A Graph-Partitioning-Based Approach for Multi-Layer Constrained Via Minimization *

Yih-Chih Chou Youn-Long Lin
Department of Computer Science, Tsing Hua University
Hsin-Chu, Taiwan 300, R.O.C.
{ycchou,ylin}@theda23.cs.nthu.edu.tw

Abstract

We propose a new layer assignment approach for the k-layer Constrained Via Minimization (CVM) problem. We transform the problem into a constrained k-way graph partitioning one. Practical issues such as pin-out constraint, over-the-cell constraint, and overlapping between wire segments of the same net, have all been taken into consideration. We propose a modified simulated-annealing program for the problem. A set of large routing results generated by a commercial three-layer router has been used to test the effectiveness of the program. Up to 70% reduction of vias has been observed. Assuming an additional fourth layer is available, more reduction is achieved. This work is the first to demonstrate the feasibility of via minimization for practical-sized multi-layer layout. It is also applicable to future design with more layers.

1 INTRODUCTION

The via minimization problem is to, performed after detailed routing and before compaction, assign wire segments to interconnection layers such that the number of vias is minimized while all design rule constraints are satisfied. Vias should be used only when they are absolutely necessary because they not only occupy large area but also degrade performance and reliability. The Constrained Via Minimization (CVM) problem limits the possible via locations to the ends of a wire segment.

For two-layer CVM, much research has been conducted [1][2][3][4] and the problem is considered solved. Nowadays, as the fabrication technology progresses, more and more layers are available for interconnection. It is not uncommon to see foundry service providing technology with 5 or even 6 layers of metal [5][6]. Therefore, there is a need for the EDA community to research on a multi-layer via minimizer.

The multi-layer CVM problem has been shown NP-hard [7][12]. Recently, some heuristic methods have been proposed [7][8][9][10][11]. But they do not support the use of stacked vias.

* Supported in part by the National Science Council, R.O.C., under a contract no. NSC87-2215-E-007-017.

In this paper, we propose a new approach for the k-layer CVM (CVM-K) problem. We transform the problem into a constrained graph k-partitioning one and consider the existence of stacked vias. We address several practical issues such as pin-out constraint, over-the-cell constraint, and overlapping between wire segments of the same net. A modified simulated-annealing program has been implemented. This work is the first to demonstrate the feasibility of via minimization for practical-sized multi-layer layout. It is also applicable to future design with more layers.

The rest of this paper is organized as following. In section 2, we formulate the CVM-K problem and show how it can be transformed into a constrained k-way graph partitioning one. In section 3, we propose a heuristic similar to, but more effective than, simulated annealing for the problem. A solution to the partitioning problem implies a solution to the CVM-K problem. Some experimental results are presented in section 4. Finally, in section 5 we draw a conclusion and point to possible directions for future research.

2 PROBLEM FORMULATION

First, we define some terminologies. A *net* is a group of electrically-connected wire segments. A *via* is a contact connecting two wire segments of the same net assigned to different layers. Vias can be stacked. For example, a connection from metal-1 to metal-3 needs two vias which are stacked together. Two wire segments are *mutually exclusive* if they cannot be assigned to the same layer. For example, two overlapping wire segments of different nets are mutually exclusive. Two overlapping wire segments are *connected* if they are not mutually exclusive, i.e., they belong to the same net.

Now we define the CVM-K problem as: Given an initial layout with n wire segment, W_1, W_2, ..., W_n, each can be assigned to a set of k different layers, L_1, L_2, ..., L_k. Let α_i denote the assignment of the ith wire segment ($1 \leq \alpha_i \leq$ k). Find a feasible assignment $\langle \alpha_1, \alpha_2, ..., \alpha_n \rangle$ such that the number of vias is minimized while all design rules are satisfied.

If two wire segments, W_i and W_j, are connected and $\alpha_i \neq \alpha_j$, a via (or stacked vias) is needed. Therefore, we should assign as many connected pairs of wire segments to the same layer as possible. This can be viewed as a partitioning problem as following.

First, we construct a graph G(V, $E_1 \cup E_2$). Each node $v_i \in$ V represents a wire segment i. E_1 is the set of exclusive edges while E_2 is the set of connected edges. An *exclusive edge* links two nodes if their corresponding wire segments are mutually exclusive. A *connected edge* links two nodes if their corresponding wire segments are electrically connected. Then, we partition the graph into k parts such that all wire segments in the same layer will have their corresponding nodes in the same partitioning. Figure 1 shows a small example illustrating this graph model.

An edge is *cut* if its two terminals lie in different partitions. A cut connected edge implies the need of a via. But, a via may serve the need of multiple cut connected edges. For example, in Figure 1(c), both connected edges $e_{2,3}$ and $e_{2,4}$ are cut. But, they correspond to the same via in Figure 1(a). Therefore, calculating the number of vias needed from the graph is not trivial and will be detailed in the next section.

A partitioning is *feasible* if, and only if, all exclusive edges are cut. Moreover, since stacked vias are allowed, we have to check the validity of all vias to make sure that the final solution is correct. The checking process will be described in the next section.

The quality of a partitioning (in terms of the number of vias needed) can be improved by moving nodes from one partition to another. A move is feasible so long as it keeps all exclusive edges cut as depicted in Figure 2. Therefore, our CVM-K problem can be treated as a constrained k-way graph partitioning problem.

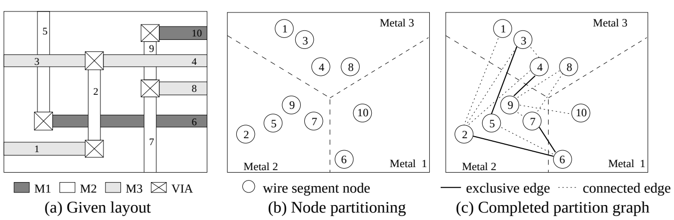


Figure 1: Constructing a partition graph

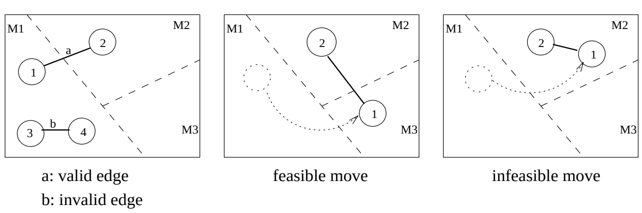


Figure 2: Example of the definition

3 A Heuristic Method

As we stated above, a feasible partitioning does not imply a feasible layer assignment. As shown in Figure 3, although the partitioning is feasible, the via between segments 1 and 3 (assigned to metal 1 and metal 3, respectively) is run through by segment 2 (assigned to metal 2). This will cause a circuit short. So we have to check whether the partitioning (assignment) will lead to illegal vias whenever it is modified. By definition, we can get the following result:

Observation 3.1 *An assignment is feasible if and only if its corresponding partitioning is feasible and causes no illegal vias.*

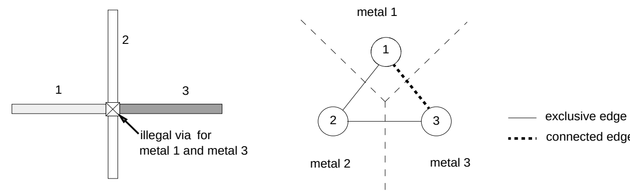


Figure 3: An example of illegal assignment

The checking method is applied after every feasible move of a node from one partition to another. For the moved node i, we will mark every c_i of its exclusive neighbor set C_i. For each c_i, we will check each cc_i of its connected neighbor set CC_i. If cc_i is marked (i.e., it is also an exclusive neighbor of i), we will check the layer assignment of i, c_i and cc_i. If i is assigned to a layer between the layers of c_i and cc_i, this move will be discarded because it will cause an illegal via.

We modify a simulated annealing-based [13] optimization heuristic for our partitioning problem. A pseudo code description of the proposed approach is given in Figure 4.

Our modification is in the inner loop. In a standard simulated annealing, the inner loop iteratively performs random moves until an equilibrium condition is reached, i.e., sufficient time has passed after the last positive move. In our approach, every node will be tried to move to every possible partitions sequentially, not randomly. A positive move is definitely accepted, while a negative move is accepted probabilistically as in a standard simulated annealing process. In addition, these movements should also be checked to see if it will cause any illegal via. The initial temperature of this simulated annealing process is 5000 degree. The freezing temperature is zero. We decrease the temperature by 20% over each iteration.

Some improvement can only be achieved with simultaneous movement of multiple nodes. For example, in Figure 5(a), the via can only be eliminated by swapping the layer assignment of wire segments a1 and a2 as depicted in Figure 5(b). The second portion of the inner loop implements this optimization. According to experiments to be presented later, about 5% more minimization can be achieved with this feature.

We count the via number by checking nodes of each net from bottom layer to top layer, and marking all traversed nodes and their neighbors to prevent redundant counting.

In our algorithm, when we move a node from one partition to another, we do not have to recompute the number of vias again. All we have to do is to make some modification to the via number. When a node is moved from partition A to partition B, we have to check all of its connected neighbors in both partitions. If the connected neighbor is in partition A(B) and is *unmarked*, we will increase(decrease) the via number by 1 and mark all its connected neighbors. Otherwise, the via number is kept the same. An example is shown in Figure 6. Node 1 is moved from partiton A to partition B. Node 2 is checked first. Since node 2 is unmarked, we will increase the via number by 1 and mark all its connected neighbors. Thus we can avoid increasing the via number redundantly when we check node 3. Similarly, if a moved node causes a reduction in the via number, we can also modify the via number correctly.

Theorem 3.1 *The modification process stated above is correct.*

Algorithm KLA

```
begin
   construct graph G;
   calculate #vias N;
   Temp = Initial_Temp;
   while (Temp > Freezing_Temp)
      for each node n
         for each feasible move m of node n
            M = #vias after applying move m;
            δ = N - M;
            if (δ > 0) then
               accept move m;
               N = M;
            else
               if (RANDOM(0,1) < exp(δ *
               Initial_Temp / Temp)) then
                  accept move m;
                  N = M;
         end for
      end for
      for each exclusive edge e
         x = the move by swapping the pair of
         nodes linked by e;
         M = #vias after applying move x;
         δ = N - M;
         if (δ > 0) then
            accept move x;
            N = M;
      end for
      Temp = Temp * α;
   end while
end
```

Figure 4: The proposed layer assignment algorithm

<proof> When a node is moved from one partition to another, the changing to the via number must be dependent of nodes in both partitions. Suppose that node 1 is moved from partition A to partition B. We will check its connected neighbors in partition A(B) to make modification to the via number. If we touched a unmarked connected neighbor in partition A(B), we have to increase(decrease) the via number by 1 since this is the first time we meet this via. On the other hand, if a marked connected neighbor of node 1 is checked, it must form a 3-clique with node 1 and the node which marks it. By observation, they must share the same via. Because the via is counted before, we do not need to make any modification. □

We have to pay special attention to some practice used by state of the art routers. An overlapping pair of wire segments could belong to either same or different nets as depicted in Figure 7. We can handle all these overlapping situation in our program. By checking the corresponding coordinate values of wire segments and their connectivities, we can insert the correct edges into our graph model. Moreover, we take the pin-out constraint into consideration. For example, a wire segment connects to an I/O pin of a standard cell can only be in a particular layer. Also, in a layout generated by an area-router, available layers to an over-the-cell wire segment is restricted. We deal with these two situations by specifying the allowable partitions for each node.

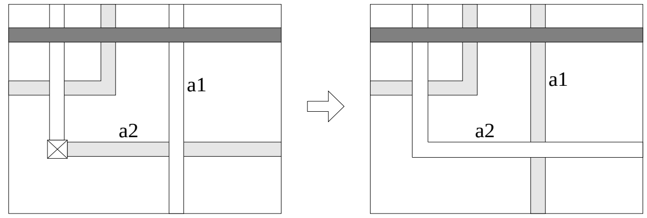


Figure 5: Swapping assignments of two wire segments(a1 and a2)

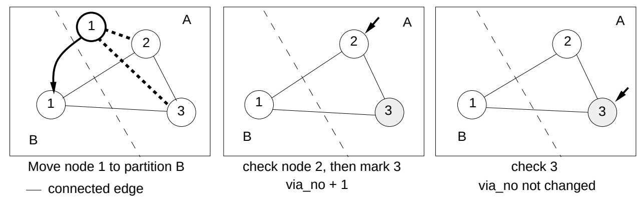


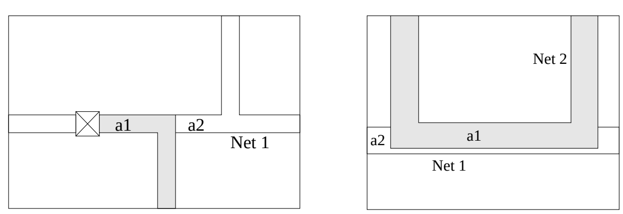
Figure 6: An example of via number updating process

4 EXPERIMENTAL RESULTS

We have implemented the proposed algorithm in a C program running on a SUN SPARC-20 workstation. We use a full-adder and some circuits from the ISCAS89 benchmark [16] to test our approach. Our experiment is conducted in a cell-based ASIC design flow. First, we use the Cadence Cell Ensemble tool [14] to create the layout in the EDIF [15] format. Then we translate it into the partition graph and run our program.

The results of all benchmarks are shown in Table 1. On the average, about 50% of via reduction has been achieved. Table 2 compares results with and without performing node-swapping, i.e., the second portion of the inner loop. Node-swapping can help to reduce about 5% more vias.

We also compare the proposed algorithm with a standard simulated annealing approach (SA) and a greedy approach (Greedy). The greedy approach was obtained by disabling the probabilistic acceptance part of our algorithm, i.e., only good moves are accepted. The results are tabulated in Table 3. Our program produced the best results in all cases. Figure 8 depicts the change in via number during the optimization process by both the standard simulated annealing and our method for test case s27. It demonstrates that our approach is more effective in traveling the solution space.



(a) wire segments a1 and a2 are overlapped but belong to the same net (Net 1)
(b) wire segments a1 and a2 are overlapped and belong to different nets

Figure 7: Overlap consideration

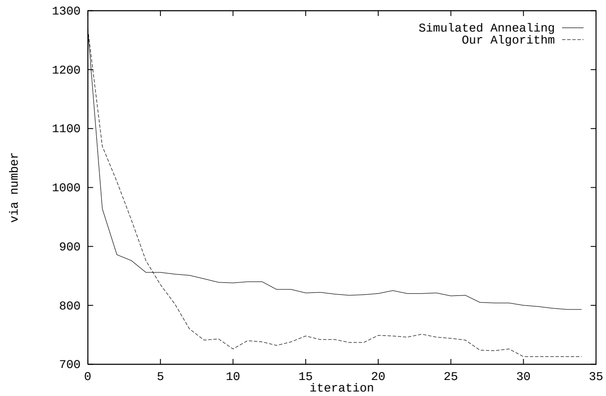


Figure 8: Via reduction process of s27

Benchmark	Initial Layout		Assignment Results		
	# wire segments	# vias	# layers	# vias reduced	% vias reduced
fullAdder	118	39	3	28	71.8
			4	31	79.5
s208	644	255	3	170	66.7
			4	203	79.6
s510	1804	823	3	356	43.3
			4	505	61.4
s27	2729	1269	3	546	43.0
			4	807	63.6
s9234	9868	4208	3	2008	47.7
			4	2920	69.4
s38584	95794	41747	3	16723	40.1
			4	24109	57.8

Table 1: Experimental Results (Initial 3 layers)

5 CONCLUSIONS

We have proposed formulating the multi-layer constrained via minimization(CVM-K) problem as a constrained k-way graph partitioning problem. We modified the simulated annealing based heuristic to solve this problem. The experimental results show that our algorithm achieves high reduction ratio for vias and can handle large circuits. It is more effective than the standard simulated annealing procedure with comparable CPU time.

In the future, we will back-annotate the assignment result by **edif-in** process to complete the entire flow. And we will extend the proposed approach for crosstalk, antenna effect, and delay minimization.

Benchmark	# vias reduced		Improve	
	w/o node-swapping	w/ node-swapping	# vias	%
fullAdder	25	28	3	12.0
s208	156	170	14	9.0
s510	328	356	28	8.5
s27	516	546	30	5.8
s9234	1910	2008	98	5.1
s38584	15910	16723	813	5.1

Table 2: Impact of node-swapping (for 3-layer design)

Benchmark	# vias reduced		
	Greedy	SA	Ours
fullAdder	22	23	28
s208	135	160	170
s510	290	322	356
s27	423	480	546
s9234	1501	1626	2008
s38584	12965	14238	16723

Table 3: Comparison with Simulated Annealing and Greedy Method (for 3-layer design)

References

[1] R. W. Chen, Y. Kajitani, and S. H. Chan, "Graph theoretic via minimization algorithm for two layer printer circuit boards," *IEEE Trans. Circuit Syst.*, vol. CAS-30, pp. 284-299, May 1983.

[2] M. J. Ciesielski, and E. Kinnen, "An optimum layer assignment for routing in IC's and PCBs," *Proc. ACM IEEE 18th Design Automation Conf.* , pp. 733-737, 1981.

[3] K. R. Stevens and W. M. Van Cleemput, "Global via elimination in a generalized routing environment," *Proc. Int. Symp. Circuits and Systems*, , pp. 689-692, 1979.

[4] A. Hashimoto and J. Stevens, "Wire routing by optimizing channel assignment within large apertures," *Proc. 8th Design Automation Workshop*, , pp. 155-169, 1971.

[5] *http://developer.intel.com/design/tech.htm*

[6] *http://www.tsmc.com.tw/Technology*

[7] K. C. Chang and H. C. Du, "Layer assignment problem for three-layer routing," *IEEE Transactions on Computers*, vol. 37, no. 5, pp. 625-632, May 1988.

[8] K. C. Chang and H. C. Du, "Efficient algorithms for layer assignment problems," *IEEE Transactions on Computer-Aided Design*, vol. CAD-6, pp. 67-78, Jan 1987.

[9] C. C. Chang and Jason Cong, "An efficient approach to multi-layer layer assignment with application to via minimization," *Proc. 34th Design Automation Conference*, pp. -, 1997.

[10] J. Cong and C. L. Liu, "On k-layer planar subset and topological via minimization problems," *IEEE Transactions on Computer-Aided Design*, vol. 10, no. 8, pp. 972-981, Aug. 1991.

[11] S. C. Fang, K. E. Chang, W. S. Feng and S. J. Chen, "Constrained via minimization with practical considerations for multi-layer VLSI/PCB routing problems," *Proc. 28th Design Automation Conference*, pp. 60-65, 1991.

[12] K. Ahn and S. Sahni, "Constrained via minimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 2, pp. 273-282, Feb. 1993.

[13] Naveed Sherwani, *Algorithms for VLSI Physical Design Automation*, Toppan Kluwer, 1995, ch. 4.

[14] CADENCE, *Preview Cell Ensemble Reference Manual*, Version 4.2.1, Sep. 1992.

[15] EDIF Schematic Technical Subcommittee, *EDIF 2 0 0 User Guide*, Sep. 1987.

[16] F. Brglez, D. Bryan, K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," *International Symposium on Circuits and Systems*, ISCAS'89, 1989, pp.1929-1934.