Fast and Optimal Redundant Via Insertion

Kuang-Yao Lee, Cheng-Kok Koh, *Senior Member, IEEE*, Ting-Chi Wang, and Kai-Yuan Chao, *Senior Member, IEEE*

***Abstract*—Redundant via insertion is highly effective in improving chip yield and reliability. In this paper, we study the problem of *double-cut via insertion* (DVI) in a postrouting stage, where a single via can have, at most, one redundant via inserted next to it and the goal is to insert as many redundant vias as possible. The DVI problem can be naturally formulated as a zero–one integer linear program (0–1 ILP). Our main contributions are acceleration methods for reducing the problem size and the number of constraints. Moreover, we extend the 0–1 ILP formulation to handle via density constraints. Experimental results show that our 0–1 ILP is very efficient in computing an optimal DVI solution, with up to 73.98 times speedup over existing heuristic algorithms.**

***Index Terms*—Integer linear program (ILP), redundant via insertion, via density.**

I. Introduction

THE SCALING of manufacturing technology has made integrated circuits (ICs) more sensitive to process variations. As a consequence, the yield of ICs has suffered [1]. In particular, via-open defects have been identified as one of the main causes of chip failures [2]. Therefore, it is crucial to reduce the yield loss due to via defects, which may be caused by several reasons, such as electromigration, cut misalignment, and thermal-stress-induced voiding effects [3]–[5]. Partially failed vias add unexpected delay, and completely failed vias leave open nets in a circuit. Considering that a redundant via provides an alternative current path, redundant via insertion has become one of the well-known and highly recommended methods to improve via yield and reliability [2], [6]–[9]. Redundant vias can provide a 6% increase in the yield over the wafers with single-cut vias, a significant improvement considering that a 1% yield loss in a 300-mm wafer fab costs a chipmaker about five million dollars on an annual basis [9].

In general, the redundant via insertion problem can be tackled during routing or in a postrouting stage. In [10]–[13], redundant vias are inserted in a postrouting stage. Lee and Wang [10] formulated the problem as a maximum independent set (MIS) problem, which considered all single vias simultaneously. To solve the large-sized MIS problem, a divide-and-conquer heuristic was proposed to partition the whole problem into several smaller MIS problems that were solved one at a time. However, optimality was not guaranteed. In [11], the single vias of a design were considered individually for redundant via insertion. Therefore, only locally optimal solutions could be obtained. In [12], the redundant via insertion problem was formulated as a bipartite matching problem that could be solved optimally only when the design is grid-based and involves, at most, three routing layers. All these works ignored via density constraints, the violation of which could adversely affect the yield and reliability of a design. Such constraints were considered in [13], which solved the problem with a two-stage heuristic approach but with no guarantee on optimality. Some previous works, such as [14] and [15], considered the redundant via insertion problem in the routing stage. Nonetheless, for further improvement of the yield and reliability of vias and electrical characteristics, it may still be desirable to perform additional redundant via insertion after the routing stage.

In this paper, we study the problem of *double-cut via insertion* (DVI) in a postrouting stage, where a single via can have, at most, one redundant via inserted and the goal is to insert as many redundant vias as possible. As it is a MIS problem, the DVI problem can be naturally formulated as a zero–one integer linear program (0–1 ILP) [16]. Our main contributions are acceleration methods that exploit the properties of the DVI problem for reducing the problem size and the number of constraints without sacrificing the optimality. Moreover, we extend the 0–1 ILP formulation to consider also via density constraints. Experimental results show that our formulation is more complete than the bipartite matching formulation in [12]. Although a MIS formulation is, in general, not scalable, the runtimes of our accelerated 0–1 ILP approach are comparable with those of a bipartite matching algorithm reported in [12]. Moreover, our 0–1 ILP approach can always generate optimal solutions, with runtimes that are up to 73.98 times faster than the approach in [10], which was also based on a MIS formulation. When via density constraints are considered, our approach can insert more redundant vias than an approach adapted from the two-stage approach in [13], with runtimes that are up to 41.9 times faster.

The rest of this paper is organized as follows. In Section II, we give the definition of the DVI problem. In Section III, we present a 0–1 ILP formulation and the accelerated methods for the DVI problem. In Section IV, we extend the 0–1 ILP formulation to consider via density constraints. We report the

Manuscript received May 16, 2008; revised July 21, 2008. Current version published November 19, 2008. This work was supported in part by the National Science Council of Taiwan under Grant NSC 96-2917-I-007-018, Grant NSC 97-2220-E-007-015, Grant NSC 97-2220-E-007-030, and Grant NSC 97-2220-E-007-032. This paper was recommended by Associate Editor D. Z. Pan.

K.-Y. Lee and T.-C. Wang are with the Department of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan (e-mail: d924347@oz.nthu.edu.tw; tcwnag@cs.nthu.edu.tw).

C.-K. Koh is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: chengkok@purdue.edu).

K.-Y. Chao is with the Enterprise Microprocessor Group, Intel Corporation, Hillsboro, OR 97124 USA (e-mail: kaiyuan.chao@intel.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2008.2006151

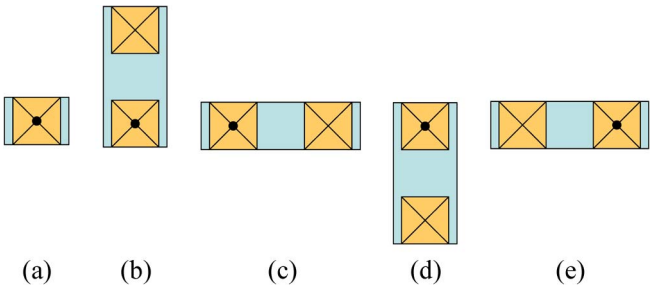


Fig. 1. Double-cut via types [10]. (a) Single via. (b) Type 1. (c) Type 2. (d) Type 3. (e) Type 4.

experimental results in Section V and conclude this paper in Section VI. Parts of the results presented in this paper appeared in [17].

II. Double-Cut Via Insertion

Without the loss of generality, we assume that a redundant via can be inserted next to a single via in one of four positions. Therefore, as in [10], four types of double-cut vias can be defined, as follows.

Definition 1 (Double-Cut Via): A single via and a redundant via inserted next to it are defined as a double-cut via. Based on the position of the redundant via, the double-cut via is one of the four types, as shown in Fig. 1. Given a single via i, its double-cut via of type j ($j \in \{1, 2, 3, 4\}$) is denoted by $\mathrm{dv}(i, j)$. Aside from this, a double-cut via is said to be feasible if replacing the single via with the double-cut via does not violate any design rule and does not overlap with any critical area, assuming that other single vias are kept intact; otherwise, the double-cut via is an infeasible one.

It should be noted that the via structures shown in Fig. 1 are only for illustration. In our experiments, different sets of test circuits have different via-related rules, such as via cut size, enclosure rule, and via cut spacing rule. These differences are all accounted for in the experiments. The DVI problem is formally defined as follows.

DVI: Given a routed design and a user-specified set of single vias, the problem of DVI is that of replacing as many single vias in the given via set as possible with double-cut vias, without violating any design rule and without rerouting any net.

Note that redundant via insertion may change the timing behavior of a design [11]. Aside from this, inserting redundant vias in congested areas may also hurt the yield and reliability of a design [13]. We assume that designers have taken these into consideration when they specify the set of single vias to be considered for redundant via insertion. In particular, our formulation in Section IV deals with via density constraints directly.

III. 0–1 ILP Formulation for DVI

In this section, we shall describe the formulation of the DVI problem as a 0–1 ILP. We also present methods to improve the efficiency of our 0–1 ILP formulation.

TABLE I
Notation

Notation	Description
$\mathrm{fdv}_{i,j}$	Feasible double-cut via $\mathrm{dv}(i,j)$
FDV	The set of all $\mathrm{fdv}_{i,j}$'s
$\mathrm{fdv}_{i,j} \leftrightarrow \mathrm{fdv}_{i',j'}$	$\mathrm{fdv}_{i,j}$ and $\mathrm{fdv}_{i',j'}$ cannot be simultaneously inserted (i.e., they have a conflict relation)
sv_i	single via i that has at least one feasible double-cut via
SV	The set of all sv_i's
$v_{i,j}$	The vertex of a conflict graph corresponding to $\mathrm{fdv}_{i,j}$
V_i	The set of vertices $v_{i,j}$'s of single via sv_i
#RVI	The number of inserted double-cut vias

A. Preliminaries

We follow the notation and definitions used in [10]. For ease of reference, these notation and definitions are provided in Table I and Definition 2.

Definition 2 (Conflict Graph): A conflict graph $G(V, E)$ is an undirected graph constructed from a detailed routing solution and a user-specified set of single vias. Each feasible double-cut via $\mathrm{fdv}_{i,j}$ is associated with a vertex $v_{i,j}$ in V. An edge $(v_{i,j}, v_{i',j'}) \in E$ if and only if $\mathrm{fdv}_{i,j} \leftrightarrow \mathrm{fdv}_{i',j'}$.

B. 0–1 ILP Formulation

Before constructing the 0–1 ILP problem, we have to compute all sv_i's, all $\mathrm{fdv}_{i,j}$'s, and the conflict relation between any pair of feasible double-cut vias. All information can be obtained by constructing a conflict graph. In a conflict graph, two vertices are connected by an edge if the corresponding feasible double-cut vias cannot be simultaneously inserted into the design due to some design rules (an external conflict relation) or they originate from the same single via (an internal conflict relation).

We use the algorithm proposed in [10] to construct the corresponding conflict graph $G(V, E)$ from a given design and the user-specified set of single vias. To formulate the problem as a 0–1 ILP, we associate each $v_{i,j} \in V$ with a binary variable $R_{i,j}$. If $R_{i,j} = 1$ in the 0–1 ILP solution, the corresponding $\mathrm{fdv}_{i,j}$ is inserted into the design. Therefore, we should add the following *conflict constraints* into our 0–1 ILP formulation:

$$\text{Conflict constraint}: \quad R_{i,j} + R_{i',j'} \leq 1 \qquad \forall (v_{i,j}, v_{i',j'}) \in E \tag{1}$$

which means that, at most, one of the $\mathrm{fdv}_{i,j}$ and $\mathrm{fdv}_{i',j'}$ can be inserted into the design if $\mathrm{fdv}_{i,j} \leftrightarrow \mathrm{fdv}_{i',j'}$. Considering that the objective of the DVI problem is to maximize the number of inserted double-cut vias (i.e., *#RVI*, as defined in Table I), we can now formally define our 0–1 ILP formulation, as follows:

$$\begin{aligned} \text{Maximize} \quad & \sum_{v_{i,j} \in V} R_{i,j} \\ \text{subject to} \quad & R_{i,j} + R_{i',j'} \leq 1 \qquad \forall (v_{i,j}, v_{i',j'}) \in E \\ & R_{i,j} \in \{0, 1\} \qquad \forall v_{i,j} \in V. \end{aligned} \tag{2}$$

This formulation has been used in [16] to find a MIS of G. However, the time complexity to solve a 0–1 ILP problem is,

in general, quite high. Therefore, we cannot afford to apply the 0–1 ILP approach to a large design directly. In the following, we will describe three acceleration methods that exploit the properties of the DVI problem to speed up the 0–1 ILP solver without sacrificing the optimality.

C. Speedup

1) Reducing the Number of Constraints: Given a conflict graph $G(V, E)$, the edge set E can be divided into two disjoint subsets $E_I = \{(v_{i,j}, v_{i',j'}) | \forall (v_{i,j}, v_{i',j'}) \in E \text{ with } i = i'\}$ (internal edge set) and $E_X = \{(v_{i,j}, v_{i',j'}) | \forall (v_{i,j}, v_{i',j'}) \in E \text{ with } i \neq i'\}$ (external edge set). Considering that each single via sv_i has, at most, four feasible double-cut via types, it contributes to G a clique that has, at most, four vertices. Therefore, there are, at most, six edges in E_I [and, at most, six constraints of the form in (1)] for each sv_i. However, those internal conflict constraints can be simplified by a single *double-cut constraint* that limits the single via sv_i to have, at most, one redundant via inserted next to it:

$$\text{Double-cut constraint}: \sum_{\text{fdv}_{i,j}\text{'s of sv}_i} R_{i,j} \leq 1. \quad (3)$$

In fact, this technique can be applied to any cliques of a conflict graph so that all conflict constraints associated with a clique can be combined into a single constraint. However, the complexity for identifying cliques in a graph is, in general, quite high. In the conflict graph of a DVI problem instance, the cliques that are formed by internal edges can be identified readily and efficiently. Therefore, we take advantage of such a property of the DVI problem and replace the conflict constraints of internal edges with the double-cut constraints.

It should be noted that we do not have to construct the double-cut constraint for a single via that has only one feasible redundant via. With the double-cut constraint (3), we only have to construct the conflict constraints (1) for the edges of E_X in the following modified but equivalent 0–1 ILP formulation:

$$\begin{aligned} \text{Maximize} \quad & \sum_{v_{i,j} \in V} R_{i,j} \\ \text{subject to} \quad & \sum_{\text{fdv}_{i,j}\text{'s of sv}_i} R_{i,j} \leq 1 \qquad \forall \text{sv}_i \in \text{SV} \\ & R_{i,j} + R_{i',j'} \leq 1 \qquad \forall (v_{i,j}, v_{i',j'}) \in E_X \\ & R_{i,j} \in \{0, 1\} \qquad \forall v_{i,j} \in V. \end{aligned} \quad (4)$$

From our experience, there are many more edges in E_I than in E_X. Therefore, the number of constraints of our 0–1 ILP formulation can be significantly reduced. The numbers of constraints and binary variables of the 0–1 ILP are $|E_X| + |\text{SV}|$ and $|V|$ (i.e., $|\text{FDV}|$), respectively.

Fig. 2 shows an example of the new 0–1 ILP formulation. Given the routed design, as shown in Fig. 2(a), we assume that the single vias 1, 2, and 3 are allowed to have one redundant via inserted. We further assume that the top redundant via of single via 1 and the left redundant via of single via 2 (the right redundant via of single via 1 and the bottom redundant

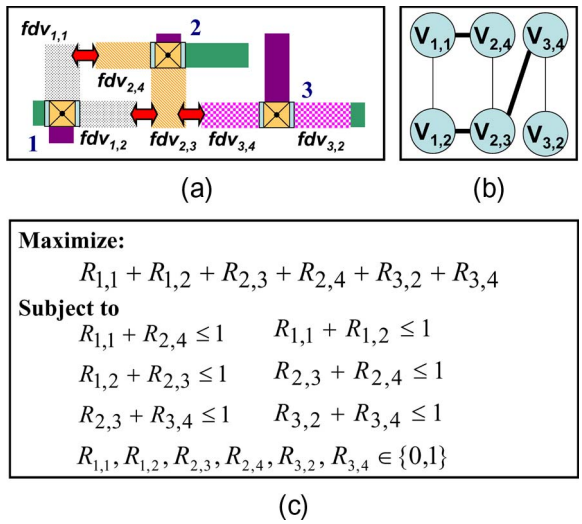


Fig. 2. Example of the DVI problem.

via of single via 2, the bottom redundant via of single via 2 and the left redundant via of single via 3) cannot be inserted simultaneously due to design rules. The corresponding conflict graph is shown in Fig. 2(b), with the edges of E_X shown in bold. Considering that $\text{fdv}_{1,1} \leftrightarrow \text{fdv}_{2,4}$, the conflict constraint $R_{1,1} + R_{2,4} \leq 1$ is included in the 0–1 ILP problem, as shown in Fig. 2(c). Similarly, $R_{1,2} + R_{2,3} \leq 1$ and $R_{2,3} + R_{3,4} \leq 1$ are also included. For the single vias 1, 2, and 3, the double-cut constraints $R_{1,1} + R_{1,2} \leq 1$, $R_{2,3} + R_{2,4} \leq 1$, and $R_{3,2} + R_{3,4} \leq 1$ are also included.

2) Preselection: Given a conflict graph $G(V, E = E_I \cup E_X)$, for each single via sv_i, let V_i denote the set of vertices $v_{i,j}$'s of sv_i. We compute the vertex subset $\text{CV}_i = \{v_{i,j} | v_{i,j} \in V_i$ has no incident edges that belong to $E_X\}$. Suppose the CV_i of sv_i is not empty. Considering that the vertices in CV_i have no conflict relation with the vertices originating from the other single vias, we can arbitrarily include any $v_{i,m} \in \text{CV}_i$ in a MIS of G. Aside from this, as a single via can have, at most, one redundant via inserted next to it, we can now delete all vertices $v_{i,j}$ originating from sv_i and all corresponding edges from the conflict graph.

Let $\text{RV}_{\text{PS}}(G)$ denote the set of preselected vertices of G. Let the reduced conflict graph be denoted by $\tilde{G}(\tilde{V}, \tilde{E} = \tilde{E}_I \cup \tilde{E}_X)$, where

$$\begin{aligned} \tilde{V} &= V - \bigcup_{\text{sv}_i \in \text{SV}} \{v_{i,j} | v_{i,j} \in V_i \wedge |\text{CV}_i| > 0\} \\ \tilde{E}_I &= \left\{(v_{i,j}, v_{i',j'}) | v_{i,j}, v_{i',j'} \in \tilde{V} \wedge (v_{i,j}, v_{i',j'}) \in E_I\right\} \\ \tilde{E}_X &= \left\{(v_{i,j}, v_{i',j'}) | v_{i,j}, v_{i',j'} \in \tilde{V} \wedge (v_{i,j}, v_{i',j'}) \in E_X\right\}. \end{aligned} \quad (5)$$

Considering that this method can decrease the numbers of variables and constraints of the 0–1 ILP problem, the efficiency of our approach can be improved. It can be shown that the preselection method does not hurt the optimality.

Lemma 1: Let $\text{RV}_{\text{PS}}(G)$ and $\tilde{G}$ denote the preselected vertex set of conflict graph G and the reduced conflict graph,

Fig. 3. Example of the preselection method.

respectively. Aside from this, let $\mathrm{RV}_{01}(G)$ denote a MIS of G [or a 0–1 ILP solution obtained from (4)]. Then, $\mathrm{RV}_{\mathrm{PS}}(G) \cup \mathrm{RV}_{01}(\tilde{G})$ is an independent set of G, and $|\mathrm{RV}_{\mathrm{PS}}(G)| + |\mathrm{RV}_{01}(\tilde{G})| = |\mathrm{RV}_{01}(G)|$.

Proof: First we show that if a vertex $v_{i,j} \in \mathrm{RV}_{\mathrm{PS}}(G)$ and $v_{i,j} \notin \mathrm{RV}_{01}(G)$, we can always create another MIS M of G such that $|M| = |\mathrm{RV}_{01}(G)|$ and $v_{i,j} \in M$. Considering that the vertices originating from single via sv_i form a clique in G, there must exist one $v_{i,k} \in \mathrm{RV}_{01}(G)$, where $k \neq j$. Otherwise, $\mathrm{RV}_{01}(G) \cup \{v_{i,j}\}$ is an independent set, which contradicts the fact that $\mathrm{RV}_{01}(G)$ is a MIS of G. In other words, we can always form a MIS M that contains $\mathrm{RV}_{\mathrm{PS}}(G)$ as a subset.

For $v_{i,j}$, $v_{i',j'} \in \tilde{G}$ and G, $(v_{i,j}, v_{i',j'}) \in \tilde{E}$ if and only if $(v_{i,j}, v_{i',j'}) \in E$. Therefore, it is trivial that $M - \mathrm{RV}_{\mathrm{PS}}(G) \subseteq \tilde{V}$ is an independent set of $\tilde{G}$. Hence

$$|M| - |\mathrm{RV}_{\mathrm{PS}}(G)| \leq \left|\mathrm{RV}_{01}(\tilde{G})\right|.$$

Similarly, $\mathrm{RV}_{01}(\tilde{G})$ is also an independent set of G. Moreover, considering that there are no edges between $v_{p,q} \in \mathrm{RV}_{01}(\tilde{G})$ and $v_{p',q'} \in \mathrm{RV}_{\mathrm{PS}}(G)$ in E, $\mathrm{RV}_{01}(\tilde{G}) \cup \mathrm{RV}_{\mathrm{PS}}(G)$ is an independent set of G. Consequently

$$\left|\mathrm{RV}_{01}(\tilde{G})\right| + |\mathrm{RV}_{\mathrm{PS}}(G)| \leq |\mathrm{RV}_{01}(G)|.$$

Therefore, we can conclude that

$$|M| \leq \left|\mathrm{RV}_{01}(\tilde{G})\right| + |\mathrm{RV}_{\mathrm{PS}}(G)| \leq |\mathrm{RV}_{01}(G)|.$$

As $|M| = |\mathrm{RV}_{01}(G)|$, $|\mathrm{RV}_{\mathrm{PS}}(G)| + |\mathrm{RV}_{01}(\tilde{G})| = |\mathrm{RV}_{01}(G)|$. ■

Fig. 3 shows an example for the preselection method. In the given conflict graph, $v_{1,1}$ and $v_{1,4}$ have no incident edge belonging to E_X; $v_{1,2}$ and $v_{1,3}$ each has an incident edge that belongs to E_X and connects to subgraphs SG_2 and SG_3. The computed vertex subset $\mathrm{CV}_1 = \{v_{1,1}, v_{1,4}\}$. Assuming that $v_{1,1}$ is preselected, $v_{1,2}$, $v_{1,3}$, $v_{1,4}$, and the associated edges shall be deleted from the conflict graph. The reduced conflict graph includes only the subgraphs SG_2 and SG_3.

As we remove the preselected vertices (and the associated vertices and edges) from G, more vertices may become available for preselection. In other words, some vertices from $\tilde{G}$, the reduced graph, may not have external edges. This suggests a recursive application of the proposed preselection method. Instead of a recursive implementation, we present an iterative procedure, as follows.

1) Construct a doubly linked list $L = \{v | v \in V$ has no external neighbors$\}$. In addition, we define a vertex set $\mathrm{RV}_{\mathrm{PS}}(G)$ to be the preselected vertex set of G. Initially, $\mathrm{RV}_{\mathrm{PS}}(G)$ is an empty set.

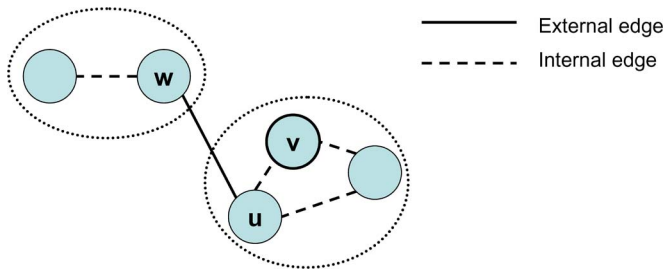


Fig. 4. Vertex v is preselected, and vertex w may be preselected later.

2) If L is empty, return $\mathrm{RV}_{\mathrm{PS}}(G)$ and G.
3) Extract the first vertex v in L and add it to $\mathrm{RV}_{\mathrm{PS}}(G)$.
4) For each internal neighbor u that belongs to the same single via as v:
   a) For each external neighbor w of u, if w has no external edges other than the one connecting it to u, as shown in Fig. 4, add w to L.
   b) If u is in L, delete it from L (considering that v is already preselected).
   c) Delete u and its associated edges from G.
5) Delete v from G.
6) Go to 2.

For an efficient implementation of the preselection procedure, we augment the graph data structure with the doubly linked list data structure. In other words, each vertex has two pointers, namely "next" and "previous," that facilitate its insertion into the doubly linked list L with $O(1)$ time complexity. Hence, the presence of non-NULL pointers in these two fields of a vertex indicates that the vertex is currently in L. Consequently, it takes $O(1)$ time complexity to determine whether a vertex is in L. Naturally, it takes also $O(1)$ time complexity to remove such a vertex from L.

Now, let us analyze the time complexity of the preselection procedure. The computation of L in step 1) can be performed using a depth-first graph traversal algorithm, which is $O(|V| + |E|)$. We traverse an edge of v to u to remove each internal neighbor of v in step 4). Moreover, we traverse an edge of u in step 4a) (to check on w, the other endpoint of the edge) right before we remove u and its edges in step 4c). In other words, we traverse each edge, at most, once in the iterative procedure of steps 2)–6). As each vertex is inserted into L and subsequently deleted from L, at most, once, the overall complexity of the preselection procedure is therefore $O(|V| + |E|)$.

It is clear to see that Lemma 1 still holds after performing our preselection procedure to generate $\mathrm{RV}_{\mathrm{PS}}(G)$ and the reduced conflict graph.

3) Computing Connected Components: From the real circuits used in the experiments of [10], we observed that their corresponding conflict graphs are all sparse graphs (see Section V-B for details). For example, the conflict graph of circuit C5 has 574 142 vertices; however, it has 257 015 connected components,[1] and the largest connected component (in terms of the sum of the numbers of vertices and edges) contains

[1]In an undirected graph, two vertices are in the same connected component if and only if there exists a path between them.

only 24 vertices and 37 edges. Moreover, after performing the preselection procedure, a conflict graph may be decomposed into even more connected components due to the preselected and subsequently removed vertices. Therefore, our 0–1 ILP approach can also be accelerated by first computing the connected components of a (reduced) conflict graph and then solving a smaller 0–1 ILP problem associated with each connected component. The union of the solutions for individual connected components (and the preselected double-cut vias) is the overall solution to the DVI problem. It can be proved that the accelerated 0–1 ILP approach described here also solves the DVI problem optimally. Note that the conflict graph in Lemma 2 could be the original (G) or the reduced ($\tilde{G}$) one.

Lemma 2: Let $\mathrm{RV}_{01}(G)$ denote the MIS obtained from a 0–1 ILP solution of G. Then, $\bigcup_{G_{\mathrm{cc}} \in G} \mathrm{RV}_{01}(G_{\mathrm{cc}})$ is an independent set of G and $|\mathrm{RV}_{01}(G)| = \sum_{G_{\mathrm{cc}} \in G} |\mathrm{RV}_{01}(G_{\mathrm{cc}})|$, where G_{cc} is a connected component of G.

Proof: Without the loss of generality, we assume that a conflict graph $G(V, E)$ is composed of two connected components $G_{\mathrm{cc1}}(V_{\mathrm{cc1}}, E_{\mathrm{cc1}})$ and $G_{\mathrm{cc2}}(V_{\mathrm{cc2}}, E_{\mathrm{cc2}})$. Similarly, let $\mathrm{RV}_{01}(G_{\mathrm{cc1}})$ and $\mathrm{RV}_{01}(G_{\mathrm{cc2}})$ denote the MISs obtained from the 0–1 ILP solutions of G_{cc1} and G_{cc2}, respectively. Based on the definition of a connected component, if two vertices $v_{i,j}$ and $v_{i',j'}$ belong to different connected components, they are not connected by any edge. Therefore, $\mathrm{RV}_{01}(G_{\mathrm{cc1}}) \cup \mathrm{RV}_{01}(G_{\mathrm{cc2}})$ is an independent set of G.

By contradiction, assume that

$$|\mathrm{RV}_{01}(G_{\mathrm{cc1}})| + |\mathrm{RV}_{01}(G_{\mathrm{cc2}})| < |\mathrm{RV}_{01}(G)|\,.$$

We can partition $\mathrm{RV}_{01}(G)$ into two disjoint subsets

$$M_1 = \{v_{i,j} | v_{i,j} \in \mathrm{RV}_{01}(G) \text{ and } v_{i,j} \in V_{\mathrm{cc1}}\}$$
$$M_2 = \{v_{i,j} | v_{i,j} \in \mathrm{RV}_{01}(G) \text{ and } v_{i,j} \in V_{\mathrm{cc2}}\}\,.$$

Therefore

$$|\mathrm{RV}_{01}(G)| = |M_1| + |M_2| > |\mathrm{RV}_{01}(G_{\mathrm{cc1}})| + |\mathrm{RV}_{01}(G_{\mathrm{cc2}})|\,.$$

Hence, $|M_1| > |\mathrm{RV}_{01}(G_{\mathrm{cc1}})|$ or $|M_2| > |\mathrm{RV}_{01}(G_{\mathrm{cc2}})|$, contradicting that $\mathrm{RV}_{01}(G_{\mathrm{cc1}})$ and $\mathrm{RV}_{01}(G_{\mathrm{cc2}})$ are, respectively, the MISs of G_{cc1} and G_{cc2}. Therefore, $|\mathrm{RV}_{01}(G)| = \sum_{G_{\mathrm{cc}} \in G} |\mathrm{RV}_{01}(G_{\mathrm{cc}})|$. ■

We use the depth-first graph traversal algorithm [18], which has $O(|V| + |E|)$ time complexity, to determine the connected components of a (reduced) conflict graph. Consider a connected component $G_{\mathrm{cc}}(V_{\mathrm{cc}}, E_{\mathrm{cc}})$. Let $\mathrm{SV}_{\mathrm{cc}} \subseteq \mathrm{SV}$ be the set of single vias such that $\mathrm{sv}_i \in \mathrm{SV}_{\mathrm{cc}}$ if it has, at least, one feasible double-cut via $\mathrm{fdv}_{i,j}$ and $v_{i,j} \in V_{\mathrm{cc}}$. The 0–1 ILP formulation for G_{cc} is as follows:

$$\begin{aligned}
\text{Maximize} \quad & \sum_{v_{i,j} \in V_{\mathrm{cc}}} R_{i,j} \\
\text{Subject to} \quad & \sum_{\mathrm{fdv}_{i,j}\text{'s of } \mathrm{sv}_i} R_{i,j} \le 1 \quad \forall \mathrm{sv}_i \in \mathrm{SV}_{\mathrm{cc}} \\
& R_{i,j} + R_{i',j'} \le 1 \quad \forall (v_{i,j}, v_{i',j'}) \in E_{\mathrm{cc}} \text{ and } i \ne i' \\
& R_{i,j} \in \{0, 1\} \qquad \forall v_{i,j} \in V_{\mathrm{cc}}. \qquad (6)
\end{aligned}$$

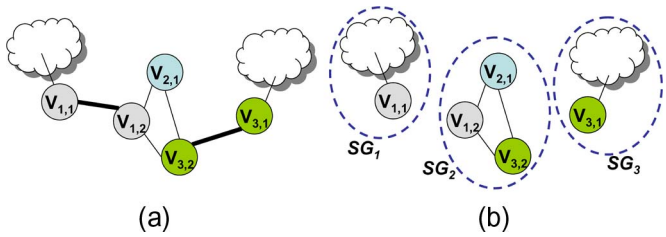


Fig. 5. Improper breaking of the original problem.

Note that when computing the connected components of G ($\tilde{G}$), we consider both E_I and E_X ($\tilde{E}_I$ and $\tilde{E}_X$). Otherwise, the accelerated 0–1 ILP approach may not solve the original problem optimally or correctly. In the conflict graph, as shown in Fig. 5(a), if we ignore the edges $(v_{1,1}, v_{1,2})$ and $(v_{3,1}, v_{3,2})$, both of which represent internal conflicts, there will be three connected components, as shown in Fig. 5(b). Suppose $v_{1,1}$ and $v_{3,1}$ in connected components SG_1 and SG_3 are included in the final solution. When solving the 0–1 ILP problem of the connected component SG_2, any of $v_{1,2}$, $v_{2,1}$, and $v_{3,2}$ can be in the solution. However, only $v_{2,1}$ can lead to the optimal overall solution while the other two choices lead to illegal solutions. Therefore, the edges $(v_{1,1}, v_{1,2})$ and $(v_{3,1}, v_{3,2})$ should not be ignored during the computation of connected components.

D. Overall Approach

Given a conflict graph $G(V, E)$, we first preselect a set $\mathrm{RV}_{\mathrm{PS}}(G)$ of redundant vias and then compute the connected components of the reduced conflict graph $\tilde{G}$. Next, we construct and solve the 0–1 ILP problem of each connected component. Let $\mathrm{RV}_{01}(\tilde{G})$ denote the set of inserted redundant vias obtained by taking the union of all 0–1 ILP solutions. The final DVI solution is $\mathrm{RV}_{\mathrm{PS}}(G) \cup \mathrm{RV}_{01}(\tilde{G})$.

Now, we have the following theorem regarding the optimality of our approach.

Theorem 1: Given a conflict graph G, the DVI problem can be solved optimally by first preselecting a set $\mathrm{RV}_{\mathrm{PS}}(G)$ of redundant vias, then computing a MIS for each connected component of the reduced conflict graph $\tilde{G}$, and finally taking the union of $\mathrm{RV}_{\mathrm{PS}}(G)$ and all the computed MISs.

Proof: Based on Lemma 1, we know that $\mathrm{RV}_{\mathrm{PS}}(G) \cup \mathrm{RV}_{01}(\tilde{G})$ is an independent set of G and

$$|\mathrm{RV}_{01}(G)| = |\mathrm{RV}_{\mathrm{PS}}(G)| + \left|\mathrm{RV}_{01}(\tilde{G})\right|$$

where $\tilde{G}$ is the reduced graph after performing the preselection. From Lemma 2, we know that $\bigcup_{\tilde{G}_{\mathrm{cc}} \in \tilde{G}} \mathrm{RV}_{01}(\tilde{G}_{\mathrm{cc}})$ is an independent set of $\tilde{G}$ and

$$\left|\mathrm{RV}_{01}(\tilde{G})\right| = \sum_{\tilde{G}_{\mathrm{cc}} \in \tilde{G}} \left|\mathrm{RV}_{01}(\tilde{G}_{\mathrm{cc}})\right|$$

where $\tilde{G}_{\mathrm{cc}}$ is a connected component of $\tilde{G}$. Therefore, $\mathrm{RV}_{\mathrm{PS}}(G) \cup \bigcup_{\tilde{G}_{\mathrm{cc}} \in \tilde{G}} \mathrm{RV}_{01}(\tilde{G}_{\mathrm{cc}})$ is also an independent set of G, and $|\mathrm{RV}_{01}(G)| = |\mathrm{RV}_{\mathrm{PS}}(G)| + \sum_{\tilde{G}_{\mathrm{cc}} \in \tilde{G}} |\mathrm{RV}_{01}(\tilde{G}_{\mathrm{cc}})|$. ■

IV. Via Density Consideration

In this section, we study a variant of the DVI problem, called DVI/VD, taking into account the maximum via density constraint. As pointed out in [13], if the number of inserted redundant vias is not well controlled, it may violate the maximum via density constraint and adversely affect the yield and reliability of a design. We shall now extend our 0–1 ILP formulation to tackle this problem.

Definition 3 (Maximum Via Density Constraint): Each via layer is partitioned into a set of overlapping rectangular regions, each of which has the same width W and height H, where W and H are process-dependent constants. For each region, its via density is defined as the number of vias located in it. The maximum via density constraint places an upper bound constraint U on the via density, where U is also a process-dependent constant.

Definition 4 (Candidate Violating Region): For a via region r_p, if the summation of the number of single vias in r_p and the number of single vias, which each has at least one feasible redundant via in r_p, is greater than the maximum via density constraint U, r_p is a candidate violating region. Note that a single via that is not in r_p may contribute a feasible redundant via to r_p in the computation of candidate violating regions.

Given a conflict graph $G(V, E = E_I \cup E_X)$, a candidate violating region r_p can be associated with a two-tuple (V_p, K_p), where $V_p \subseteq V$ and $K_p < |V_p|$. V_p denotes the vertex set of fdv$_{i,j}$'s that may be inserted in the region r_p and, at most, K_p of them can be simultaneously inserted into the design in order to meet the maximum via density constraint. The set of candidate violating regions, denoted R, in a design can be computed by the method proposed in [19] (with some modifications), which is a sliding-window-based approach to identify the density violation regions. Given G and R, we should add *density constraints* (7) into the 0–1 ILP formulation (4) in order to handle the maximum density constraint, as follows:

$$\text{Density constraint}: \quad \sum_{v_{i,j} \in V_p} R_{i,j} \leq K_p \qquad \forall r_p \in R. \quad (7)$$

We refer to this extended 0–1 ILP formulation as *01VD*. To efficiently solve the DVI/VD problem, we adopt the approach in Section III. However, we first have to modify the definitions of CV$_i$ and $G_{\rm cc}$ of G. The vertex subset CV$_i$ is redefined to be $\{v_{i,j} | v_{i,j} \in V_i$ is not in any candidate violating region and has no incident edges that belong to $E_X\}$. Steps 1) and 4a) of the preselection procedure shown in Section III-C2 are also modified as follows.

1) The doubly linked list L is redefined to be $\{v | v \in V$is not in any candidate violating region and has no external neighbors$\}$.
2) The vertex w is added to L only when it is not in any candidate violating region and has no other external edges except the one connecting it to the vertex u.

Lemma 3: Let $\text{RV}_{\rm PS}(G)$ and $\tilde{G}$ denote the preselected vertex set of the conflict graph G and the reduced conflict graph, respectively [$\text{RV}_{\rm PS}(G)$ is computed by the modified preselection procedure]. Moreover, let $\text{RV}_{01\text{VD}}(G)$ denote the independent set of G obtained from an optimal solution to *01VD*. Then, $\text{RV}_{\rm PS}(G) \cup \text{RV}_{01\text{VD}}(\tilde{G})$ is an independent set of G, and $|\text{RV}_{\rm PS}(G)| + |\text{RV}_{01\text{VD}}(\tilde{G})| = |\text{RV}_{01\text{VD}}(G)|$. Aside from this, the corresponding double-cut vias of $\text{RV}_{\rm PS}(G) \cup \text{RV}_{01\text{VD}}(\tilde{G})$ can be simultaneously inserted into the layout without violating the maximum via density constraint.

Proof sketch: Considering that the corresponding double-cut vias of $\text{RV}_{\rm PS}(G)$ are not in any candidate violating regions and the corresponding double-cut via set of $\text{RV}_{01\text{VD}}(\tilde{G})$ must meet the via density constraint (7), the corresponding double-cut vias of $\text{RV}_{\rm PS}(G) \cup \text{RV}_{01\text{VD}}(\tilde{G})$ can be simultaneously inserted into the layout without violating the maximum via density constraint. In addition, we can follow a similar reasoning as in Lemma 1 to show that $\text{RV}_{\rm PS}(G) \cup \text{RV}_{01\text{VD}}(\tilde{G})$ is an independent set of G and $|\text{RV}_{\rm PS}(G)| + |\text{RV}_{01\text{VD}}(\tilde{G})| = |\text{RV}_{01\text{VD}}(G)|$. ■

Given a conflict graph $G(V, E = E_I \cup E_X)$ and a set R of candidate violating regions, we define a density constraint graph $G^+(V, E \cup E^+)$, where $E^+ = \{(v_{i,j}, v_{i',j'}) | \exists r_p = (V_p, K_p) \in R$ such that $v_{i,j}$ and $v_{i',j'} \in V_p\}$. Let $G^+_{\rm cc}(V^+_{\rm cc}, E^+_{\rm cc})$ be a connected component of G^+. Then, we redefine a $G_{\rm cc}$ of G to be

$$G_{\rm cc}\left(V_{\rm cc} = V^+_{\rm cc}, E_{\rm cc} = E^+_{\rm cc} \cap E\right). \quad (8)$$

Note that, now, each $G_{\rm cc}$ may contain several connected components of G.

Given a $G_{\rm cc}(V_{\rm cc}, E_{\rm cc} \subseteq E)$ and the subset $R_{\rm cc} = \{r_p | r_p \in R$ and $V_p \cap V_{\rm cc} \neq \emptyset\}$ of the candidate violating regions, we should add the following constraint into the 0–1 ILP formulation (6) in order to handle the maximum density constraint:

$$\sum_{v_{i,j} \in V_p} R_{i,j} \leq K_p \qquad \forall r_p \in R_{\rm cc}. \quad (9)$$

Lemma 4: Let $\text{RV}_{01\text{VD}}(G)$ denote the independent set of $G(V, E)$ obtained from an optimal solution to *01VD*. Aside from this, let $G^+(V, E \cup E^+)$ be the corresponding density constraint graph. Then, $\bigcup_{G_{\rm cc} \in G} \text{RV}_{01\text{VD}}(G_{\rm cc})$ is an independent set of G, and $|\text{RV}_{01\text{VD}}(G)| = \sum_{G_{\rm cc} \in G} |\text{RV}_{01\text{VD}}(G_{\rm cc})|$, where $G_{\rm cc}$ is computed by (8). Moreover, the corresponding double-cut vias of $\bigcup_{G_{\rm cc} \in G} \text{RV}_{01\text{VD}}(G_{\rm cc})$ can be simultaneously inserted into the layout without violating the maximum via density constraint.

Proof sketch: For any two connected components $G^+_{\rm cc1}$ and $G^+_{\rm cc2}$ of G^+, there are no paths between any $v_1 \in G^+_{\rm cc1}$ and any $v_2 \in G^+_{\rm cc2}$ in G^+. Hence, for all $v_1 \in G^+_{\rm cc1}$ and $v_2 \in G^+_{\rm cc2}$, $(v_1, v_2) \notin E$ and $(v_1, v_2) \notin E^+$. In other words, v_1 and v_2 do not have any conflict relation, and the corresponding candidate violating regions of $G^+_{\rm cc1}$ and $G^+_{\rm cc2}$ do not overlap. Moreover, for each $G_{\rm cc}$ derived from $G^+_{\rm cc}$ [see (8)], the corresponding double-cut via set of $\text{RV}_{01\text{VD}}(G_{\rm cc})$ must meet the via density constraint (9). Therefore, the corresponding double-cut vias of $\bigcup_{G_{\rm cc} \in G} \text{RV}_{01\text{VD}}(G_{\rm cc})$ can be simultaneously inserted into the layout without violating the maximum via density constraint.

As $G_{\rm cc}$ here is a collection of connected components of G in Lemma 2, we can follow a similar line of argument as in

TABLE II
TEST CASES

	Circuit	CU(%)	#Nets	#I/Os	#Layers	#Vias	#A-Vias
[12]	struct	100	1920	0	3	7598	7580
	primary1	100	904	0	3	5536	5497
	primary2	100	3029	0	3	23154	22881
	s5378	100	1694	0	3	6739	6554
	s9234	100	1478	0	3	5365	5229
	s13207	100	3778	0	3	13972	13622
	s15850	100	4471	0	3	16922	16441
	s38417	100	11309	0	3	40942	39927
	s38584	100	14754	0	3	55381	53914
	mcc1	100	802	0	4	5948	5702
	mcc2	100	7118	0	4	34376	32376
[10]	C1	98	4309	20	5	24594	19796
	C2	70	5252	211	5	41157	31464
	C3	70	18157	85	5	127059	99142
	C4	95	17692	415	5	151912	112076
	C5	70	44720	99	5	357386	276032
[12]	dma_dfm	95	13256	950	6	108401	100699
	dsp1_dfm	91	28447	846	6	223550	182326
	dsp2_dfm	90	28431	846	6	232613	191614
	risc1_dfm	86	34034	629	6	344391	294973
	risc2_dfm	95	34034	629	6	350558	294500

Lemma 2 to show that $\bigcup_{G_{\rm cc}\in G} \mathrm{RV}_{01\mathrm{VD}}(G_{\rm cc})$ is an independent set of G and $|\mathrm{RV}_{01\mathrm{VD}}(G)| = \sum_{G_{\rm cc}\in G} |\mathrm{RV}_{01\mathrm{VD}}(G_{\rm cc})|$. ■

Note that Lemma 4 still holds when G is a reduced graph. With the aforementioned modifications, we can also use the flow described in Section III-D to solve efficiently and optimally the DVI/VD problem.

V. EXPERIMENTAL RESULTS

We used three sets of circuits to evaluate our 0–1 ILP approaches. The first set of test circuits was from [12]. The second set of test circuits was from [10] and was implemented with a 0.18-μm technology that has five metal layers. The third set of test circuits was also from [12]. However, as the circuits in the third set were not design rule check clean [20], we replaced and rerouted them by Cadence SoC Encounter [21]. The detailed information of all test circuits is shown in Table II. For each test circuit, "Circuit" shows the circuit name, "CU(%)" gives the core utilization,[2] "#Nets" shows the number of nets, "#I/Os" gives the number of I/O pins, "#Layers" indicates the number of routing layers, "#Vias" shows the total number of single vias, and "#A-Vias" gives the number of *alive single vias* (i.e., the ones that have at least one feasible double-cut via each).

There are some differences in these numbers reported in this paper and in [12] and [17]. For the first set of test circuits, as the program of [12] could identify only a subset of the feasible double-cut vias that we could identify [20], the numbers of "#A-Vias" shown in Table II are larger than those reported in [12]. It should be also noted that in [12], a stack via is counted as one via, even if it consists of two or more single vias. Hence, for the first set of test circuits, the numbers of "#Vias" in Table II are larger than those reported in [12] (we will show the number of stack vias in Table III when we compare our formulation with the formulation of [12] in Section V-A). For the second set of test circuits, because we consider different design rules when identifying feasible double-cut vias, the numbers of "#A-Vias" shown in Table II and the sizes of the corresponding conflict graphs shown in Table IV are larger than those reported in [17]. Despite their differences, the results in [17] and in this paper are all optimal with respect to the constructed conflict graphs.

We used CPLEX [22] as the solver for 0–1 ILP. All the experiments were conducted on a Linux-based machine with 2.4-GHz processor and 4-GB memory. In order to demonstrate the efficiency of our approaches in handling large-sized problems, we assumed that all single vias in each test circuit were considered to be replaced with double-cut vias. For the DVI/VD problem, because different sets of test circuits had different technologies, we used regions of different dimensions W and H to check for via density violations. For each test circuit, we used $W = H = 8$ times the largest spacing rule among all layers as the region dimensions. The maximum via density bound U for a test circuit was specified to be the maximum number of single vias among all regions in that circuit. That allowed us to stress test our formulation with the DVI/VD problem instances with large numbers of candidate violating regions.

For the DVI and DVI/VD problems, we shall show the results of our 0–1 ILP approaches described in Sections III-D and IV. Considering that our 0–1 ILP approaches use the algorithm proposed in [10] to construct a conflict graph, the reported CPU times (in seconds) are measured after the completion of graph construction.

A. Completeness of 0–1 ILP Formulation

Chen *et al.* [12] pointed out that if a grid-based design has up to three metal layers, the DVI problem can be modeled as a maximum bipartite matching (MBM) problem under the constraint that the redundant vias for the stacked single vias

[2]Considering that the circuits of the first set are all flattened designs, each of them has only one macro, and the core dimensions are equal to the macro dimensions. Therefore, all the core utilizations of the circuits in the first set are 100%.

TABLE III
INADEQUACY OF THE MBM FORMULATION

Circuit	TDVI [12]			#Vias	#Stack vias	#SVias	#A-SVias	TDVI_ours			01ILP-SV			0-1 ILP
	#SDVI	#DVI	T(s)[§]					#SDVI	#DVI	T(s)	#SDVI	#DVI	T(s)	#DVI
struct	7037	7445	0.23	7598	517	7081	7058	7058	7564	0.05	7058	7564	0.03	7580
primary1	5244	5417	0.16	5536	224	5312	5272	5270	5488	0.07	5270	5489	3.12	5495
primary2	21842	22486	0.75	23154	848	22306	22037	22034	22836	0.31	22034	22837	3.27	22877
s5378	5806	6048	0.20	6739	352	6387	6199	6177	6477	0.08	6182	6482	3.13	6538
s9234	4710	4854	0.16	5365	236	5129	4990	4976	5181	0.07	4979	5185	3.12	5216
s13207	12356	12752	0.43	13972	575	13397	13040	13011	13518	0.18	13015	13525	3.19	13594
s15850	14718	15256	0.52	16922	802	16120	15628	15590	16272	0.22	15600	16288	3.22	16406
s38417	36170	37223	1.37	40942	1597	39345	38298	38198	39605	0.54	38229	39645	3.44	39855
s38584	48478	50133	1.98	55381	2487	52894	51370	51223	53340	0.76	51262	53395	3.57	53802
mcc1[†]	5011	5324	0.26	5948	402	5501	5241	-	-	-	5225	5579	3.11	5682
mcc2[†]	29099	30684	1.74	34376	1955	32148	30205	-	-	-	30026	31718	3.29	32177
C1	-	-	-	24594	4064	20432	15547	-	-	-	15505	17943	3.28	19754
C2	-	-	-	41157	5860	35020	25119	-	-	-	25081	28234	3.37	31411
C3	-	-	-	127059	17159	109046	80744	-	-	-	80538	90550	4.01	98888
C4	-	-	-	151912	24076	126457	85959	-	-	-	85623	96532	4.10	111657
C5	-	-	-	357386	47162	308464	226470	-	-	-	225937	251532	5.69	275437
dma_dfm[‡]	-	71331	4.74	108401	17167	90285	80432	-	-	-	80064	92671	3.92	100116
dsp1_dfm[‡]	-	133198	11.34	223550	39696	182310	136075	-	-	-	135313	156200	4.75	180941
dsp2_dfm[‡]	-	131546	10.76	232613	38458	192505	146721	-	-	-	145864	167067	4.85	190204
risc1_dfm[‡]	-	202487	19.51	344391	55610	285494	228263	-	-	-	226743	260045	5.74	292796
risc2_dfm[‡]	-	206919	19.94	350558	55429	292560	227219	-	-	-	225397	254733	5.79	291769

[†] These circuits have 4 metal layers (see Table II for details).
[‡] These circuits have been re-placed and re-routed (see the beginning of Section V for details).
[§] The runtime was measured from a 1.2GHz SUN Blade-2000 workstation with 8GB memory.

TABLE IV
STATISTICS ON CONFLICT GRAPHS

Circuit	$\|V\|$	$\|E_I\|$	$\|E_X\|$
struct	24112	29159	281
primary1	15922	17631	214
primary2	63410	66753	864
s5378	16479	15367	1460
s9234	13408	12802	1041
s13207	34536	32290	2595
s15850	41153	38110	3217
s38417	100942	94541	7736
s38584	135146	124678	11326
mcc1	13770	12306	906
mcc2	75730	65247	3770
C1	43246	35267	1447
C2	67312	51700	2676
C3	215647	170923	8384
C4	220538	149119	10572
C5	574142	425220	19534
dma_dfm	214961	163898	18730
dsp1_dfm	367578	270231	26945
dsp2_dfm	385204	278282	27851
risc1_dfm	584185	411876	36036
risc2_dfm	562965	377225	43235

are all simultaneously inserted on the same side. For a design that has more than three metal layers, the MBM-based approach reported in [12], called *TDVI*, uses a layer partitioning method to divide the design into several layer groups, each of which is composed of up to three metal layers. Then, *TDVI* iteratively applies the MBM formulation to each layer group. In Table III, the columns under "TDVI [12]" give the related results reported in [12] that were generated by *TDVI*. "#SDVI" gives the numbers of inserted double-cut vias, in which each stack of double-cut vias is counted as one double-cut via. "#DVI" gives the numbers of inserted double-cut vias, where each double-cut via in a stack is counted individually. "T(s)" gives the runtimes.

For the circuits in Table II, however, the MBM formulation is inadequate, as these circuits all have gridless layouts and many of them have more than three metal layers. On the contrary, our 0–1 ILP formulation can optimally solve the DVI problems of general designs. In order to support the completeness of our 0–1 ILP formulation, we also adapted our 0–1 ILP formulation to consider the same problem as [12]. The modifications are as follows.

Let single vias $\{\text{sv}_1, \text{sv}_2, \ldots, \text{sv}_n\}$ form a stack via SV_t. If there exists a sv_i in SV_t whose double-cut via of type j is infeasible, we delete all $v_{i,j}$'s from the conflict graph for each sv_i in SV_t. If, for all single vias in SV_t, each has a feasible double-cut via of type j, we merge all corresponding vertices $v_{i,j}$'s, $\forall \text{sv}_i \in \text{SV}_t$, into a vertex $v_{t,j}$. We then apply our 0–1 ILP-based approach on the modified conflict graph. If the DVI solution contains a merged vertex $v_{t,j}$, the double-cut vias $\text{dv}_{i,j}$'s, $\forall \text{sv}_i \in \text{SV}_t$, will be inserted into the design. This methodology is called *01ILP-SV*.

In Table III, "#Vias" shows the numbers of single vias, which are the same as those numbers shown in Table II. "#Stack vias" indicates the numbers of stack vias. "SVias" and "A-SVias," respectively, give the numbers of single vias and the numbers of alive single vias; here, each stack via is counted as one via. "01ILP-SV" shows the results generated by *01ILP-SV*. "0–1 ILP" gives the results of our 0–1 ILP approach for the general DVI problem, where we are free to insert redundant vias of different types to the single vias in a stack via.

As in this paper we could identify more alive single vias than in [12], we implemented a version of the *TDVI* approach called *TDVI_ours*. In Table III, the columns under "TDVI_ours" give the results of *TDVI_ours*. Considering that different implementations of the layer partitioning method affect the solution quality and performance, it is unfair to compare the results of those circuits that have more than three metal layers. Moreover, the MBM formulation is optimal only for

TABLE V
STATISTICS ON REDUCED GRAPHS AND CONNECTED COMPONENTS AFTER PERFORMING PRESELECTION AND CONNECTED-COMPONENT COMPUTATION

Circuit	DVI							DVI/VD						
				Small-CC		Max-CC					Small-CC		Max-CC	
	$\lvert\tilde{V}\rvert$	$\lvert\tilde{E}\rvert$	#CC	NUM	%	$\lvert V_{cc}\rvert$	$\lvert E_{cc}\rvert$	$\lvert\tilde{V}\rvert$	$\lvert\tilde{E}\rvert$	#CC	NUM	%	$\lvert V_{cc}\rvert$	$\lvert E_{cc}\rvert$
struct	0	0	0	0	0	0	0	1421	1388	177	176	99	22	30
primary1	5	3	2	2	100	3	2	781	683	91	90	99	29	26
primary2	9	5	4	4	100	3	2	4646	3805	603	601	99	26	30
s5378	92	109	23	23	100	8	15	1366	1168	203	199	98	39	60
s9234	34	26	13	13	100	6	6	3851	3332	359	328	91	46	57
s13207	118	110	39	39	100	7	11	8707	7467	833	770	92	67	76
s15850	134	122	44	44	100	9	13	3026	2427	482	475	99	38	55
s38417	346	355	99	99	100	14	22	15234	12603	1961	1913	98	46	49
s38584	509	518	152	152	100	15	29	11859	10202	1709	1682	98	46	63
mcc1	58	39	21	21	100	6	7	1891	1563	302	299	99	38	41
mcc2	563	403	198	198	100	14	13	2044	1556	429	428	99	36	37
C1	89	47	42	42	100	4	3	5118	3690	776	770	99	41	45
C2	128	78	56	56	100	7	10	7764	4996	1256	1255	99	26	30
C3	566	319	259	259	100	6	9	14146	9171	2554	2550	99	32	29
C4	919	508	420	420	100	6	5	11032	6525	2309	2309	100	28	24
C5	1334	751	609	609	100	9	11	11096	6297	2660	2660	100	23	25
dma_dfm	3001	3719	838	838	100	14	32	54659	39768	1625	1166	72	1414	941
dsp1_dfm	6455	7748	1888	1888	100	15	33	60092	49341	2774	2113	76	2156	1475
dsp2_dfm	6122	7036	1861	1861	100	12	23	79617	62879	3330	2538	76	2172	1486
risc1_dfm	7363	7040	2544	2544	100	13	26	89981	65164	5379	4370	81	313	197
risc2_dfm	10575	11193	3392	3392	100	14	27	83486	62307	4874	4025	83	1877	1375

the DVI problem that is grid-based and has up to three metal layers. Therefore, for *TDVI_ours*, only the results of the circuits that each has, at most, three metal layers are presented in Table III.

Considering that *TDVI_ours* identifies more alive vias, it always generates better results than *TDVI*. On account that all circuits have gridless layouts, we can see that, as compared with *TDVI_ours*, *01ILP-SV* can insert more double-cut vias for almost all circuits, except for the circuits struct, primary1, and primary2, for which *TDVI_ours* and *01ILP-SV* have the same numbers of double-cut vias inserted (see the results in the corresponding "#SDVI" columns). Therefore, we can conclude that the 0–1 ILP formulation is more complete than the MBM formulation. Although *01ILP-SV* is slower than *TDVI_ours*, it is still very efficient. Moreover, we can see from column "0–1 ILP" that when we are free to insert redundant vias of different types to the single vias in a stack via, our 0–1 ILP approach can always generate a better solution than *TDVI*, *TDVI_ours*, and *01ILP-SV* (see the results in the corresponding "#DVI" columns). The other detailed results of our 0–1 ILP approach for the DVI problem will be presented in Section V-C.

B. Efficiency Improvement

For each test case, the statistics on the given conflict graph are shown in Table IV, where "$|V|$" gives the number of vertices in the conflict graph (i.e., the number of feasible double-cut vias); "$|E_I|$" and "$|E_X|$" show the numbers of edges $(v_{i,j}, v_{i',j'})$, with $i = i'$ and $i \neq i'$, respectively. We can see that most of the edges are in E_I. By comparing "$|E_I|$" with the column "#A-Vias," as shown in Table II, for most of the test cases, the number of single vias that have at least one feasible double-cut via each is about 26%–78% of the number of edges of E_I. Therefore, using the double-cut constraints instead of the conflict constraints corresponding to the edges of E_I, the number of constraints of our 0–1 ILP can be reduced significantly.

Table V gives the statistics on the reduced graphs and connected components after performing the preselection procedure and connected-component computation. The columns below "DVI" and "DVI/VD" show the statistics of the connected components of each test case for the DVI and DVI/VD problems, respectively. "$|\tilde{V}|$" and "$|\tilde{E}|$," respectively, give the numbers of vertices and edges of the reduced conflict graph $\tilde{G}$. "#CC" shows the number of connected components of $\tilde{G}$. The size of a connected component is measured by the summation of the number of vertices and the number of edges. "NUM" and "%" under "Small-CC," respectively, show the number and the ratio of the connected components with sizes less than 50. Aside from this, "Max-CC" shows the numbers of vertices and edges of the largest connected component, denoted by $|V_{\rm cc}|$ and $|E_{\rm cc}|$, respectively.

Without via density constraints, we can see from Tables IV and V that after performing the preselection, the sizes of all conflict graphs are reduced significantly. Moreover, the sizes of all connected components are very small. However, when via density constraints are considered, the ratio of the small connected components is decreased, and the size of the largest connected component is increased. Although the sizes of the largest connected components are now larger than those in the DVI problem, they are still much smaller than those of the original conflict graphs. On the average, $|V_{\rm cc}|$ and $|E_{\rm cc}|$ are only 0.2% and 0.2% of $|V|$ and $|E|$, respectively. In other words, after performing the preselection and connected-component computation, the DVI/VD problem is decomposed into much smaller subproblems. Therefore, as we shall see in Tables VI and VIII and Section V-D, our 0–1 ILP approach is also very efficient in computing an optimal DVI/VD solution.

TABLE VI
RESULTS OF SPEEDUP METHODS

Case	DVI		DVI/VD	
	Without acceleration	With acceleration	Without acceleration	With acceleration
struct	3.90	0.83	10.82	3.87
primary1	3.47	3.11	5.80	3.44
primary2	7.36	3.21	49.77	5.16
s5378	3.73	3.12	4.82	3.29
s9234	3.47	3.11	5.77	3.63
s13207	5.04	3.18	14.60	4.34
s15850	5.88	3.19	10.11	3.58
s38417	20.44	3.36	69.57	5.39
s38584	47.81	3.46	102.41	5.04
mcc1	3.56	3.01	4.73	3.35
mcc2	11.93	3.30	18.49	3.62
C1	5.30	3.23	10.84	4.02
C2	8.08	3.30	19.45	4.54
C3	92.62	3.81	216.33	6.75
C4	93.44	3.87	178.86	5.70
C5	843.24	5.14	1191.83	9.12
dma_dfm	118.76	3.95	653.75	16.80
dsp1_dfm	386.64	5.32	1163.91	20.45
dsp2_dfm	389.49	4.99	1840.52	33.51
risc1_dfm	1011.40	5.59	3703.73	22.42
risc2_dfm	1055.20	5.98	3072.40	26.26
Normalized	52.80	1	63.57	1

Table VI shows the efficiency improvement of our speedup methods when applied to the 0–1 ILP approaches for the DVI and DVI/VD problems.[3] "With acceleration" shows the CPU time of our 0–1 ILP approach; on the other hand, "Without acceleration" shows the CPU time of our 0–1 ILP approach without speedup methods. We can see that the acceleration methods help to reduce the runtime of our 0–1 ILP approach significantly. For the DVI problem, with the acceleration methods, the solution for risc2_dfm can be generated in 5.98 s. On the other hand, without the acceleration methods, it takes 1055.2 s to generate a similarly optimal solution. For the DVI/VD problem, with the acceleration methods, the solution for risc1_dfm can be generated in 22.42 s. However, without the acceleration methods, it takes 3703.73 s to generate a similarly optimal solution.

C. Comparing With an Existing Method for the DVI Problem

For comparative studies in the DVI problem, we implemented the MIS-based approach [10], called *H2K*. *H2K* solves the MIS problem on a conflict graph in an iterative manner. In each iteration, *H2K* extracts a subset of vertices, and finds an independent set of the subgraph composed of the extracted vertices. When all vertices in the given conflict graph have been considered, *H2K* terminates. We used the qualex-ms [23] as the MIS solver and limited the subgraph extracted at each iteration of *H2K* to consist of at most 225 vertices. The reported CPU time of *H2K* is also measured after the completion of graph construction.

Table VII shows the results of the DVI problem. The columns "H2K" and "0–1 ILP," respectively, show the results generated by *H2K* and our 0–1 ILP approach. "#RVI" gives the number of inserted double-cut vias, and "#D-RVI" indicates the difference between the numbers of double-cut vias inserted by *H2K* and our 0–1 ILP approach. "Rate(%)" gives the insertion rates of double-cut vias. "Time(s)" gives the runtimes, and "Speed-up" gives the runtime improvements of our 0–1 ILP approach over *H2K*.

For the DVI problem, our 0–1 ILP approach is always able to generate a solution in which the number of inserted double-cut vias is the same or larger (i.e., *#RVI* is the same or larger), as compared with *H2K*. In other words, our approach always generated an optimal solution for each test case; however, *H2K* did not. Considering that the problem size that *H2K* handled in each iteration is relatively larger compared with the size of each connected component that our 0–1 ILP approach handled, our 0–1 ILP approach ran much faster than *H2K*, particularly for the second and third sets of test circuits. In particular, for C5, our 0–1 ILP approach is 73.98 times faster than *H2K*. Although the runtimes of our 0–1 ILP approach are slightly longer than those of *H2K* in four test circuits (mcc1, primary1, s5378, and s9234), our approach is still very efficient in these test circuits. It is worth noting that if we apply the two acceleration methods shown in Sections III-C2 and C3 to *H2K*, similar speedup results can be observed. However, considering that the accelerated *H2K*, such as *H2K*, is still a heuristic and has no guarantee of optimality, we do not report its detailed results here.

D. Comparing With an Existing Method for the DVI/VD Problem

For comparative studies in the DVI/VD problem, we adapted the approach proposed in [13] and implemented a two-stage heuristic. First, it uses *H2K* to insert as many double-cut vias as possible. Then, it uses a 0–1 ILP approach proposed

[3]For the second set of test circuits, considering that the conflict graphs and the maximum via density rule used in this paper are different from those in [17], the CPU times shown in Table VI are different from those reported in [17].

TABLE VII
RESULTS OF THE DVI PROBLEM

Case	H2K			0-1 ILP			
	#RVI	Rate(%)	Time(s)	#D-RVI	Rate(%)	Time(s)	Speed-up
struct	7580	99.76	5.74	+0	99.76	0.83	6.96X
primary1	5495	99.26	2.98	+0	99.26	3.11	0.96X
primary2	22877	98.80	12.50	+0	98.80	3.21	3.89X
s5378	6537	97.00	1.76	+1	97.02	3.12	0.56X
s9234	5215	97.20	1.70	+1	97.22	3.11	0.55X
s13207	13594	97.29	4.12	+0	97.29	3.18	1.30X
s15850	16404	96.94	4.69	+2	96.95	3.19	1.47X
s38417	39853	97.34	15.53	+2	97.35	3.36	4.62X
s38584	53802	97.15	23.98	+0	97.15	3.46	6.93X
mcc1	5682	95.53	1.41	+0	95.53	3.01	0.47X
mcc2	32177	93.60	9.55	+2	93.61	3.30	2.90X
C1	19754	80.32	5.25	+0	80.32	3.23	1.63X
C2	31411	76.32	9.40	+0	76.32	3.31	2.85X
C3	98888	77.83	55.39	+0	77.83	3.81	14.55X
C4	111657	73.50	56.85	+0	73.50	3.87	14.69X
C5	275436	77.07	380.35	+1	77.07	5.14	73.98X
dma_dfm	100114	92.36	46.18	+2	92.36	3.95	11.70X
dsp1_dfm	180934	80.94	145.29	+7	80.94	5.32	27.33X
dsp2_dfm	190199	81.77	142.82	+5	81.77	4.99	28.62X
risc1_dfm	292792	85.02	353.28	+4	85.02	5.59	63.25X
risc2_dfm	291759	83.23	347.31	+10	83.23	5.98	58.06X

TABLE VIII
RESULTS OF THE DVI/VD PROBLEM

Case	Two-Stage			0-1 ILP			
	#RVI	Rate(%)	Time(s)	#D-RVI	Rate(%)	Time(s)	Speed-up
struct	7351	96.75	6.52	+30	97.14	3.87	1.68X
primary1	5346	96.57	3.30	+26	97.04	3.44	0.96X
primary2	21885	94.52	13.58	+184	95.31	5.16	2.63X
s5378	6430	95.41	1.92	+66	96.39	3.29	0.58X
s9234	4648	86.64	1.98	+288	92.00	3.63	0.55X
s13207	12361	88.47	4.89	+619	92.90	4.34	1.13X
s15850	16171	95.56	5.06	+144	96.41	3.58	1.41X
s38417	38095	93.05	16.78	+1020	95.54	5.39	3.11X
s38584	52761	95.27	24.14	+656	96.45	5.04	4.79X
mcc1	5328	89.58	1.55	+151	92.11	3.35	0.46X
mcc2	31948	92.94	9.99	+106	93.25	3.62	2.76X
C1	18868	76.72	5.45	+442	78.52	4.02	1.35X
C2	29982	72.85	9.51	+636	74.39	4.54	2.09X
C3	97032	76.37	56.15	+1008	77.16	6.75	8.31X
C4	110575	72.79	57.13	+681	73.24	5.70	10.02X
C5	274206	76.73	382.17	+715	76.93	9.12	41.90X
dma_dfm	95028	87.66	50.20	+1925	89.44	16.80	2.99X
dsp1_dfm	176690	79.04	148.77	+1624	79.76	20.45	7.28X
dsp2_dfm	183278	78.79	150.77	+2444	79.84	33.51	4.50X
risc1_dfm	283855	82.42	359.61	+2883	83.26	22.42	16.04X
risc2_dfm	285319	81.39	351.55	+2182	82.01	26.26	13.39X

in [13] to eliminate redundant vias for satisfying the maximum via density constraint. We refer to this method as *Two-Stage*. Moreover, the reported CPU time of *Two-Stage* is measured after the completion of graph construction.

The results of the DVI/VD problem are shown in Table VIII. The columns "Two-Stage" and "0–1 ILP," respectively, show the results generated by *Two-Stage* and our 0–1 ILP approach. "#RVI" gives the number of inserted double-cut vias, and "#D-RVI" gives the difference between the numbers of double-cut vias inserted by *Two-Stage* and our 0–1 ILP approach. "Rate(%)" gives the insertion rates of double-cut vias. "Times(s)" gives the runtimes, and "Speed-up" gives the runtime improvements of our 0–1 ILP approach over *Two-Stage*.

For the DVI/VD problem, our 0–1 ILP approach is also always able to generate an optimal solution. It can insert more double-cut vias than *Two-Stage* for each test case, with runtimes that are up to 41.9 times faster. Similarly, for the four test circuits mcc1, primary1, s5378, and s9234, the runtimes of our 0–1 ILP approach are slightly longer than those of *Two-Stage*; however, our approach is still very efficient.

VI. CONCLUSION

In this paper, we have studied the DVI problem with and without via density constraints. We have shown that the DVI problems can be formulated as a set of smaller 0–1 ILP problems. Each smaller problem can be solved independently and efficiently without sacrificing the optimality. Although the 0–1 ILP formulation is, in general, intractable, experimental results show that our accelerated approaches are efficient in optimally solving the DVI and DVI/VD problems.

REFERENCES

[1] K. Rajkanan, "Yield analysis methodology for low defectivity wafer fabs," in *Proc. Int. Workshop Memory Technol., Des. Testing*, 2000, p. 65.

[2] N. Harrison, "A simple via duplication tool for yield enhancement," in *Proc. Int. Symp. Defect Fault-Tolerance VLSI Syst.*, 2001, pp. 39–47.

[3] S. Raghvendra and P. Hurat, "DFM: Linking design and manufacturing," in *Proc. Int. Conf. VLSI Des.*, 2005, pp. 705–708.

[4] T. Pompl, C. Schlünder, M. Hommel, H. Nielen, and J. Schneider, "Practical aspects of reliability analysis for IC designs," in *Proc. Conf. Des. Autom.*, 2006, pp. 193–198.

[5] C. Christiansen, B. Li, J. Gill, R. Filippi, and M. Angyal, "Via-depletion electromigration in copper interconnects," *IEEE Trans. Device Mater. Rel.*, vol. 6, no. 2, pp. 163–168, Jun. 2006.

[6] *Reference Flow 5.0 and Reference Flow 6.0*, Taiwan Semicond. Manuf. Company (TSMC).

[7] Y. Zorian, D. Gizopoulos, C. Vandenberg, and P. Magarshack, "Guest editors' introduction: Design for yield and reliability," *IEEE Des. Test Comput.*, vol. 21, no. 3, pp. 177–182, May/Jun. 2004.

[8] D. Abercrombie, "Via doubling can help to stem yield loss," *Electron. Eng. Times*, no. 1392, pp. 72–74, Oct. 10, 2005.

[9] J. Wilson and W. Ng, *Improving Design Robustness With Via Doubling*. Oak Brook, IL: Semicond. Int., Jul. 1, 2006.

[10] K.-Y. Lee and T.-C. Wang, "Post-routing redundant via insertion for yield/reliability improvement," in *Proc. Conf. Asia South Pacific Des. Autom.*, 2006, pp. 303–308.

[11] F. Luo, Y. Jia, and W. W.-M. Dai, "Yield-preferred via insertion based on novel geotopological technology," in *Proc. Conf. Asia South Pacific Des. Autom.*, 2006, pp. 730–735.

[12] H.-Y. Chen, M.-F. Chiang, Y.-W. Chang, L. Chen, and B. Han, "Full-chip routing considering double-via insertion," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 5, pp. 844–857, May 2008.

[13] K.-Y. Lee, T.-C. Wang, and K.-Y. Chao, "Post-routing redundant via insertion and line end extension with via density consideration," in *Proc. Int. Conf. Comput.-Aided Des.*, 2006, pp. 633–640.

[14] G. Xu, L.-D. Huang, D. Z. Pan, and M. D. F. Wong, "Redundant-via enhanced maze routing for yield improvement," in *Proc. Conf. Asia South Pacific Des. Autom.*, 2005, pp. 1148–1151.

[15] H. Yao, Y. Cai, X. Hong, and Q. Zhou, "Improved multilevel routing with redundant via placement for yield and reliability," in *Proc. Great Lakes Symp. VLSI*, 2005, pp. 143–146.

[16] F. Eisenbrand, S. Funke, N. Garg, and J. Könemann, "A combinatorial algorithm for computing a maximum independent set in a t-perfect graph," in *Proc. Symp. Discr. Algorithms*, 2003, pp. 517–522.

[17] K.-Y. Lee, C.-K. Koh, T.-C. Wang, and K.-Y. Chao, "Optimal post-routing redundant via insertion," in *Proc. Int. Symp. Phys. Des.*, 2008, pp. 111–117.

[18] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: MIT Press, 2001.

[19] H. Xiang, K.-Y. Chao, R. Puri, and M. D. F. Wong, "Is your layout density verification exact?: A fast exact algorithm for density calculation," in *Proc. Int. Symp. Phys. Des.*, 2007, pp. 19–26.

[20] H.-Y. Chen, M.-F. Chiang, and Y.-W. Chang, Dec. 2007–Jul. 2008, private communication.

[21] *Cadence Design Systems*. [Online]. Available: http://www.cadence.com

[22] *CPLEX*. [Online]. Available. http://www.ilog.com

[23] *QUALEX-MS Package*. [Online]. Available: http://www.busygin.dp.ua/npc.html

Kuang-Yao Lee received the B.S. degree in computer science and engineering from Yuan Ze University, Chungli, Taiwan, in 2003. He is currently working toward the Ph.D. degree in the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan.

His research interests include manufacturability-aware physical design for VLSI circuits.

Mr. Lee was the recipient of the Best Paper Award at the 2006 Asia and South Pacific Design Automation Conference (ASP-DAC) and the Best Poster Award of the Student Forum at the 2007 ASP-DAC.

Cheng-Kok Koh (S'92–M'98–SM'06) received the B.S. (with first-class honors) and M.S. degrees in computer science from the National University of Singapore, Kent Ridge, Singapore, in 1992 and 1996, respectively, and the Ph.D. degree in computer science from the University of California, Los Angeles (UCLA), in 1998.

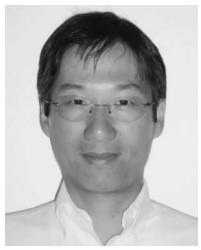
He is currently an Associate Professor of electrical and computer engineering with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. His research interests include physical design of VLSI circuits and modeling and analysis of large-scale systems.

Dr. Koh was the recipient of the Lim Soo Peng Book Prize for Best Computer Science Student from the National University of Singapore in 1990; the Tan Kah Kee Foundation Postgraduate Scholarship in 1993 and 1994; the General Telephone and Electronics Fellowship and the Chorafas Foundation Prize from the UCLA, in 1995 and 1996, respectively; the Association for Computing Machinery (ACM) Special Interest Group on Design Automation (SIGDA) Meritorious Service Award in 1998; the Chicago Alumni Award from Purdue University in 1999; the National Science Foundation CAREER Award in 2000; the ACM/SIGDA Distinguished Service Award in 2002; and the Semiconductor Research Corporation Inventor Recognition Award in 2005.

Ting-Chi Wang received the B.S. degree in computer science and information engineering from National Taiwan University, Taipei, Taiwan, and the M.S. and Ph.D. degrees in computer sciences from University of Texas, Austin.

He is currently an Associate Professor with the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan. His research interests include very large scale integration (VLSI) design automation.

Dr. Wang was the recipient of the Best Paper Award at the 2006 Asia and South Pacific Design Automation Conference (ASP-DAC) for his work on redundant via insertion. He supervised a team to win the first place at the 2008 International Symposium on Physical Design (ISPD) Global Routing Contest. He has served on the technical program committees of several conferences, including ASP-DAC, International Conference on Field Programmable Logic and Applications, International Conference on Field-Programmable Technology, International Conference on Computer-Aided Design, and Workshop on Synthesis and System Integration of Mixed Information Technologies.



Kai-Yuan Chao (M'95–SM'07) received the B.S. degree in nuclear engineering from the National Tsing Hua University, Hsinchu, Taiwan, in 1986, the M.S. degree in medical engineering from the National Yang-Ming Medical College, Taipei, Taiwan, in 1988, and the M.S.E. and Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin in 1992 and 1995, respectively.

He was a Research Assistant with the Nuclear Medicine Department, Veteran General Hospital, Taipei, from 1986 to 1988 and a licensed Medical Physicist Officer with the Radiology Oncology Department, Tri-Service General Hospital, Taipei, from 1988 to 1990. He is currently a Principal Engineer with the Enterprise Microprocessor Group, Intel Corporation, Hillsboro, OR. He has been leading and managing floor plan and assembly design automation for all Pentium 4 processor development projects since 1995. He designed the full-chip management system and repeater netlist-to-DFM flow for Intel's multiple-core modular Core i7 Nehalem/Westmere CPU projects. He has authored more than 40 technical papers and two book chapters in the areas of VLSI/CAD, packaging, and radiology. His current research interests include architecture/design convergence, deep-submicrometer design/manufacture/packaging interface, and design collaboration/management system.

Dr. Chao served on the Technical Program Committee for the 2006-2008 International Symposium on Physical Design.