The Constrained Via Minimization Problem for PCB and VLSI Design

Xiao-Ming Xiong and Ernest S. Kuh

Electronics Research Laboratory
Department of EECS
University of California, Berkeley
Berkeley, California 94720

ABSTRACT

A new via minimization approach is presented for two layer routing of printed circuit boards and VLSI chips. We have analyzed and characterized different aspects of the problem and have derived an equivalent graph model for the problem from the linear programming formulation. Based on the analysis of our unified formulation, we posed a practical heuristic algorithm. The algorithm can handle both grid-based and gridless routing. Also, an arbitrary number of wires is allowed to intersect at a via and we allow both Manhattan and knock-knee routings.

1. Introduction

Multilayer routing is essential in the present and future design of VLSI and PCBs (Printed Circuit Boards). The problem of layer assignment and its related problem of via minimization are addressed in this paper. Although physical constraints on layer may require special performance considerations in layer assignment, the primary criterion to layer assignment in this paper is via minimization. There are several reasons for this. First, vias increase manufacturing cost; second, vias cause problem in reliability; and third, the physical dimension of vias is usually larger than the width of connecting wires, thus minimizing vias increases space usage.

In this paper, we consider only the two-layer problem and we assume that terminals can be accessed from either layer. The assumption is made to simplify our discussion only and we will see in section 5 that it can be released by constraining the wire connecting the terminal on a specific layer. We will treat the practical problem of constrained via minimization from a specified routing geometry only. We formulate the problem as a {0, 1} integer programming problem and propose a simple heuristic implementation. We allow multi-terminal nets and multiple wires to intersect at any via. During the via minimization process, we consider only Manhattan routing style but include knock-knee routings.

The constrained via minimization problem originated in the pioneer PCB design work of Hashimoto and Stevens ([1]) in 1971. In 1980, Kajitani ([2]) proposed a polynomial-time algorithm for a special case of the problem. Ciesielski and Kinnen ([3]) introduced a integer programming formulation to the problem with a solution which is exponential in time complexity. Chan, Kajitani and Chan ([4], [5] and [6]) extended Kajitani's earlier work to a more general, but still restricted situation and proposed an optimal solution. Independently, Pinter ([7]) found a polynomial-time algorithm for the same situation where each via is to connect at most three wires. As far as we known the algorithms proposed in [6] and [7] were not implemented. In 1986, Du and Chang ([8]) proposed another heuristic algorithm for via minimization based on bi-partitioning of a graph.

In this paper, we first discuss the planar representation of a two-layer routing in section 2. We formulate the constrained via minimization (CVM) problem in section 3. Based on the formulation, the necessary and sufficient conditions for layer assignment for two-layer routing is discussed in section 4. The algorithm implementation is given in section 5. Experimental results are illustrated in section 6. In section 7, we discuss some generalization and practical implications.

2. The Planar Representation of Two-Layer Routing

A well-known scheme in two-layer routing is to have one layer strictly used for horizontal wires and the other for vertical wires. Interconnections between the two layers are made through vias wherever necessary. A simple example is shown in Fig. 1a where dotted vertical lines represent wire segments on one layer and solid horizontal lines represent wire segments on the other. Vias designated by circles are inserted to make the necessary interconnections. It is well-known that such a V-H routing scheme can guarantee 100% routing completion provided that there are no restrictions on space and tracks. The disadvantage of V-H routing is of cause the large number of vias and the possibility of excess space required. In Fig. 1b we show an alternative routing of the same example which uses only one via in comparison with seven required in the V-H routing shown in Fig. 1a. Our problem is to determine, in general, how via minimization in two-layer routing can be achieved.

For the present example, it should be noted that both routings lead to the shortest total wire length, and they have the same planar projection. This suggests that we may consider the input to our constrained via minimization problem in the form of a planar representation of a proposed two-layer routing. The planar representation of the same example is given in Fig. 1c. We define a *cross point* as the intersection of two nets in such a representation. In the present example, there are four cross points marked by crosses in the figure. Next we introduce the con-

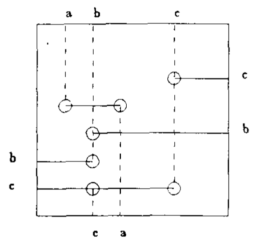

Fig. 1a

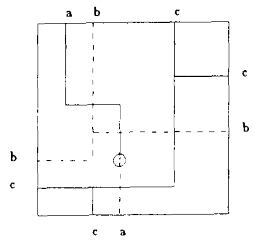

Fig. 1b

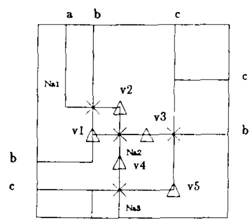

Fig. 1c

This research is sponsored by the Semiconductor Research Cooperation under SRC 82-11-008.

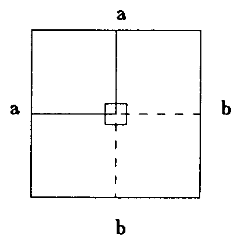

Fig. 2a

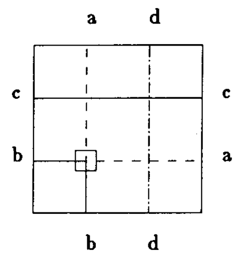

Fig. 2b

cept of a *via candidate*. It is defined as a location where a via might be introduced. Obviously, a via candidate must be located on a net; thus it cannot be at a cross point where two nets meet. Since we assume that terminals can be accessed from either layer, there is no need to consider via candidate on any part of a net which originates from a terminal and leads to only a single cross point. Thus for the present example, there are five possible via locations v_1, v_2, v_3, v_4, and v_5 marked as triangles.

In order to precisely formulate the via minimization problem, we need to introduce the definition of a ***net segment*** as part of a net which spans from a via candidate to either another via candidate or a terminal. For the present example Net a has three net segments, designated as N_{a1}, N_{a2} and N_{a3}. Notice N_{a1} starts from the top terminal a to via candidate v_2, N_{a2} is from v_2 to v_4 and N_{a3} is from v_4 to the bottom terminal a. We define the ***degree*** of a via candidate as the number of net segments intersect at the via candidates. Thus the degree of v_1 is three while the degree of the other via candidates is two.

3. Problem Formulation of CVM

Given the planar representation of a geometric routing, we first determine the cross points and all possible via candidates. We state the following Lemma.

Lemma 3.1:

A layer assignment is feasible if and only if all the intersecting net segments of different nets are assigned to different layers.

Proof:

Necessity: If any two net segments which belong to different nets and intersect are assigned to the same layer, the two nets will short-circuit at the intersecting point. Thus the layer assignment is illegal.

Sufficiency: If all the intersecting net segments of different nets are assigned to different layers, then no short-circuit will occur. We can use vias to connect all the net segments which belong to the same net but assigned to different layers. So there is no open-circuit in the layout; hence the layer assignment is feasible.

Q.E.D.

Corollary 3.1:

A layer assignment is feasible if and only if the net segments attached to the same cross point are assigned to different layers.

Now to each net segment, we assign a {0, 1} variable x_{nl}, where the first subscript n designates the net n and the second subscript l is the label of the net segment. The value of x_{nl} determines the layer (either 0 or 1) on which the net segment will be placed. By corollary 3.1, we have a constraint for every cross point as follows:

$$x_{ni} + x_{mj} = 1, \qquad n \neq m \qquad (1)$$

We can directly obtain the following corollary from corollary 3.1.

Corollary 3.2:

A layer assignment is feasible if and only if the linear equations in (1) hold.

Notice that there are exactly two net segments intersecting at a cross point. Therefore, for each constraint, exactly two variables will be involved, and there are altogether t constraint equations, where t is the number of cross points.

Ultimately, a via candidate may or may not be a via. If all the net segments attached to a via candidate are assigned on the same layer, the via candidate will not be a via. On the other hand, if there is at least one net segment at a via candidate which is assigned to a different layer, a via will result. Thus, if we consider a pair of net segments at a via candidate, the existence of the non-zero term

$$| x_{ni} - x_{nj} |, \qquad i \neq j \qquad (2)$$

signifies that a via is present. If all via candidates are of degree two, then the sum of such terms in (2) over all via candidates will constitute our objective function. It is to be minimized subject to the t constraints in (1) over all cross points.

In case of via candidates with degree k > 2, we need to introduce a weighting function to properly weight the contribution of the term in (2) due to a pair of net segments as

$$W(k) \times | x_{ni} - x_{nj} |, \qquad i \neq j \qquad (3)$$

Since for a k-degree via candidate, there are $\frac{k(k-1)}{2}$ such terms, a proper choice of the weighting function is $\frac{2}{k(k-1)}$. To summarize, our via minimization problem is reduced to the following {0, 1} integer programming problem:

Minimize the function f

$$f = \sum_{\text{all via candidates}} W(k) \times | x_{ni} - x_{nj} |, \qquad i \neq j \qquad (4)$$

Subject to the constraints

$$x_{ni} + x_{mj} = 1, \qquad n \neq m \qquad \text{for all cross points} \qquad (5)$$

Knock-Knee Connection

A particular two-layer connection pattern which is not included in our discussion so far is the knock-knee connection shown in Fig. 2a. Note that N_{na} and N_{nb} are placed at two different layers but meet at the knock-knee point marked by a square. Knock-knee connection introduces further flexibility in Manhattan routing and is of theoretical interest. For example, it is easy to see that a 100% routing cannot be always achieved with only two layers if knock-knee connection is allowed. Such an example is shown in Fig. 2b.

For our present discussion of two-layer routing, there is no difficulty in including knock-knee points in our formulation for two-layer routing. We can treat a knock-knee point exactly the same as a

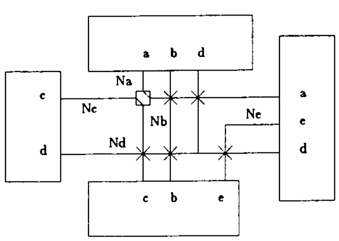

Fig. 3a

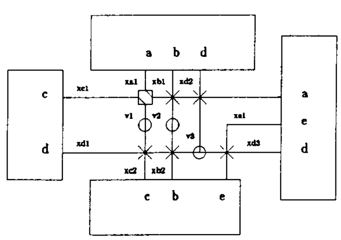

Fig. 3b

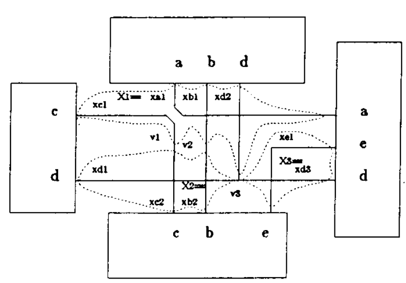

Fig. 3c

cross point. Thus we also use the term *intersecting point* in the planar representation to designate either a cross point or a lock-knee point.

EXAMPLE

Consider the example of Fig. 3a where there are four two-terminal nets, N_a, N_b, N_c and N_e and one 3-terminal net, N_d. There are five cross points and one lock-knee point. In Fig. 3b, we show the three via candidates v_1, v_2, and v_3 and denote the net segments by assigning boolean variables. The six constraint equations for the six intersecting points are:

$$\begin{aligned} x_{a1} + x_{c1} &= 1 \\ x_{a1} + x_{b1} &= 1 \\ x_{a1} + x_{d2} &= 1 \\ x_{c2} + x_{d1} &= 1 \\ x_{b2} + x_{d1} &= 1 \\ x_{d3} + x_{e1} &= 1 \end{aligned} \qquad (6)$$

For the via candidate v_3, the contributing terms in the objective function are

$$\frac{1}{3} \times | x_{a1} - x_{d3} | + \frac{1}{3} \times | x_{d3} - x_{d1} | + \frac{1}{3} \times | x_{d1} - x_{d2} |$$

The other two terms for via candidates v_1 and v_2 are

$1 \times | x_{b1} - x_{b2} |$ and $1 \times | x_{c1} - x_{c2} |$

Thus the objective function which we wish to minimize is

$$\begin{aligned} f = \; & \frac{1}{3} \times | x_{a1} - x_{d3} | + \frac{1}{3} \times | x_{d3} - x_{d1} | \\ & + \frac{1}{3} \times | x_{d1} - x_{d2} | + 1 \times | x_{b1} - x_{b2} | \\ & + 1 \times | x_{c1} - x_{c2} | \end{aligned} \qquad (7)$$

Unfortunately, the function is not linear, and the solution is not easily found. In section 5, we will convert this to a linear integer programming problem and propose a graph theoretical solution.

4. Existence of a Feasible Solution and the Constraint Graph

As mentioned, the objective function (4) is not a linear function because of the presence of absolute value sign. Thus the solution, in general, is exceedingly difficult to obtain. Fortunately, it is possible to transform the problem into one which has both a linear objective function and linear constraint equations.

First let us take a look at the constraint equations in (1). There are exactly t constraint equations where t represents the number of intersecting points. For each constraint, exactly two variables representing two different net segments, are involved. Furthermore, the values of the two variables in the constraint are always complementary because the two net segments must be assigned to two different layers. We introduce the definition of a *cluster* as a subset of constraints, in which every constraint shares at least one variable with at least another constraint in the subset. A *maximal cluster* is then a cluster which shares no variables with the constraints outside the cluster. In the example of Fig. 3b which is redrawn in Fig. 3c, there are three maximal clusters { N_{a1}, N_{c1}, N_{b1}, N_{d2} }, { N_{c2}, N_{d1}, N_{b2} }, and { N_{d3}, N_{e1} }. Here, we also use the terms cluster and maximal cluster to refer the subset of variables.

We say that a cluster is *feasible* if and only if all constraints in the cluster can be satisfied. That is, the values of all variables in the cluster can be determined. The following theorem gives the necessary and sufficient conditions for layer assignment for two-layer routings:

Theorem 4.1:

The via minimization problem has a feasible (not necessarily optimal) solution if and only if all clusters are feasible.

Proof:

Necessity: If a cluster is not feasible, at least one constraint cannot be satisfied. So there are intersecting net segments of different nets which cannot be assigned on different layers. By lemma 3.1, we do not have a feasible solution.

Sufficiency: If all clusters are feasible, we have a unique value assignment for the variables involved. We can assign all remaining variables outside the clusters arbitrary {0, 1} values. The layer assignment is feasible by corollary 3.2.

Q.E.D.

Another way to look at the problem is by means of a constraint graph. In the graph, a net segment corresponds to a node, and an edge exists between two nodes if and only if the two corresponding net segments meet at a intersecting point. Fig. 4 gives the constraint graph for the example given in Fig. 3. Then a maximal cluster corresponds to a connected component in the graph. A cluster is feasible if and only if the corresponding connected component in the constraint graph is two-colorable. Thus we state the following corollary:

Corollary 4.1:

The via minimization problem formulated has a feasible (not necessarily optimal) solution if and only if the corresponding constraint graph is two-colorable (or contains no odd cycles).

We state the following property of a maximal cluster as a lemma.

Lemma 4.1:

If the value of any one variable in a maximal feasible cluster is fixed, the values of all variables in the cluster are fixed.

Proof:

Let v be the variable whose value is fixed. Because the values of the two variables in any given constraint are complementary, the variable(s) in the same constraint(s) as v has a fixed value. By the definition of a cluster, every constraint in a cluster must share at least one variable with at least one other constraint in the cluster. Thus, the values of all variables in the cluster will be fixed.

Q.E.D.

In the constraint graph, a cluster corresponds to a connected sub-graph and to assign the layer to net segments is to color the constraint graph with two colors. Once a node is colored, the color of all the other nodes in the connected sub-graph is fixed.

We will see in next section how to develop a solution based on the consideration of maximal clusters.

5. Solution Method Based on Weighted Cluster Graph

We now deal with maximal cluster in the constrained graph of section 4. Let x^i be the variable in the maximal cluster i, which represents a net segment assignment of any vertex in i. Similarly, let x^j

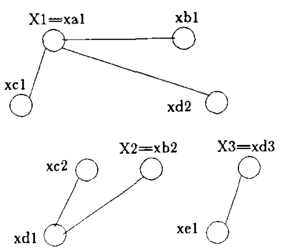

Fig. 4

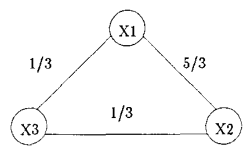

Fig. 5

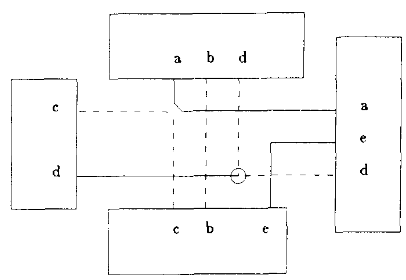

Fig. 6

be in the maximal cluster j. We introduce a new {0, 1} variable y_{ij} according to the following:

$$y_{ij} = 0 \quad \text{if } x^i \text{ and } x^j \text{ have the same value}$$
$$\phantom{y_{ij}} = 1 \quad \text{otherwise}$$

Then the objective function f introduced in section 3 can be written in terms of {0, 1} variable y_{ij} as:

$$f = \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} M_{ij} \qquad (8)$$

where q is the total number of maximal clusters, and M_{ij} is the part of f contributed by clusters i and j only.

Let us define:

D_{ij} = cost contributed by the clusters i and j if x^i and x^j are assigned the same values, i.e. $y_{ij} = 0$

C_{ij} = cost contributed by the clusters i and j if x^i and x^j are assign different values, i.e. $y_{ij} = 1$

Then:

$$\begin{aligned} M_{ij} &= M_{ij}\ |_{y_{ij}=0} \text{ or } M_{ij}\ |_{y_{ij}=1} \\ &= D_{ij} \times (1 - y_{ij}) + C_{ij} \times y_{ij} \end{aligned}$$

The crucial point to be noticed is that all C_{ij}'s and D_{ij}'s are coefficiencies which can be pre-calculated because all x^i's can be arbitrarily assigned the value without affect each other. Then the objective function f to be minimized in equation (8) can be simplified as:

$$\begin{aligned} f &= \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} M_{ij} \\ &= \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} \{ C_{ij} \times y_{ij} + D_{ij} \times (1 - y_{ij}) \} \\ &= \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} \{ (C_{ij} - D_{ij}) \times y_{ij} + D_{ij} \} \\ &= \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} C_{ij}' \times y_{ij} + \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} D_{ij} \\ &= \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} C_{ij}' \times y_{ij} + D \end{aligned}$$

By dropping the constant term D, we obtained a linear objective function

$$f' = \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} C_{ij}' \times y_{ij} \qquad (9)$$

The minimization of f is then subject to the constraints

$$x^i + x^j - y_{ij} = 2 \times z_{ij} \qquad (10)$$

where z_{ij} is a dummy boolean variable. By means of this manipulation, we have converted our original integer programming problem to a linear one.

EXAMPLE

Let us refer to Fig. 3c. Since there are three maximal clusters, let us choose $x^1 = x_{a1}$, $x^2 = x_{b2}$ and $x^3 = x_{d3}$ arbitrarily.

To obtain D_{12}, we assign $x^1 = 1$ and let $x^2 = x^1 = 1$. Notice that the values of the variables in the same cluster as x^1 are determined by (6). We have: $x_{b1} = x_{d2} = x_{c1} = 0$. Also the values of the variables in the same cluster as x^2 are determined. We have $x_{d1} = 0$ and $x_{c2} = 1$. Dropping the terms containing variables not in the maximal clusters 1 and 2, i.e. those containing x_{d3} and x_{e1} in cluster 3, we have from the function f in equation (7),

$$\begin{aligned} D_{12} &= M_{12}|_{y_{12}=0} \\ &= \frac{1}{3} \times | x_{d1} - x_{d2} | + 1 \times | x_{b1} - x_{b2} | + 1 \times | x_{c1} - x_{c2} | \\ &= \frac{1}{3} \times | 0 - 0 | + 1 \times | 0 - 1 | + 1 \times | 0 - 1 | \\ &= 2. \end{aligned}$$

Similarly, we have the following:

$D_{13} = 1/3$

$D_{23} = 1/3$

To obtain C_{12}, we need to get M_{12} for $y_{12} = 1$. We assign $x^1 = 1$ and $x^2 = 0$. Thus $x_{b1} = x_{d2} = x_{c1} = 0$, $x_{d1} = 0$ and $x_{c2} = 0$. dropping the terms containing x_{d3} and x_{e1} in cluster 3, we have:

$$\begin{aligned} C_{12} &= M_{12}|_{y_{12}=1} \\ &= \frac{1}{3} \times | x_{d1} - x_{d2} | + 1 \times | x_{b1} - x_{b2} | + 1 \times | x_{c1} - x_{c2} | \\ &= \frac{1}{3} \times | 1 - 0 | + 1 \times | 1 - 1 | + 1 \times | 0 - 0 | \\ &= \frac{1}{3}. \end{aligned}$$

Thus we obtain:

$$C_{12}' = C_{12} - D_{12} = \frac{1}{3} - 2 = -\frac{5}{3}.$$

Similarly, we get:

$$C_{13}' = C_{13} - D_{13} = 0 - \frac{1}{3} = -\frac{1}{3}$$

$$C_{23}' = C_{23} - D_{23} = 0 - \frac{1}{3} = -\frac{1}{3}.$$

Physical Constraints

In IC layout, the layers may not be equal in performance. Some net segments may have pre-assigned layers. For each of these net segments, we need to add another constraint:

x_p = K,

where x_p is the corresponding variable and K is a constant, 0 or 1.

By means of this kind of constraints, we can release the assumption which we made in section 1. If a terminal can only be accessed on one layer, we simply add a constraint for the net segment connecting the terminal.

To optimize the chip performance, we may want to limit the number of contacts on a critical net: say the number of contacts on net n must be less than a constant N. Then we can pose the following constraints to take care of the case:

$$\sum_{\textit{all via candicates of net n}} | x_i - x_j | < N ,$$

where x_{ni} and x_{nj} belong to net n and share a via candidate.

With a more complicated cost function, we can also reflect the trade-offs between the number of contacts and the wire length on a preferred layer or we can maximize the wire length on the preferred layer as a secondary goal.

Weighted Cluster Graph

Because of the large number of variables in the formulation, any kind of known constraint relaxation technique is not practical due to the space and/or run-time problems. We propose a graph theoretical approach to solve our problem. First, let us build a weighted cluster graph G(V, E) as follows:

A node, v_i, in V has a one-to-one correspondence to a maximal cluster in the formulation. An edge, e = $\{v_i, v_j\}$, in E has a one-to-one correspondence to the boolean variable y_{ij} in the problem formulation. A weight, $w_{ij} = -C_{ij}'$, is assigned to the corresponding edge. The weighted cluster graph for the example given in Fig. 3c is shown in Fig. 5.

The problem now becomes of finding a maximal cut in the weighted cluster graph which will divide the vertex set into two parts. The vertices in one part will be colored in one color while the vertices in the other part will be colored in the other color. If the weighted cluster graph is planar, we can find the optimal solution in polynomial time ([9] and [10]). Unfortunately, the weighted cluster graph is not planar in general and we know that to find a maximal cut in a general graph is NP-complete ([11] and [12]). So we will try some heuristics to solve the problem.

Special Case of the Problem

Let us conduct some special cases of the problem from the graph theory point of view.

Lemma 6.1:

If the weighted cluster graph is a bipartite graph, the optimal solution can be found in O(| E |) time.

This lemma comes from the fact that the maximum cut of a bipartite graph is the set of the edges with positive weights in the graph.

Lemma 6.2:

If the weighted cluster graph is a planar graph, the optimal solution can be found in O($|V|^{\frac{1}{2}} \times |E|$) time ([9], [10], [13] and [14]).

A graph is called *biconnected* if and only if there exist at least two different paths connecting each pair of nodes. The *biconnected components* of a graph are maximal *biconnected* subgraphs of the graph. Notice that two biconnected components of a graph are either disconnected or connected by a node, an edge or a simple path. Then we have the following:

Lemma 6.3:

The maximum cut in a biconnected component of the weighted cluster graph is a part of the maximum cut for the graph.

Theorem 6.1:

The biconnected components of the weighted cluster graph can be handled independently.

The above theorem gives a way to partition the original problem into smaller ones if it is possible.

The Algorithm

step 1:Get input data; identify all cross points, knock-knee points and via candidates; and build the weighted cluster graph G = (V, E);

step 2:Sort the edges of G by their weights in decreasing order in E and use the "edge-pick-up" procedure described below to get an initial solution.

step 3:Improve the initial result by swapping the nodes.

edge-pick-up procedure

Input:weighted cluster graph G = (V, E, W)

Output:

an edge cut, C, partitioning the node set V into two disjoint sets, V_s and V_t

Objective:

maximize $W = \Sigma\, w_e$

{

While ((($e = \{v_i, v_j\}) \in E$) = NULL)

{

if (both v_i and v_j NOT marked)

{

mark $v_i \in V_s$ and $v_j \in V_t$;

put $e \in C$;

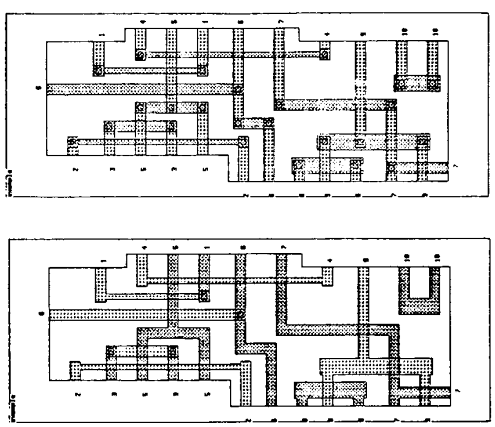

Fig. 7

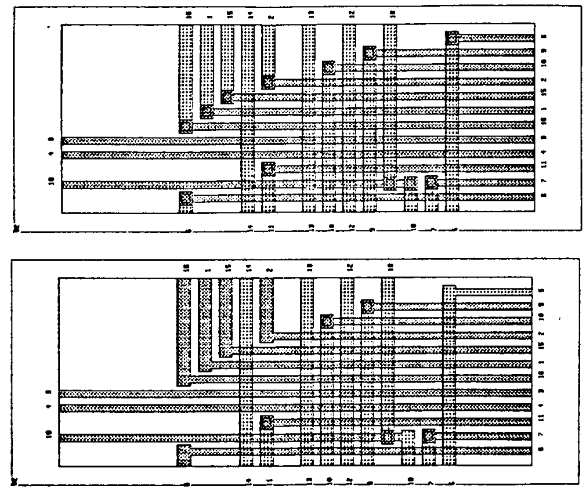

Fig. 8

}

else if (both v_i and v_j MARKED)

do nothing;

else

{

mark the unmarked node $\in$ the different set to the marked node;

put $e \in C$;

}

go to next edge in E;

}

}

Time Complexity

To implement step one, we can use the plane-sweep technique to find all cross points, knock-knee points and via candidates; the time complexity will be O(nlogn + k) where n is the number of layout objects in the input chip and k is the total number of cross points, knock-knee points and via candidates [15]. In step two, we need O(| E | log| E |) time to sort the edges and O(| E |) to find the cut. So the time complexity for step two is O(| E | log| E |) where | E | is the number of edges in the weighted cluster graph.

In order to improve the initial result obtained by the above procedure, we propose the following iterative improvement approach. After the initial solution is obtained, the node set V of the weighted cluster graph is divided into two sub-node-sets, V_s and V_t, by the cut C. For each node v, w_{vs} is the sum of the weights of the edges incident to v and those nodes which belong to the same sub-node-set as v; w_{vt} is the sum of the weights of the edges incident to v and those nodes which belong to the complementary sub-node-set. If w_{vs} is larger than w_{vt} for a node v, then we will move the node, v, from its original sub-node-set to the complementary sub-node-set.

To further improve the result, we can group the nodes which are connected by the minimal weighted edges in the same sub-node-set to form pseudo nodes and repeat the above process.

Each iteration of such an improvement takes O(| V |) time where | V | is the number of the nodes in the weighted cluster graph.

Usually, the number of layout objects is much larger than the number of edges, | E | , and the number of nodes, | V | , in the weighted cluster graph. So the overall time complexity of our algorithm is O(nlogn + k).

6. Results

Following the above algorithm, we find (X_1, X_2) and (X_1, X_3) as the maximal cut for the example given in Fig. 3c. The layer assignment is shown in Fig. 6. Fig. 7 and Fig. 8 show the layer assignment of two other examples. In all the examples given here, the optimal solution is obtained. In other words, the number of vias in these examples is the minimum possible. Notice that the number of vias is the minimum possible under the given topology. As an example, if we allow to re-route the channel given in Fig. 7. Fig. 9 shows the topological routing result with only one via, and Fig. 10 is a geometric realization of the topology.

We have used our algorithm to minimize the number of vias in D. N. Deutsch's difficult channel routing example which was routed with 406 vias. The via minimization step is done in about 10 minutes on a microVax II work-station. The final number of vias is 335, reduced by 17.5 percents.

7. Concluding Remarks

We used a unified {0, 1} linear programming formulation for the constrained via minimization problem. Based on the analysis of the formulation and the use of graph theory, we developed an efficient algorithm to solve the general practical via minimization problem.

The rapid advance in VLSI fabrication technology has made it possible to use more than two layers for routing. So, how to minimize the number of vias in routing for more than two layer is an important problem. We propose to extend our approach to handle the problem involving more than two routing layers.

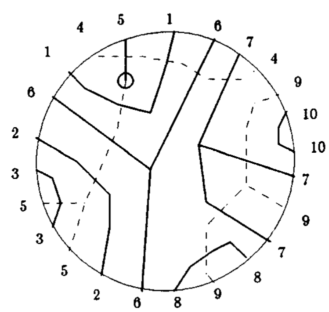

Fig. 9

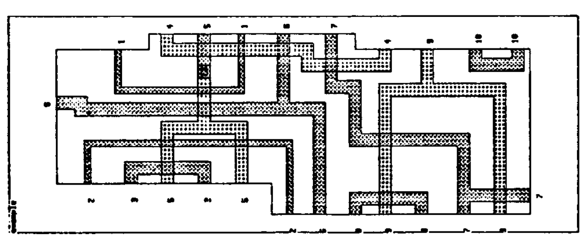

Fig. 10

References

[1]. Hashimoto, A., and Stevens, J., "Wire Routing by Optimizing Channel Assignment within Large Apparatus", Proc. 8th Design Automation Workshop, 1977, pp 155-169.

[2]. Kajitani, Y., "On Via Hole Minimization of Routing on a 2-layer Boards", Proc. 1980 ICCC, 1980, pp 295-298.

[3]. Ciesielski, M. J. and Kinnen, E., "An Optimum Layer Assignment for Routing in ICs and PCBs", Proc. 18th Design Automation Conference, 1981, pp 733-737.

[4]. Chan, R. W., Kajitani, Y., and Chan, S. P., "On the Via Minimization Problem for a Two-Layer Printed Circuit Board", Conference Records of the 15th Asilomar Conference on Circuits, systems and Computers, 1981, pp 22-26.

[5]. Chan, R. W., Kajitani, Y., and Chan, S. P., "Topological considerations of the Via Minimization Problem for two Layer PC. Boards", Proc. ISCAS, 1982, pp 968-971.

[6]. Chan, R. W., Kajitani, Y., and Chan, S. P., "A Graph-Theoretic Via Minimization Algorithm for Two-Layer Printed Circuit Boards", IEEE Transactions on Circuits and Systems, Vol. CAS-30, No. 5, May 1983, pp 284-299.

[7]. Pinter, R. Y., "Optimal Layer Assignment Interconnect", Proc. ICCC, 1982, pp 398-401.

[8]. Chang, K. C. and Du, D. H-C., "Efficient algorithms for layer assignment problem", IEEE Transactions on CAD of Integrated Circuits and Systems, Vol. CAD-6, no. 1, pp. 67-78, Jan. 1987.

[9]. Hadlock, F., "Finding a Max. Cut of a Planar Graph in Polynomial Time", SIAM J. Comp., vol. 6, 1, 1977, pp 86.

[10]. Aoshima, K. and Iri, M., "Comments on F. Hadlock's paper: 'Finding a Maximal Cut of a Planar Graph in Poly. time' ", SIAM J. Comp., vol. 6, 1, Mar. 1977, pp 86-87.

[11]. Garey, M. R. and Johnson, D. S., "Computers and Intractability -- Guide to the Theory of NP-Completeness", W. H. Freeman and Co., San Francisco, 1979.

[12]. Orlova, G. I. and Dorfman, Y. G., "Finding the Maximal cut in a graph", Eng. Cybernetics, vol. 10, 1972, pp 502-506.

[13]. Edmonds, J., "Maximum matching and a polyhedron with 0,1 -- vertices", J. Res. Nat. Bur. Stand, B69, June 1965, pp 125-130.

[14]. Micali, S. and Vazirani, V. V., "An $O(|V|^{\frac{1}{2}} \times |E|)$ Algorithm for Finding Maximum Matching in General Graphs", Proceedings of 21st Annual Symposium on Foundations of Computer Science, Oct. 1981, pp 17-27.

[15]. Nievergelt, J. and Preparata, F. P., "Plane-Sweep Algorithms for Intersecting Geometric Figures", Communications of the ACM, Volume 25, Number 10, October 1982, pp. 739-747.