Integrated Interlayer Via Planning and Pin Assignment for 3D ICs

Xu He[1], Sheqin Dong[1], Xianlong Hong[1], Satoshi Goto[2]
[1]Dept. of Computer Science & Technology, Tsinghua University, Beijing, China, 100084
[2]Information Production and Systems, Waseda University, Kitakyushu, Japan, 808-0135
E-mail: hexu07@mails.tsinghua.edu.cn, dongsq@mail.tsinghua.edu.cn

ABSTRACT
As technology advances, 3D ICs are introduced for alleviating the interconnect problem coming with shrinking feature size and increasing integration density. In 3D ICs, one of the key challenges is the vertical interlayer via used for different device layers connection. In this paper, we use min-cost maximum flow algorithm for integrated interlayer via planning and pin assignment for all two-pin nets from one source block to all the other blocks, which make sure interlayer via is inserted as successfully as possible with the shortest wire length. By iteratively using this algorithm with other auxiliary methods on each block, we can deal with the problem for all nets among blocks in 3D ICs. Experimental results show its efficiency and effectiveness. To our knowledge, this is the first algorithm of interlayer via planning with pin assignment for 3D ICs.

Categories and Subject Descriptors
B.7.2 [**Integrated Circuits**]: Design Aids – *Placement and routing*.

General Terms
Algorithms, Design, Experimentation.

1. INTRODUCTION
With the persistent shrinking of device size and increasing of integration density, interconnection delay has become one of the most important issues in VLSI design. 3D ICs, which are composed of several separate device layers connected by short and vertical interlayer interconnections, can significantly reduce the interconnect delay and achieve high circuit performance.

In 3D ICs, the nets which cross multiple layers need vertical interlayer interconnections. Generally, the vertical connections are implemented by interlayer vias (signal through-the-silicon via) that goes through a device layer, connecting the pins of the same net distributed on different device layers. However, unlike the regular via (pitch≤0.5μm×0.5μm) located in metal layers, the interlayer via is very large (pitch≈5μm×5μm), and is usually placed at the whitespace between the macro blocks in the device layer, as shown in Fig. 1. In addition, interlayer via locations are the input information of the following thermal via planning, aiming at reducing the chip temperature to a satisfactory level [1]. Therefore, interlayer vias need a method to be planned efficiently.

SLIP'09, July 26–27, 2009, San Francisco, California, USA.

Many researches concerning vias in 3D ICs have been proposed. They can be classified into two categories: 1) Considering via number minimization only during floorplanning or placement [7][8][9], and 2) Integrating via planning in physical design[1][2][3][4][5][6]. In the first category, the via location is not decided. In the second category, many algorithms mainly focus on thermal via planning [1][2][3][4]. In [5], the interlayer via under consideration is placed in any place on the chip. In [6], a net-by-net approach is performed, and a heuristic interlayer via planning method is used for every net to satisfy timing requirements. As far as we known, there is no existing work that considers pin assignment during interlayer via planning.

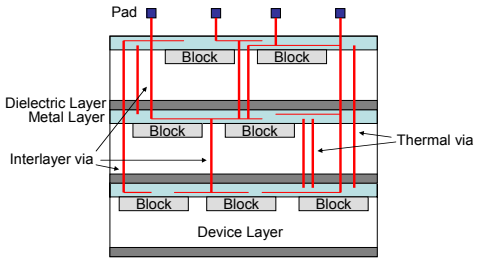

Fig. 1. Interlayer via in 3D IC stack

Pin assignment is a traditional method used for optimizing wire length. Previous algorithms of pin assignment mainly focus on 2D ICs [10][11][12][13]. If interlayer via planning process is integrated with pin assignment, the location of interlayer via will be much more flexible and the wire length will be further reduced. There are two examples shown in Fig. 2 to illustrate this method's advantage. In Fig. 2, a net (source, sink) which crosses 2 layers, needs an interlayer via to connect. In Fig. 2 (a), the wire length is long because of the limitation of whitespace for interlayer via insertion. In Fig. 2 (b), since the pins are reassigned by considering via location limitation, the wire length is shorter.

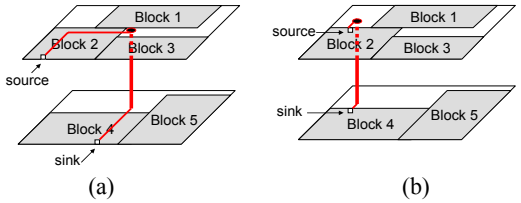

Fig. 2. Pin assignment influences interlayer via location

In this paper, we first consider simultaneous pin assignment and interlayer via planning for the two-pin nets connecting one source

block to all of its sink blocks. This problem is solved by min-cost maximum flow based algorithm, which runs in polynomial time. Then, some effective auxiliary solutions for the pin assignment and interlayer via planning among blocks are given. The process can be considered in many steps of physical design, such as during or after floorplanning or placement, or during global routing.

The rest of the paper is organized as follows. Section 2 gives the problem formulation. In Section 3, the pin assignment and interlayer via planning problem for all the two-pin nets from one source block to all of its sink blocks is solved by min-cost maximum flow algorithm. The details of solving the pin assignment and interlayer via planning problem among blocks is demonstrated in Section 4. The experimental results are shown in Section 5. Finally, the conclusion and future work are given in Section 6.

2. PROBLEM FORMULATION

In this paper, our algorithm is discussed at post floorplanning phase. The application used in other physical design phases is similar.

The problem of 3D ICs pin assignment and interlayer via planning is defined as follows:

- Problem: Pin assignment and interlayer via planning for 3D ICs.
- Objective: Interlayer via of multi-layer net can be located as successfully as possible while total wire length is minimized.
- Input: A 3D floorplan, net list, the available pin locations of blocks
- Output: The final pin positions of each net, interlayer via locations of multi-layer nets.

The basic assumptions are as follows:

- All the macro blocks are hard ones, which means each of them has fixed size and aspect ratio.
- All the nets are two-pin nets, and multi-pin nets are split into a set of two-pin nets.
- Interlayer via can only be inserted into whitespace.
- A net cannot cross the same device layer more than once. Therefore, the interlayer via amount of multi-layer net is decided when the 3D floorplan is given. In addition, the wire length of vertical segments is also determined, and cannot be further optimized. For this reason, the wire length considered is only planar wire length.

3. PIN ASSIGNMENT AND INTERLAYER VIA PLANNING FOR ONE BLOCK(PAVPO)

In this section, we demonstrate the solution of the pin assignment and interlayer via planning problem for nets from one source block to other sink blocks.

3.1 Using Grid Structure to Calculate Wire Length and Whitespace

Since interlayer vias can only be inserted into whitespace and the reduction of total wire length is the objective of our problem, a gird structure is introduced to calculate the whitespace area and to estimate the wire length. Given a 3D floorplan, each layer is divided into a set of grids ($X \times Y$), as shown in Fig. 3. To trade off the wire length estimation precision and running time, the grid size is determined both by the distance between adjacent pins of the same block and by the chip width and height.

It is obvious that the more grid edges the net crosses, the longer the wire length is. Each pin is connected with its nearby grid node. Vertical edges in Fig. 3 are interlayer via candidate insertion sites. As shown in Fig. 3, there is a two-pin net (source, sink) crossing two layers. After pin assignment and interlayer via planning, a path (bold marked) is got for the net. It should be mentioned that the path is not the net's final routing because it is used at post floorplanning phase; it is only a method to assign pin and locate interlayer via. The wire length at this phase is calculated according to the Manhattan distance of every segment from pin to its next via, or via to via, or via to its next pin along the path. However, the path can be seen as the net's routing if the application is used in routing phase, in which the capacity and size of each grid is given by manufacture parameter.

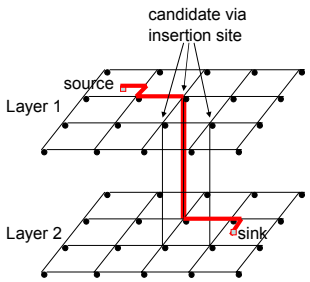

Fig. 3. Grid structure for wire length estimation and interlayer via capacity calculation

Interlayer vias must be inserted within the whitespace of device layers, as show in Fig. 1. Fig. 4 shows the region scope around a grid node. The capacity of interlayer vias in the region of a grid node is equal to ws(i)/Av, where ws(i) is the whitespace area of the region scope of grid node i, and Av is the size of an interlayer via. Only if the interlayer via capacity is more than zero, will there be a vertical connection from this grid node to its corresponding node in the lower adjacent layer. As shown in Fig. 3, there are only 3 vertical edges between the two layers because of the whitespace limitation in the upper layer.

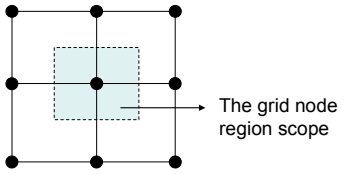

Fig. 4. The region scope of grid node

3.2 Using Min-Cost Maximum Flow Algorithm to Assign Pin and Locate Interlayer Via

The input information of this step is:

- $B = \{b_s, b_1, ..., b_n\}$: b_s is the source block, and the others are sink blocks which is connected with b_s.
- $N = N_1 \bigcup ... \bigcup N_n$: N_i (i=1,…, n) is the set of two-pin nets connecting between block b_s and block b_i.
- $P = P_s \bigcup P_1 \bigcup ... \bigcup P_n$: P_i (i=s, 1,…, n) is the set of available pin positions of block b_i.

To solve the problem of pin assignment and interlayer via planning, we need to construct a network graph, then apply min-cost maximum flow algorithm to get the solution. The network graph G=(V, E) with edge capacity and edge cost is constructed as follows:

- $V = \{s, t, t_1, ..., t_n\} \cup P \cup GN$, where s is the source node, t is the sink node, t_i (i=1,…, n) is subsink node. GN is all the grid nodes of layers crossed by N.
- $E = E_s \cup E_t \cup E_g \cup E_v \cup E_{pg} \cup E_{pt}$, where

  E_s: from s to all pin positions of P_s.

  E_t: from subsink node t_i (i=1,…,n) to t.

  E_g: the grid edges of each layer.

  E_v: edge between upper layer grid node and its corresponding grid node in the lower adjacent layer. If the interlayer via capacity of the grid node of upper layer is equal to zero after calculating, there will be no edge connecting these two nodes. In addition, the direction of edges in E_v is determined by the relative position of grid node's layer and the layer of block b_s. For example, if the layer of upper layer grid node of the edge is higher than the block b_s's layer, the edge is in upward direction, otherwise, the edge is in downward direction.

  E_{pg}: edges between the pin positions of P_i (i=s, 1,…,n) and their nearby grid node. For example, if a pin is in a grid node's region scope (shown in Fig. 4), there will be a connection between the pin and the grid node. The direction of the edge is from pins of block b_s to grid node, or grid node to pins of block b_i (i=1,…,n).

  E_{pt}: from the pins of P_i (i=1,…,n) to t_i.

  The edges in E_g are undirected, while others are directed.

- Capacity of edge:
  For edges in $E_s \cup E_{pg} \cup E_{pt}$, the capacity is 1.

  For edges (t_i, t) in E_t, the capacity is equal to $|N_i|$ (i=1,…,n)

For edges in E_v, the capacity is the upper layer grid node's interlayer via capacity.

For edges in E_g, the capacity has no constraint, and we set the capacity infinite. It should be mentioned that, the grid structure is mainly used to roughly estimate the wire length, and we do not care about the final routing from pin to pin, pin to via, or via to via. Therefore, the capacity of edges in E_g is infinite. If the algorithm is used during the routing phase, the capacity will be set according to the 3D manufacture parameter.

- Cost of edge:
  For edges in $E_s \cup E_t \cup E_{pt} \cup E_v$, the cost is zero.

  The cost of edges in $E_g \cup E_{pg}$ is according to the Manhattan distance of the two vertexes of the edge. For example, the cost of $e \in E_g$ is equal to the grid's width or height; the cost of $e \in E_{pg}$ is the Manhattan distance from the pin to its connected grid node.

Fig. 5 (b) illustrates the constructed network graph for the PAVPO problem in Fig. 5 (a). The source block s has two nets connecting block 1 and block 2, all available pin positions are pointed out on each block. Because the whitespaces are limited, there are only three vertical edges connecting grid nodes from adjacent layers. The direction of the vertical edge is downward because the upper grid node of vertical edge is in the same layer as block s.

The flow in G can be mapped to a PAVPO solution for given nets. The used capacity of edges in E_v is the number of interlayer vias needed for the solution. The Fig. 6 (b) shows the solution of flow f ($|f|=2$) illustrated in Fig. 6 (a). If a flow f exists and $|f|=|N|$, then we can find a feasible solution of pin assignment and interlayer via planning for all the nets in N. On the other hand, given a solution of pin assignment and interlayer via planning for n nets, a flow f ($|f|=n$) can always be found on the constructed flow network. Because the total capacities of edges going into sink node t are $\sum_{i=1}^{n} Capacity(t_i, t) = \sum_{i=1}^{n} |N_i| = |N|$, the maximum flow f_{max} in G, $|f_{max}| \le |N|$. If the flow $|f|<|N|$, then there is no feasible solution to the original PAVPO problem. Furthermore, the cost of the flow is also the cost of pin assignment and interlayer via planning solution. Therefore, if there is a flow f for G after using min-cost maximum flow algorithm, a solution of pin assignment and via planning for as many nets as possible with minimum total cost can be always found, as explained in the following theorem:

Theorem 1. *A min-cost maximum flow f ($|f|=|N|$) in G corresponds to a pin assignment and interlayer via planning solution for all nets in N with minimum total cost (which roughly denotes the total wire length). If $|f|<|N|$, then it gives the solution for maximum amount of nets of N, where the total cost of these subset nets is minimum.*

Although the edges in E_g are undirected, the flow got from min-cost maximum flow algorithm is directed [14]. This ensure that we can find a path from the pins of block b_s to pins of block b_i (i=1,…n) according to the flow. When the grid node has many outgoing arcs, the path connecting a pair of pins may have more than one choice in deciding which grid edge to go next. But the total cost is the same. If there are many outgoing arcs from a grid node, we randomly select one of them as the next going arc and subtract the using capacity of the arc in the flow. The final result can be derived net-by-net from the flow solution.

The whole process for solving the PAVPO problem is below:

Algorithm PAVPO (B, N, P)
1. Divide the 3D layout into grid, and calculate the capacity of interlayer via in each grid node region.
2. Construct the network graph $G(V, E)$, assign cost and capacity for each edge.
3. Use min-cost maximum flow algorithm on $G(V, E)$.
4. Get the pin positions and interlayer via locations for each net according to the flow.

Finding a min-cost maximum flow in a network is a classical problem, which has several polynomial time optimal algorithms to solve [14]. We adopt a min-cost maximum flow solver CS2, as described in [15], whose time complexity is $O(|V|^2 |E| \log(C|V|))$ for $G=(V, E)$, where C is the maximum cost value of arcs.

According to the graph construction rules, $|V|=2+n+|P|+|GN|$, the upper bound of edges is $|E|=2|P|+3|GN|+n$ (consider grid edges having two directions). In fact, the number of edges is smaller due to the limitation of interlayer via insertion candidate sites.

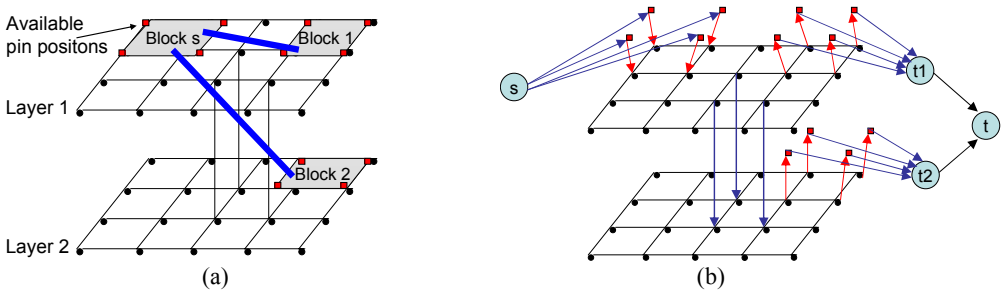

Fig. 5. (a) A PAVPO problem for two nets connecting between block s and other two blocks. (b) The corresponding network graph.

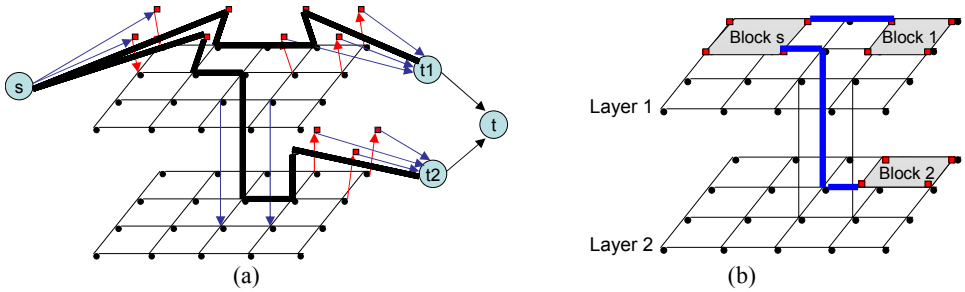

Fig. 6. (a) A flow *f* in the network in Fig. 5 (b), |*f*|=2. (b) The corresponding solution of interlayer via planning and pin assignment to the PAVPO problem in Fig. 5 (a).

4. PIN ASSIGNMENT AND INTERLAYER VIA PLANNING (PAVP)

In this section, we discuss how to extend the solution given above to apply to PAVP problem of all the nets and macro blocks.

The coarse framework to solve PAVP problem is using PAVPO algorithm iteratively for each block one by one, and the order of blocks is random. After a block having applied PAVPO algorithm, the interlayer via capacity of grid nodes and the remaining available pin positions of its sink blocks are updated, and all the two-pin nets connecting with the block are also marked. The next block will check whether its connected two-pin nets have been marked before. Only those unmarked nets connecting with this block will be assigned pin and allocated interlayer via.

More details are discussed in the following sub-sections.

4.1 Decompose Multi-pin Net

Since the input net must be a two-pin net, all the multi-pin nets in reality need to be decomposed. First, we build a complete graph, whose vertices are the pins' initial positions of the multi-pin net. Then we use Prim's algorithm to get the minimum spanning tree for this graph, so as to decompose it into a set of two-pin nets. The cost of an edge in the complete graph is according to the distance between the connected two pins. The distance is composed of planar Manhattan distance and vertical distance. The planar Manhattan distance is calculated using the half-perimeter model, and if the two pins come from different layers, we project them onto the same plane before calculation. The vertical distance is calculated by multiplying a weight c with the number of crossing layers. The weight c has been set to a large value. Therefore, the total wire length and the interlayer via amount needed are reduced.

4.2 The Solution for Problem Caused by Decomposing Multi-pin Net

As shown in Fig 7, a pin may connect to several pins from other blocks after decomposing a multi-pin net. However, the capacity of edges from E_s is 1, which means a pin position in the source block will be assigned to one net at most according to the flow f. As a result, the solution for PAVPO problem cannot be applied directly on block s in this situation.

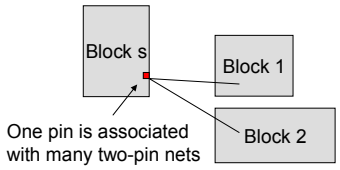

Fig. 7. One pin connecting with more than one two-pin nets

Because the critical problem is that, the number of available pin positions supported by each block is less than the number of two-pin nets connecting with the block, we should append sufficient pin positions in each block before applying PAVPO algorithm on this block. After pin assignment and interlayer via allocation, the interlayer via positions for multi-layer nets will not be changed, but the pins should be mapped to the original pin positions.

The increased pin position amount in each block is equal to the number of two-pin nets connecting with the block. In order to append available pin positions, we randomly choose original pin positions from the block and duplicate them.

The process of mapping pins to original pin positions is formulated as a min-cost maximum flow problem, and applied on each block one by one with random order. Fig. 8 (a) shows the pin assignment result on current block after using PAVPO algorithm. There are 3 original available pin positions (left column, black points); and the

number of two-pin nets is 4, two of which coming from the same net *n1* after decomposing. To apply PAVPO algorithm, there is a pin appending in the current block (left column, white point). The 4 corresponding sink pins are in the right column.

Fig. 8 (b) shows the network flow graph for mapping. Each net connected with current block is represented by a node n_i (two-pin nets from the same multi-pin net *i* share the same node n_i). Each original available pin positions are indicated by node p_j (j=1,…,|*P*|, |*P*| is the number of original pin positions amount). Every node n_i has a directed edge to p_j, and the capacity is 1, the cost is $cost(n_i, p_j)$, calculated by the following formula:

$$\cos t(n_i, p_j) = \sum_{k=1}^{m} (d_{jk} - d_k) \qquad (1)$$

where *m* is the number of two-pin nets connecting with the current block and decomposed from net *i*; d_{jk} is the Manhattan distance from the *j*th original pin to the first crossed interlayer via of the two-pin net, or the sink pin if the two pins are on the same layer; d_k is the distance between the pin assigned by PAVPO algorithm and the first interlayer via, or the sink pin if the two pins are on the same layer.

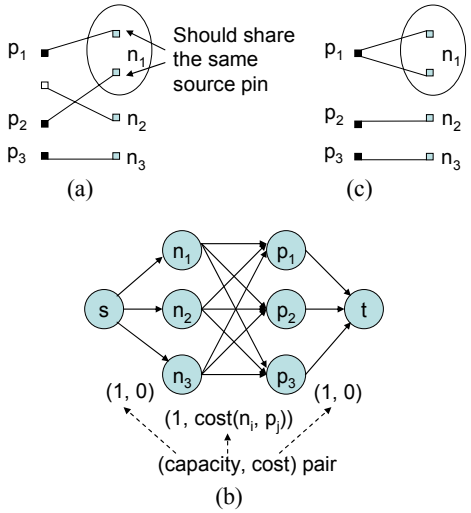

Fig. 8. Mapping pin process using min-cost maximum flow algorithm

Fig. 8 (b) also includes a source node *s* and a target node *t*. There is an edge from *s* to every n_i with capacity 1 and cost 0; an edge is built from every node p_j to target *t* with capacity 1 and cost 0 as well.

Fig. 8 (c) shows the final pin assignment result on the current block. During this pin mapping process, the position p_1 is used for the net n_1, which is composed of 2 two-pin nets.

After finishing the pin mapping process on the current block, the corresponding two-pin nets will update their pin positions and wire length.

4.3 The Framework for PAVP Problem

The framework for solving the PAVP problem is:

Algorithm PAVP	
1.	Increase the amount of available pin positions in each block.
2.	For each block, apply PAVPO(B, N, P) algorithm.
3.	Get pin's final positions by using pin mapping process on every block, and update wire length of corresponding net.

5. EXPERIMENTS AND RESULTS

We have implemented our algorithm in the C++ programming language, and all experiments are performed on a Pentium Dual 1.86 GHz PC machine. We have tested our algorithms on 2 MCNC benchmarks (ami33, ami49) and 4 generated benchmarks [16] (generated by duplicating the module and network information in MCNC benchmarks, such as M65, M99, M198 are derived from ami33, and M147 is derived from ami49) with more modules (Shown in Table 1). All input 3D floorplans have 4 device layers, and are generated from a SA-based floorplanning algorithm [8]. All the multi-pin nets are decomposed into a set of two-pin nets; while all the single-pin nets are ignored.

Table 1 Benchmarks

Circuit	Module	Net	Two-pin net	Area (μm×μm×layer)
ami33	33	123	357	623 × 616 × 4
ami49	49	408	523	3430 × 3486 × 4
M65	65	123	829	854 × 868 × 4
M99	99	123	1312	1050 × 1127 × 4
M147	147	408	2385	6104 × 5922 × 4
M198	198	123	2759	1540 × 1610 × 4

We compare our algorithm PAVP (section 4.3) with a net-by-net approach VPNN, which considers only one net each time and finds a shortest path between source pin and sink pin to allocate interlayer via. The net order in VPNN is at random. The pin positions of nets in VPNN are set according to the benchmarks' original information; while these pin positions are only seen as available positions for pin assignment in PAVP. PAVP and VPNN are run under two different scenarios: one is DMST where all the multi-pin nets are decomposed using minimum spanning tree model (section 4.1); and the other is DSS where multi-pin nets are decomposed by choosing a pin to be the source and all the others to be sinks. The results for the four different algorithmatic combinations are summarized in Table 2, where PAVP/DMST means PAVP algorithm applied to scenario DMST, and so on.

For each test circuit, these approaches are repeated 5 times. Table 2 lists the average result of these 5 times. In table 2, we report for each algorithm combination: 1) Total wire length in mm (Wire). If the wire detours, the length is acquired by calculating the Manhattan distance of each segment from source pin to its next via, or via to next via, or via to its sink pin. 2) Total interlayer via insertion amount (Via). In these experiments, interlayer via's successful insertion ratios is 100% (wire can detours to find via insertion site), so we only list the via insertion amount. If the whitespace is not enough for interlayer via insertion, we can still get the maximum insertion amount of interlayer via according to Section 3.2. 3) The percentage of the chip area occupied by interlayer vias (Area ratio) 4) The CPU time in seconds (CPU). From Table 2, we can see that:

- Under the same DMST scenario (i.e., using MST model to decompose multi-pin net), the PAVP and VPNN have the same total via insertion amount. However, the wire length in PAVP is 6.9% shorter than VPNN has. The run time in PAVP is much less than VPNN. That's because PAVP is using block-by-block iteration algorithm; while VPNN is using net-by-net algorithm. The same conclusion on the comparison of PAVP versus VPNN holds for the DSS scenario.

- Using the same algorithm, it is obvious that the PAVP/DMST needs much less interlayer via and shorter wire length than PAVP/DSS does. The same conclusion can be drawn in the comparison of VPNN/DMST and VPNN/DSS. Due to the less interlayer via amount and shorter wire length, the run time is also shorter in DMST scenario.

To summarize, it is evident that the PAVP/DMST is the best combination among these four in wire length, with less interlayer via amount and CPU time.

6. CONCLUSION AND FUTURE WORK

In this paper, we have addressed the issue of simultaneous interlayer via planning and pin assignment which is new for 3D. Experimental results have shown that our algorithm can effectively optimize the total wire length and interlayer via amount.

Since interlayer vias can also be used for dissipating thermal in 3D ICs, locating interlayer vias in the right place and appropriately assigning pins can reduce the total thermal via requirement amount while optimizing wire length. Therefore, thermal analysis of interlayer via will be discussed in the future work.

7. REFERENCES

[1] J. Cong and Y. Zhang, "Thermal-driven multilevel routing for 3-D ICs," in Proc. Asia South Pacific Des. Autom. Conf., Jan. 2005, pp. 121–126.

[2] T.Zhang, Y.Zhan, and S.S.Sapatnekar, "Temperature-aware routing in 3D ICs," in Proc. ASPDAC, Jan. 2006, pp. 309–314.

[3] J. Cong, Yan Zhang, "Thermal via planning for 3-D ICs", Proc. ICCAD, p.745-752, November 06-10, 2005.

[4] Z. Li, X. Hong, et. al, "Efficient thermal via planning approach and its application in 3D floorplanning," IEEE Trans. Computer-Aided Design, 2007.

[5] V. F. Pavlidis and E. G. Friedman, "Via placement for minimum interconnect delay in three-dimensional (3-D) circuits", in Proc. IEEE ISCAS, pp.4587-4590, 2006.

[6] J. Lu, S. Chen, et. al, "Performance maximized interlayer via planning for 3D ICs", in Proc. IEEE ASICON, pp 1096-1099, 2007.

[7] Y. Deng and W. Maly, "Interconnect characteristics of 2.5-d system integration schemes," in Proc. ACM ISPD, pp. 341-345, 2001.

[8] Z. Li, X. Hong, et. al, "Hierarchical 3D Floorplanning Algorithm For Wirelength Optimization". in IEEE Trans on Circuits And Systems-I: Fundamental Theory and Applications, 2637-2646, 2006.

[9] P. H. Shiu, R. Ravichandran, S. Easwar, and S. K. Lim, "Multi-layer Floorplanning for Reliable System-on-Package", in Proc. IEEE International Conference on Circuits and Systems (ISCAS), 2004.

[10] H. Xiang, X. P. Tang, and D. F. Wong, "An algorithm for integrated pin assignment and buffer planning," in Proc. ACM/IEEE Design Automation Conf. (DAC), pp. 584–589, 2002.

[11] L. E. Liu and C. Sechen, "Multilayer pin assignment for macro cell circuits," IEEE Trans. Computer-Aided Design, vol. 18, pp. 1452-1461, Oct. 1999.

[12] X. Yao, M. Yamada, and C. L. Liu, "A new approach to the pin assignment problem," in Proc. ACM/IEEE Design Automation Conf., 1988, pp. 566-572.

[13] H. Xiang, X. P. Tang and D. F. Wong, "Min-cost flow-based algorithm for simultaneous pin assignment and routing", IEEE Trans. Computer-Aided Design, vol. 22, pp 870-878, July. 2003.

[14] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice Hall/Pearson, 2005.

[15] A. V. Goldberg. "An efficient implementation of a scaling minimum-cost flow algorithm". Journal of Algorithms, 22(1):1–29, 1997.

[16] X. He, S. Dong, et. al, "Simultaneous Buffer and Interlayer Via Planning for 3D Floorplanning", in Proc. ISQED, March. 2009, pp. 740-745.

Table 2 The comparison of interlayer via planning with pin assignment in 3D ICs

Circuit	Wire	Via	Area ratio	CPU	Wire	Via	Area ratio	CPU
	VPNN/DSS				**VPNN/DMST**			
ami33	152.58	368	0.60%	3.18	64.0	122	0.20%	1.13
ami49	1262.4	696	0.04%	2.65	1201.3	572	0.03%	1.86
M65	483.2	771	0.65%	4.21	199.5	191	0.16%	1.13
M99	913.7	1156	0.61%	7.80	328.9	202	0.11%	1.14
M147	9778.4	2458	0.04%	7.88	6917.4	960	0.02%	3.63
M198	2671.9	2593	0.65%	17.6	858.5	276	0.07%	1.90
	PAVP/DSS				**PAVP/DMST**			
ami33	123.7	368	0.60%	0.53	61.2	122	0.20%	0.50
ami49	1047.4	696	0.04%	0.59	990.0	572	0.03%	0.52
M65	418.6	771	0.65%	0.75	194.3	191	0.16%	0.62
M99	765.4	1156	0.61%	1.41	323.6	202	0.11%	1.08
M147	9113.8	2458	0.04%	1.32	6493.2	960	0.02%	1.10
M198	2434.3	2593	0.65%	2.62	850.3	276	0.07%	2.31
	PAVP/DMST vs. VPNN/DMST				**PAVP/DMST vs. PAVP/DSS**			
Imp.(%)	6.9%	0%	0%	43.2%	35.9%	71.1%	71.1%	15.1%