Implementation of Multi-Way Partitioning Algorithm

Kulpreet S. Sikand, Sandeep S. Gill, R. Chandel, and A. Chandel

***Abstract*—This paper presents a discussion of methods to solve partitioning problems and advocates the use of multi-way partitioning algorithms. The paper gives an implementation of a multi-way partitioning algorithm based on partitioning without size constraint and iterative improvement. A top-down clustering technique is employed to deal with the local minima problems faced in common heuristics and a primal-dual approach is used to enhance the iterative improvement. The Fiduccia-Mattheyses (FM) algorithm has been taken as the core algorithm which has been subjected to iterations, clustering and primal-dual iterations. The algorithm has been implemented in a way that it gives netlist files for each partitioned block. These netlists can further be used to implement actual hardware or detailed analysis. The results obtained were compared to the results obtained from the traditional FM algorithm. The results show good improvements.**

***Index Terms*—Benchmarks, Cells, Clustering, Hypergraph, Net, Netlist, Nodes, Pads, Partitioning, Primal-Dual.**

I. Introduction

The Electronic Design Automation (EDA) involves automation of all those tasks that are used in fabrication of electronic circuits on silicon. Circuit designers specify their circuit requirements in programming languages like Hardware Description Languages (HDL) (Commonly used languages are VHDL, Verilog and Analog VHDL). Specifications are analyzed and modifications are made based on varying requirements. Once the requirements are complete, the HDL based codes are synthesized into gate-level netlist. The netlist mainly contains gates (such as AND gates, OR gates etc.) widely called as cells, and the interconnecting wires (that interconnect gates) widely called as nets. This netlist is then subjected to physical design automation (back-end flow), where cells are assigned different areas on actual silicon (placement) and the actual routes or paths that each connection (interconnecting wire) should take to connect the cells are identified (routing). During the placement various factors like wire length, path delay, congestion (when a local region contains more nets than the available routing tracks the region is said to be congested) etc. are considered. Most of these factors can be resolved using various hypergraph partitioning algorithms.

Consider a system; partitioning will divide the whole circuit from the system level to the board level, from the board level to the chip level, and from the chip level to the macro-cell level. At each level, circuits are further divided into smaller sub-circuits. A good partitioning will work to significantly reduce the complexity of the problem and improve both the reliability and the timing performance of the system.

Historic research data reveals that the choice of the objective function is usually set to minimize the number of nets connecting the two final subsets (called blocks i.e in case of a bi-partitioning method). In case of different designs of multiple blocks partitioning, often the demand is for a different objective function or functions. For example if we consider silicon physical layout, the partitioning of a circuit must guarantee that the resultant sub-circuits have a number of IO pins (or pads etc.) that are within the physical limit requirements. So**,** to ensure a feasible implementation one objective function could be to minimize the maximum number of IO pins. A second objective function that can be considered for physical layout is to simplify the routing problem. For example if a net is connected exactly to say x blocks, then the cost function can be assigned a value of x. The objective function in this case is to minimize the sum of all costs assigned to each net. A third possible objective function for silicon physical layout from the architectural point of view is to have minimal interface signals among the blocks resulting from partitioning. So, clearly the objective function in this case is to minimize the number of nets connecting more than two blocks. Many more objective functions can be derived from the variable requirements and problems. As all these problems and requirements optimize on different objective functions the traditional two-way partitioning algorithms cannot be applied directly to solve them. Hence the need for multi-way and multi-objective partitioning algorithm is evident. In this paper an attempt has been made to advocate the use of multi-way partitioning algorithms over the two-way partitioning algorithms based on their performance. The only addition to the existing multi-way partitioning algorithm which this paper proposes is the addition of pads to the output netlist files to make them self-revealing standalone files, this easies further analysis of these files. The organization of the paper is as follows:

Section 2 gives a brief review of previous research.

Section 3 introduces a formal definition of the problem.

This section presents an iterative improvement algorithm for partitioning. The algorithm utilizes a top-down clustering technique and a Primal-Dual iteration to enhance the partitioning result.

Section 4 contains experimental results & discussions.

Section 5 contains the conclusion.

II. Previous Work

Attempts have been made to solve graph and network related partitioning problems with specified bound on the

Manuscript received August 14, 2012; revised September 20, 2012.

Kulpreet S. Sikand and Sandeep S. Gill are with the Guru Nanak Dev Engg. College, Ludhiana, Punjab, India (e-mail: kpsikand@gmail.com, ssg@gndec.ac.in).

R. Chandel and A. Chandel are with the NIT Hamirpur, Hamirpur, India (e-mail: rchandel@yahoo.com, ashwani@nithm.ac.in).

sizes of the resulting subsets. These attempts have concentrated on finding approximate solutions in polynomial time. Several approaches and several algorithms have been devised to find out approximate solutions**.** B. W. Kernighan and S. Lin [1] proposed a two-way partitioning algorithm with constraints on the final subset sizes**.** The algorithm applied swapping iterations on all pairs of nodes to find the best improvement on the existing partition, the swapping was done pair-wise. D. G. Schweikert and B. W. Kernighan [3] proposed a net cut model for two-way partitioning. The concept of multi-pin nets was defined and used for partitioning. C. M. Fiduccia and R. M. Mattheyses [4] further improved this algorithm. They were able to developed useful data structures that helped in reducing time complexity of the algorithm. The complexity was reduced to O(P), where P is the total number of pins. Much of the research was directed to the problem of multi-pin net models. C. Sechen and D. Chen [7] proposed the net crossing model derived from row-based layout where probability analysis is used to estimate the gain of a move. B. Krishnamurthy [5] introduced the multiple level gain model for multi-pin nets. L. A. Sanchis [9] used the multiple level gain concepts to introduce a new model of multiple-way partitioning. C. W. Yeh, C. K Cheng and T. T. Lin [11] further suggested an improved multi-way partitioning algorithm based on traditional two-way portioning algorithm.

III. Problem Formulation

Let us consider a Hypergraph denoted by $H(V, E)$, where H stands for Hypergraph, V stands for set of nodes ($V = \{v_i \mid i = 1, 2, \ldots, n\}$) and E stands for set of nets ($E = \{e_u \mid u = 1, 2, \ldots, m\}$). Each net e_u is a subset of V with cardinality $\mid e_u \mid \geq 2$. A k-way partition is a partition that assigns v_i into k non-empty blocks as $V_1, V_2, \ldots, V_k$. Let us consider a term "SPAN" of a net, which is zero if the net connects exactly to one block and say s if it connects exactly to s blocks. (consider $s \geq 2$). Different objectives [11, 13, 15] can be considered for this k-way partitioning problem:

Objective 1:

$$\min \max_{s \in (1 \ldots k)} imun \mid \{e_u \mid \text{span}(e_u) \geq 2,\ e_u \cap V_s \neq \cap \phi \tag{1}$$

Objective 2:

$$\min \mid \{e_u \mid \text{span}(e_u) \geq 2\} \mid \tag{2}$$

Objective 3:

$$\min \sum_{\text{eu} \in \text{E}} \text{span}(e_u) \tag{3}$$

Subject to: $C_m \leq |V_b| \leq C_M$ (4)

where C_m , C_M are two constants that set the size limit of each block, $0 < C_m < C_M < |V|$.

So the objective is not only to reduce the net cut but to reduce the span of each net and its cardinality. As mentioned earlier in the abstract that the core algorithm used is FM algorithm [4], some of its basic equations are:

$$g_i = D_{ai} + D_{bi} - 2c_{\,ai\,bi} \tag{5}$$

where a_i and b_i are the nodes of two partitions A and B respectively, c_{aibi} is the cost function. D is the difference between the external and internal edge cost.

$$Dx = Ex - Ix \tag{6}$$

E is the external edge cost, which measures the connections from node a to b or vice-versa.

$$E_a = \sum_{y \in B} c_{ay} \tag{7}$$

I is the internal edge cost to measure the internal connections to a (or internal connections to b).

$$I_a = \sum_{z \in a} c_{az} \tag{8}$$

Pseudocode for updating gain is given as [4]:

1) Begin /* Move Base cell and update neighbors' gains */
2) F :- the Front Block of the base cell.
3) T :- the To Block of the base cell.
4) Lock the base cell and complement its block.
5) For each net n on the base cell do /* check critical nets before the move */
6) If $T(n)$=0 then increment gains of all free cells on n; elseif $T(n) = 1$ then decrement gain of the only T cell on n, if it is free /* change $F(n)$ and $T(n)$ to reflect the move */
7) $F(n) <= F(n) - 1$; $T(n) <= T(n) + 1$; /* check critical nets after the move */
8) If $F(n) = 0$ the decrement gains of all free cells on n elseif $F(n) = 1$ then increment gain of the only F cell on n, if it is free.
9) End.

Data structures are used for updating gain. As traditional FM algorithm is a bi-partitioning algorithm hence it gives only two partitions as output. Clustering and iterations are used to obtain multi-way and multi-objective partitions.

The primal dual approach is based upon the concept of duality and consists of three major parts: Top-Down clustering, Uniform Multi-pin net model and primal iteration.

A. Top Down Clustering

The Kernighan-Lin (KL) based algorithms share the common weakness that they are often trapped by local minima when the size of the circuit is very large. One way to overcome this difficulty is to group highly connected sub-circuits into clusters and then condense these clusters into single nodes prior to the execution of the KL based algorithms. The complexity of the problem is thus dramatically reduced, which in turn improves the performance of the algorithm [5]. Traditionally, clustering has been carried out in a bottom-up fashion. This approach lacks the global view of the entire network and so is prone to produce incorrect grouping. Recently, top-down clustering technique has been introduced by employing the clustering nature of a ratio-cut and repeatedly applying the two-way ratio-cut algorithm to partition the network into highly connected groups. The top-down clustering procedure [11] is as follows:

Consider a hypergraph $H(V, E)$, and a predefined cluster size limit C_s,

- Consider $\alpha = \{V\}$ where V is set of nodes.

- Chose a subset $V^* \in \alpha$ such that $|V^*| = \max_{Vi \in \alpha} |Vi|$ If ($|V^*| \leq C_s$) then exit.
- Set $\alpha = \alpha - \{V^*\}$
- Apply Ratio cut algorithm to V^* to get a cut (A, A') where $V^* = A \cup A$'

Set $\alpha = \alpha \cup \{A, A'\}$, and go to step2.

B. Multi-Pin Net Model

As mentioned earlier most of the traditional algorithms (KL algorithm etc.) were based on multi-pin net model. A good multi-pin net model can not only correctly reflect the immediate gain of a move but it can also calculate and give the potential gain of a move. The calculation of the potential move forms the basis of the multi-pin net model. This look-ahead mechanism increases the probability of choosing the best move. If we consider a hypergraph, then each net connecting to more than two nodes in a hypergraph corresponds to a multi-pin net and this hypergraph model is called multi-pin net model.

All the existing multi-pin net models intend to estimate the "goodness" or "badness" of moving one single node at a time, and they have achieved excellent results, but in some cases this still is not satisfactory. For example consider a situation as shown in Fig. 1. Suppose that during the execution of the algorithm, node D, A and B have not been locked, i.e., they are allowed to be moved to the other blocks. Suppose the nets on D, A, and B are as shown as in Fig. 1. Moving D would remove nets e_i, e_k and e_l from the cut set and would introduce net e_n to be cut. When calculated the gain for this move come out to 2. Moving A would not remove any net. But if we move node B and node A together, it would help in removing the nets e_k, e_j, and e_i. But various other models like the net cut model, the level gain model or the probabilistic model will always the support the step of moving D and definitely stop the movement of A and B.

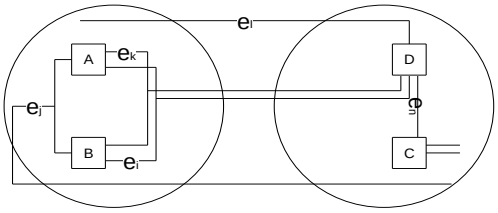

Fig. 1. Multi-pin net model example.

Let's say if a freedom of choosing a node among node D, node A and node B is given or it is allowed to form a cluster of the nodes A and B and further it is allowed to move them as a cluster. Then the latter move (i.e move A and B as cluster) will always be preferred. In other words, the algorithm would have a better judgment if it had the freedom to move more than one node at a time. The question then arises as which nodes should be clubbed together to form a cluster and then the cluster is moved. The answer lies in the positioning of nets, the focus should be on removing nets and not on removing nodes. If net e_j needs to be eliminated from the cut set, nodes A and B have to be moved together. This would also remove the nets e_i, and e_k at the same time. Thus the calculated gain would be 3. On the other hand, if it's decided to remove net e_l from the cut-set only D will be moved, and the calculated gain would be 2. A comparison of the gains of the nets shows that net e_j has the largest gain among all of the nets. This supports that A and B should be moved together. Therefore the ambiguity associated with selecting moves would be greatly reduced, if a "move" is viewed as initiated by a net instead of a node. Consider a net e_u and a block b. Let us define two sets, the first one critical set of e_u and the second complimentary critical set of e_u. Critical set is given as [11]:

$$S_{ub} = \{ v \mid v \in e_u \text{ and } v \in V_b \} \quad (9)$$

Complimentary critical set is given as:

$$S_u\overline{b} = \{v \mid v \in e_u \text{ and } v \in \overline{V}_b\} \quad (10)$$

The objectives mentioned earlier can be explained using the critical and complimentary critical sets. Objective 1 and 3 can be understood as placing the critical set S_{ub} into a block other than b. In objective 2 a move associated with e_u is defined by placing the complimentary critical set $S_u\overline{b}$ into block b. The gain of each move can be calculated based on the change in cost brought by the movement of critical and complimentary critical sets.

C. The Iteration

As mentioned earlier the FM algorithm is utilized as primal process. The adaptations [11], [13], [15] of the algorithm to multiple-way partitioning problem consist of the following:

1) For each block b, a sorted list of moves is kept which shifts nodes from block b to each of the other blocks. This sorted list is called a "bucket" and bears the same structure as that in the FM algorithm. The gains of moves are computed according to the objective function.
2) In order to assure the convergence of the algorithm, a "rejecting" mechanism is imposed which prohibits a node from being moved to a block if this node had resided in the same block before.
3) During each trial of move, the best move among all buckets is selected and performed. This procedure continues until either all possible moves are "rejected, or none of the remaining moves will satisfy the size constraint.

The Dual process [11] is similar to the primal process except instead of shifting a single node, the whole critical or complimentary critical sets are shifted as explained in multi-pin net model. The flow chart [11] of the whole algorithm is shown in Fig. 2.

The last step is to make netlist files of the partitions created by the algorithm. First consider there are only two partitions. The total net cut can be found from the gain data structure. The total net cut is the total no. of nets being cut after final partitioning, as shown in Fig. 4. Only for further analysis sometimes these partitions may be required as standalone partitions as if they represent a complete circuit. A circuit which is complete in itself shall not have any net cuts, hence for a partition to represent a complete circuit the net cuts have to be replaced by pins or pads (external IOs) as shown in Fig. 3 extra pads P4, P5, P6 have been added. So, two partitions will represent two independent circuits. Similar thing can be repeated for other partitions that have been formed by a multi

way partitioning algorithm.

IV. Results and Discussions

Use A number of benchmark circuits like IC67.net, IBM0.net, IBM1.net and one randomly generated circuit form a VHDL file Test_kp.net (sample.net) were used to compare the algorithm outputs. The algorithms were implemented in c/c++ and run on a dual core Turion machine. The algorithms were run for atleast 20 times for each netlist file and there averages were tabulated (the objective considered was: minimize the connections between the partitions/blocks i.e min cut).

The data in the Table I show much greater improvement for primal approach then for the traditional FM algorithm. The algorithms were run for 2 partitions, 4 partitions, 8 partitions (shown as 0 level, 1 level, 2 level partitions). The average improvements are .058%, 26.8%, 29.43% respectively. For zero level partition all the algorithms show identical results, as both algorithms are based on same traditional algorithm i.e FM. Except for 0-level partitioning, where both algorithms reach the same value for most of the cases, almost all cases experience a noticeable improvement.

A multiple-way network partitioning algorithm was implemented which covered multiple objectives and showed improved results. Standalone partitions were made useful for further analysis.

TABLE I: Comparison of Tradition Algorithm and Multi-way Partitioning Algorithm

Level 0 Partitioning			
Netlist File	Traditional Algorithm	Primal Dual Approach	Improvement %
IC 67	38	38	0
IBM0	574	573	.174
Sample	3	3	0
Level 1 Partitioning			
Netlist File	Traditional Algorithm	Primal Dual Approach	Improvement %
IC 67	48	42	12.5
IBM0	646	530	17.9
Sample	4	2	50
Level 2 Partitioning			
Netlist File	Traditional Algorithm	Primal Dual Approach	Improvement %
IC 67	44	31	29.54
IBM0	704	524	25.56
Sample	3	2	33.33

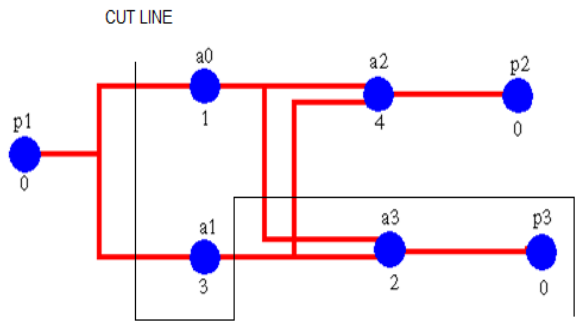

Fig. 4. Netlist graph representation with cutline.

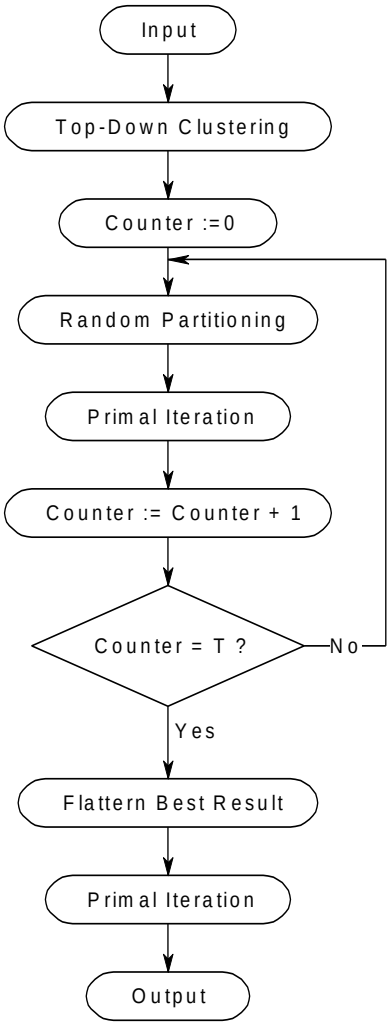

Fig. 2. Primal iteration algorithm.

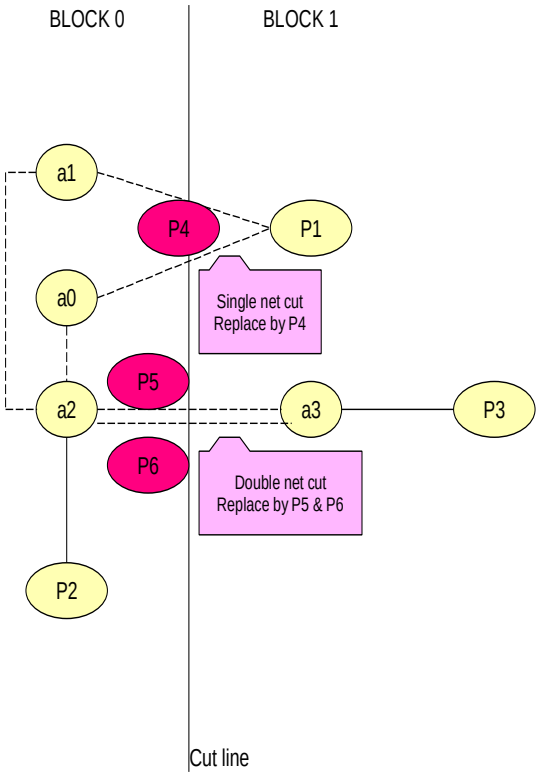

Fig. 3. Partitioning and netlist generation example.

V. Conclusions

FM algorithm is primarily a bi-partitioning technique; it is capable of dividing a single netlist file into two equally sized partitions. With the advancements in the technology we are coming up with more & more complex IC's day by day, this creates bigger circuits which in hand requires the circuit to be partitioned into more than two partitions with more than one objectives and constraints; this is not possible with the core FM technique.

A multiple-way network partitioning algorithm unlike FM algorithm can handle & cover more than one objective function. It is very evident that the field of circuit design has tremendously grown, the performance now not only depends on nets (link between modules) but it depends on various factors & partitioning constraints like the time delay, thermal constraint, and noise isolation etc. These constraints can only be handled by a multi-way & multi-objective algorithm. FM algorithm can only act as aid but is insufficient to meet all the constraints of partitioning.

References

[1] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell System Technical Journal*, vol. 49, no. 2, Feb. 1970, pp. 291-307.

[2] M. A. Breuer, "Recent developments in automated design and analysis of digital systems," *Proc. IEEE*, vol. 60, January, 1972, pp. 12-27.

[3] D. *G.* Schweikert and B. W. Kernighan, "A proper model for the partitioning of electrical circuits," *9th Design Automation Workshop*, 1972, pp. 57-62.

[4] C. M. Fiduccia and R. M. Mattheyses, "A linear time heuristic for improving network partitions," *Proc 19th ACM/IEEE Design Automation Conference*, 1982, pp. 175-181.

[5] B. Krishnamurthy, "An improved min-cut algorithm for partitioning VLSI networks," *IEEE Trans. on Computers*, vol. C-33, May 1984. pp. 438-446.

[6] S. Meyer, "A data structure for circuit netlists," in *proceedings of 25th ACM/IEEE Design Automation Conference*, vol. 39, no. 1, 1988.

[7] C. Sechen and D. Chen, "An improved objective function for mincut circuit partitioning," *Proc. Int Conf. on Computer-Aided Design*, 1988. pp. 502-505.

[8] A. B. Kahng, "Fast Hypergraph partitioning," *Proc. of 26th ACM/IEEE Design Automation Conference*, vol. 43, no. 2, 1989.

[9] L. A. Sanchis, "Multi-way network partitioning," *IEEE Trans. on Computers*. vo1.38, no.1. Jan 1989. pp. 62-81.

[10] Y.-C. Wei and C.-K. Cheng, "A two-level two-way partitioning algorithm," *Proc. Of IEEE conference* 1990.

[11] C. W. Yeh, C. K. Cheng, and T. T. Lin, "A general purpose multiple-way partitioning algorithm," in *Proceedings of the Design Automation Conference*, pp. 421-426. 1991.

[12] J. Cong, L. Hagen, and A. Kahng, "Net partitions yield better module partitions," in *Proceedings of 29th ACM/IEEE Design Automation Conference*, vol. 5, no. 1, 1992.

[13] J. Cong, W. Labio, and N. Shivkumar, "Multi-way VLSI circuit partitioning based on dual net representation," *Proc. of International Conference for Computer Aided Design*, pp. 56-62, Nov. 1994.

[14] Y.-P. Chen, "Algorithms for VLSI partitioning and routing," *Dissertation for the Degree of Doctor of Philosophy*, August, 1996.

[15] X. Tan, J. Tong, P. Tan, N. Park, and F. Lombardi, "An efficient multi-way algorithm for balanced partitioning of VLSI circuits," *Proceedings of International Conference on Computer Design*, 1997.

[16] G. Karypis and V. Kumar, "Multilevel k-way hypergraph partitioning," in *Proceedings of the Design and Automation Conference*, 1999.

[17] S.-J. Chen and C.-K. Cheng, "Tutorial on VLSI partitioning," *VLSI Design*, vol. 11, no. 3, pp. 175-218, 2000.

[18] Z. Zhao, L. Tao, and Y. Zhao, "An effective algorithm for multi-way hypergraph partitioning," *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, vol. 49, no. 8, August 2002.

[19] S. N. Adya, M. C. Yildiz, I. L. Markov, P. G. Villarrubia, P. N. Parakh, and P. H. Madden, "Benchmarking for large-scale placement and beyond," *Paper in Proc. of ISPD*, April 2003.