Trapezoidal Algorithm with Weighted Aggregation Scheme for floorplanning in VLSI

Amrutha K.P
VLSI Student
Amrita University
Ettimadai, Coimbatore, India

R.Sundararajan
Professor .Department of ECE
Amrita University
Ettimadai, Coimbatore, India

ABSTRACT
An important step in the automation of electronic design is the assignment of the physical components on the target semiconductor die. The major aim of floorplanning is to distribute the modules of a circuit onto a chip to optimize its area, wire length and timing. As the density of very large scale integrated (VLSI) circuits enhance, the need for faster floorplanning algorithms also grows. The goal of this work is to produce a fast method for developing wire-optimized floorplan. This uses trapezoidal algorithm for floorplanning. The generalized trapezoidal algorithm combines the major physical design steps in one algorithm. By the use of connectivity grouping, simple geometry, and a constrained brute-force approach, trapezoidal algorithm achieves lesser wire estimate than simulated annealing (SA) in orders of magnitude less time. It derives its advantages from the two important concepts of mathematics ie constraint brute-force-approach and divide-and-conquer approach. This generalised trapezoidal algorithm considers the softblocks.In addition; this algorithm implemented the multilevel partitioning with weighted aggregation scheme (WAG) scheme. The speed of this algorithm allows the designers to get a quick feedback about the design of larger circuits.

Keywords
Floorplanning, Trapezoidal algorithm, multilevel partitioning, weighted aggregation scheme

1. INTRODUCTION
Floorplanning is a major step in the physical design of VLSI circuits. A good layout is one, which occupies minimum area, uses short wires for interconnection and uses as few vias as possible. Layout design is complicated by different constraints. The possible constraints may be minimize area, minimize wirelength, minimize delays etc.There are different methods using for floorplanning [1].Three general classes are iterative approaches, constructive approaches and knowledge based approaches. Iterative approaches start with some initial solution and further steps it is refining this initial solution. Constructive approaches are starting with a seed module and in the following steps, another module is added to this. If a solution selected it will not change in the following steps. It will add to the final solution. In knowledge-based approaches, the following elements include(1) a knowledge set that contains the elements to which floorplan design specified (2) a set of rules for altering the data. (3) an inference engine for controlling the knowledge base.

Simulated annealing and genetic algorithms are coming under the iterative approaches. Constructive approaches include connectivity clustering; cluster growth etc .Routing, circuit mining etc is coming under the knowledge-based approaches.

2. WHY TRAPEZOIDAL ALGORITHM?
As the circuit integration increases, the requirement for high-speed floorplanning algorithms also increases. For the increased speed, the data structure that implements the algorithm should be very fast. However, all classical floorplanning algorithms will take longer run time [2]. In genetic algorithm, convergence is very slow and choosing the implementation of encoding and fitness function can be difficult. In simulated annealing global minimum guaranteed only when the temperature schedule has selected properly. This is very slow. In addition, the data structures using for the simulated annealing is very difficult. It will take more run time.Trapezoidal algorithm achieves lower wire estimate than simulated annealing.This facilitates the development of trapezoidal algorithm. The speed and quality of this algorithm is significant.

3. GENERALIZED TRAPEZOIDAL ALGORITHM
First phase of the trapezoidal algorithm consists of global connectivity phase and the second phase consists of local connectivity phase. Connectivity phase handles the wire optimization problem. The connectivity phase restricts the whole solution space in to a limited number of solutions. In the global connectivity phase, multilevel partitioning is used for partitioning the blocks. After the multilevel partitioning, the partitions are arranged linearly. Linear ordering of partitions also ensures lower wire and lower cost. For reducing, the length of wires trapezoidal algorithm binds highly connected pairs. Highly connected pairs are the blocks, which are connected by more number of interblock nets. Highly connected blocks will lock so that they may not rotate. The local connectivity again reduces wire estimate. The execution time for the wire optimization is minimum compared to other optimization algorithms [3].

3.1 Global Connectivity phase
A circuit is divided into a number of smaller subcircuits using circuit partitioning techniques. The partitioning should satisfy the given area, wire, or pin constraints. The aim of partitioning is to reduce the number of interconnections among the subcircuits, which has a direct effect on the final
chip performance in terms of area minimization as well as wire minimization.

3.1.1 Multilevel partitioning:
The usual graph partitioning algorithms are very slow; also, they produce poor quality solutions. The recursive bisection can use for

partitioning the graph in to k parts. But the complexity is very high ie.O (E) logk.In multilevel partitioning it is reduced to O(E) while coarsening performed[4].The multilevel partitioning algorithm consists of three stages. The three stages are coarsening, initial partitioning, and refinement.

In the coarsening phase, the graph is converted in to sequence of smaller graphs. The partition of the coarsest graph is calculated in the initial partitioning phase. And in the uncoarsening phase the initial partition is projected back to the original graph by going through intermediate partitions. Multiilevel partitioning scheme is shown in the figure1.

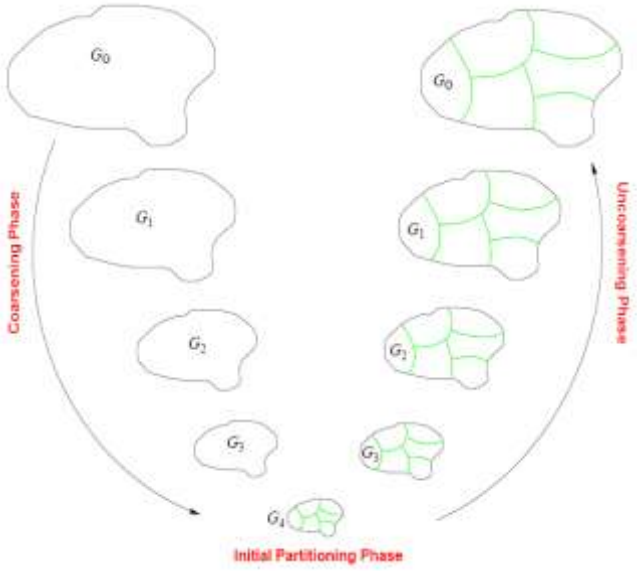

Fig.1.Multilevel partitioning

3.1.2 Coarsening phase: There are mainly two techniques using for coarsening (1) Strict Aggregation Scheme (SAG) (2) Weighted Aggregation Scheme (WAG)

Strict Aggregation Scheme (SAG):There are two different methods available for getting coarser graphs. The first method is based on calculating a random matching and merging the matched vertices into a multinode. The second method is based on creating multinodes that are highly connected .Thus, the edge between two vertices is merged and a multinode consisting of these two vertices is created. The SAG scheme is shown in the fig2. The aim of collapsing vertices using matchings is to decrease the size of the graph. The matching can be of four types. (1)Random matching (RM) (2) Heavy edge matching (HEM) (3) Light edge matching (LEM) (4) Heavy clique matching (HCM).

In random matching ,the vertices are visited in random order. For an unmatched vertex u, we arbitrarily choose one of its unmatched adjacent vertices. If such a vertex v exists, we incorporate the edge (u, v) in the matching and mark vertices u and v as being matched. If there is no unmatched adjacent vertex v exists, then vertex u left as unmatched in the random matching. For minimizing the edge-cut we use heavy edge matching. For a graph Gi = (Vi, Ei), a matching Mi that is used to coarsen Gi, and its coarser graph Gi+1 = (Vi+1, Ei+1) induced by Mi, then W (Ei+1) = W (Ei) -W (Mi)

In this scheme we match a vertex with another vertex such that the weight of the edge between these vertices is maximum compared to other incident edges. In Light edge matching (LEM), we maximize the total edge-weight of the coarser graph. The heavy clique matching method calculates the matching by merging vertices that has high edge density.

The coarsening preserves the two properties (1) the edge-cut of a partition in a coarser graph is equal to the edge-cut of the same partition in the finer graph; (2) a balanced partitioning of the coarser graphs leads to a balanced partitioning of the finer graph

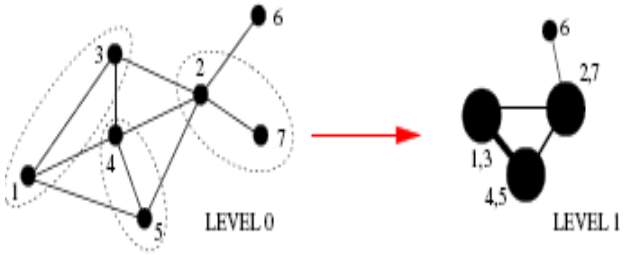

Fig.2.SAG scheme

3.1.3Weighted aggregation scheme (WAG):
In this scheme, each vertex can be divided into fractions. There are a number of aggregates forming. Different fractions of these vertices belong to different aggregates. The entire graph will be covered by small intersecting subsets of V.The nodes that belong to more than one subset will be divided among the corresponding coarse aggregates [4]. The creation of coarse graph is divided in to three stages. (1) A subset of the given nodes is selected as the seeds of the aggregates (2) The rules for interpolation are determined (3) The weights of the edges between the aggregates are calculated. WAG scheme is shown in fig3.

WAG coarse nodes: The creation of the set of seeds C selected and its complement F selected. Starting from C=∅ and F=V transfer nodes from F to C until all remaining $i \in F$ satisfy the equation 1.Where θ is .5

$$\sum_{j \epsilon C} w_{ij} \Big/ \sum_{j \epsilon V} w_{ji} \geq \theta \qquad (1)$$

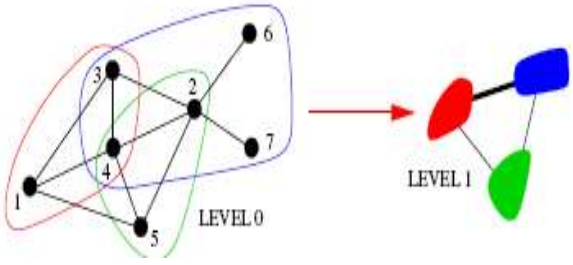

Fig.3.WAG scheme

WAG coarse nodes: Define for every element in F a coarse neighborhood Ni consisting of C-nodes to which i is associated. Let I (j) be the ordinal number in the coarse graph of the node that represents the aggregate around a seed whose ordinal number at the

fine level is j. The interpolation matrix P is defined by set of equations2

$$P_{iI(j)}=\begin{cases} w_{ij} \Big/ \sum_{k\epsilon N_i} w_{ik} & for\ i\epsilon F, j\epsilon N_i \\ 1 & for\ i\epsilon C, j=i \\ 0 & otherwise \end{cases} \quad (2)$$

$P_{iI(j)}$ thus represents the likelihood of i to belong to the I(j)th aggregate.

3.1.4 WAG coarse edges: Allocate the edge connecting two coarse aggregates p and q with the weight by the equation 3

$w_{pq} = \sum_{k\neq l} P_{kp} w_{kl} P_{lq}$ (3)

3.1.5 Initial partitioning phase:

The second stage of a multilevel algorithm computes high-quality partitions. Any recursive bisection can use for initial partitioning stage [5]

*3.1.6 Uncoarsening phase:*During the uncoarsening phase, the partition *Pm* of the coarser graph *Gm* is projected back to the original graph by going through the graphs *Gm*-1,*Gm*-2, …… *G*1.

3.3 Local connectivity phase

For reducing the wires this algorithm binds the highly connected blocks. We expand the shorter block to the height of the taller. And they will remain identical height in the floorplan.Because of these highly connected blocks close together they will contribute to less wire estimate. Locking down too many bits restricts the exploration of large design space.

4. RESULTS

The evaluations are done on an AMD Athlon64 processor 1.76GHz with 448 MB of RAM. The code of this algorithm is in C++.The benchmark circuits used for the experiments are GSRC, IBM and MCNC The Table 1 Shows the results net count and block count for various bench mark circuits.Table2 shows the number of partitions required for global partitioning.

And the Table3 shows the initial wire length estimate obtained by using semi perimeter method. Table4 shows the wirelength after local connectivity of the algorithm. Wirelength is reduced for all the circuits.

Table1
Net and block count for different benchmark circuits

Circuit	blocks	net count
n10	10	118
n30	30	349
n50	50	489
n100	100	885
n200	200	1585
n300	300	1893
ami	33	123
xerox	10	203
apte	9	97

Table2
Number of partitions for different benchmark circuits

circuit	Number of partitions
n10	No partitioning
n30	2
n50	3
n100	6
n200	9
n300	11
ami	2
xerox	No partitioning
apte	No partitioning

Table3
Initial wire-length estimation for different benchmark circuits

circuit	Wire length(in mm)
n10	20
n30	51.02
n50	121.01
n100	181.05

Table4
Wire-length after local connectivity

circuit	Wire-length(in mm)
n10	16.56
n30	48.34
n50	108.67
n100	161.05

Table 5 shows the wire-length after the second phase of the algorithm. Wire-length is again reduced after this global connectivity. Table6 tabulates the results after both optimizations. The wire-length for larger circuits has tremendously decreased.

Table 5
Wire-length after global connectivity

circuit	Wire- length(in mm)
n10	16
n30	40.03
n50	80.23
n100	123.45

Table 6
Wire-length after both connectivity in generalized trapezoidal algorithm

circuit	Wire-length(in mm)
n10	13.8
n30	35.5
n50	84.00
n100	118.76

Table 7
Comparison of wire-length (in mm)
with trapezoidal algorithm and simulated annealing

circuit	WL(Initial)	WL(GTA)	WL(TA)	WL(SA)
n10	20	13.8	14	17
n30	51.02	35.5	37	40
n50	121.01	84.00	85	98
n100	181.05	118.76	122	130

Table8
Comparison of CPU runtime (in seconds) with trapezoidal and simulated annealing algorithm

circuit	cpu(GTA)	cpu(TA)	cpu(SA)
n10	0.0	0.0	0
n30	0.8	1.0	3
n50	1.0	1.0	10
n100	2.2	2.0	40

Table7 shows the comparison of generalized trapezoidal algorithm with trapezoidal and simulated annealing algorithm. The generalized algorithm achieves lower wire estimate than other two algorithms.Table8 gives the comparison of runtime with trapezoidal algorithm and simulated annealing.

5. CONCLUSION

The net count and block count for different benchmark circuits are found. And the numbers of partitions, wirelength estimates are determined. From the results, it can be concluded that the generalized trapezoidal algorithm achieves lower wire estimate than simulated annealing and trapezoidal algorithm. From the runtime comparison, the runtime for smaller circuit has decreased. And for the larger circuits there is a slight increase in runtime but there is a tremendous decrease in wirelength compared totrapezoidal.The runtime for all the circuits is less compared to simulated annealing.For wire length, semiperimeter method is used. This generalised trapezoidal algorithm is implemented using softblocks.The softblocks are the blocks, which can change shape, and direction.

These softblocks play a major role in the design of VLSI circuit.This algorithm has all the advantages of multilevel partitioning.To improve the efficiency of multilevel partitioning, coarsening scheme implemented with weighted aggregation scheme. This WAG scheme stops itself from making the local decision before gathering global information .It gives lower wire estimate than other algorithms. The constraint brute-force approach of the algorithm increases its speed.

6.REFERNCES

[1] Sadiq M. Sait and Habib Youssef, "VLSI Physical Design Automation-Theory and Practice", IEEE Press,Vol 3,1995.pp90-92

[2] S.Kirkpatrick,C.D.Gelatt,Jr,M.P.Vecchi"Optimizationby SimulatedAnnealing"science, vol220, No.4598, May1983, pp671-680

[3] Peter G. Sassone, *Student Member, IEEE*, and Sung Kyu Lim, "Traffic: A Novel Geometric Algorithm for Fast Wire-Optimized Floorplanning", vol. 25, No. 6, June 2006

[4] C´edric Chevalier_ Ilya Safro†,"Weighted aggregation for multi-level graph partitioning", Dagstuhl Seminar Proceedings 09061, Nov2008

[5] George karypis and vipin kumar," A fast and high quality multilevel scheme for partiioning irregular graphs" SIAMJ. SCI.COMPUT.,vol.20,No.1.1998.pp.359-392

[6] J. Cong and S. K. Lim, "Edge separability based circuit clustering with application to multi-level circuit partitioning," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 23, no. 3, pp. 346–357, Mar. 2004.

[7] L. Stockmeyer, "Optimal orientation of cells in slicing floorplan designs,"Inf. Control, vol. 57, no. 2, pp. 91–101, 1983.

[8] T. Hamada, C. K. Cheng, and P. M. Chau, "A wire length estimation technique utilizing neighborhood density equations," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 15, no. 8, pp. 912–922, Aug. 1996.