	18965	6172	67.45	8935	52.89	4157	78.1
IBM17D	41472	11707	3593	69.31	5334	54.44	3802	67.52

9. H. Shin, and C. Kim, "A Simple Yet effective technique for partitioning", IEEE Transactions OnVery Large Scale Integration (VLSI) SYSTEMS, Vol. 1, No. 3, September 1993, pp-380-386.
10. S. Dutt, "A New Faster Kernighan- Lin Type Graph Partitioning Algorithm", In proc. IEEE Intel Conf. computer Aided Design, 1993, pp-370-377.
11. Cong and smith, "A parallel bottom-up clustering algorithm with application to VLSI partitioning", InProc. ACM/IEEE Design Automation Conference, 1993, pp-755-760.
12. A. G. Hoffmann, "The Dynamic Locking Heuristic-A new Graph Partitioning algorithm", In Proc. IEEE Intel Sym. Circuit and Systems, 1994, pp-173-176.
13. L.W. hagen, D.J. Haung,A.BKahng, "On Implementation choices for iterative improvement partitioning algorithms", European Design Automation Conference, 1995, pp-144-149.
14. C.J. Alpert, A. B Kahng, "Recent directions in netlist partitioning: a survey", Elsevier the VLSI Integration Journal, Vol.19, 1995, pp- 1-81.
15. G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multi-level hyper graph partitioning algorithm: Application in VLSI domain", 34th Design Automation Conference, Vol. 7, No.1, 1997.
16. C.J. Alpert, J.H. Huang, and A.B. Kahng, "Multi-level circuit partitioning" IEEE Transaction on Computer-Aided Design of Integrated Circuit and Systems, Vol. 17, No. 8, 1998.
17. Y. Saab, "A new 2-way multi-level partitioning algorithm", VLSI Design Overseas Publishers Association, Vol. 2, No. 3, 2000, pp. 301-310.
18. G. Karypis, V. Kumar, "Multi-level K-way hyper graph partitioning", VLSI Design Overseas Publishers Association, 2000, pp. 1 – 16.
19. J. Congand S.K. Lim, "Edge separability based circuit clustering with application to multi-level circuit partitioning", IEEE Transactions On Computer-Aided Design Of Integrated Circuits and Systems, Vol. 23, No. 3, 2004.
20. M. Rajesh and R. Manikandan, "Efficient FM Algorithm for VLSI Circuit Partitioning", International Journal of Engineering and Technology, Vol. 5, No 2, May 2013, pp-968-972.
21. M. Sinha, R. Mohanty, P. Tripathy, S. Pattanaik "Experimental study of a novel variant of Fiduccia Mattheyses(FM) partitioning algorithm" (2016). pp-1820-1825. DOI:10.1109/ICACCI.2016.7732313.

AUTHORS PROFILE

Mrs. Mitali Sinha, Biju patnaik university of technology (BPUT) (mitalisinha148@gmail.com), Parala maharaja engineering college. She is working as assistant professor in parala maharaja engineering college in the department of computer science and engineering. She published the paper in 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI).