Partitioning by Probability Condensation

John Blanks
Mentor Graphics Corporation

A partitioning model is formulated in which components are assigned probabilities of being placed in bins separated by partitions. The expected number of nets crossing partitions is a quadratic function of these probabilities. Minimization of this expected value forces condensation of the probabilities into a "definite" state representing a very good partitioning. The bipartitioning case is treated explicitly.

Partitioning and placement are plagued by the problem of local minima. This is true in spite of the development of good heuristics [KER70] for the partitioning problem. This difficulty has prompted the development of such techniques as simulated annealing which exploit some "uncertainty" at the beginning of the placement process.

These methods are very powerful, being known to tend to global optimality asymptotically [ROM84). But their demand upon resources is vast, and this consideration prompted the author to wonder if one could explicitly imbed this uncertainty requirement in the problem statement.

Consider the problem of partitioning N cells into a number M of bins, so as to minimize nets crossing the partitions. One might characterize the partitioning at any phase of the process by a set of probabilities, for each cell, of being in any bin. Then it is possible to write down the expected number of crossings over the partitions. The probability distribution now carries the uncertainty of the partitioning, and one hopes that these probabilities may be intelligently forced to the necessary 0's and 1's which characterize a real partitioning. This represents a "condensation" in probability space.

Consider the special case of bipartitioning with no fixed cells.

Let p_i be the probability of cell i being on the "left" side of the partition, and so $1\text{-}p_i$ will be the probability of it being on the right. In addition, let C_{ij} be the number of connections between cell i and cell j. In the case of 2-node nets, C_{ij} is simply the number of nets connecting cell i and cell j. When nets have more than 2 nodes, the nets may be approximately decomposed as a collection of 2-node nets which correspond to the edges of the complete graph constructed on the net's nodes with altered connectivities. Then the number of crossings due to i and j will be zero if i and j are on the same side, and C_{ij} if they are on different sides. The expected number of crossings due to the ij pair will be

(1) $p_ip_j(0) + (1\text{-}p_i)p_j(C_{ij}) + p_i(1\text{-}p_j)(C_{ij}) + (1\text{-}p_i)\ (1\text{-}p_j)\ (0)$
$= C_{ij}\,[p_i + p_j\text{-}2p_ip_j]$

Thus the total expected number of crossings due to all cells will be

$$\Gamma = \tfrac{1}{2} \sum_{i,j}^{N} C_{ij}\,[p_i + p_j\text{-}2p_ip_j]$$

The factor of ½ comes from counting pairs twice. Now make the substitution

$$p_i = \tfrac{1}{2} + \tfrac{1}{2}\,\varepsilon_i$$

Since $p_i \in [0,1]$ then $\varepsilon_i \in [-1,1]$. Then (1) may be written:

$$\Gamma = \tfrac{1}{2} \sum_{i,j}^{N} C_{ij}\,(\tfrac{1}{2}\text{-}\tfrac{1}{2}\,\varepsilon_i\varepsilon_j)$$

Thus minimizing Γ amounts to maximizing

$$\Gamma' = \sum_{i,j}^{N} C_{ij}\,\varepsilon_i\varepsilon_j$$

The constraint of finishing with N/2 cells on each side of the partition can be imposed by requiring the expected number of cells on the left be N/2 at each phase of the partitioning:

$$N/2 = \sum_{i}^{N} p_i = \sum_{i}^{N} (\tfrac{1}{2} + \tfrac{1}{2}\varepsilon_i) = N/2 + \tfrac{1}{2} \sum_{i}^{N} \varepsilon_i \text{ or } \sum_{i}^{N} \varepsilon_i = 0$$

This is not, however, the most difficult constraint. That is the requirement that $p_i \in [0,1]$ or $\varepsilon_i \in [-1,1]$. This is addressed in the next section.

Geometric ideas in partitioning

The problem is, maximize $\Gamma' = \sum_{i,j}^{N} C_{ij}\varepsilon_i\varepsilon_j$ subject to $\varepsilon_i \in [-1,1]$.

This suggests figure 1 where a three-dimensional version of the N dimensional space is portrayed. A point in the N-space represents the entire state of the partitioning at some time. The concentric ellipsoids represent surfaces of constant $\Gamma' = \Sigma\, C_{ij}\varepsilon_i\varepsilon_j$ with values increasing with expanding ellipsoids. (Summations are from 1 to N if not explicitly stated otherwise). The vertices of the N-cube represent "certain" partitionings, since they have all ε_i as either 1 or -1, corresponding to the p_i being either 1 or 0. The subset of vertices which satisfy $\Sigma\varepsilon_i = 0$ will be the "legal" partitionings, since that constraint corresponds to $\Sigma p_i = N/2$.

Figure 1 is subtly misleading, because the ellipsoids correspond to positive eigenvalues of $[C_{ij}]$, whereas $[C_{ij}]$ may have negative eigenvalues. The correct figure will then be an N-dimensional hyperboloid, as shown in figure 2. The reader is advised to consider both figures in the discussion which follows.

The space in which the optimization takes place, the N-cube, is convex. This suggests gradient optimization on the quadratic objective fuction [BER82], [WAL79]. Thus in figure 3, one starts

at a point A and follows the gradient outward (in the direction of increasing Γ'). When a constraint is encountered (a face of the N-cube), a new gradient is calculated, and its projection along the face of the cube is followed for the next iteration (point B, figure 3). To calculate the gradient (g_i), simply use

$$g_i = \frac{\partial \Gamma'}{\partial \varepsilon_i} = 2 \sum_j C_{ij} \varepsilon_j$$

This will not give the global maximum, since the quadratic objective function discussed here is not convex. Thus, if the process is started at a different point (C), the final vertex reached may be different. However, it does appear that the process will terminate with the partitioning state at some vertex. Provided the $\Sigma \varepsilon_i = 0$ constraint has been adhered to, this will be an acceptable partitioning.

But one can actually do much better. Since the starting point is variable, it can be chosen in a way which enhances the gradient search. Consider figure 4, where the N-cube has been replaced with an N-sphere. Then the constraints are that $\Sigma \varepsilon_i^2 = R_0^2$, where R_0 is the radius of the N-sphere, so the maximum of $\Sigma C_{ij} \varepsilon_i \varepsilon_j$ occurs along the eigenvector corresponding to the largest eigenvalue λ_N of $[C]$, with value $\lambda_N R_0^2$. This is shown in [NOB69], for instance. It is assumed here that [C] is nonsingular. A connection between eigenvectors and partitioning is shown in [DON88] but without the use of a probabilistic scheme.

Thus, a strategy emerges. Find the largest-eigenvalued eigenvector, probe out from the origin in this direction, and find where this ray intersects the constraint cube. Then reevaluate the gradient from this point, project along the face of the cube, follow until another constraint surface is reached, repeat, etc. This constraint method will terminate at an N-cube vertex, which defines the required partitioning.

Unfortunately, the eigenvector found will not obey the constraint $\Sigma \varepsilon_o = 0$. But it is possible to find a maximizing "ray" which is "close" to this eigenvector, and which does maximize the form $\Sigma C_{ij} \varepsilon_i \varepsilon_j$ over the surface of the sphere while satisfying $\Sigma \varepsilon_i = 0$. Space does not permit a proof of this, but the construction of this maximizing ray will be explained in the implementation section which follows.

Implementation Issues

Two implementation issues deserve special discussion.

A. Solving for the maximizing ray.

This is accomplished by a variation of the power method [FAD63]. In this method, a random vector is chosen, and it is repeatedly multiplied by the matrix $[C_{ij}]$. This will asymptotically yield the largest-magnitude eigenvalue and its associated eigenvector. If that eigenvalue happens to be negative, the power method can still be used by "shifting" the eigenvalue spectrum to make all eigenvalues positive.

A modification is necessary to the classic power method. The constraint $\Sigma \varepsilon_i = 0$ requires the ε_i vector should be perpendicular to the vector $1^T = (1,1,1,\ldots 1)$. So after every multiplication, the 1 component of the e vector can be subtracted from itself. This method will generally construct the vector which minimizes $\Sigma C_{ij} \varepsilon_i \varepsilon_j$ subject to the $\Sigma \varepsilon_i = 0$ constraint. The vector so constructed will in general *not* be an eigenvector of C. For a demonstration, see [BLA85].

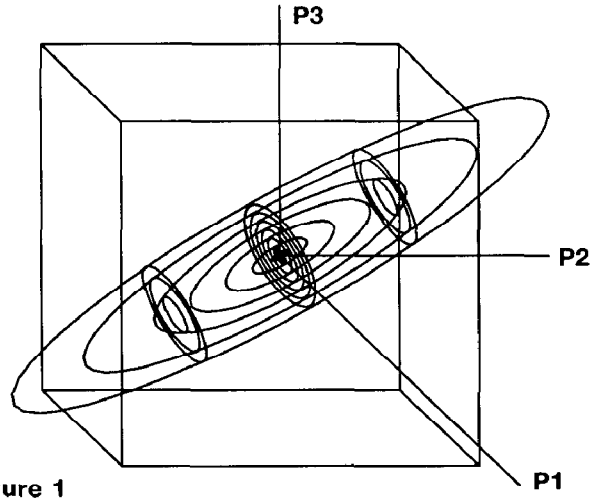

Figure 1

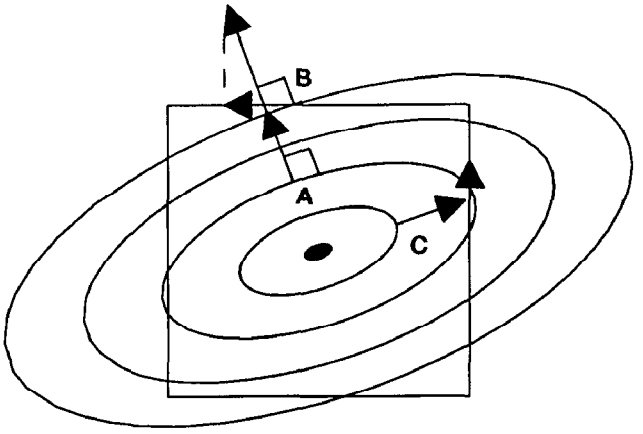

Figure 3

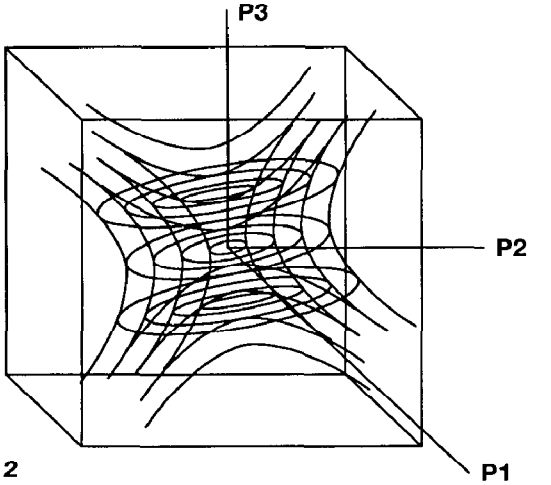

Figure 2

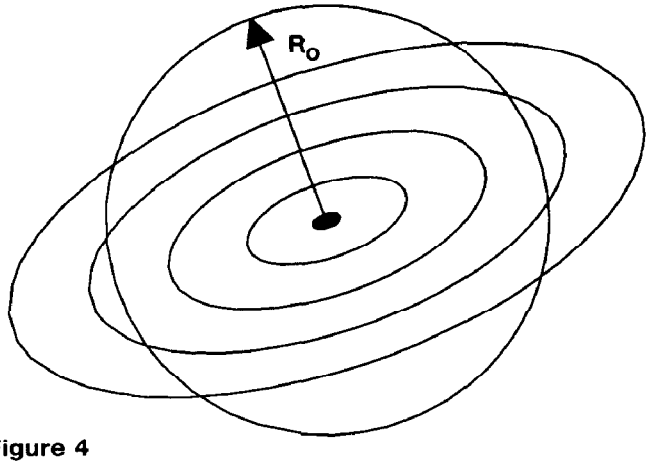

Figure 4

B. The constrained gradient method.

There are some subtleties here. First, it is useful to consider the causes of suboptimality in the gradient search method outlined above.

(1) the function Γ' is not convex, so no global optimum can be claimed. This has already been discussed.

(2) the gradient should really be followed only for infinitesimal distances in the partitioning space. However, the gradient is expensive to calculate, and with 2N constraint surfaces, each gradient projection will only be followed a short distance anyway before another face of the N-cube is hit.

(3) This is a problem of constrained vs. free gradients. The gradient may not always point to the "outside" of the N-cube from a face. It certainly will when that constraint is invoked, but it need not during later phases of the partitioning (because the objective function is not linear.) Thus, one might argue that the gradient should be tested at each iteration on the constraints which are "active", to see if they can be relaxed. However, this ignores the $\Sigma\varepsilon_i = 0$ constraint. It is very difficult to satisfy this while simultaneously shifting "face" constraints. The best solution found was to keep a "face" constraint in place permanently after being invoked. Once the solution point hits a face of the N-cube, it is "stuck" there. Future research may treat this problem differently. With this last simplification, the implementation of the constrained gradient method is clear. At each iteration, there will be a set of N' free g_i and a set of constrained g_i. If these constraints are fixed, then the $\Sigma g_i = 0$ constraint must be absorbed by the remaining free g_i. Thus, for the projection of the gradient onto the allowable constraint face,

$$g'_i\,(\text{free}) = g_i\,(\text{free}) - \frac{\Sigma g_i\,(\text{free})}{N'} \text{ and } g_i'\,(\text{fixed}) = 0$$

This clearly satisfies the constraints.

Experimental Results

The experimental results of bipartitioning on four test cases of various sizes are shown in Table 1. These are actual circuits although bipartitioning is not necessarily indicated as a layout step for them.

Some explanation is necessary here.

(1) The first score column shows the full-blown scheme as outlined above: eigenvector calculation, followed by gradient ascent to the solution vertex.

(2) The second score column gives the score result after a sequence of N^2 trial pairwise interchanges followed each case. In only one case out of four was the pairwise exchange able to improve the score by even 1%. The failure of the pairwise interchange phase provides the strongest support of probabilistic partitioning.

(3) Next, the effectiveness of using the eigenvector as a starting direction was tested. A random vector on the surface of the N-sphere was used to start the partitioning. Column 3 gives the score resulting from that (random) partitioning. This score is bad, as expected. Next, column 4 shows this random partitioning followed by the pairwise interchange sequence. In every case, this is still much worse than "ordered" partitioning. Column 5 shows the scores resulting from a random vector followed by gradient ascent, and column 6 illustrates a random vector followed by gradient ascent followed by pairwise exchange; this seems roughly comparable to column 4, so there seems little reason to perform gradient ascent without the benefit of the eigenvector starting direction.

(4) Finally, the effectiveness of using gradient ascent was tested. Column 7 shows the eigenvector solution followed by pairwise exchange, with gradient ascent excluded from the job stream. In case #1 and #2 this process worked almost as well as the full-blown probabilistic partitioning, but not so in the larger cases.

The conclusion in the case of bipartitioning seems to be that the sequence of eigenvector calculation followed by gradient ascent works better than any other subset of that method, even when augmented by massive pairwise exchange.

Complexity

N iterations are needed (one for each constraint) and each one requires a gradient calculation, which is 0(NB) where B is the number of cells to which an average cell has connections. So the complexity of the algorithm is $O(N^2B)$. The actual CPU times for the tests are shown in Table 1. Tests were run on an Apollo DN3000.

For the above, the gradient is recalculated after each face is hit. But it might be adequate to calculate the gradient after every N/100 faces had been hit, for instance. Thus only 100 gradients would have to be calculated, suggesting the possibility of a linear-time partitioner. This idea will be examined later.

Extensions

The probabilistic partitioning algorithm discussed here is obviously restricted in its applications. Several extensions are being investigated:

A. Partitioning into >2 bins, with different required "weights" in the partitions:

Each cell i can be assigned a probability p_{ij} of being in bin j, and the required partitioning objective function can be shown to be quadratic in the (p_{ij}). The difficulties of this extension are:

(1) The time and space required for storing and manipulating the variables (p_{ij}).

(2) Now the constraints are more difficult. For the bipartitioning case, the assignment of each cell was dictated by one variable ε_i varying between -1 and 1. For M bins, M-1 variables are required, with coupled, complicated constraint equations.

In spite of these difficulties, the success of the bipartitioning case suggests the extension can be made to work.

B. Handling "fixed" cells (ie, cells permanently assigned to a given bin):

This causes a linear term in addition to the quadratic portion of the objective function. Methods such as those used in [BLA85] are expected to solve this problem.

C. Nets with >2 nodes.

While many -node nets can be approximated by the techniques discussed earlier, it would be preferable to have an explicit technique for handling them.

Table 1 **Partitioning Scores**

Case	# Devices	CPU Seconds	(1) Eigenvector and Gradient Ascent	(2) Eigenvector & Gradient & Pairwise Exchange	(3) Random	(4) Random & Pairwise Exchange	(5) Random & Gradient	(6) Random & Gradient & Pairwise	(7) Eigenvector & Pairwise Exchange
1	32	37	31	31	106	44	58	48	32
2	509	1572	299	295	1458	390	421	322	302
3	788	3661	354	354	2833	481	650	454	508
4	1491	11528	639	636	5291	1247	1994	1417	714

The effect of these nets can be included in the objective functions in terms which are cubic, quartic, or of higher order. It may be that perturbation theory can be used here.

Conclusion

The probabilistic partitioning scheme developed here appears quite promising. Two reasons may be hypothesized for its success:

(1) The formulation of the problem in continuous variables (probabilities) rather than discrete or binary variables appear to lend flexibility and fluidity to the convergence process, and

(2) The objective function obtained in this formulation is quadratic, and thus is well-behaved.

Generalization of the technique to multipartitioning (and other extensions) is certainly not straightforward. But the initial success argues the algorithm can be made to work well for those problems.

References:

[BER82] Bertsekas, Dimitri; *Constrained Optimization and Lagrange Multiplier Methods,* Academic Press, Inc., New York, New York, Copyright 1982.

[BLA85] Blanks, John; *Use of a Quadratic Objective Function for the Placement Problem in VLSI Design,* Dissertation, University of Texas at Austin, Copyright April 1985.

[DON88] Donath, W.E., "Logic Partitioning", p. 65, *Physical Design Automation of VLSI Systems,* Benjamin/Cummings Publishing Company, Inc., 1988.

[FAD63] Faddeev, D.K. and Faddeeva, V.N.; *Computational Methods of Linear Algebra,* W.H. Freeman and Company, San Francisco and London, 1963.

[KER70] Kernighan, B.W. and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs", Bell System Tech, Journal, Vol. 49, February 1970, 291-307.

[NOB69] Noble, Ben; *Applied Linear Algebra,* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1969.

[ROM84] Romeo; Vincentelli; and Sechen; "Research on Simulated Annealing at Berkeley", Proceedings of the ICCD-IEEE – October, 1984, pp. 652-657.

[WAL79] Walsh, G.R.; *Methods of Optimization,* John Wiley and Sons Ltd., Copyright 1975, Reprinted December 1977, Revised reprint December 1979.