Clustered Fault Tolerance TSV Planning for 3-D Integrated Circuits

Qi Xu, Song Chen, *Member, IEEE*, Xiaodong Xu, and Bei Yu, *Member, IEEE*

***Abstract*—In 3-D integrated circuits (3-D ICs), through silicon via (TSV) is a critical technique to provide vertical connections. However, the yield and reliability challenge of TSV in industry is one of key obstacles to adopt the 3-D ICs technology. Various fault-tolerance structures by using additional spare TSVs (s-TSVs) to repair faulty functional TSVs (f-TSVs) have been proposed in literature for yield and reliability enhancement. However, these structures are formed in standard cell placement stage where all the f-TSVs are already placed. In reality, since the s-TSVs can be only inserted into the whitespace, the quality of the generated repair solution is strongly dependent on the whitespace distribution. In this paper, we propose an efficient TSV planning and repair framework in floorplanning stage, which takes nonuniform TSV distribution and clustered TSV defect-distribution into account. The proposed framework mainly consists of four stages: 1) a whitespace redistribution algorithm that uses a probability-based strategy to make the whitespace distribution more reasonable for the f-TSV planning. Subsequently, a convex-cost flow-based model for f-TSV allocation considering the fault clustering; 2) a top-down globally partitioning combined with a bottom-up locally merging to partition f-TSVs into groups with minimum hardware cost; 3) the min-cost max-flow algorithm for s-TSV allocation with minimum wirelength overhead; and 4) an integer linear programming-based model to form a fault-tolerance structure with minimum multiplexer delay overhead. The experimental results demonstrate that the proposed repair framework can improve the yield with minimum hardware cost and multiplexer delay overhead.**

***Index Terms*—3-D integrated circuit (3-D IC), fault-tolerance, through silicon via (TSV) planning, TSV repair, yield enhancement.**

I. Introduction

As DEVICE feature sizes continue to rapidly decrease, the interconnect delay is becoming a bottleneck limiting IC performance. 3-D integrated circuits (3-D ICs) technology involves vertically stacking multiple dies connected by through silicon vias (TSVs), providing a promising way to alleviate the interconnect problem and achieve a significant reduction in chip area, wire-length, and interconnect power [1]. Theory indicates that the average wire-length of a 3-D IC varies according to the square root of the number of layers [2]. Moreover, 3-D ICs also offer the potential for heterogeneous integration, which is essential for more than Moore technology [3].

However, yield of TSVs-based 3-D ICs is limited under current manufacturing process. In general, there are two types of yield losses in 3-D ICs due to defects in stacked dies or defects occurred during the assembling process [4]. For the former case, it is critical to conduct prebond testing to avoid the stacking of defective dies [5]. A number of die/wafer matching and interdie repair strategies have also been proposed to increase the stack yield [6]–[8]. In the latter case, adding spare TSVs (s-TSVs) to repair fault functional TSVs (f-TSVs) is an effective method for enhancing yield.

Two different TSV placement styles, namely uniform TSV placement and nonuniform TSV placement, are considered in [9]. For the uniform TSV placement, f-TSVs are placed uniformly, and s-TSVs and multiplexers are inserted after f-TSV planning but prior to the placement of logic cells and detailed routing [10]. Therefore, it is not required to consider whitespace during s-TSV allocation. In contrast, for nonuniform TSV placement, the f-TSVs and logic cells need to be placed simultaneously. In order to avoid significant distortion to the routing of signal nets during inserting s-TSVs, the s-TSVs have to be allocated after placement stage, and the whitespace constraint must be taken into account [11]. Although uniform TSV placement offers advantages in heat dissipation and package bonding, nonuniform TSV placement provides more design flexibility, and lower latency [12]. In this paper, we consider the nonuniform TSV placement style.

A. Previous Work

Several s-TSV allocation mechanisms have been proposed to increase reliability. Nain *et al*. [13] attempted to improve the yield by providing wireless redundant TSVs, and performed Monte Carlo simulation under different TSV defect rates to estimate the chip yield. However, huge extra costs including transmitter and receiver circuits may be introduced to ensure the functionality of the employed wireless redundant TSVs. Zhao *et al*. [14] tried to determine the optimal grouping ratio, i.e., the number of f-TSVs to the number of s-TSVs, in fault-tolerance structures to achieve high chip yield while minimizing hardware costs. Hsieh and Hwang [15] presented a repair mechanism, which partitions f-TSVs into TSV groups and assigns each TSV group with one s-TSV for repairing the faulty link in that TSV group. Jiang *et al*. [16] proposed a TSV redundancy architecture using dedicated switches to handle clustered TSV faults. The proposed technique enables faulty TSVs to be repaired by s-TSVs that are distant rather than

Manuscript received July 22, 2016; revised October 15, 2016 and December 29, 2016; accepted February 9, 2017. Date of publication March 10, 2017; date of current version July 17, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61674133 and Grant 61404123, in part by the Anhui Provincial Natural Science Foundation, China, under Grant 1508085MF134, and in part by the Chinese University of Hong Kong Direct Grant for Research. This paper was recommended by Associate Editor C. C.-N. Chu. *(Corresponding author: Song Chen.)*

Q. Xu, S. Chen, and X. Xu are with the Department of Electronic Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: xuqi@mail.ustc.edu.cn; songch@ustc.edu.cn; xxd0210@mail.ustc.edu.cn).

B. Yu is with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong (e-mail: byu@cse.cuhk.edu.hk).

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.2017.2681080

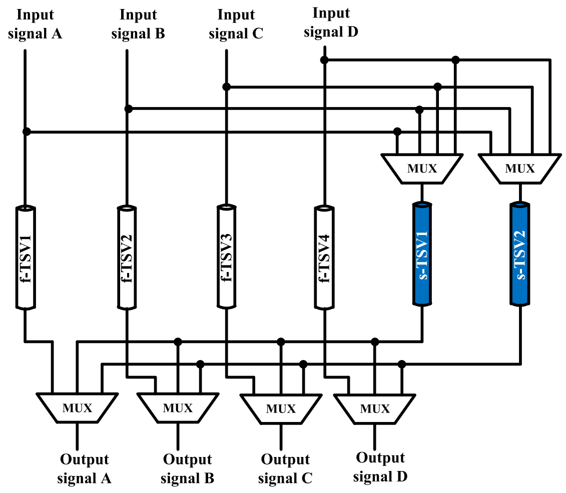


Fig. 1. Fault-tolerance structure with large delay overhead.

by the neighboring s-TSVs, thus being suitable for repairing the clustered TSV faults. Lo *et al*. [17] proposed a ring-based redundant TSV architecture, which places s-TSVs at the edges of the f-TSV grid with multiple rings. Simulation results show that the ring-based architecture can efficiently repair clustered faulty TSVs with low area overhead. However, these methods are only suitable for uniform TSV designs, where TSVs are placed in a regular structure on the die.

Recently there are investigations on s-TSV allocation in nonuniform TSV-based design. Ye and Chakrabarty [18] proposed an integer linear programming (ILP)-based algorithm to minimize the wirelength overhead due to signal rerouting by optimally allocating s-TSVs to f-TSVs. Unfortunately, the chip yield, which is a primary motivation for s-TSV allocation, was not explicitly considered. Chen *et al*. [11] studied an optimal assignment of s-TSVs under yield and timing constraints to minimize the total area overhead, where at most one s-TSV can be assigned into a TSV group. Wang *et al*. [12] presented a fault-tolerance technique that can repair faulty TSVs based on a realistic clustered defect model. It showed that the hardware cost is proportional to the number of f-TSV groups, so a greedy algorithm is first used to partition the f-TSVs into several groups with minimizing the hardware cost. Then an ILP-based algorithm is utilized to determine the exact locations of the inserted s-TSVs to minimize the delay overhead.

However, most existing nonuniform TSV allocation works suffer from one or more of the following drawbacks.

1) Some only allow one s-TSV to be inserted in each TSV group, thus the proposed fault-tolerance structure cannot be repaired in case of more than one faulty f-TSV.
2) Usually a chain structure for a group with n s-TSVs includes multiplexers with at most $(n + 1)$ input ports, which has the smallest delay overhead. Without the consideration of fault-tolerance structure generation, some previous works may suffer from unreasonable TSV locations. For instance, as shown in Fig. 1, without appropriate design, four f-TSVs are partitioned into a group but they cannot transfer each other's signals due to high wirelength cost. Therefore, each f-TSV has to directly connect to an s-TSV, introducing large (4-to-1) multiplexers.
3) The delay overhead by multiplexers, which are used for rerouting signals in the generated fault-tolerance structures, is not considered. Table I shows the relation between delay and the type of multiplexer, which is estimated based on an industry 40-nm library. If fault-tolerance structure is not well designed, the input port number of a multiplexer could be the f-TSV number in the group, which introduces large delay overhead.

TABLE I
RELATION BETWEEN DELAY AND THE INPUT PORT NUMBER OF MULTIPLEXER

Type	Delay (ns)	Area (um^2)
MUX2_1	0.0557	4.7628
MUX3_1	0.1152	14.8176
MUX4_1	0.1338	5.8212
MUX5_1	0.2199	12.1716
MUX6_1	0.2205	17.1108
MUX7_1	0.2253	21.5208
MUX8_1	0.2253	25.2252

From previous work, we notice that all fault-tolerance structures are formed in standard cell placement stage when all the f-TSVs are already placed. In reality, since the s-TSVs can be only inserted into the whitespace, the quality of the generated repair solution is strongly dependent on the whitespace distribution. It should be noted that although some previous f-TSV planning methods (e.g., [19]–[22]) can be extended to allocate f-TSVs, the extension may not be trivial and purely f-TSV planning itself is hard to achieve reasonable fault tolerance. That is, with an inappropriate whitespace and f-TSV distribution, we might be fail to generate fault tolerance structure thus the chip yield could be greatly impacted. To construct an available TSV repairable structure, we should make the whitespace distribution in the generated 3-D floorplan more reasonable for the subsequent TSV insertion.

B. Our Contributions

Motivated by the above argument, in this paper we propose an effective TSV planning and repair framework, which is flexible to repair clustered TSV faults with minimum hardware cost and delay overhead. Our framework is aware of the yield constraint under the defect clustering. Some key technical contributions of this paper are listed as follows.

1) f-TSV Planning: f-TSVs are inserted layer by layer through solving a set of convex-cost flow problems. The convex function associated with the whitespace distribution makes the allocation of the f-TSVs scattered to reduce the TSV fault clustering effect and can potentially reserve some whitespace for s-TSVs. Before f-TSVs are inserted, to make the whitespace distribution in the given floorplan more balanced for the TSV planning, we iteratively insert whitespace blocks into the region that contains the maximum probability number of f-TSVs, estimated using a probabilistic analysis, until the maximum probability number is smaller than 1 all over the chip. A partitioned sequence pair (P-SP) [23] is used to represent multilayer 3-D floorplans. The relative positions of the blocks will be kept unchanged and the fixed-outline constraint will not be violated during the whitespace redistribution.

2) s-TSV Planning: First, to minimize the hardware costs, we propose to use a top-down partitioning followed by a bottom-up merging (clustering) to form f-TSV groups with enough s-TSV candidates, considering the yield constraint under the TSV defect clustering. The f-TSVs in the same group will share same s-TSVs for the fault-tolerance structure. In the top-down stage, we are able to globally partition the f-TSVs into groups by recursive min-cut bi-partitioning algorithm until the chip yield is higher than the target yield. In the bottom-up merging stage, we locally merge the available two f-TSV groups with the highest yield to reduce the group number under the target yield constraints. After the f-TSV grouping, f-TSVs are already partitioned into groups and the f-TSVs in each group can have a reasonable number of common s-TSV candidates for constructing fault-tolerance structure under the target yield constraint. Then an efficient min-cost max-flow (MCMF)-based method is used to assign s-TSV candidates to f-TSV groups with minimization of the wirelength costs.

3) Fault-Tolerance Structure Construction: To generate n-fault tolerance structures that use n s-TSVs for a group of f-TSVs, we need to find n independent shifting paths from each f-TSV to n s-TSVs. Because the multiplexers are used to reroute signals, the delay of a multiplexer is increased along with the number of input ports. Therefore, minimizing the delay overhead of multiplexers is equivalent to minimizing the maximum input number of multiplexers. In this paper, we present an ILP formulation to generate n-fault tolerance structures minimizing the maximum input number of all multiplexers. To the best of our knowledge, this is the first work for generating n-fault ($n > 1$) tolerance structures that uses n s-TSVs for a group of f-TSVs, considering the delay overhead of multiplexers. The experimental results show that the proposed ILP formulation for generating n-fault tolerance structures can effectively minimize the maximum input number of multiplexers.

The remainder of this paper is organized as follows. Section II formulates the TSV yield problem and presents the background and motivation of this paper. Section III describes the overview of the proposed framework. The corresponding proposed f-TSV and s-TSV planning algorithms are described in detail in Sections IV and V. Section VI presents an ILP formulation for generating the fault tolerance structures. The experimental results are provided in Section VII, followed by the conclusions in Section VIII.

II. Preliminaries and Problem Formulation

A. Chip Yield and TSV Yield

According to the cumulative yield property, the yield of a 3-D chip Y_{chip} can be formulated as follows [4]:

$$Y_{\text{chip}} = Y_{\text{stack}} \cdot \prod_{i=1}^{t-1} Y_{\text{bonding}(i)} \cdot \prod_{i=1}^{t-1} Y_{\text{TSV}(i)} \tag{1}$$

where t is the number of device layers in the 3-D chip, Y_{stack} is the stacking yield, $Y_{\text{bonding}(i)}$ is the yield of the ith bonding step, and $Y_{\text{TSV}(i)}$ denotes the TSV yield in the i^{th} layer. In this paper, we focus on the yield enhancement of 3-D chip in terms of overall TSV yield $Y_{\text{TSV}(i)}$ [12]. The TSV yield $Y_{\text{TSV}(i)}$ calculation is shown in Section V-A.

B. TSV Fault-Tolerance Structure

In this paper, we focus on an effective fault-tolerance TSV planning, where a set of f-TSVs are partitioned into several groups, and one or multiple s-TSV(s) are assigned to each group to provide redundancy. By inserting the control circuit (i.e., multiplexers) and carefully designing the reconfigurable routing paths, the s-TSVs can be used to transfer signals in the presence of faulty f-TSVs [24]. Fig. 2 shows one such structure with four f-TSVs and two s-TSVs, which is constructed by our algorithm, and in this structure we could repair the group in case of at most two faulty f-TSVs through multiplexer rerouting. When all f-TSVs are fault-free, each signal passes through their corresponding f-TSV. Once two f-TSVs are faulty, the reconfigurable routing paths are shown in Fig. 2.

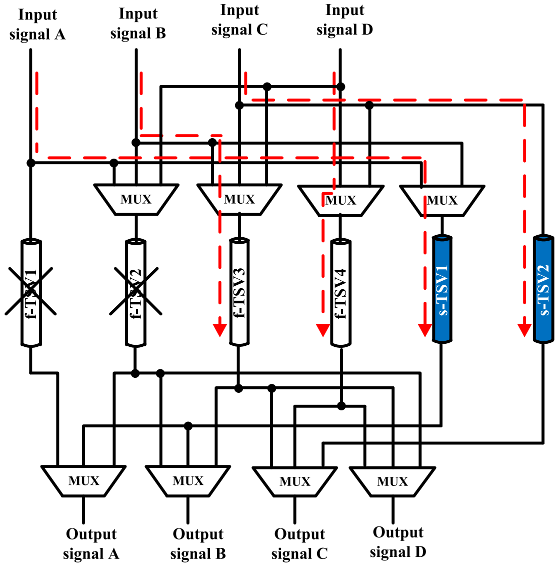


Fig. 2. Fault-tolerance structure and the signal routing when f-TSVs 1 and 2 fail.

C. Problem Formulation

Based on the above definitions, the problem formulation of *clustered fault-tolerance TSV planning under yield constraints* is as follows.

Input: 1) A 3-D IC floorplan result where the relative positions of blocks have been determined; 2) target yield of the chip; and 3) defect model parameters.

Constraints: 1) All blocks are packed into the specified region (fixed outline); 2) two blocks within the same layer do not overlap; 3) the generated s-TSV allocation solution should satisfy the target yield; and 4) f-TSVs and s-TSVs should only be inserted into the whitespace surrounding the placed blocks.

Output: A TSV fault-tolerance structure, including: 1) which f-TSVs are assigned to the same s-TSV(s); 2) the exact positions of the allocated s-TSVs; and 3) the connection relation of TSVs.

Objective: Minimize the hardware cost and delay overhead induced by the fault-tolerance structure.

III. Overview of the Proposed Framework

Fig. 3 illustrates the overall flow of the proposed TSV planning framework, which mainly consists of five stages: 1) whitespace redistribution; 2) f-TSV insertion; 3) f-TSV grouping; 4) s-TSV allocation; and 5) fault-tolerance structure construction.

First, given a 3-D floorplan, to make the whitespace distribution in the given floorplan more balanced for the TSV planning, we iteratively insert whitespace blocks into the region that contains the maximum probability number of f-TSVs, estimated using a probabilistic analysis, until the maximum probability number is smaller than 1 all over the chip, which potentially alleviates the dense distribution of the f-TSVs in some local regions. In this stage, the P-SP [23] is used to represent multilayer 3-D floorplans. The relative

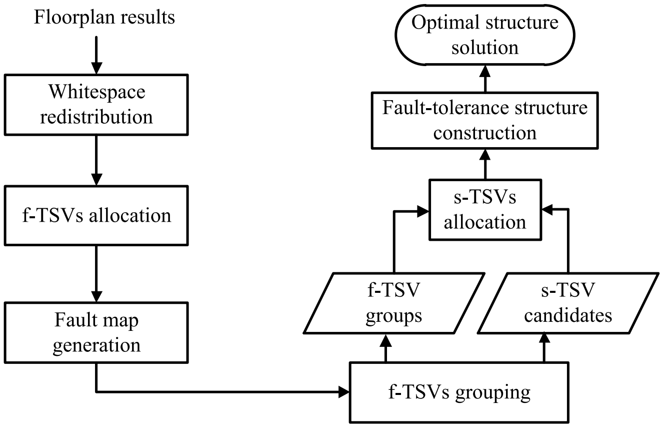


Fig. 3. Overall flow of the proposed framework.

positions of the blocks will be kept unchanged and the fixed-outline constraint will not be violated during the whitespace redistribution.

Second, the f-TSVs are inserted layer by layer through solving a set of convex-cost flow problems. The convex edge cost function associated with the whitespace distribution makes the allocation of the f-TSVs scattered to reduce the TSV fault clustering effect and can potentially reserve some whitespaces around f-TSVs for s-TSVs.

Third, to minimize the hardware costs, we use a top-down partitioning followed by a bottom-up clustering to form f-TSV groups with enough s-TSV candidates, considering the yield constraint under the TSV defect clustering. Moreover, considering the delay overhead introduced by the multiplexers, we also take into account the replaceable relations between f-TSVs during the grouping for potentially reducing the maximum size of the multiplexers. The f-TSVs in the same group will share same s-TSVs for the fault-tolerance structure. In the top-down stage, we are able to globally partition the f-TSVs into groups by recursive min-cut bi-partitioning algorithm until the chip yield is higher than the target yield. In the bottom-up merging stage, we locally merge the available two f-TSV groups with the highest yield to reduce the group number until the chip yield is decreased to the target yield. Compared with previous grouping method in [12], which includes only a bottom-up clustering stage, the proposed group method has a more global view and potentially is able to generate f-TSV groups with lower hardware costs.

Fourth, after the f-TSVs are grouped, an efficient MCMF-based method is used to assign fixed number of s-TSVs to all the f-TSV groups, simultaneously, with minimization of the wirelength costs.

Finally, we propose an ILP-based method to generate n-fault tolerance structures for each group by finding independent signal shifting paths from each f-TSV to n s-TSVs and the objective is to minimize the delay overhead introduced by the multiplexers for rerouting signals, i.e., the maximum size of the multiplexers.

IV. Functional TSV Planning

In this stage, a whitespace redistribution algorithm that uses a probability-based strategy is first performed to make the whitespace distribution more reasonable for the subsequent f-TSV planning. Then, f-TSVs are assigned by solving a set of convex-cost flow problems layer by layer. The convex edge cost function associated with the whitespace distribution is proposed to guide a scattered allocation of f-TSVs to reduce the TSV fault clustering effect and reserve some whitespaces around each f-TSV.

A. Whitespace Redistribution

We use P-SP extended from the well-studied 2-D floorplan representation sequence pair [23] to represent 3-D floorplan. During the chip-level floorplanning, we adopt an efficient fixed-outline multilayer floorplanner called insertion-after-remove multi-layer floorplanning (IAR-MLFP) [25], [26] to determine block positions while minimizing the wirelength and the number of f-TSVs. Then a whitespace redistribution algorithm is performed based on P-SP representation, which can make the whitespace distribution in the above generated floorplan be aware of the following f-TSV planning and s-TSV allocation. In our whitespace redistribution, we first compute candidate positions of each f-TSV, and then iteratively insert a whitespace block into a position which contains the maximum number of candidate f-TSVs. Therefore, the whitespace redistribution method can avoid a dense TSV distribution in some positions, and the TSV fault clustering effect would be also reduced. The fixed-outline constraint will not be violated during the whitespace insertion.

On all device layers, an even grid structure is used, whose size P×Q is determined by a specified individual grid size. Given a multilayer floorplan result, we compute the whitespace distribution on each layer by calculating the amount of whitespace in each grid g, denoted as $ws(g)$. Let A_v be the area of a TSV. The capacity cap(g) of a grid g, i.e., the number of TSVs that can be located at g, is defined as $\lfloor ws(g)/A_v \rfloor$. Therefore, the whitespace distribution is related to the candidate positions of each f-TSV. Here we use a probability-based heuristic for computing candidate positions of each f-TSV [27].

First, given an f-TSV, we set all the grids with nonzero capacity that are covered by the bounding box of the corresponding net as the initial candidate positions of the f-TSV. The bounding box of a net is a rectangular region encapsulating all the pins of the net and can be defined by the maximum and minimum $x-/y-$coordinates of the net pins. If a net spans multiple device layers, we project all the net pins onto one device layer and calculate the bounding box.

Second, for an f-TSV v of a net nt, let CP(v) be the set of all the grids with nonzero capacity that are covered by the bounding box of nt. And for a grid with nonzero capacity, CV(g) be the set of f-TSVs of the nets whose bounding box cover g, that is, $\mathrm{CV}(g) = \{v|\mathrm{CP}(v) \text{ includes } g\}$. We calculate the expected position number of v denoted as epn(v), and the expected via number in g denoted as evn(g), as follows:

$$\mathrm{epn}(v) = \sum_{g \in \mathrm{CP}(v)} \frac{\mathrm{cap}(g)}{|\mathrm{CV}(g)|}, \quad \mathrm{evn}(g) = \sum_{v \in \mathrm{CV}(g)} \frac{1}{|\mathrm{CP}(v)|}. \tag{2}$$

If there is an f-TSV v with $\mathrm{epn}(v) < 1$, the CP(v) will be extended by iteratively extending the net bounding box one unit in each direction until $\mathrm{epn}(v) \geq 1$. The evn(g) can be assumed as the congestion coefficient for each grid g. And CV(g) is also updated.

A whitespace is assumed as a virtual block with the same size as a TSV during the insertion. After inserting the whitespace block into the floorplan, if the new chip area does not violate the fixed outline constraint, this insertion point will be assumed as a candidate position of whitespace block insertion. According to P-SP, we calculate the position of inserted whitespace block and find the corresponding grid g in grid structure. The insertion point is evaluated by the congestion coefficient evn(g) of the corresponding grid g which is calculated by (2). Given a P-SP for t layers and n_i (>0) blocks on layer i, the number of insertion points for the whitespace block equals to $\sum_{i=1}^{t} n_i^2$ [25]. After evaluating all insertion points, the candidate positions are sorted and the one with the highest congestion coefficient will be selected as the insertion point. Then we insert the whitespace block into P-SP and update P-SP to set new positions for all blocks. In our experiments, the algorithm will be iterated until the highest congestion coefficient is smaller than 1 or the iteration number is larger than 1200. Therefore, the whitespace distribution is more reasonable for the following f-TSVs insertion and s-TSVs allocation.

B. f-TSV Allocation

As shown in [19], the f-TSV allocation problem for 3-D IC is $\mathcal{NP}$-complete. Although there are many previous f-TSV allocation studies, to consider the clustered TSV defect, a modified f-TSV planning is necessary. According to (4), the presence of a TSV fault increases the probability of more defective TSVs in close vicinity. Hence, the scattered distribution of the TSVs will increase the probability to reduce TSV fault clustering effect. In addition, an appropriate whitespace distribution could make the TSV distribution more reasonable and ensure that there are enough whitespaces around f-TSVs for s-TSV planning. Inspired by Liu *et al.* [19] and Chen *et al.* [28], in this paper, f-TSV positions are determined by solving a convex-cost flow problem, in which the TSV congestion is considered [29]. One-layer assignment algorithm finds the optimal solution for two-layer chips. If there are more than two device layers in a chip, we can assign f-TSVs layer by layer by applying a set of one-layer assignments. One-layer assignment algorithm is as follows.

To assign f-TSVs into grids, a directed graph $G_f(V_f, E_f)$ is constructed. Vertex set V_f contains four portions $V_f = s \cup V_{f1} \cup V_{f2} \cup t$, where s is the start vertex, V_{f1} is the set of all f-TSVs, V_{f2} is the set of grids with nonzero capacity, and t is the end vertex. Besides, edge set $E_f = \{s \to V_{f1}\} \cup \{v_i \to g_j | v_i \in V_{f1} \wedge g_j \in V_{f2}$ is a candidate position of $v_i\} \cup \{V_{f2} \to t\}$. We define all the edge *capacities* as follows: the capacity of one edge from V_{f2} to node t is the capacity of the corresponding grid cap(g); while the capacities of all other edges are set to 1. We define all the edge *costs* as follows:

$$ec_f(v, w) = \begin{cases} 0, & \forall (v, w) \in \left\{s \to V_{f1}\right\} \\ wl\left(\mathrm{snt}_0^k\right) + wl\left(\mathrm{snt}_1^k\right), & \forall (v, w) \in \left\{V_{f1} \to V_{f2}\right\} \\ & \text{if } v \text{ is TSV of net } \mathrm{nt}_k \\ f_{pl}(x_{vw}), & \forall (v, w) \in \left\{V_{f2} \to t\right\} \\ & \text{if } x_{vw} \text{ is flow on } (v, w) \end{cases} \tag{3}$$

where $wl(\mathrm{snt}_i^k)$ is the half-perimeter wirelength (HPWL) of net snt_i^k, the subnet of net nt_k, on device layer i. In order to get HPWL estimation, we adopt the method in [28] to decompose the nets spanning multiple device layers into subnets, one on each device layer, by introducing dummy pins corresponding to TSVs.

$f_{pl}(x)$ is piecewise linear and convex. Let $0 = d_{vw}^0 < d_{vw}^1 \cdots < d_{vw}^j = \mathrm{cap}(g)$ denote the breakpoints of the function and the cost varies linearly in the interval $[d_{vw}^{i-1}, d_{vw}^i]$. Let c_{vw}^i denote the linear cost coefficient in the interval $[d_{vw}^{i-1}, d_{vw}^i]$. In our experiment, we set $f_{pl}(x) = 10 \cdot e^{x/\mathrm{cap}(g)}$ and the interval between adjacent breakpoints to 1. This $f_{pl}(x)$ function would allocate f-TSVs scattered to reduce the TSV fault clustering effect and reserve some whitespaces around each f-TSV. In the experiment, by using this $f_{pl}(x)$ function, the s-TSV allocation success rate reaches 100%. Fig. 4 shows an example of convex-cost flow model. Note that all of the edges in the network have a capacity and a cost.

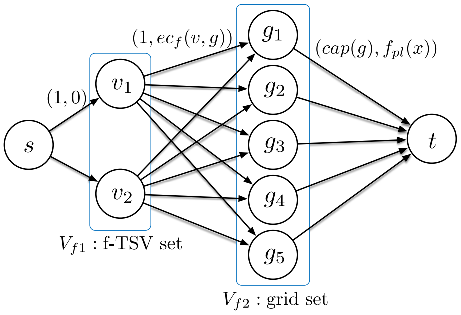


Fig. 4. Example of convex-cost flow network.

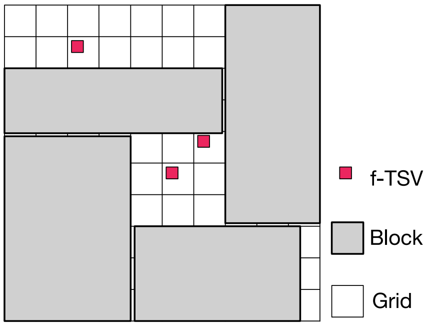


Fig. 5. Example of f-TSV allocation layout result.

Such a convex-cost flow problem can be easily transformed into a traditional min-cost flow problem by replacing an edge, whose edge cost function is piecewise-linear and convex, by a set of edges [30]. The layout example of f-TSV allocation is shown in Fig. 5.

After determining f-TSVs position, we calculate current whitespace distribution as all s-TSVs can only be placed into the whitespace.

V. Spare TSV Planning

In this section, we first introduce two TSV defect-distribution models and present the calculation method of defect probability of TSVs under the clustered defect-distribution model. Then, a top-down and bottom-up partitioning strategy is proposed to cluster f-TSVs into several groups under the target yield constraint. During f-TSV grouping, we ensure each f-TSV group can be assigned with a reasonable number of s-TSV candidates. Finally, the MCMF-based method is used to assign s-TSV candidates to f-TSV groups with minimal wirelength cost.

A. Fault-Map Generation

TSV defect-distribution models can be divided into two types, namely uniform defect-distribution and clustered defect-distribution. For the uniform defect-distribution model, each TSV fails independently. This assumption is valid for certain random defects such as void formation [31] and lamination due to thermal induced stress [32]. However, many types of TSV defects appear during the imperfect bonding process. The bond surface roughness, the height variation of the TSVs, and cleanliness of dies also impact the bonding process [12], [16]. Therefore, it is likely that the presence of a TSV fault increases the probability of more defective TSVs in close vicinity; that is called clustered defect-distribution model [10], [14]. In this paper, we take the clustered defect-distribution into account.

In the clustered defect-distribution model, the presence of a TSV fault increases the probability of more defective TSVs in close vicinity. Approximately, this defect probability is inversely proportional to distance from existing defects (regarded as cluster centers) [33]. If there are already N_c cluster centers, the defect probability of TSV$_i$, P_i, can be expressed as [12]

$$P_i = p \cdot \left(1 + \sum_{j=1}^{N_c} \left(\frac{1}{d_{ij}}\right)^{\alpha}\right) \tag{4}$$

where p is the single TSV failure rate, and d_{ij} is the distance between TSV$_i$ and the jth cluster center. α is the clustering coefficient indicating clustering extent, i.e., a larger α implies higher clustering.

In this paper, we take the clustered defect-distribution into account. To find the defect-cluster centers, the compound Poisson distribution is widely accepted [34], where cluster center numbers follow Poisson distribution and the distribution of defect density is presented by a Gamma function. After determining the cluster centers among f-TSVs and s-TSVs, the defect probability is calculated by (4). If an s-TSV is identified as a cluster center, it will not be chosen to repair faulty f-TSV due to the high defect probability.

B. f-TSV Grouping

In this stage, f-TSVs are divided into groups under the chip yield constraint. The work in [12] pointed that the hardware cost induced by a fault-tolerance solution is related to the number of f-TSV groups. In this paper, we first use the recursive min-cut bi-partitioning algorithm to globally partition the f-TSVs into groups until the chip yield is higher than the target yield (*partitioning* step). In order to further minimize the hardware cost, we then locally merge the two available f-TSV groups with the highest yield together to reduce the group numbers until the chip yield is reduced to target yield (*merging* step). During the *partitioning* and *merging*, we ensure each f-TSV group can share a reasonable number of s-TSV candidates ($\geq N_{gs}$).

Y_{TSV} under clustering defects is calculated by multiplying all f-TSV groups yield Y_{gk}, Y_{TSV} as follows:

$$Y_{\text{TSV}} = \prod_{k=1}^{N} Y_{gk} \tag{5}$$

where N is the number of f-TSV groups. The algorithm in [12] is adopted to calculate group yield Y_{gk}.

The *partitioning* is based on the complete graph of all the f-TSVs, where any two different f-TSVs i, j form an edge e_{ij}. The weight w_{ij} of edge e_{ij} is defined as

$$\begin{aligned} w_{ij} = \rho \cdot \frac{\text{Yield}_{ij}}{\text{MaxYield}} + \beta \cdot \frac{\#\text{CandStsv}_{ij}}{\#\text{MaxCandStsv}} \\ + (1 - \rho - \beta) \cdot \frac{\text{RRCost}_{ij}}{\text{MaxRRCost}} \end{aligned} \tag{6}$$

where the coefficients ρ, β are experience parameters. In the experiment, ρ and β are set to 0.2 and 0.3, respectively. Yield$_{ij}$ is the yield of group which only contains two f-TSVs i and j. Since the s-TSVs for each group will be assigned in the next s-TSV allocation stage, during the *partitioning* step, all the candidate s-TSVs are assumed to have the maximum defect probability among all the candidate s-TSVs except for those cluster centers. #CandStsv$_{ij}$ denotes the number of common candidate s-TSVs between f-TSVs i and j. The term “candidate s-TSVs” means that, for f-TSV i, if the s-TSV s_k is covered by the bounding box of the corresponding net, then s_k is identified as the candidate s-TSV for f-TSV i, and the wirelength overhead is zero when we replace i by s-TSV s_k. In order to ensure that there are enough candidate s-TSVs for each f-TSV group, in the experiment, if the number of candidate s-TSVs for an f-TSV is less than $2N_{gs}$, we iteratively extend the net bounding box one unit in each direction until the number is more than $2N_{gs}$. MaxYield and #MaxCandStsv, respectively, denote the maximum yield and maximum common candidate s-TSV numbers over all the edges.

RRCost$_{ij}$ denotes the replaceable relation cost between the f-TSVs. For f-TSVs i and j, if f-TSV i is covered by the bounding box of the corresponding net of f-TSV j, that is, i can transfer signal for j, we say that j can be replaced by i, and i and j have a replaceable relation. In order to decrease the delay overhead, we tend to place two f-TSVs that can transfer each other’s signals into the same group and RRCost$_{ij}$ denotes the replaceable relation cost. In the f-TSV grouping stage, those f-TSVs that have replaceable relations tend to be assigned to the same group. Because, for a group with m f-TSVs, the summation of the port number of all the multiplexers in the N_{gs}-fault tolerance structure is at most $m \times (N_{gs}+1)$. Considering the delay, in the worst-case scenario, the fault-tolerance structure includes a full connection, constructed by N_{gs} m-port multiplexers, between the set of f-TSVs and the set of s-TSVs. Fig. 1 shows an example. By making use of the replaceable relations between f-TSVs in a group, we can build proper connections between f-TSVs to replace part of the direct connections from the f-TSVs to the s-TSVs, for reducing the input number of the individual multiplexers. In the experiment, for f-TSVs i and j, if f-TSV i is covered by the bounding box of the corresponding net of f-TSV j, meanwhile f-TSV j is also covered by the bounding box of the net of f-TSV i, then RRCost$_{ij}$ is set to 10; if f-TSV i is covered by the bounding box of the corresponding net of f-TSV j but f-TSV j is not covered by the bounding box of the net of f-TSV i, then RRCost$_{ij}$ is set to 5; otherwise the RRCost$_{ij}$ is set to 0. The values of RRCost$_{ij}$ are chosen by the experimental results in Section VII-A.

According to (6), those f-TSVs with higher yield, having replaceable relations, and sharing more common candidate s-TSVs tend to be assigned to the same group. The *partitioning* step continues until the chip yield is higher than the target yield and each partitioned f-TSV group have a reasonable number of common candidate s-TSVs ($\geq N_{gs}$).

In the *merging* step, in order to minimize the hardware cost, we merge the two available f-TSV groups with the highest yield together in each step to reduce the group numbers by 1 until the chip yield is reduced to target yield. The term “the two available groups” means that the number of common candidate s-TSVs for the merged groups should at least equals to N_{gs}. During the *merging* step, all the assigned candidate s-TSVs for a group are assumed to have the maximum defect probability among all s-TSV candidates for the group, which have been determined in the *partitioning* step. In our *merging* step, a greedy algorithm [12] is used to reduce f-TSV group numbers.

C. s-TSV Allocation

With the f-TSV grouping stage, f-TSVs are already partitioned into groups and the f-TSVs in each group can have a reasonable number of common s-TSV candidates ($\geq N_{gs}$) for constructing fault-tolerance structure under the target yield constraint. The s-TSV allocation assigns N_{gs} s-TSVs for each f-TSV group from the s-TSV candidates and the objective is to minimize the wirelength induced by the possible connections between the f-TSVs and the s-TSVs. The s-TSVs cannot be shared between f-TSV groups and are assigned, layer by layer, as the same to the following one-layer algorithm. The similar problem is formulated as time-consuming ILP problems in [12] and [18]. In this paper, we proposed an efficient MCMF-based method for allocating s-TSVs.

In this paper, we formulate the s-TSV allocation into MCMF problem. The objective of s-TSV allocation is to minimize the total wirelength induced by the possible connections between the f-TSVs and the s-TSVs. In [12], the s-TSV allocation is formulated as an ILP problem. And the objective is to minimize the maximum delay overhead incurred after TSV repair by replacing f-TSVs by s-TSVs. In order to observe the effect of s-TSV allocation, we compare chip yield, runtime, the total incremental wirelength and maximum incremental wirelength incurred by the fault-tolerance structure of the two methods. Table VIII shows the experimental results. It can be noticed that compared with ILP, the proposed MCMF can achieve almost same chip yield and maximum incremental wirelength. However, the total incremental wirelength and the runtime of the MCMF are quite better than the ILP.

To represent all the possible assignments from s-TSVs to groups, a directed graph $G_s(V_s, E_s)$ is constructed, which contains four sets of vertices $V_s = s \cup V_{s1} \cup V_{s2} \cup t$, where s is the start vertex, V_{s1} is the s-TSVs set, V_{s2} is the group set, and t is the end vertex. Besides, Edge set $E_s = \{s \rightarrow V_{s1}\}$ $\cup$ $\{s_i \rightarrow r_j | r_j \in V_{s2} \wedge s_i \in V_{s1}$ is an s-TSV candidate for $r_j\}$ $\cup \{V_{s2} \rightarrow t\}$. We define all the edge *capacities* as follows: the capacity of one edge from V_{s2} to node t equals to N_{gs}; while the capacities of all other edges are set to 1. We define all the edge *costs* as follows:

$$ec_s(v, w) = \begin{cases} 0, \forall(v, w) \in \{s \rightarrow V_{s1}\} \cup \{V_{s2} \rightarrow t\} \\ \sum_{i=1}^{N_f} \left[wl\left(\text{snt}_0^k\right) + wl\left(\text{snt}_1^k\right)\right], \forall(v, w) \in \{V_{s1} \rightarrow V_{s2}\} \\ \text{if the fTSV } i \text{ of net nt}_k \text{ is replaced by sTSV } v \end{cases} \tag{7}$$

where N_f is the number of f-TSV in group w, $wl(\text{snt}_j^k)$ is the HPWL of net when the f-TSV i on the net snt_j^k is replaced by the s-TSV v. Since each TSV impact two subnets, when an f-TSV is replaced by an s-TSV, we should consider the wirelength overhead on two subnets. Therefore, we tend to choose an s-TSV which resulting less wirelength overhead incurred after replacing f-TSVs by the s-TSV.

VI. Fault-Tolerance Structure Construction

After s-TSV planning, TSV groups are generated and each group is associated with multiple f-TSVs and one or multiple s-TSVs. This section further discusses in each group how f-TSVs are connected to s-TSVs to generate a fault tolerance structure. As shown in Table I, a multiplexer is used to reroute signals, thus its delay is increased along with the number of input ports. Therefore, in the fault tolerance structure construction, the objective is to minimize delay overhead occurred by multiplexer.

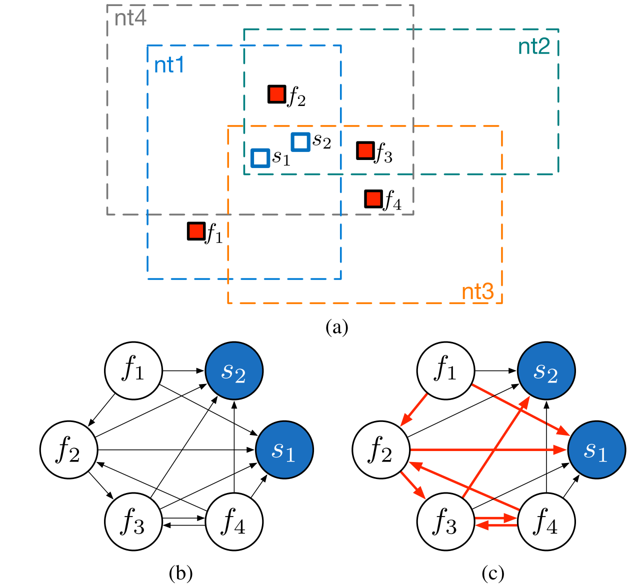


Fig. 6. (a) Layout example of four f-TSVs and two s-TSVs. (b) Corresponding directed graph G. (c) Two-fault tolerance structure.

In [11], a minimum spanning tree (MST)-based method is developed to search for 1-fault tolerance structure. However, in each group one and at most one s-TSV can be considered thus the proposed method is hard to be extended to general cases with multiple s-TSVs. In [12], a chain structure is developed to build up n-fault tolerance structure, where n is the s-TSV number in the group. Since each multiplexer is bounded with at most $n + 1$ input ports, the proposed chain structure can achieve the smallest delay overhead. However, the chain structures cannot always be found due to unreasonable TSV locations. In this paper, to overcome all the above limitations, we consider a general and practical fault tolerance structure minimizing multiplexer delay overhead. To the best of our knowledge, this is the first work generating n-fault ($n > 1$) tolerance structures meanwhile considering multiplexer delay overhead.

A. Graph Construction

Given a TSV group with m f-TSVs and n s-TSVs, we generate a directed graph $G(V, E)$. The vertex set $V = V_1 \cup V_2$, where $V_1 = \{f_i | i = 1, \ldots, m\}$ is the f-TSVs set and $V_2 = \{s_i | i = 1, \ldots, n\}$ is the s-TSVs set. Besides, the edge set $E = \{(u, v) | u \in V_1 \wedge v \in V \wedge u$ can be replaced by $v\}$. Here we say f-TSV u can be replaced by TSV v if and only if v is covered by the bounding box of the net that corresponds to u, thus replacing u by v has no wirelength overhead. Note here TSV v can be either f-TSV or s-TSV.

Fig. 6(a) gives a layout example of four f-TSVs and two s-TSVs. There are four nets nt1, nt2, nt3, and nt4, and the bounding boxes of different nets are shown in rectangles with dashed lines. The corresponding directed graph $G(V, E)$ is shown in Fig. 6(b). From Fig. 6(a), we can see that f_2 is covered by the bounding box of nt_1, thus f_1 can be replaced by f_2 without extra wirelength overhead. Therefore, there exists a directed edge from f_1 to f_2. Similarly, the directed edges from f_2 to f_3, f_3 to f_4, f_4 to f_2, and f_4 to f_3 are generated for

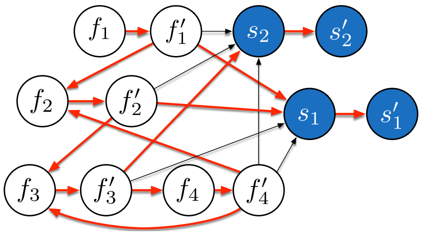


Fig. 7. Corresponding splitting graph G', where two edge-disjoint paths begin with the split f-TSVs, f_1': $\{f_1' \to s_1 \to s_1'\}$ and $\{f_1' \to f_2 \to f_2' \to f_3 \to f_3' \to s_2 \to s_2'\}$, f_2': $\{f_2' \to s_1 \to s_1'\}$ and $\{f_2' \to f_3 \to f_3' \to s_2 \to s_2'\}$, f_3': $\{f_3' \to f_4 \to f_4' \to f_2 \to f_2' \to s_1 \to s_1'\}$ and $\{f_3' \to s_2 \to s_2'\}$, and f_4': $\{f_4' \to f_2 \to f_2' \to s_1 \to s_1'\}$ and $\{f_4' \to f_3 \to f_3' \to s_2 \to s_2'\}$.

the same reason. Since s-TSVs s_1 and s_2 are located in the overlap region of the bounding box of the four nets, thus all four f-TSVs can be replaced by s-TSVs. Therefore, there is a directed edge from each f-TSV node to each s-TSV node in the graph.

In a TSV group with n s-TSVs, we build up n-fault tolerance structure, where each f-TSV has n paths connecting to n s-TSVs. For each f-TSV, the paths are *node-disjoint* except for the source node, so that each fault in f-TSV can only affect one path. Fig. 6(c) illustrates the output of 2-fault tolerance structure, given the layout in Fig. 6(a). The node-disjoint paths for each f-TSV are as follows, where the maximum multiplexer input port number is 2:

$$\begin{aligned}
&f_1\colon \{f_1 \to s_1\},\ \{f_1 \to f_2 \to f_3 \to s_2\}\\
&f_2\colon \{f_2 \to s_1\},\ \{f_2 \to f_3 \to s_2\}\\
&f_3\colon \{f_3 \to f_4 \to f_2 \to s_1\},\ \{f_3 \to s_2\}\\
&f_4\colon \{f_4 \to f_2 \to s_1\},\ \{f_4 \to f_3 \to s_2\}.
\end{aligned}$$

To reduce the problem to a standard edge flow form of the network flow problem, we perform node splitting transformation on $G(V, E)$. Each node $u \in V$ is split into two nodes u and u', respectively, corresponding to the node's input and output. An extra edge (u, u') with zero cost and infinite capacity is also added. A new directed graph $G'(V', E')$ is constructed as follows.

1) The vertex set $V' = V \cup V_1' \cup V_2'$, where V_1' is the split node set of V_1 and V_2' is the split node set of V_2.
2) The edge set $E' = E_1' \cup E_2'$, where $E_1' = \{(u, u')|u \in V \wedge u'$ is the corresponding split node of $u\}$ and $E_2' = \{(u', v)|(u, v) \in E(G) \wedge u'$ is the corresponding split node of $u\}$. If there is a directed edge from u to v in $E(G)$, a corresponding directed edge from u' to v is added in $E'(G')$.

Through the node splitting, the original node-disjoint path problem in $G(V, E)$ is transformed into an *edge-disjoint* path problem in $G'(V', E')$. In other words, we need to find a repairable structure including $m \times n$ paths, which begin with each split f-TSV s in V_1' and end with each split s-TSV t in V_2'. In addition, all the paths sharing one same source node should be edge-disjoint. For instance, the red lines in Fig. 7 present edge-disjoint paths for each split f-TSV, and the corresponding generated fault-tolerance structure is shown in Fig. 2.

B. ILP Formulation

In the following, we will discuss how the edge-disjoint path search problem can be formulated as an integer programming. To facilitate the discussions, some notations are first defined. Given a unit flow (path) from source $s \in V_1'$ to sink $t \in V_2'$, a binary variable $x_{uv}^{(s,t)}$ is defined whether the path goes through edge $(u, v) \in E'$. If the path goes through edge (u, v), $x_{uv}^{(s,t)} = 1$; otherwise $x_{uv}^{(s,t)} = 0$.

Besides, to model the delay of each multiplexer, it is of importance calculating indegree of each node $u \in V$, which is not trivial. As shown in Fig. 7, the edge (f_2', f_3) is on the path from f_1' to s_2', as well as the path from f_2' to s_2'. Although the same edge is traversed by two paths, it only increases the indegree of f_3 by one. Meanwhile, there may be several edges directed into same TSV node on the paths. For instance, due to edges (f_4', f_3) and (f_2', f_3), the indegree of f_3 should be increased by two. Given a node u in the split directed graph G', its indegree is calculated by the following equation:

$$\text{indegree}(u) = \sum_{v:(v,u)\in E'} \min\left(\sum_{s\in V_1', t\in V_2'} x_{vu}^{(s,t)}, 1\right). \tag{8}$$

Based on the above notations, the edge-disjoint path search problem can be formulated as the following integer programming:

$$\begin{aligned}
&\mathbf{min} \ \max_{u\in V}\left\{\sum_{v:(v,u)\in E'} \min\left(\sum_{s\in V_1', t\in V_2'} x_{vu}^{(s,t)}, 1\right)\right\} && \text{(9a)}\\
&\mathbf{s.t.} \ \sum_{v:(u,v)\in E'} x_{uv}^{(s,t)} - \sum_{v:(v,u)\in E'} x_{vu}^{(s,t)}\\
&\qquad = \begin{cases} 1, & \text{if } u = s\\ 0, & \text{if } u \in V' - \{s, t\}, \quad \forall s \in V_1', t \in V_2'\\ -1, & \text{if } u = t \end{cases} && \text{(9b)}\\
&\qquad \sum_{t\in V_2'} x_{uu'}^{(s,t)} \le 1, \quad \forall s \in V_1', \left(u, u'\right) \in E_1' && \text{(9c)}\\
&\qquad x_{uv}^{(s,t)} \in \{0, 1\}, \quad \forall (u, v) \in E', s \in V_1', t \in V_2'. && \text{(9d)}
\end{aligned}$$

The objective function (9a) is to minimize the maximum indegree of all the nodes. The number of binary variables $x_{uv}^{(s,t)}$ is $m \times n \times |E'|$, where m is the number of f-TSVs, n is the number of s-TSVs, while $|E'|$ is the number of edges in split directed graph G'. The constraint (9b) defines a unit flow from $s \in V_1'$ to $t \in V_2'$, which corresponds a path from s, an f-TSV, to t, an s-TSV. The number of this set of constraints is $m \times n \times |V'|$. The constraint (9c) ensures that a set of V_2' paths, which have the same source $s \in V_1'$, are edge-disjoint. For example, considering the f_1' in Fig. 7, we have to search for two edge-disjoint paths, which end to s_1' and s_2', respectively. Constraint (9c) implies an edge-disjoint constraint to all edge $(u, u') \in E_1'$ that $x_{uu'}^{(f_1',s_1')} + x_{uu'}^{(f_1',s_2')} \le 1$. The number of this set of constraints is $m \times (m + n)$.

Formula (9) is nonlinear due to the min-max-min operation in the objective function (9a). Through linearizing the objective function, (9) can be transformed into an ILP

$$\begin{aligned}
&\mathbf{min} \ \lambda && \text{(10a)}\\
&\mathbf{s.t.} \ d_{vu} \ge x_{vu}^{(s,t)}, \quad \forall s \in V_1', t \in V_2', (v, u) \in E' && \text{(10b)}\\
&\qquad d_{vu} \le \sum_{s\in V_1', t\in V_2'} x_{vu}^{(s,t)}, \quad \forall (v, u) \in E' && \text{(10c)}\\
&\qquad d_{vu} \in \{0, 1\}, \quad \forall (v, u) \in E' && \text{(10d)}\\
&\qquad \sum_{v:(v,u)\in E'} d_{vu} \le \lambda, \quad \forall u \in V && \text{(10e)}\\
&\qquad \text{(9b)–(9d).}
\end{aligned}$$

To implement a linear function with linear constraints, variable λ is introduced to linearize the max operation. Meanwhile, as shown in (10b)–(10d), for each edge $(v, u) \in E'$ a binary variable d_{vu} is to linearize the min operation. In other words, d_{vu} is to help calculate the indegree of node u as follows:

$$d_{vu} = \min\left(\sum_{s \in V'_1, t \in V'_2} x_{vu}^{(s,t)}, 1 \right). \tag{11}$$

Combining (8) and (11), constraint (10e) ensures that the indegrees of all TSVs will not be greater than λ.

Considering the edge (f'_1, f_2) in Fig. 7, we give a detailed explanation how to calculate $d_{f'_1 f_2}$. First, per constraint (10b), for each split f-TSV in V'_1 and each split s-TSV in V'_2, we have

$$\begin{cases} d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_1, s'_1)}; \quad d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_1, s'_2)} \\ d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_2, s'_1)}; \quad d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_2, s'_2)} \\ d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_3, s'_1)}; \quad d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_3, s'_2)} \\ d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_4, s'_1)}; \quad d_{f'_1 f_2} \geq x_{f'_1 f_2}^{(f'_4, s'_2)}. \end{cases}$$

Second, per constraint (10c), we have the following equation:

$$d_{f'_1 f2} \leq \Big(x_{f'_1 f_2}^{(f'_1, s'_1)} + x_{f'_1 f_2}^{(f'_1, s'_2)} + x_{f'_1 f_2}^{(f'_2, s'_1)} + x_{f'_1 f_2}^{(f'_2, s'_2)} + x_{f'_1 f_2}^{(f'_3, s'_1)} + x_{f'_1 f_2}^{(f'_3, s'_2)} + x_{f'_1 f_2}^{(f'_4, s'_1)} + x_{f'_1 f_2}^{(f'_4, s'_2)} \Big).$$

As shown in Fig. 7, we have

$$\begin{cases} x_{f'_1 f_2}^{(f'_1, s'_1)} = 0; \quad x_{f'_1 f_2}^{(f'_1, s'_2)} = 1 \\ x_{f'_1 f_2}^{(f'_2, s'_1)} = 0; \quad x_{f'_1 f_2}^{(f'_2, s'_2)} = 0 \\ x_{f'_1 f_2}^{(f'_3, s'_1)} = 0; \quad x_{f'_1 f_2}^{(f'_3, s'_2)} = 0 \\ x_{f'_1 f_2}^{(f'_4, s'_1)} = 0; \quad x_{f'_1 f_2}^{(f'_4, s'_2)} = 0. \end{cases}$$

According to the constraints (10b)–(10c), $d_{f'_1 f_2} = 1$.

VII. Experimental Results

The proposed framework has been implemented in C++ language on a Linux 64-bit workstation (Intel 2.0 GHz, 62 GB RAM). The experiments were tested on MCNC and GSRC benchmarks, including two MCNC circuits (`ami33` and `ami49`), and four GSRC circuits (`n50`, `n100`, `n200`, and `n300`). The layer number is set to 3. The white space percentage and the aspect ratio are set to 20% and 1, respectively. The TSV pitch is assumed to be 5 × 5 um [3]. The multilayer floorplans are generated by a fixed-outline multilayer floorplanner IAR-MLFP [25], whose objective is a linear combination of the wire, the f-TSV number and the area cost. During the top-down *partitioning* step in the TSV grouping, hMetis [35] is used to partition f-TSVs into groups. The algorithmic solution software LEDA [36] is adopted to solve the MCMF problem. GLPK [37] is used as the ILP solver.

TABLE II
Impact of Replaceable Relation Between f-TSVs on Delay Overhead

Bench	w/o. *RRCost*		w. *RRCost*	
	#max_port	#g_f-TSVs	#max_port	#g_f-TSVs
`ami49`	7	8	3	7
`n100`	7	8	5	8
`n300`	8	9	6	8

A. Impact of Replaceable Relations Between f-TSVs on Delay Overhead

As shown in Section VI, the fault-tolerance structure for an f-TSV group is greatly affected by the replaceable relations between f-TSVs. In the f-TSV grouping stage, those f-TSVs that have replaceable relations tend to be assigned to the same group. By making use of the replaceable relations between f-TSVs in a group, we can build proper connections between f-TSVs to replace part of the direct connections from the f-TSVs to the s-TSVs, for reducing the input number of the multiplexers.

In the first experiment, we group the f-TSVs with and without consideration of the replaceable relations between f-TSVs, RRCost item in (6). The results are, respectively, shown in the columns "w. RRCost" and "w/o. RRCost" in Table II. Here the TSV defect probability p is set to 0.01 and N_{gs} is set to 3. Column "#max_port" represents the maximum port number of multiplexers among all groups. Column "#g_f-TSVs" gives the corresponding f-TSVs number in the group. As shown in Table II, with considering the RRCost item in the f-TSV grouping stage, the maximum port number of multiplexers among all groups is decreased, which demonstrates the effectiveness of the grouping method for reducing the multiplexer delay overhead.

The value of RRCost$_{ij}$ in (6) is set through the experimental results. For f-TSVs i and j, if f-TSV i is covered by the bounding box of the corresponding net of f-TSV j, meanwhile f-TSV j is also covered by the bounding box of the net of f-TSV i, the RRCost$_{ij}$ is labeled as RRCost$_{ij1}$; if f-TSV i is covered by the bounding box of the corresponding net of f-TSV j but f-TSV j is not covered by the bounding box of the net of f-TSV i, then RRCost$_{ij}$ is labeled as RRCost$_{ij2}$. The experiment is performed on `n100` benchmark. The TSV defect probability p is set to 0.001 and N_{gs} is set to 3. In the experiment, if we set RRCost$_{ij2}$ to a fixed value 5, the s-TSV numbers and maximum input number of multiplexers varied with RRCost$_{ij1}$, which is shown in Fig. 8(a). We noticed that as the RRCost$_{ij1}$ increases, the s-TSV numbers are rapidly increasing, resulting a higher hardware cost. In addition, with the RRCost$_{ij1}$ increases, the maximum input number of multiplexers is stabilized at a value. And if we set RRCost$_{ij1}$ to a fixed value 10, the s-TSV numbers and maximum input number of multiplexers varied with RRCost$_{ij2}$, which is shown in Fig. 8(b). Therefore, according to Fig. 8, RRCost$_{ij1} = 10$ and RRCost$_{ij2} = 5$ achieve relatively less s-TSV numbers and less input port number of multiplexers.

B. Impact of N_{gs} on s-TSV Allocation

N_{gs} denotes the number of inserted s-TSV(s) in one TSV group. In this paper, the same number of s-TSV(s), N_{gs}, is assigned to each group. A group cannot be repaired if and only if the number of faulty f-TSVs is more than the number of nonfaulty s-TSVs. In order to ensure a successful repair when there are multiple faulty f-TSVs in a same group, it is

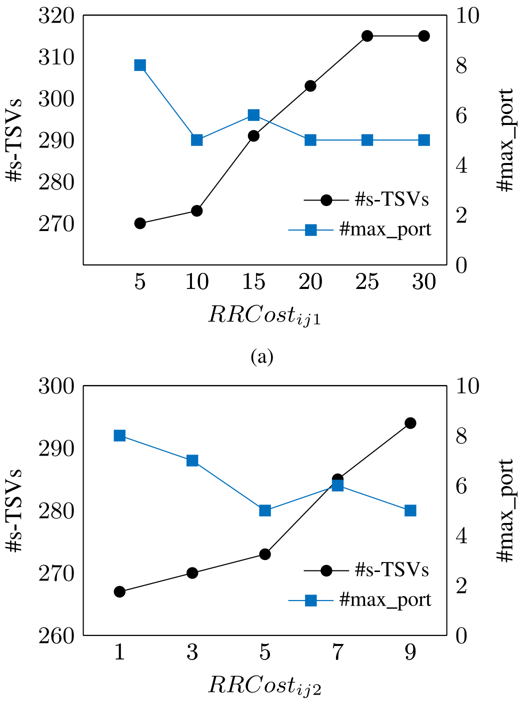
TABLE III
TSV YIELD AND HARDWARE COST UNDER DIFFERENT DEFECT PROBABILITY p AND N_{gs}

Bench	Defect probability	N_{gs}	#f-TSVs	#s-TSVs	Wire (um)	MUX		Yield
						#max_port	#g_f-TSVs	
ami33	$p = 0.01$	1	55	55	32575.54	1	1	99.39%
		2	56	28	32145.85	5	8	99.96%
		3	54	51	33256.59	6	8	100%
	$p = 0.001$	1	51	14	35854.87	4	8	99.98%
		2	58	24	30529.58	5	7	100%
		3	52	39	35092.97	6	7	100%
	$p = 0.0001$	1	53	12	34295.30	4	7	100%
		2	56	20	32843.39	5	8	100%
		3	58	36	29973.33	6	8	100%
ami49	$p = 0.01$	1	NF	NF	NF	NF	NF	NF
		2	136	68	278857.42	5	7	99.92%
		3	132	75	283672.70	3	7	100%
	$p = 0.001$	1	135	37	279268.67	4	7	99.96%
		2	131	46	285152.40	2	4	100%
		3	138	63	273860.61	6	7	100%
	$p = 0.0001$	1	133	25	280665.49	4	8	100%
		2	137	44	275408.94	4	8	100%
		3	135	60	279524.37	5	8	100%
n50	$p = 0.01$	1	NF	NF	NF	NF	NF	NF
		2	377	158	58025.25	5	7	99.79%
		3	385	207	56962.52	6	8	100%
	$p = 0.001$	1	386	129	56642.98	3	7	99.90%
		2	380	142	57452.93	5	8	100%
		3	387	201	55960.22	5	8	100%
	$p = 0.0001$	1	374	69	59599.68	4	7	100%
		2	380	130	57038.25	4	7	100%
		3	386	180	56059.73	5	7	100%
n100	$p = 0.01$	1	NF	NF	NF	NF	NF	NF
		2	592	236	76935.25	5	7	99.64%
		3	591	285	77827.28	5	8	99.99%
	$p = 0.001$	1	593	154	75507.74	3	5	99.81%
		2	592	226	76722.78	5	8	100%
		3	592	273	76245.32	5	7	100%
	$p = 0.0001$	1	596	106	74729.76	4	8	100%
		2	595	218	75096.09	5	8	100%
		3	590	237	79489.22	6	8	100%
n200	$p = 0.01$	1	NF	NF	NF	NF	NF	NF
		2	1148	378	158898.94	4	7	99.48%
		3	1143	453	165821.16	6	7	99.98%
	$p = 0.001$	1	1145	373	161908.82	3	8	99.64%
		2	1146	366	160703.73	5	8	100%
		3	1144	408	163348.96	6	7	100%
	$p = 0.0001$	1	1142	196	166300.51	5	8	99.99%
		2	1143	300	166222.97	5	8	100%
		3	1149	372	155706.39	7	8	100%
n300	$p = 0.01$	1	NF	NF	NF	NF	NF	NF
		2	1244	522	214307.47	5	7	99.19%
		3	1244	549	215567.84	6	8	99.97%
	$p = 0.001$	1	1238	384	222546.42	4	7	99.62%
		2	1242	420	217300.98	7	8	100%
		3	1246	525	212860.39	6	7	100%
	$p = 0.0001$	1	1247	210	210164.89	6	8	99.99%
		2	1240	394	220639.52	6	8	100%
		3	1248	483	206651.37	6	7	100%

beneficial to increase N_{gs}, but it also leads to higher hardware cost. Therefore, it is necessary to explore a tradeoff analysis between TSV yield and hardware cost.

The experiments are performed on all test cases, which are shown in Table III. N_{gs} is varied from 1 to 3 to achieve the target yield. The yield results in experiment are accurate to the fourth decimal place. Three options for the TSV defect probability (p) are considered: 0.01, 0.001, and 0.0001 [12], and the clustering coefficient α equals to 2. The wire is the sum of HPWL of subnets obtained by decomposing nets spanning multiple device layers (Section IV-B). We execute the proposed method 20 times independently for each cast, and list the average statistic result. Columns "#f-TSVs," "#s-TSVs," "Wire," and "Yield" list the total f-TSVs number, total number of allocated s-TSVs, wirelength, and yield, separately. Besides, "#max_port" represents the maximum port number of multiplexers among all groups, and "#g_f-TSVs" gives the corresponding f-TSVs number in the group. The term *NF* represents that the chip yield cannot satisfy the target yield.

As shown in the tables, with the increasing f-TSV numbers, the chip yield drops and more s-TSVs are needed to generate a repair solution. For ami33, ami49, n50, and n100 benchmarks, the chip yield can reach 100% under a lower defect probability ($p = 0.0001$). According to [12], the hardware cost is related to the f-TSV group numbers and the N_{gs}. Therefore, due to lower hardware cost, $N_{gs} = 1$ is a more efficient solution



(a)

(b)

Fig. 8. Effect of (a) RRCost_{ij1} and (b) RRCost_{ij2} on s-TSV numbers and maximum input number of multiplexers.

for `ami33`, `ami49`, `n50`, and `n100` benchmark under low defect probability scenarios. However, for `n200` and `n300` benchmarks, $N_{gs} = 2$ is a better option under a lower defect probability due to its higher chip yield.

For a higher defect probability ($p = 0.01$ and 0.001), the repair configuration with $N_{gs} = 1$ cannot achieve an optimal yield solution, thus it is beneficial to allocate more s-TSVs for each group in this situation. However, increasing N_{gs} does not always result in higher hardware cost. For example, in case `n200` with $p = 0.001$, the repair configuration with $N_{gs} = 2$ can achieve higher yield with lower hardware cost. Because, under $N_{gs} = 1$, in order to guarantee the chip yield, more f-TSV groups are partitioned. Although only one s-TSV is assigned for each group, the total s-TSV number will be more. Therefore, it is necessary to choose an appropriate N_{gs} for different benchmarks during s-TSV allocation.

C. Impact of Clustering

As discussed in Section II-B, in reality, faulty TSVs tend to cluster together rather than being uniformly distributed. It is necessary to compare the generated repair solution under the two TSV defect-distribution model.

The experiment is performed on `n100` benchmark. The TSV defect probability p is set to 0.001 and N_{gs} is set to 1. With different values of clustering coefficient α, the proposed method is applied for different target yields. Table IV shows the results. Compared with the realistic clustered TSV defect-distribution, the repair solution generated by uniform TSV defect-distribution model ($\alpha = 0$) underestimates the allocated s-TSV numbers. For example, to achieve the target yield $Y_{\text{TSV}} = 99.7\%$, 120 s-TSVs are allocated to generate a repair solution under the uniform defect-distribution model, which is less than the required number under the clustered TSV defect-distribution model. Thus, the generated repair solution cannot satisfy the yield constraint.

TABLE IV
DIFFERENCE OF ALLOCATED S-TSV NUMBERS FOR UNIFORM AND CLUSTERED TSV DEFECT-DISTRIBUTION MODELS

Clustering coefficient	Y_{TSV}		
	99.8%	99.7%	99.6%
α = 0	125	115	66
α = 1	149	129	75
α = 2	154	137	88
α = 3	165	150	100

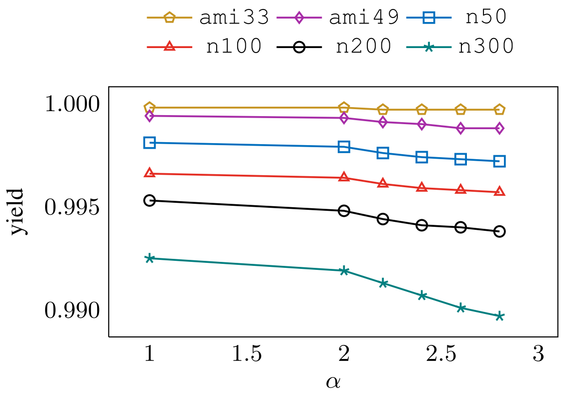


Fig. 9. Relation between yield and the clustering coefficient α.

TABLE V
EFFECTIVENESS OF WHITESPACE REDISTRIBUTION

Bench	w. WR		w/o. WR	
	Wire (um)	Yield	Wire (um)	Yield
`ami33`	32523.47	99.97%	32693.38	99.78%
`ami49`	280457.33	99.95%	283181.17	99.72%
`n50`	571792.36	99.88%	570076.98	99.69%
`n100`	76578.72	99.78%	76281.63	99.57%
`n200`	160721.16	99.59%	161348.97	99.36%
`n300`	211360.56	99.32%	209846.39	99.08%
Avg.	222238.93	99.75%	222238.09	99.53%

We also use Fig. 9 to show the relation between the yield and the clustering coefficient α under the clustered TSV defect-distribution model. The experiments are performed on all test benchmarks. And the TSV defect probability p is set to 0.01 and N_{gs} is set to 2. It can be observed from the figure that, with the increasing clustering effect, the chip yield drop. Because a larger α implies larger possibility to incur faulty TSVs around the existing faulty TSVs. And the yield drop becomes more visible on benchmarks with more f-TSVs.

D. Impact of Whitespace Redistribution on Yield and Performance

As the s-TSVs can be only allocated into the whitespace, the quality of the generated repair solution is strongly dependent on the whitespace distribution. In this experiment, we compare the framework with and without adopting the whitespace redistribution strategy to study the impact of whitespace redistribution on the chip yield and wirelength. The wire is the sum of HPWL of subnets obtained by decomposing nets spanning multiple device layers (Section IV-B). The experiment is performed on all benchmarks. The TSV defect probability p is set to 0.005 and N_{gs} is set to 2. And we execute the experiment ten times independently for each benchmark, and list the average statistic results. The results are, respectively, shown in columns "w. *WR*" and "w/o. *WR*" of Table V. As shown

TABLE VI
COMPARISON AMONG MCMF METHOD WITHOUT WHITESPACE REDISTRIBUTION (MCMF), SHORTEST-PATH-BASED HEURISTIC WITHOUT WHITESPACE REDISTRIBUTION (PATH), AND THE PROPOSED CONVEX-COST FLOW WITH WHITESPACE REDISTRIBUTION (CCF_WR)

Bench	MCMF			PATH			CCF_WR		
	#Succ	Wire (um)	Yield	#Succ	Wire (um)	Yield	#Succ	Wire (um)	Yield
ami33	93%	32411.77	99.69%	97%	31776.08	99.77%	100%	31923.61	99.95%
ami49	93%	281413.71	99.54%	97%	274079.23	99.72%	100%	279216.82	99.93%
n50	90%	58535.47	99.50%	90%	57393.27	99.57%	100%	58137.42	99.78%
n100	90%	77493.24	99.32%	90%	76308.73	99.38%	100%	76634.45	99.62%
n200	80%	160825.16	99.18%	83%	158785.81	99.20%	100%	160714.38	99.47%
n300	73%	215328.34	98.87%	80%	210625.72	98.97%	100%	213642.19	99.19%
Avg.	86%	137667.95(+0.69%)	99.35%	89%	134828.14(-1.38%)	99.44%	100%	136711.48(1.00)	99.66%

in table, the wirelength can be changed by (−1%, 1%) on average, which means that the impact of whitespace redistribution strategy on wirelength is insignificant. We also notice that the whitespace redistribution can improve the yield of fault-tolerance solution, which demonstrates that the proposed strategy potentially avoids a dense TSV distribution in some regions and reduces the effect of TSV fault clustering.

E. Impact of f-TSV Planning on Yield

In the contrast experiment, without considering the whitespace redistribution and convex-cost flow model in f-TSV planning, we directly adopt the MCMF model in [19] and [28] to allocate f-TSVs layer by layer. Besides, considering the suboptimality of allocating the TSVs layer by layer, we also implement a shortest path-based heuristic similar to that in [19] for f-TSV allocation. The experiment is performed on all benchmarks. Each case is executed 30 times independently, and the average statistic results are listed. Here the TSV defect probability p is set to 0.01 and N_{gs} is set to 2. We compare the success rate in finding s-TSVs for generating the fault tolerance structure of these methods, which are shown in the column “#Succ” of Table VI. The experimental results show that the convex-cost flow model, combined with whitespace redistribution, can achieve 100% success rate in finding s-TSVs for f-TSV groups on penalty of 1.38% wirelength, whereas a directly application of MCMF-based or shortest-path-based heuristic (PATH) can only achieve 86% and 89% success rate, respectively, and, hence, the chip yields are greatly lowered.

F. Comparison With Previous Work

In the proposed f-TSV grouping method, we first use the recursive min-cut bi-partitioning algorithm to globally partition the f-TSVs into several groups until the chip yield is higher than the target yield, then locally merge the two available f-TSV groups with the highest yield together to reduce the group numbers until the chip yield is reduced to target yield. In the previous work [12], only a locally greedy method is used to partition f-TSVs into groups. In order to observe the effect of grouping, we compare the number of allocated s-TSVs and chip yield of the two methods.

The experiment is performed on all benchmarks. The clustering coefficient α equals to 2, the TSV defect probability p is set to 0.01, and N_{gs} is set to 2. Based on the same f-TSV planning result, we run the partitioning method in [12] and the proposed grouping method, respectively. Table VII shows the experimental results. It can be noticed that the proposed top-down and bottom-up partitioning strategy can achieve lower hardware cost with higher chip yield.

TABLE VII
COMPARISON BETWEEN THE LOCAL PARTITION METHOD [12] AND THE PROPOSED F-TSV GROUPING METHOD

Bench	#f-TSVs	Local partition [12]		Ours	
		#s-TSVs	Yield	#s-TSVs	Yield
ami33	55	30	99.96%	32	99.96%
ami49	138	74	99.80%	70	99.91%
n50	380	162	99.77%	156	99.77%
n100	595	246	99.59%	242	99.63%
n200	1148	388	99.43%	378	99.48%
n300	1249	546	99.12%	532	99.17%

After the f-TSV grouping stage, f-TSVs are already partitioned into groups and the f-TSVs in each group can have a reasonable number of common s-TSV candidates ($\geq N_{gs}$) for constructing fault-tolerance structure under the target yield constraint. Then the s-TSV allocation assigns N_{gs} s-TSVs for each f-TSV group from the s-TSV candidates. In this paper, we formulate the s-TSV allocation as the MCMF problem. The objective is to minimize the total wirelength induced by the possible connections between the f-TSVs and the s-TSVs. Meanwhile, in order to reduce the maximum wirelength overhead among all f-TSVs, we changed the edge cost in (7) from the linear superposition to the square superposition in this experiment. In [12], it formulates the s-TSV allocation as an ILP problem. And the objective is to minimize the maximum delay overhead incurred after TSV repair by replacing f-TSVs by s-TSVs. In order to observe the effect of s-TSV allocation, we compare chip yield, runtime, the total incremental wirelength and maximum incremental wirelength incurred by the fault-tolerance structure of the two methods.

The experiment is performed on all benchmarks. The clustering coefficient α equals to 2, the TSV defect probability p is set to 0.01, and N_{gs} is set to 2. Based on the same f-TSV grouping results, we run the ILP in [12] and the proposed MCMF method, respectively. We execute the experiment ten times independently for each benchmark, and list the average statistic results. Table VIII shows the experimental results. Columns “RT,” “M_IWire,” and “T_IWire” represent the runtime, maximum and total incremental wirelength incurred by the fault-tolerance structure. It can be noticed that compared with ILP, the proposed MCMF can achieve almost same chip yield and maximum incremental wirelength. However, the total incremental wirelength and the runtime of the MCMF is quite better than the ILP.

In this paper, an ILP-based model is proposed to form a fault-tolerance structure with minimization of the maximum input number of multiplexers. To the best of our knowledge, this is the first work for generating n-fault ($n > 1$) tolerance

TABLE VIII
COMPARISON BETWEEN THE ILP [12] AND THE MCMF IN S-TSV ALLOCATION

Bench	#f-TSVs	#s-TSVs	ILP [12]				Ours			
			Yield	M_IWire (um)	T_IWire (um)	RT (s)	Yield	M_IWire (um)	T_IWire (um)	RT (s)
ami33	56	32	99.96%	726.24	1855.21	1.347	99.96%	699.92	1689.76	0.006
ami49	138	70	99.91%	2950.01	15609.38	27.752	99.91%	2950.01	14234.97	0.010
n50	378	160	99.78%	411.91	6791.82	36.051	99.78%	402.52	6308.49	0.012
n100	594	238	99.62%	479.18	10462.53	68.35	99.63%	486.32	10008.36	0.018
n200	1148	378	99.48%	513.64	24000.84	259.06	99.48%	500.33	19986.01	0.021
n300	1250	530	–	–	–	> 7200	99.17%	800.06	26305.60	0.031

TABLE IX
COMPARISON BETWEEN MST METHOD [11] AND PROPOSED ILP MODEL FOR GENERATING FAULT-TOLERANCE STRUCTURE

Bench	MST [11]		Ours	
	#max_port	#g_f-TSVs	#max_port	#g_f-TSVs
ami33	7	8	4	8
ami49	7	8	4	7
n50	6	7	3	7
n100	6	6	3	5
n200	6	8	3	8
n300	7	8	4	7

TABLE X
RUNTIME OF THE PROPOSED REPAIR FRAMEWORK FOR ALL TEST BENCHMARKS

Bench	T_{group} (s)	T_{allo} (s)	T_{struct} (s)
ami33	1.27	0.009	0.017
ami49	3.05	0.010	0.325∼3951.071
n50	10.47	0.011	0.378
n100	14.53	0.017	0.627
n200	51.24	0.019	1.103
n300	214.36	0.030	2.194

structures that uses n s-TSVs for a group of f-TSVs, considering the delay overhead of multiplexers. In [11], 1-fault tolerance structures are generated using MST-based method. However, it is difficult to apply the method to structures using more than one s-TSVs. In addition, the delay overhead introduced by the multiplexers, which are used for rerouting signals in the generated fault-tolerance structures, is not considered. In the worst-case the input port number of a multiplexer could be the number of f-TSVs in the group if the tree is a star structure, which introduces large delay overhead. We compare the proposed ILP-based model with MST method in [11] under the 1-fault tolerance structure case.

The experiment is performed on all benchmarks. The clustering coefficient α equals to 2, the TSV defect probability p is set to 0.001, and N_{gs} is set to 1. Based on the same s-TSV planning result, we run the MST method in [11] and the proposed ILP model, respectively. We execute the experiment ten times independently for each benchmark, and list the average statistic results. Column "#max_port" represents the maximum port number of multiplexers among all groups, while column "#g_f-TSVs" gives the corresponding f-TSVs number in the group. As shown in Table IX, compared with [11], the proposed ILP-based model can reduce the maximum port number of multiplexers among all groups, which demonstrates the effectiveness of the proposed ILP model for reducing the multiplexer delay overhead.

G. Runtime Analysis

In the proposed TSV repair framework, we mainly consider the runtime for f-TSV grouping, s-TSV allocation, and fault-tolerance structure construction stage. Table X shows the average runtime on all test benchmarks by varying the defect probability, target yield, clustering coefficient, and N_{gs}. Columns "T_{group}," "T_{allo}," and "T_{struct}" represent runtimes for f-TSV grouping, s-TSV allocation, and structure construction, respectively. For ami49 case with $N_{gs} = 3$, the runtime of the structure construction stage is very long. One possible reason is that when the directed graph for some groups have a large amount of edges, solving the ILP model may be very runtime consuming.

VIII. CONCLUSION

In this paper, we have proposed an efficient TSV planning and repair framework during floorplanning stage. A set of novel algorithms, e.g., f-TSV grouping, s-TSV allocation, and fault-tolerance structure construction, are developed to provide yield awareness in TSV planning. Experimental results demonstrate the proposed strategy can effectively repair nonuniformly placed f-TSVs under clustered TSV defect distribution. This paper is the first work for generating double or even multiple fault tolerance structures. As continuing growth of technology node, 3-D IC turns out to be a promising solution to further scaling, we believe this paper will stimulate more research on yield aware 3-D IC design.

ACKNOWLEDGMENT

The authors would like to thank the Information Science Laboratory Center of USTC for hardware and software services. They would also like to thank Dr. S. Wang from the Karlsruhe Institute of Technology and Prof. L. Jiang from Shanghai Jiaotong University for their help on the yield modeling.

REFERENCES

[1] S. J. Souri, K. Banerjee, A. Mehrotra, and K. C. Saraswat, "Multiple Si layer ICs: Motivation, performance analysis, and design implications," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, Los Angeles, CA, USA, 2000, pp. 213–220.

[2] J. W. Joyner, P. Zarkesh-Ha, and J. D. Meindl, "A global interconnect design window for a three-dimensional system-on-a-chip," in *Proc. IEEE Int. Interconnect Technol. Conf. (IITC)*, Burlingame, CA, USA, Jun. 2001, pp. 154–156.

[3] (2008). *International Technology Roadmap for Semiconductors*. [Online]. Available: http://www.itrs2.net

[4] Q. Xu, L. Jiang, H. Li, and B. Eklow, "Yield enhancement for 3D-stacked ICs: Recent advances and challenges," in *Proc. IEEE/ACM Asia South Pac. Design Autom. Conf. (ASPDAC)*, Sydney, NSW, Australia, Feb. 2012, pp. 731–737.

[5] H.-H. S. Lee and K. Chakrabarty, "Test challenges for 3D integrated circuits," *IEEE Des. Test. Comput.*, vol. 26, no. 5, pp. 26–35, Sep./Oct. 2009.

[6] C. Ferri, S. Reda, and R. I. Bahar, "Strategies for improving the parametric yield and profits of 3D ICs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2007, pp. 220–226.

[7] C.-W. Chou, Y.-J. Huang, and J.-F. Li, "Yield-enhancement techniques for 3D random access memories," in *Proc. Int. Symp. VLSI Design Autom. Test (VLSI DAT)*, Hsinchu, Taiwan, Apr. 2010, pp. 104–107.

[8] L. Jiang, R. Ye, and Q. Xu, "Yield enhancement for 3D-stacked memory by redundancy sharing across dies," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2010, pp. 230–234.

[9] D. H. Kim, K. Athikulwongse, and S. K. Lim, "A study of through-silicon-via impact on the 3D stacked IC layout," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2009, pp. 674–680.

[10] L. Jiang, Q. Xu, and B. Eklow, "On effective TSV repair for 3D-stacked ICs," in *Proc. IEEE/ACM Design Autom. Test Eurpoe (DATE)*, Dresden, Germany, Mar. 2012, pp. 793–798.

[11] Y.-G. Chen, W.-Y. Wen, Y. Shi, W.-K. Hon, and S.-C. Chang, "Novel spare TSV deployment for 3-D ICs considering yield and timing constraints," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 4, pp. 577–588, Apr. 2015.

[12] S. Wang, M. B. Tahoori, and K. Chakrabarty, "Defect clustering-aware spare-TSV allocation for 3D ICs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Austin, TX, USA, Nov. 2015, pp. 307–314.

[13] R. K. Nain, S. Pinge, and M. Chrzanowska-Jeske, "Yield improvement of 3D ICs in the presence of defects in through signal vias," in *Proc. IEEE Int. Symp. Qual. Electron. Design (ISQED)*, San Jose, CA, USA, Mar. 2010, pp. 598–605.

[14] Y. Zhao, S. Khursheed, and B. M. Al-Hashimi, "Cost-effective TSV grouping for yield improvement of 3D-ICs," in *Proc. IEEE Asian Test Symp. (ATS)*, New Delhi, India, Nov. 2011, pp. 201–206.

[15] A.-C. Hsieh and T. Hwang, "TSV redundancy: Architecture and design issues in 3-D IC," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 4, pp. 711–722, Apr. 2012.

[16] L. Jiang, Q. Xu, and B. Eklow, "On effective through-silicon via repair for 3-D-stacked ICs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 4, pp. 559–571, Apr. 2013.

[17] W.-H. Lo, K. Chi, and T. Hwang, "Architecture of ring-based redundant TSV for clustered faults," in *Proc. IEEE/ACM Design Autom. Test Europe (DATE)*, Grenoble, France, Mar. 2015, pp. 848–853.

[18] F. Ye and K. Chakrabarty, "TSV open defects in 3D integrated circuits: Characterization, test, and optimal spare allocation," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, San Francisco, CA, USA, Jun. 2012, pp. 1024–1030.

[19] X. Liu, Y. Zhang, G. Yeap, and X. Zeng, "An integrated algorithm for 3D-IC TSV assignment," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, San Diego, CA, USA, 2011, pp. 652–657.

[20] M.-C. Tsai, T.-C. Wang, and T. Hwang, "Through-silicon via planning in 3-D floorplanning," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 8, pp. 1448–1457, Aug. 2011.

[21] C.-R. Li, W.-K. Mak, and T.-C. Wang, "Fast fixed-outline 3-D IC floorplanning with TSV co-placement," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 3, pp. 523–532, Mar. 2013.

[22] X. Liu, G. Yeap, J. Tao, and X. Zeng, "Integrated algorithm for 3-D IC through-silicon via assignment," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 3, pp. 655–666, Mar. 2014.

[23] P. H. Shiu, R. Ravichandran, S. Easwar, and S. K. Lim, "Multi-layer floorplanning for reliable system-on-package," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 5. Vancouver, BC, Canada, May 2004, pp. V-69–V-72.

[24] I. Loi, S. Mitra, T. H. Lee, S. Fujita, and L. Benini, "A low-overhead fault tolerance scheme for TSV-based 3D network on chip links," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2008, pp. 598–602.

[25] S. Chen and T. Yoshimura, "Multi-layer floorplanning for stacked ICs: Configuration number and fixed-outline constraints," *Integr. VLSI J.*, vol. 43, no. 4, pp. 378–388, 2010.

[26] S. Chen and T. Yoshimura, "Fixed-outline floorplanning: Block-position enumeration and a new method for calculating area costs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 5, pp. 858–871, May 2008.

[27] W. Zhong, S. Chen, and T. Yoshimura, "Whitespace insertion for through-silicon via planning on 3-D SoCs," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Paris, France, May/Jun. 2010, pp. 913–916.

[28] S. Chen, L. Ge, M.-F. Chiang, and T. Yoshimura, "Lagrangian relaxation based inter-layer signal via assignment for 3-D ICs," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E92-A, no. 4, pp. 1080–1087, Apr. 2009.

[29] S. Chen, J. Shen, W. Guo, M.-F. Chiang, and T. Yoshimura, "Redundant via insertion: Removing design rule conflicts and balancing via density," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E93-A, no. 12, pp. 2372–2379, Dec. 2010.

[30] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.

[31] B. Kim, C. Sharbono, T. Ritzdorf, and D. Schmauch, "Factors affecting copper filling process within high aspect ratio deep vias for 3D chip stacking," in *Proc. IEEE Electron. Compon. Technol. Conf.*, San Diego, CA, USA, May 2006, pp. 838–843.

[32] A. P. Karmarkar, X. Xu, and V. Moroz, "Performanace and reliability analysis of 3D-integration structures employing through silicon via (TSV)," in *Proc. IEEE Int. Rel. Phys. Symp. (IRPS)*, Montreal, QC, Canada, Apr. 2009, pp. 682–687.

[33] M. B. Tahoori, "Defects, yield, and design in sublithographic nano-electronics," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. (DFT)*, Monterey, CA, USA, Oct. 2005, pp. 3–11.

[34] I. Koren and Z. Koren, "Defect tolerance in VLSI circuits: Techniques and yield analysis," *Proc. IEEE*, vol. 86, no. 9, pp. 1819–1838, Sep. 1998.

[35] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multilevel hypergraph partitioning: Applications in VLSI domain," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 7, no. 1, pp. 69–79, Mar. 1999.

[36] K. Mehlhorn and S. Naher, *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge, U.K.: Cambridge Univ. Press, 1999.

[37] A. Makhorin. (2008). *GLPK (GNU Linear Programming Kit)*. [Online]. Available: https://www.gnu.org/software/glpk/

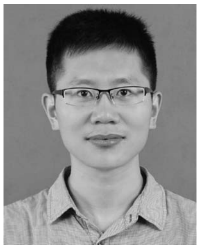
Qi Xu received the B.E. degree in microelectronics from Anhui University, Hefei, China, in 2012. He is currently pursuing the Ph.D. degree in electronic science and technology with the University of Science and Technology of China, Hefei.

His current research interests include physical design automation and design for reliability for 3-D integrated circuits.

Song Chen (M'09) received the B.S. degree from Xi'an Jiaotong University, Xi'an, China, in 2000, and the Ph.D. degree from Tsinghua University, Beijing, China, in 2005, both in computer science.

He is currently an Associate Professor with the Department of Electronic Science and Technology, University of Science and Technology of China. His current research interests include several aspects of very large-scale integration design automation, on-chip communication system, and computer-aided design for emerging technologies.

Dr. Chen is a member of IEICE.



Xiaodong Xu received the B.E. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2014. He is currently pursuing the M.S. degree with the Department of Electronic Science and Technology, University of Science and Technology of China, Hefei, China.

His current research interests include computer-aided design for very large-scale integration, high-level synthesis for field-programmable gate array, and optimization algorithms.

Bei Yu (S'11–M'14) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Texas at Austin in 2014.

He is currently an Assistant Professor with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong.

Dr. Yu was a recipient of the four Best Paper Awards at International Symposium on Physical Design 2017, the SPIE Advanced Lithography Conference 2016, the International Conference on Computer Aided Design 2013, and the Asia and South Pacific Design Automation Conference 2012. He has served in the editorial boards of *Integration*, *VLSI Journal*, and *IET Cyber-Physical Systems: Theory & Applications*.