An Efficient Multi-Way Algorithm for Balanced Partitioning of VLSI Circuits *

X. Tan, J. Tong and P. Tan
Fudan University
Department of Electronic Engineering
Shanghai 200433 PRC

N. Park and F. Lombardi
Texas A&M University
Department of Computer Science
College Station, TX77843-3112

Abstract

This paper presents an efficient algorithm for multi-way balanced partitioning of VLSI circuits. The proposed algorithm is still based on the widely used net-cut model, but its novelty is the potential gain function into the net-cut cost function to relax the single-cell-move constraint (as commonly encountered in the Kernighan-Lin algorithm) for balanced partitioning. This feature permits to move a group of cells (referred to as a Multi-Cell-Move strategy) for partitioning a circuit, while reducing its sensitivity to size constraint. The new multi-way partitioning algorithm is fully analyzed; expressions for the potential gain function (with respect to the multi-move operation) and the cost (for the min-cut objective function) are presented. The time complexity of the proposed partitioning algorithm is $O(P \times k^2 \log_2(k))$, where k is the number of blocks and P is the number of pins. Simulation results are presented and a remarkable improvement is achieved compared with existing algorithms, such as the k-DualPart algorithm.

1. Introduction

The multi-way network partitioning problem consists of dividing a VLSI circuit (specified by a set of cells and their connections in a netlist), into k disjoint subsets (blocks or modules) under some constraints. Note that usually k is specified a-priori. Partitioning must be performed such that a given objective function is optimized. The constraints are usually the size or number of I/O pins on each subset. For k=2 (i.e. the two-way case), the objective is usually to minimize the number of nets (called a cut) which have cells in the two subsets. Theoretically, the min-cut partitioning problem with a size constraint belongs to the class of NP-complete problems [1].

Kernighan and Lin [3] have proposed the group migration algorithm (KL) for the two-way graph partitioning problem which through years of use has been proved to be very efficient. The complexity of the KL algorithm is $n^2 \times \log_2(n)$, where n is the number of vertices. Schweikert and Kernighan in [4] have proposed a net-cut model which extended the KL method, and have made it possible to handle nets connecting more than two cells [8,9]. Fiduccia and Mattheyese [2] (FM) have further improved the KL algorithm by introducing an elegant bucket sorting technique with a dependency on P, where P is the number of pins. After the FM algorithm, many algorithms have been proposed such as the multi-level gain algorithm by Kirshnamurthy [5] and the probabilistic analysis algorithm by Sechen [7]. Both these algorithms take into account the potential interactions in moving free cells, such that a better sequence can be established for the move operations of free cells with equal net-cut gain. Multi-way partitioning algorithms include the recursive bipartitioning method by Kernighan and Lin [4], the multi-level gain algorithm [6], the primal-dual algorithm [10] and the dual netlist representation algorithm [12]. Both algorithms in [10, 12] relax the single cell-move restriction and allow the move of a group of cells; remarkable improvements over the level-gain multi-way partitioning algorithm of [6] were achieved. Although different methods have been investigated and proposed for solving the partitioning problem, group migration still remains the dominant heuristic criterion, because of its efficiency and ease of implementation. As partitioning techniques have been extensively used for hierarchical VLSI design [8], then time complexity is of primary importance.

A further problem which is addressed in this paper, is a procedure for obtaining balanced partitions. An optimal balanced k-way partitioning solution for a given netlist must satisfy the following conditions [12]: 1. Each module is assigned to exactly one partition. 2. The constraint (such as total area of the modules in each partition) is within the user specified bounds. 3. The number of nets being cut is minimized.

It is well documented that the qualitative figures of merit of most partitioning algorithms are very sensitive to size constraints; hence, some algorithms relax the size constraints to

*This research supported by grants from China SSTC and the Texas ATP and ARP.

obtain better results. The net-based algorithm [10,11,12] is an example of a method which assigns nets (instead of cells) into subsets. Such method improves over previous results, but it still seems to be very sensitive to size constraints. So, Yeh [10] has relaxed the size constraints to allow 50% deviation from the balanced partitions.

In this paper, we propose an algorithm that introduces a novel (discrete) expression to evaluate the potential gain between different move operations of free cells. To obtain more balanced partitions, we introduce a new multi-cell-move (MCM) algorithm which not only improves the quality of partitioning, but it also reduces its sensitivity to size constraints. The proposed algorithm also obtains more balanced partitions. Simulation results are provided.

2. Problem Definition

A netlist is commonly used in VLSI CAD to represent a circuit [12]; a netlist can be considered as a hypergraph with the vertices corresponding to the cells (or modules, gates, components or registers) and the edge corresponding to the signal nets. Let $H(V,E)$ denote a hypergraph with vertex set $V=\{v_1,v_2,...,v_m\}$ and net set $E=\{e_1,e_2,...,e_n\}$. Each net e_j is also a subset of V with $|e_j|\geq 2$.

The k-way partitioning problem consists of assigning a v_i into k non-empty subsets $V_1,V_2,..,V_k$. Let's define the *span* of a net e_j (and denoted as $span(e_j)$) as the number of subsets which net e_j connects. Note that if $span(e_j)\geq 2$, then e_j becomes a cut. Therefore, the two objectives along with the size constraint for the subsets, can be mathematically described as follows:

$$min-cut : \min|e_j|span(e_j) \geq 2| \qquad (01)$$

$$min-span : \min \sum_{e_j \in E} span(e_j) \qquad (02)$$

subject to the commonly used area constraint of

$$\frac{1}{k}W - \tau \leq Area(V_i) \leq \frac{1}{k}W + \tau \qquad (03)$$

for i=1,2, ... , k.

(01) corresponds to the min-cut objective, while (02) is the objective to minimize the sum of spans of all nets. $Area(V_i)$ is the size of subset V_i, W is the total area of the cells (given by $W=\sum_{i=1}^{m} Area(V_i)$), τ is the measure of the offset from its balanced size (referred to as *deviation factor* in this paper). As in [5,6], the *incident number* of a net e with respect to a set of cells (given by A and denoted by $\alpha_A(e)$) is defined as the number of cells in A that are on net e, i.e.

$$\alpha_A(e) = |c|c \in A and C \in e| = |A \cap e| \qquad (04)$$

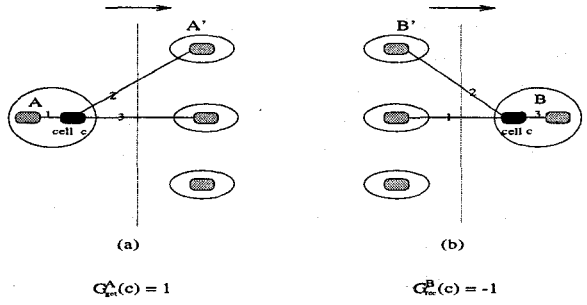

Figure 1. The Two Steps in The Calculation of Gain

where c denotes a cell. The *binding force* of a net e with respect to the set A (denoted as $\beta_A(e)$) is defined as

$$\beta_A(e) = \alpha_{A_F}(e) \quad if \quad \alpha_{A_L}(e) = 0 \qquad (05a)$$

and

$$\beta_A(e) = \infty \quad if \alpha_{A_L}(e) > 0 \qquad (05b)$$

where A_F (A_L) denotes the subset that contains all free cells of A (all locked cells of A). The binding force can be intuitively viewed as a indicator of how tightly a net is bound to a set.

3 The New Multiple-Way Partitioning Algorithm

To derive a new gain cost function, consider initially the min-span objective function. The proposed approach is based on dividing the move operation of cell c from A to B into two steps.

1. Cell c is selected from A and put into $\bar{A}$ (i.e. the complement of A, $\bar{A} = V - A$). This is shown in Figure (1a); the corresponding net cut gain (denoted as $G^A{}_{get}(c)$) is expressed as:

$$G^A{}_{get}(c) = |\{e \in E_c|\beta_A(e) = 1 \ and \ \beta_{\bar{A}}(e) > 0\}| - |\{e \in E_c|\beta_A(e) > 0 \ and \ \beta_{\bar{A}}(e) = 0\}| \qquad (06)$$

where E_c denotes the set of nets that cell c connects, i.e E_c= $\{e \mid c \in e\}$.

2. Cell c is moved from the complement of B ($\bar{B}$) to B. This is shown in Figure (1b). $G^B{}_{rec}(c)$ denotes the gain of this step which is given by

$$G^B{}_{rec}(c) = |\{e \in E_c|\beta_{\bar{B}}(e) = 1 \ and \ \beta_B(e) > 0\}| - |\{e \in E_c|\beta_{\bar{B}}(e) > 0 \ and \ \beta_B(e) = 0\}| \qquad (07)$$

Then, the total gain of the move operation of c from A to B is given by

$$G_{min-span}(c) = G^A{}_{get}(c) + G^B{}_{rec}(c) \qquad (08)$$

Note that two steps are not explicitly considered in the move operation described above; these are the steps from $\bar{A}$ to a temporary set (denoted by T) and from T to $\bar{B}$, where $T = V - (A \cup B)$. As these two steps do not contribute to the cell's gain, then they are not considered in the gain expression.

As it can be proved that the total decrease of spans of all nets in E_c is equal to the negative value of $G_{min-span}$, i.e. $\sum_{e \in E_c} \Delta span(e) = -G_{min-span}$, then the terms that only contribute to the change of spans in (06) and (07), can be eliminated. The cost expression for the min-cut objective function is therefore identical to [6], i.e.

$$G_{min-cut}(c) = |\{e \in E_c | \beta_{\bar{B}}(e) = 1 \text{ and } \beta_B(e) > 0\}| - |\{e \in E_c | \beta_A(e) > 0 \text{ and } \beta_{\bar{A}}(e) = 0\}| \quad (09)$$

(08) and (09) can be viewed as the cost functions of a traditional FM algorithm for the multi-way partitioning case; however, these expressions still do not account for the potential gain as mentioned previously. Sechen [7] has obtained a good evaluation of potential gains from a row-based VLSI layout model using a probabilistic analysis. In fact, the expressions of [7] can be viewed as a specific penalty function which provides an excellent criterion for establishing a sequence of moving operations for the free cells. Using this consideration, a novel penalty expression which has the same function of the probabilistic analysis method [7] is proposed. Its novelty is the applicability to multiple-way partitioning. The main principle of the proposed method is to introduce a control factor to adjust the free cells on any cut in a suitable sequence such that cuts can be eliminated as much as possible and balanced partitions can be found.

Let $G_P(e)$ denote the potential gain of net e imposed on its incident cells. The potential gain also consists of two parts which correspond to the two steps in a move operation, i.e. $G_P(e) = G^A{}_{Pget}(e) + G^B{}_{Prec}(e)$, where

$$G^A{}_{Pget} = 0 \text{ if } \beta_A(e) = |e| \quad (10a)$$

$$G^A{}_{Pget} = -P \text{ if } \beta_A(e) = \infty \quad (10b)$$

$$G^A{}_{Pget} = -P \times W(e) \frac{\alpha_A(e)}{|e|} \text{ if } \beta_A(e) < |e| \quad (10c)$$

and

$$G^B{}_{Prec}(e) = 0 \text{ if } \beta_B = (e) = |e| \quad (11a)$$

$$G^B{}_{Prec}(e) = P \text{ if } \beta_B(e) = \infty \quad (11b)$$

$$G^B{}_{Prec} = P \times W(e) \frac{\alpha_B(e)}{|e|} \text{ if } \beta_B(e) \; |e| \quad (11c)$$

and $W(e) = \frac{|e|}{max\_degree}$ if $|e| < max\_cell$; and $W(e) = 1$ otherwise.

P is the *penalty factor*, and max_degree is the (specified) *upper bound in the number of cells which a net connects*. When a cell is moved from A to $\bar{A}$, $G^A{}_{Pget}(e)$ can be viewed as the *resistance* (or penalty) imposed by net e on its incident free cells. The force of the resistance mainly depends on the status of net e in A. When all cells on net e are in A, i.e $\beta_A(e) = |e|$, then no resistance exists, because net e is not a cut (only a cut has such an effect on its incident cells). If there are cells on net e locked in A, i.e. $\beta_A(e) =$, then the greatest resistance is imposed by net e on its incident free cells, because net e is still a cut, unless all cells on it are moved into A. Hence, the resistance will be *dependent* to the ratio of α_A to $|e|$ as in (10c).

As it is more difficult to move all the cells on a long net into one set than for a short net, hence the resistance must also reflect the size of a net. This is specified by $W(e)$ (max_degree of $W(e)$ has generally a value of 8-9 for most standard CMOS circuits). The penalty factor P is given by *the average number of pins for the cells*, such that it may influence the cell move as the net cut gain. Based on similar considerations, $G^B{}_{Prec}(e)$ serves as the *attractive* force which draws the cells on net e into B (when moved from $\bar{B}$). Then, the total gain for the whole move operation can be expressed as

$$G(c) = G_{min-cut/span}(c) + \sum_{e \in A} G^A{}_{Pget}(e) + \sum_{e \in B} G^B{}_{Prec}(e) \quad (12)$$

A single cell move strategy assumes that after each move of a cell, the obtained partitions can possibly be the final result. So, the size constraints must be satisfied after each move. However, this limitation of the single cell move operation makes the algorithm to be very sensible to size constraints, especially when they are strict [11]. Also, this limitation severely confines the available solution space of the algorithm. This problem can be solved if such limitation is relaxed to allow several cells to be moved at the same time. During the move operation, a temporary violation of the size constraints can be allowed. However, after such a move, the size constraints are satisfied and fully restored.

For ease of implementation, we still move cells individually each time. A move operation involves exchanges of groups of cells and is referred to as a $macro - step$. The gain of the macro-step is equal to the sum of the gains of the cells involved. In each subset, cells are sorted according to their gain value using a novel bucket sorting technique (whose data structure is presented in the next section). The set of a moved cell before its move operation is referred to as the "From" set and after its move as the "To" set. The proposed algorithm begins with an initial partition and can be expressed as follows.

Algorithm 1:

Step 1: Input the set of free cells, $F = \{v_1,..,v_m\}$.
Store in subset[i] (i=1,..,k) the maximum gain value and current size of subset V_i.
Step 2:
while($|F|\neq 0$) do {
Begin a new macro-step
/* Select a Ccell (candidate cell) by ignoring the size constraints in its From and To sets */
(1) Sort subset[i] in decreasing order according to the gain value and select Ccell according to the highest gain from V_{From}, where $From$ is the smallest index in $i,..,k$ with $|V_{From-free}|\neq 0$
$V_{From-free}$ denotes the set of free cells in V_{From}.
/* to balance the To set of Ccell if the move of Ccell causes an unbalance in the To set*/
(2) Put Ccell into its To set
if Size(V_{To} + {Ccell}) > $((\frac{1}{k})W + \tau)$
then try to move the cells with the highest gain out of the To set until there is no violation of size constraints in the To set end
/* to balance the From set of Ccell if the move of Ccell causes an unbalance in the From set */
(3) if $(\frac{1}{k})W - \tau >$ Size(V_{From} - {Ccell})
then Sort subset[i] in decreasing order by its size and select a new Ccell from V_i, where i is the smallest index in 1,...,k, so that the To set of the new Ccell is the From set of the old Ccell.
Recursively repeat substeps (1), (2), (3) until there is no violation of the size constraints in the From set of the old Ccell.
end
/* stop a macro-step */
(4) Lock Ccell; record the current cut size and add Ccell to the current Macro-step }

In every macro-step, the first selected cell is always free from the size constraints, but following selections of cell(s) are determined by both size-balancing and objective-optimizing considerations (provided there exist violations of size constraints). As a temporary violation of size constraints is allowed during a macro-step, then the proposed approach (referred to as the MCM Balanced Partitioning, or MCMBP) can explore a larger solution space with the specified size constraints. Convergence of the MCMBP is as same as that of the FM algorithm because a cell can be moved only once per pass and the algorithm can always reach the condition of no free cells, hence it stops.

4 Data Structure and Time Complexity

As data structure, the bucket data structure used in [2] is adopted. However, this data structure is modified to account for an expansion due to the multi-way partitioning strategy.

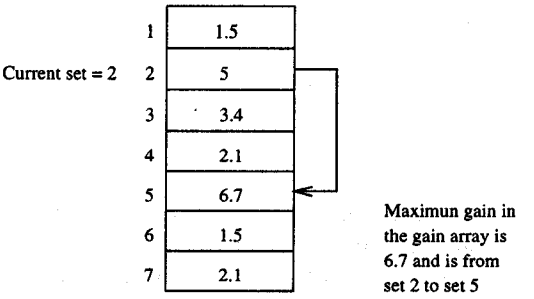

Figure 2. The Gain Array Structure

In the proposed algorithm, each subset maintains a sorted bucket. Since each free cell has k-1 move directions, so k-1 gains must be stored for a cell. In this paper, a gain array with k elements is employed to represent the gain of a cell. Each gain is stored in one element of the array (indicated by its subscript). The subscript is also the index number of the To set of a cell, except the one whose subscript is identical to the index of the cell current set (this is used as a pointer for the highest gain in the array). Figure (2) illustrates the gain array structure with k =7. An important advantage of this gain array structure is that when a free cell is selected for the move operation, then its move direction is determined at the same time.

To calculate the time complexity, assume that the maximum number of pins on a net and the maximum number of nets which a cell connects are bounded by a constant which is also significantly less than P. In the proposed algorithm, when a cell is moved, all gains of its adjacent cells will need to be adjusted. Let N denote the number of nets and $n(i)$ denote the number of cells on net e_i (i= 1 ,..., N), then the total number of adjustments for net i during a pass is

$$n(i) + (n(i) - 1) + ... + 1 = \frac{n(i)\ (n(i) - 1)}{2}$$

$$= O(\ (n(i))^2\ ) \qquad (13)$$

Hence, the total number of adjustments is given by

$$T\ =\ O(\sum_{i=1}^{N}\ (n(i))^2\ ) \qquad (14)$$

Let $n_{max} = \max\{n(1),..,n(N)\}$, then

$$\sum_{i=1}^{N}\ (n(i))^2\ \leq\ \sum_{i=1}^{N} {n_{max}}^2\ =\ N{n_{max}}^2 \qquad (15)$$

$$P = \sum_{i=1}^{N} n(i)\ \ =\ \ N \times \bar{n}(i) \qquad (16)$$

where

$$\bar{n}(i)\ = \frac{\sum_{i=1}^{N} n(i)}{N}$$

Hence,

$$O(\bar{n}(i)) = O(n_{max}) \quad (17)$$

and

$$T = O(Pn_{max}) \quad (18)$$

Note that as n_{max} is a constant (as according to a previous assumption), then T is dependent on P; however within a cell gain adjustment, k-1 gains must be recalculated. Hence, this process is $O(k)$. Using the MCMBP algorithm, each macro-step always begins by sorting k subsets. In the worse case, every macro-step corresponds to just one cell move. As sorting has a time complexity of $O(k\log_2(k))$, then $T = O(P \times k^2 \times \log_2(k))$ with respect to a single gain adjustment.

5 Experimental Results

The proposed algorithm has been implemented in a C package using a Sun-Sparc 20 workstation. The MCNC benchmark circuits with different sizes have been evaluated. Table (1) summarizes the characteristics of the benchmark circuits.

Tables (2) and (3) show the comparison results for the proposed MCMBP algorithm and previous algorithms (such as the KFM by [6], K-DualPART/DF by [12]) for k= 3 and 4. For comparison purpose with [12] the same conditions are used for both algorithms except for the reported cases. The deviation factor τ allows 10% offset from the balanced partitions for a fair comparison with the two other algorithms. However, as reported in [12], the size constraints can not be strictly satisfied in some benchmarks because the area of the largest cell is even more than the allowed partition are as. In these cases, MCMBP automatically relaxes the value of τ to a value to allow 10% more that the area of the largest cell. The results obtained in these case s are marked by an asterisk. These cases include Test02, Test04 and Test05 for k =3 and k = 4. So the values of τ allow 45%, 30% and 23% deviations in Table (2) (45%, 73% and 65% deviation in Table (3)). for these circuits.

From the results of Tables (2) and (3), it can be observed that a significant improvement is achieved by MCMBP over both the KFM [6] and K-DualPART/DF [12] for all the cases where the value of τ is strictly allowed a 10% deviation from the balanced partitions. The percentage improvement ranges from 59.9% to 29.6% for the 3-way and 4-way partitioning ca ses. This is a good indication of the effectiveness of the proposed potential gain functio n and the MCM method

6 Conclusions

This paper has proposed a polynomial time complexity algorithm for multi-way balanced VLSI network partitioning. The new algorithm extends the net cut model by taking into consideration the potential gain in moving a cell. A novel expression for the gain cost is introduced. A Multi-Cell-Move strategy is introduced to relax the single cell move limitation in the original group migration algorithm of [3]. This permits the new algorithm to explore a larger solution space under specified size constraints, thus obtaining better results, even under very strict size constraints. An heuristic implementation has been presented. Experiments show very good results as compared with other multi-way algorithms.

References

[1] A.U.Aho, J.E.Hopcroft and J.D Ullman, "The Design and Analysis of Computer Algorithms," Reading, MA: Addison-Wesley, 1976.

[2] C.M.Fiduccia and R.M.Mattheyses, "A Linear Time Heuristic for Improving Network Partitions," *Proc. 19th ACM/IEEE Design Automation Conference,* pp.175-181, 1982.

[3] B.W.Kernighan and S.Lin, "An Efficient Heuristic Procedure for Partitioning Graphs," *IEEE Transactions on Computers, Bell Syst. Tech. J.,* vol.49, pp.291-307, 1970.

[4] D.G.Schweikert and B.W.Kernighan, "A Proper Model for the Partitioning of Electric Circuits," *Proc. 9th Design Automation Workshop,* pp.57-62, 1979.

[5] B.Krishnamurthy, "An Improved Min-Cut Algorithm for Partitioning VLSI Networks," *IEEE Trans. on Computers,* vol. C33, no.5, pp.438-446, 1984.

[6] L.A.Sanchis, "Multiple-Way Network Partitioning," *IEEE Trans. on Computers,* vol.C38, no.1, pp.62-81, 1989.

[7] C.Sechen and D.Chen, "An Improved Objective Function for Mincut circuit partitioning," *Proc. IEEE ICCAD,* pp. 502-505, 1994.

[8] Y.C.Wei and C.K.Cheng, "Ratio Cut Partitioning for Hierarchical Design," *IEEE Trans. on Computer Aided Design,* vol.CAD10, no.7, pp.911-921, 1991.

[9] C.K.Cheng Y.C.A.Wei, "Am Improved Two-Way Partitioning Algorithm with Stable Performance," *IEEE Trans. on Computer Aided Design,* vol.10, no.12, pp.1502-1511, 1991.

[10] C.W.Yeh, T.Y. Lin and C.K.Cheng, "A General Purpose Multiple Way Partitioning Algorithm," *Proc. 28th ACM/IEEE Design Automation Conference,* pp.421-426, 1991.

[11] J. Cong ,L. Hagen, and A.Kahng, "Net Partiuons Yield Better Module Partitions," *Proc 29th IEEE/ACM Design Automation Conference,* pp.47-52, 1992.

[12] J. Cong, , W. Labio, and N. Shivakumar, "Multi-Way VLSI Circuit Partitioning Based On Dual Net Representation", *Proc. IEEE Int. Conf. on Computer Aided Design,* 1994.

Table 1.

Circuit	Numb. of Cells	Numb. of Nets	Numb. of Pins	Ave. net degree
Test02	1720	1663	6722	2.91
Test03	1618	1607	6527	2.79
Test04	1658	1515	6506	2.97
Test05	2750	2595	10814	3.10
Test06	1614	1752	7045	3.44
PrimGA1	902	833	3272	3.20
PrimGA2	3029	3014	12007	3.15

Table 2.

Benchmark	Best net cut			MCMBP impro over		MCMBP elapsed time seconds
	K-FM	K-DualPART/DF	MCMBP	K-FM	K-DualPART/DF	
Test02	114	81	91*	-	-	7.9
Test03	271	201	122	55.0	39.3	6.8
Test04	588	329	113*	-	-	2.7
Test05	1069	517	162*	-	-	15.6
Test06	281	252	101	64.1	59.9	7.6
PrimGA1	197	159	82	58.4	48.4	3.8
PrimGA2	704	578	417	42.2	38.6	22.8

Table 3.

Benchmark	Best net cut			MCMBP impro over		MCMBP elapsed time seconds
	K-FM	K-DualPART/DF	MCMBP	K-FM	K-DualPART/DF	
Test02	724	318	135*	-	-	13.1
Test03	368	298	189	51.0	34.6	10.2
Test04	843	514	121*	-	-	14.4
Test05	1233	863	131*	-	-	24.9
Test06	332	285	104	68.7	54.0	10.7
PrimGA1	203	187	132	35.0	29.4	6.1
PrimGA2	875	717	379	56.7	47.1	31.6