Adaptive 3D-IC TSV Fault Tolerance Structure Generation

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

***Abstract*—In 3-D integrated circuits (3D-ICs), through silicon via (TSV) is a critical technique in providing vertical connections. However, the yield is one of the key obstacles to adopt the TSV-based 3D-ICs technology in industry. Various fault-tolerance structures using spare TSVs to repair faulty functional TSVs have been proposed in literature for yield and reliability enhancement, but a valid structure cannot always be found due to the lack of effective generation methods for fault-tolerance structures. In this paper, we focus on the problem of adaptive fault-tolerance structure (AFTS) generation. Given the relations between functional TSVs and spare TSVs, we first calculate the maximum number of tolerant faults in each TSV group. Then we propose an integer linear programming-based model to construct the AFTS with minimal multiplexer delay overhead and hardware cost. We further develop a speed-up technique through an efficient min-cost-max-flow model. All the proposed methodologies are embedded in a top-down TSV planning framework to form functional TSV groups and generate AFTSs. Experimental results show that, compared with state-of-the-art, the number of spare TSVs used for fault tolerance can be effectively reduced.**

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

I. Introduction

As DEVICE feature sizes continue to rapidly decrease, the interconnect delay is becoming a bottleneck limiting IC performance. 3-D integrated circuits (3D-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]. Study indicates that the average wire-length of a 3D-IC varies according to the square root of the number of layers [2]. Moreover, 3D-ICs also offer the potential for heterogeneous integration, which is essential for More than Moore technology [3]. 3-D integration has already seen commercial applications in the form of 3-D memory but there are still significant open problems in both research and implementation [4]. In this paper, we will focus on the TSV reliability problem.

TSVs may be affected by various reliability issues, such as undercut, misalignment, or random open defects [5], [6]. Because there exist a large number of TSVs in a chip, these issues in turn lead to a low chip yield. For example, [7] reported a 60% chip yield for a chip with 20 000 TSVs and only 20% yield for 55 000 TSVs in IMEC process technology. Since yield and reliability is a primary concern in 3D-ICs design, a robust fault-tolerance structure is imperative. In general, there are two types of yield losses in 3D-ICs: 1) the yield loss due to defects in stacked dies and 2) the yield loss due to defects occurred during assembling process [8]. For the former case, it is critical to conduct prebond testing to avoid the stacking of defective dies [9]. A number of die/wafer matching and interdie repair strategies have also been proposed to increase the stack yield [10]–[13]. For the latter case, adding spare TSVs (referred to as s-TSVs) to repair fault functional TSVs (referred to as f-TSVs) is an effective method for enhancing yield.

One key problem in TSV fault-tolerance design is the fault-tolerance structure generation, where a number of functional TSVs and one or several spare TSVs are grouped together to provide redundancy. Chen *et al*. [7] proposed a minimum spanning tree-based method to group f-TSVs and form one-fault-tolerance structures. However, the method is difficult to be applied to multiple-fault-tolerance structure generation. Wang *et al*. [14] presented a regular TSV replacing chain structure that can repair faulty TSVs based on a realistic clustered defect model. Xu *et al*. [15] further considered the physical information of the TSV groups, and developed an integer linear programming (ILP) formulation for fault-tolerance structure generation. They model replaceable relations between f-TSVs, so the maximum input-port number of individual multiplexers can be effectively reduced. However, all previous works [14], [15] are under an assumption that a predetermined number of s-TSVs is assigned to each TSV group. To ensure that K common s-TSVs can be allocated to each f-TSV group, in each group f-TSV number is usually quite small, which introduces a large number of TSV groups. Since the total number of s-TSVs is proportional to the TSV group number, it may cause overuse of s-TSVs.

To overcome the above issues, in this paper we propose an adaptive fault-tolerance structure (AFTS), in which the

Manuscript received July 12, 2017; revised November 26, 2017 and January 30, 2018; accepted March 27, 2018. Date of publication April 6, 2018; date of current version April 19, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61732020 and Grant 61674133, in part by the Anhui Provincial Natural Science Foundation, China under Grant 1508085MF134, and in part by the Research Grants Council of Hong Kong under Project CUHK24209017. This paper was recommended by Associate Editor H.-G. Stratigopoulos. *(Corresponding authors: Qi Xu; Song Chen.)*

S. Chen and Q. Xu are with the Department of Electronic Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: songch@ustc.edu.cn; xuqi@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.2018.2824284

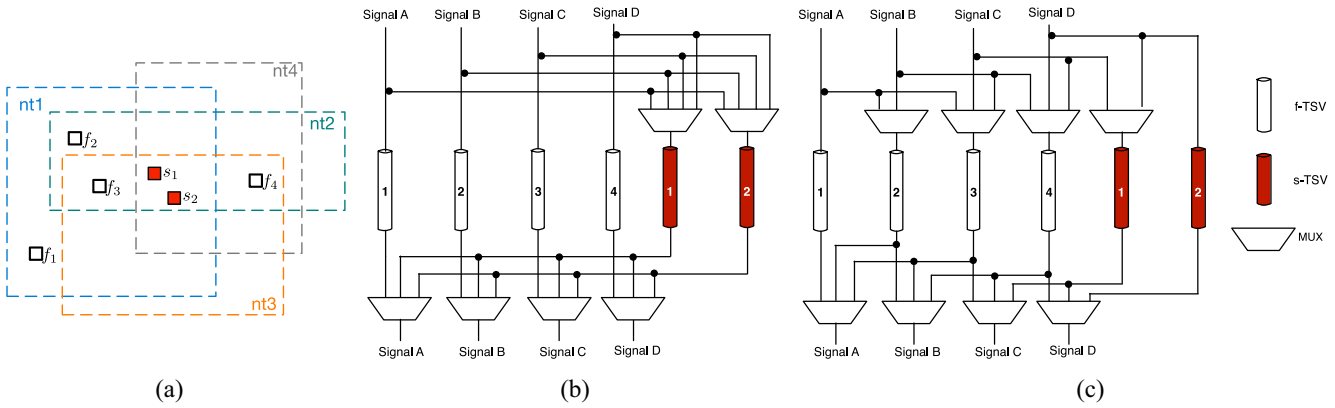


Fig. 1. (a) Example of TSV group with four f-TSVs and two s-TSVs. (b) Fault-tolerance structure with large multiplexer delay overhead. (c) Regular chain structure.

number of tolerant faults is adaptively determined by the distribution of the f-TSVs and their candidate s-TSVs. A set of s-TSVs will be selected from a large amount of candidates. Our AFTS generation method can achieve minimal multiplexer delay overhead, as well as minimal number of required s-TSVs. Key technical contributions of this paper are listed as follows.

1) We are able to determine the maximum number of tolerant faults, denoted as K, in polynomial time.
2) We present an ILP formulation in generating the adaptive K-fault tolerance structures.
3) We further propose an efficient min-cost-max-flow (MCMF)-based heuristic method to speed-up the K-fault tolerance structure generation.
4) All the proposed methodologies are embedded in a top-down TSV planning framework to form f-TSV groups and generate fault-tolerance structures.

Experimental results show that, compared with state-of-the-art, the proposed framework can reduce the number of used s-TSVs and maximum port number of multiplexers.

The remainder of this paper is organized as follows. Section II presents the motivation and gives the problem formulation. The method for determining the maximum number of tolerant faults is presented in Section III. Sections IV and V present the proposed ILP formulation and heuristic method. Section VI describes the proposed fault tolerance TSV planning methodology. Section VII provides experimental results, followed by conclusion in Section VIII.

II. Preliminaries

A. Chip Yield and TSV Yield

Consider a 3D-IC containing l layers, and the yield of ith layer die is Y_{die_i}. The yield for wafer-to-wafer stacking Y_{stack} can be roughly modeled as [8]

$$Y_{\text{stack}} = \prod_{i=1}^{l} (Y_{\text{die}_i}). \tag{1}$$

Therefore, the defects exist in each die will certainly affect the overall chip yield after stacking.

Besides, during bonding, any foreign particle caught between the wafers can lead to peeling, as well as delamination, which dramatically reduces bonding quality and yield [16]. Y_{Bonding} captures the yield loss of the chip due to faults in the bonding processes.

According to the cumulative yield property, the yield of a 3-D chip $Y_{\text{3-D}-\text{chip}}$ can be formulated as follows [8]:

$$Y_{\text{3-D}-\text{chip}} = Y_{\text{stack}} \cdot \prod_{i=1}^{l-1} \left(Y_{\text{Bonding}(i)} \cdot Y_{\text{TSV}(i)} \right) \tag{2}$$

where $Y_{\text{Bonding}(i)}$ is the yield of the ith bonding step, and $Y_{\text{TSV}(i)}$ is the TSV yield in the ith layer. In this paper, we focus on the yield enhancement of 3-D chip in terms of TSV yield Y_{TSV} [14]. The total TSV yield Y_{TSV} is calculated by multiplying all f-TSV groups yield Y_{gj} as follows:

$$Y_{\text{TSV}} = \prod_{j=1}^{N} Y_{gj} \tag{3}$$

where N is the number of f-TSV groups. In this paper we adopt the algorithm described in [14] for the calculation of group yield Y_{gj}.

B. TSV Fault-Tolerance Structure

By inserting the multiplexers (including control circuits) and carefully designing the reconfigurable TSV replacing paths, we can construct TSV fault-tolerance structures, where the s-TSVs can be used to transfer signals in the presence of faulty f-TSVs [5].

Given an f-TSV planning result, we know the number and position of all f-TSVs. Then we perform a top-down iterative f-TSV partitioning to form f-TSVs groups and allocate s-TSVs in the whitespace for each group. The number and position of used s-TSVs for each f-TSV group are determined simultaneously in the f-TSV partitioning stage. Fig. 1(a) shows an example of a TSV group with four f-TSVs ($f_1 \cdots f_4$) and two s-TSVs (s_1 and s_2). Here, $f_1 \cdots f_4$ belong to nets $nt_1 \cdots nt_4$, respectively. The dashed large rectangles represent the bounding boxes of different nets. Without loss of generality, we denote the bounding box of an f-TSV f_i as the bounding box

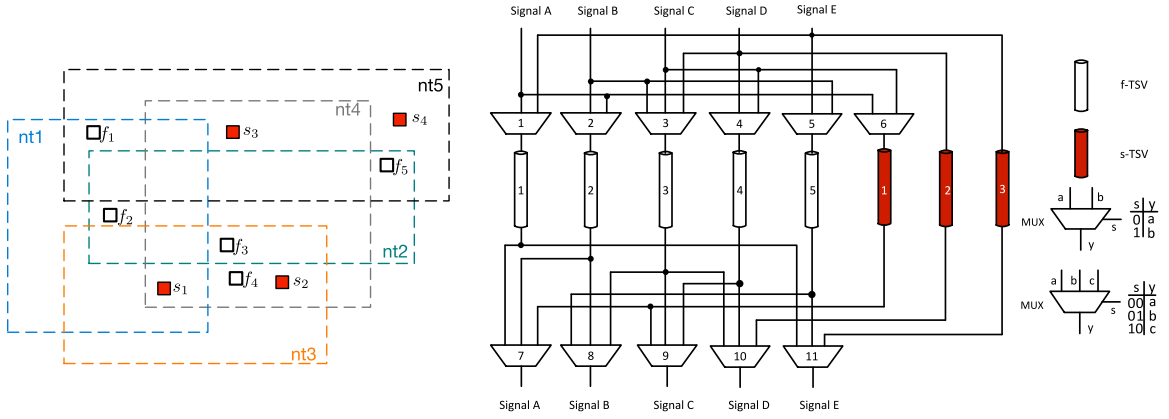


Fig. 2. (a) Example of TSV group with five f-TSVs and four s-TSVs, which cannot be handled by previous works. (b) AFTS generated by our proposed methodology.

of the net f_i belonging to. We say that an f-TSV f_i can be replaced by another TSV v, if and only if v is located inside or nearby the bounding box of f_i. Note that here the TSV v can be either f-TSV or s-TSV. For example, f_1 is replaceable by f_2, f_3, s_1, s_2, since these four TSVs are covered by the bounding box of f_1.

Given a TSV group with some f-TSVs and K s-TSVs, a K-fault tolerance structure includes K independent directed TSV-replacing paths from each f-TSV to s-TSVs. In this structure we can repair at most K faulty f-TSVs through multiplexer rerouting. For instance, for the TSV group shown in Fig. 1(a), a 2-fault tolerance structure with two s-TSVs can be generated as in Fig. 1(b), where each f-TSV is directly connected to all s-TSVs. Although the design scheme is very simple, this structure suffers from large delay overhead due to large multiplexer input size. Some recent works [14], [15] proposed a regular K-fault tolerance structure, as shown in Fig. 1(c). Here, each f-TSV is regularly connected to two right side neighboring TSVs and the rightmost f-TSVs are connected to s-TSVs. Instead of 4-port multiplexers occupied in Fig. 1(b), here only 3-port multiplexers and 2-port multiplexers are needed. For each f-TSV, the independent TSV-replacing paths are listed as follows:

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

To ensure the existence of fault-tolerance structures in TSV groups, the previous works (e.g., [14] and [15]) form TSV groups under two constraints: 1) K fault-tolerance structures use exactly K s-TSVs and 2) an f-TSV in a group can be replaced by any s-TSV within the group. Fig. 1(a) shows an example of TSV group having two-fault tolerance structures, where all the f-TSVs, f_1, f_2, f_3, and f_4, can be replaced by both s_1 and s_2 considering the net bounding boxes. Unfortunately, general cases may violate these constraints. Fig. 2(a) shows a generalized example, where five f-TSVs ($f_1 \cdots f_5$) and four s-TSVs ($s_1 \cdots s_4$) are involved. The replaceable relations between TSVs are shown in Fig. 3(a). In this TSV group, the constraint 1) is violated since we cannot find two-fault tolerance structures if only two s-TSVs are used. The constraint 2) is also violated even if the group is partitioned into smaller groups since f_2 have no replaceable s-TSVs. Consequently, the method in [14] cannot generate cost-effective fault-tolerance structures for this TSV group, because f_2 has no candidate s-TSVs. The ILP-based method in [15] cannot generate fault-tolerance structures for this TSV group since the number of tolerant faults is unknown. However, the f-TSV group definitely includes a two-fault tolerance structure as shown in Fig. 3(c), where three out of four s-TSVs are used in the fault-tolerance structure. The possible TSV replacing paths are as follows:

$$\begin{aligned}
f_1&: \quad \{f_1 \to s_1\}, \quad \{f_1 \to f_2 \to f_3 \to f_4 \to s_2\}\\
f_2&: \quad \{f_2 \to f_5 \to f_1 \to s_1\}, \quad \{f_2 \to f_3 \to f_4 \to s_2\}\\
f_3&: \quad \{f_3 \to s_1\}, \quad \{f_3 \to f_4 \to s_2\}\\
f_4&: \quad \{f_4 \to f_3 \to s_1\}, \quad \{f_4 \to s_2\}\\
f_5&: \quad \{f_5 \to f_1 \to s_1\}, \quad \{f_5 \to s_3\}.
\end{aligned}$$

In reality, there is no essential difference between f-TSVs and s-TSVs. Therefore, the existing TSV testing technique can be directly adopted to test f-TSVs and s-TSVs [17]. And the control signal of multiplexers can be set to determine the direction of signal transfer. As shown in Fig. 2(b), the control signal of 2-to-1 and 3-to-1 multiplexer are 1-bit and 2-bit, respectively. When all TSVs are fault-free or existing faulty s-TSVs, the control signals of each multiplexer are set to transfer signal through their corresponding f-TSVs. But once an f-TSV is faulty, the reconfigurable routing paths can be determined by the corresponding control signal of multiplexers. For instance, when f-TSV 1 is faulty, the control signals of multiplexer 6 and 7 are set to 0 and 10, causing s-TSV 1 to reroute the signal A.

C. Hardware Cost and Multiplexer Delay Overhead

The hardware cost incurred by the fault-tolerance structure can be divided into several parts, including the area overhead

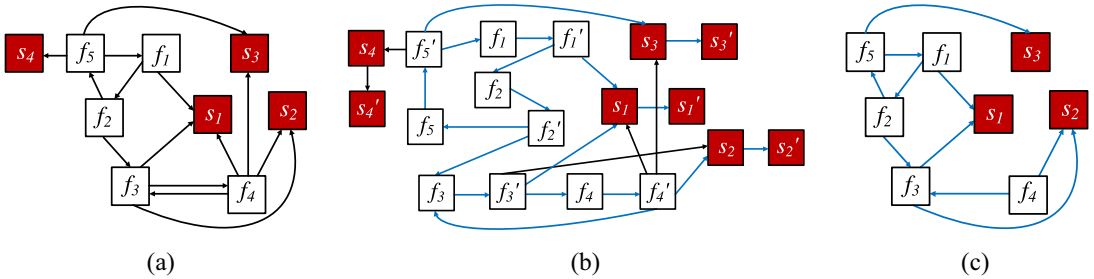


Fig. 3. (a) Corresponding directed graph G of layout in Fig. 2(a). (b) Corresponding splitting graph G'. (c) 2-fault tolerance structure on graph G.

due to inserted s-TSVs, related control logic (i.e., MUXes), and rerouting interconnect [14]. And the cost is dominated by the first two parts [13]. Jiang *et al*. [18] pointed out that the area of control logic is negligible compared with the TSV size and the TSV manufacturing cost is much larger than logic gates. Therefore, in order to reduce the hardware cost, we should reduce the number of s-TSVs used in the fault-tolerance structures.

The delay of a multiplexer is increased along with the number of ports. Therefore, a large multiplexer will introduce large delay overhead. Moreover, the proposed TSV fault tolerance planning is performed in floorplanning stage and we have no exact timing information. If we minimize the multiplexer delay overhead in this stage, we could alleviate the timing closure issue in next placement and routing stage. Therefore, in this paper, we consider the multiplexer delay overhead as one of optimization objectives.

D. Problem Formulation

From the example in Fig. 2, we can see that we confront new design challenging if not all s-TSVs can be occupied in constructing K-fault tolerance structure. Given a TSV group with m f-TSVs and n s-TSVs, we first construct a directed graph $G(V, E)$ consisting of all TSV replaceable relations. Here, 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 $v\}$. Given the TSV group in Fig. 2(a), the corresponding replaceable relation graph is shown in Fig. 3(a).

We define the problem of *TSV fault-tolerance structure generation* as follows.

Problem 1: Given a TSV group with m f-TSVs and n s-TSVs, and the directed graph $G(V, E)$, we search for the maximum number of tolerant faults K. Then we generate a K-fault tolerance structure, which includes K independent TSV replacing paths (vertex-disjoint) for each f-TSV, to minimize both the multiplexer delay overhead and the number of used s-TSVs.

Notice that the yield of the TSV group is evaluated based on the allocated s-TSVs and f-TSVs. With the yield of TSV groups, the total TSV yield can be calculated as discussed in Section II-A. If the target TSV yield is not satisfied, a TSV group will be selected and partitioned into two smaller new TSV groups, where the above *TSV fault-tolerance structure generation* problem will be solved again. New TSV groups will be iteratively generated until the target chip yield is satisfied.

III. Max Flow-Based Methodology

Given a TSV group with replaceable relation graph G, we say the TSV group has a K-fault tolerance structure if each f-TSV $f \in V_1$ has K paths to s-TSV vertices in G. Besides, for each f-TSV f, the paths are vertex-disjoint except the f itself. In this section, we develop a polynomial time algorithm to determine the K value in a TSV group. Our methodology is based on the Menger's theorem as follows.

Lemma 1 (Menger's Theorem [19]): Let G be a directed graph, and let S and T be distinct vertices in G. Then the maximum number of vertex-disjoint S-T paths is equal to the minimum size of an S-T disconnecting vertex set.

Here, the S–T disconnecting vertex set represents a vertex set whose removal will cause no paths from any vertex in S to any vertex in T. According to Lemma 1, for each f-TSV f, the number of vertex-disjoint paths $Nd(f)$ equals to the minimum size of the $\{f\}$-V_2 disconnecting vertex set in G. For example, in Fig. 3(a), $\{f_2, s_1\}$ is a minimum $\{f_1\}$-V_2 disconnecting vertex set. Therefore, the number of vertex-disjoint paths, $Nd(f_1)$, equals to 2. Based on above lemma, we reach the following theorem.

Theorem 1: Given the replaceable relation graph, the maximum number of tolerant faults, K, can be determined in polynomial time, as follows:

$$\mathrm{K} = \min_{f \in V_1} \{Nd(f)\}. \tag{4}$$

Since vertex-disjoint problem is not easy to model, we perform vertex splitting on $G(V, E)$ so that it can be transformed into an edge-disjoint problem, which can be appropriately modeled in a maximum flow problem. Each vertex $u \in V$ is split into two vertices u and u', respectively, corresponding to the vertex's input and output, and an extra edge (u, u') with zero cost 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 vertex set of V_1 and V_2' is the split vertex 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 vertex of $u\}$ and $E_2' = \{(u', v) | (u, v) \in E(G) \wedge u'$ is the corresponding split vertex 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 $E'(G')$.

TABLE I
NOTATIONS USED IN ILP

V, V'	set of f-TSVs and s-TSVs, set of split f-TSVs and split s-TSVs
V_1, V_1'	set of f-TSVs, set of split f-TSVs
V_2, V_2'	set of s-TSVs, set of split s-TSVs
f_i, f_i'	f-TSV in V_1, split f-TSV in V_1'
s_j, s_j'	s-TSV in V_2, split s-TSV in V_2'
E'	set of all edges in graph G'
E_1'	set of all splitting edges in graph G' ($f_i \rightarrow f_i'$ and $s_j \rightarrow s_j'$)
E_2'	set of all replaceable edges in graph G'
(w, w')	edge in E_1' and w in V_2
s, t	split f-TSV in V_1', split s-TSV in V_2'
$v^{(s,t)}$	binary variable; if a unit flow (path) exists from s to t then $v^{(s,t)} = 1$, otherwise $v^{(s,t)} = 0$
(v, u)	edge in E'
$x_{vu}^{(s,t)}$	binary variable; if a unit flow (path) from s to t goes through edge (v, u), then $x_{vu}^{(s,t)} = 1$, otherwise $x_{vu}^{(s,t)} = 0$
d_{vu}	binary variable on edge (v, u); if a unit flow (path) goes through edge (v, u), then $d_{vu} = 1$, otherwise $d_{vu} = 0$

Based on the splitting graph, the maximum number of tolerant faults K can be determined in polynomial time by solving a max-flow problem [19] for each f-TSV. For instance, given the replaceable relation graph $G(V, E)$ in Fig. 3(a), Fig. 3(b) illustrates the splitting graph $G'(V', E')$. The number of edge-disjoint paths for each f-TSV is as follows, $Nd(f_1) = 2$, $Nd(f_2) = 2$, $Nd(f_3) = 3$, $Nd(f_4) = 3$, and $Nd(f_5) = 3$. Since f_1 and f_2 have only two edge-disjoint paths, the maximum number of tolerant faults, K, equals to 2.

The fault-tolerance structure can be generated by finding $m \times K$ paths, which begin with each split f-TSV in V_1' and end with split s-TSV in V_2'. In addition, all the paths sharing one same source vertex should be edge-disjoint. In the next two sections, we will propose an ILP-based algorithm and an MCMF-based heuristic method to generate the K-fault tolerance structure in minimizing both the used s-TSV number and the multiplexer delay overhead.

IV. INTEGER LINEAR PROGRAMMING FORMULATION

In this section, we discuss how the K edge-disjoint path search problem can be formulated as an integer programming. For convenience, some notations used in this section are listed in Table I.

First, an integer programming formulation in [15] is given to generate the fault-tolerance structures with minimization of the multiplexer delay overhead.

To model the delay of each multiplexer, it is of importance calculating indegree of each vertex $u \in V$. As shown in Fig. 3(b), 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 vertex on the paths. For instance, due to edges (f_2', f_3) and (f_4', f_3), the indegree of f_3 should be increased by two. Given a vertex $u \in V$, its indegree is calculated by the following equation:

$$\text{indegree}(u) = \sum_{v:(v,u)\in E'} \min\left(\sum_{s\in V_1', t\in V_2'} x_{vu}^{(s,t)}, 1 \right). \tag{5}$$

The starting integer programming formulation of fault-tolerance structure generation problem in [15] is shown in (6). The objective function in (6) is to minimize the maximum indegree of all the vertices. The number of binary variables $x_{vu}^{(s,t)}$ 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 (6a) defines a unit flow from $s \in V_1'$ to $t \in V_2'$, which corresponds a path from s to t. The number of this set of constraints is $m \times n \times |V'|$. The constraint (6b) ensures that a set of V_2' paths, which have the same source $s \in V_1'$, are edge-disjoint. The number of this set of constraints is $m \times (m + n)$

$$\min \max_{u\in V} \text{indegree}(u) \tag{6}$$

$$\text{s.t.} \sum_{v:(u,v)\in E'} x_{uv}^{(s,t)} - \sum_{v:(v,u)\in E'} x_{vu}^{(s,t)} = \begin{cases} 1, & \text{if } u = s \\ 0, & \text{if } u \in V' - \{s, t\} \\ -1, & \text{if } u = t \end{cases}, \quad \forall s \in V_1', t \in V_2' \tag{6a}$$

$$\sum_{t\in V_2'} x_{uu'}^{(s,t)} \le 1, \quad \forall s \in V_1', (u, u') \in E_1' \tag{6b}$$

$$x_{vu}^{(s,t)} \in \{0, 1\}, \quad \forall (v, u) \in E', s \in V_1', t \in V_2'. \tag{6c}$$

Though the integer programming method in [15] can generate K fault-tolerance structures using K s-TSVs, the method cannot be directly applied for the generation of AFTSs, where the number of required s-TSVs might be larger than K in K fault-tolerance structures. Then a new integer programming formulation is proposed to generate AFTSs in minimizing both the used s-TSV number and the multiplexer delay overhead. The number of s-TSVs used in the structure can be calculated by

$$\text{usedstsv} = \sum_{w\in V_2} \min\left(\sum_{s\in V_1', t\in V_2'} x_{ww'}^{(s,t)}, 1 \right). \tag{7}$$

Based on the above notations, the edge-disjoint path search problem can be formulated as the following integer programming (8).

Compared with the integer programming (6), in constraint (8a) a new binary variable $v^{(s,t)}$ is introduced to indicate whether a unit flow (path) exists from source $s \in V_1'$ to sink $t \in V_2'$. Besides, a new constraint (8b) is defined to ensure that there will be K paths from each source $s \in V_1'$ to vertices in V_2'. The number of this set of constraints is m. By this way, (8) can be applied for any $K \le n$ and additionally minimize the number of required s-TSVs in the structure, while (6) can only

be applied for the case $K = n$

$$\min \left\{ \max_{u \in V} \text{indegree}(u) + \text{usedstsv} \right\} \tag{8}$$

$$\text{s.t.} \sum_{v:(u,v)\in E'} x_{uv}^{(s,t)} - \sum_{v:(v,u)\in E'} x_{vu}^{(s,t)} = \begin{cases} v^{(s,t)}, & \text{if } u = s \\ 0, & \text{if } u \in V' - \{s,t\}, \\ -v^{(s,t)}, & \text{if } u = t \end{cases} \quad \forall s \in V_1', t \in V_2' \tag{8a}$$

$$\sum_{t \in V_2'} v^{(s,t)} = K, \quad \forall s \in V_1' \tag{8b}$$

$$v^{(s,t)} \in \{0,1\}, \quad \forall s \in V_1', t \in V_2' \tag{8c}$$

$$(6b) - (6c).$$

Equation (8) is nonlinear due to the min-max-min and min-min operations in the objective function. Through linearizing the objective function, (8) can be transformed into an ILP (9). For each edge $(v, u) \in E'$, an extra binary variable d_{vu} and extra constraints (9a)–(9c) are introduced to replace the min operation in (5) and (7). Besides, the extra constraint (9d) ensures that the indegrees of all TSVs will not be greater than λ_1. Another extra constraint (9e) ensures that the number of s-TSVs used in the structure equals to λ_2

$$\min \ (\lambda_1 + \lambda_2) \tag{9}$$

$$\text{s.t.} \ d_{vu} \geq x_{vu}^{(s,t)}, \quad \forall s \in V_1', t \in V_2', (v,u) \in E' \tag{9a}$$

$$d_{vu} \leq \sum_{s \in V_1', t \in V_2'} x_{vu}^{(s,t)}, \forall (v,u) \in E' \tag{9b}$$

$$d_{vu} \in \{0,1\}, \quad \forall (v,u) \in E' \tag{9c}$$

$$\sum_{v:(v,u)\in E'} d_{vu} \leq \lambda_1, \quad \forall u \in V \tag{9d}$$

$$\sum_{(w,w')\in E_1'} d_{ww'} = \lambda_2, \quad \forall w \in V_2 \tag{9e}$$

$$(6b) - (6c), \ (8a) - (8c).$$

For instance, as shown in Fig. 3(b), the blue lines present edge-disjoint paths for each split f-TSV, and the corresponding generated 2 fault-tolerance structure is shown in Fig. 2(b).

V. Heuristic Framework

For large TSV groups, the ILP-based method is very time consuming. Consequently, in this section, we propose an MCMF-based heuristic method to solve the edge-disjoint path problem. The basic idea is to deal with f-TSVs one by one and, for each f-TSV, an MCMF algorithm is used to find K independent paths. The edge costs are defined to keep the input port number of multiplexer and the number of s-TSVs as small as possible.

A. Network Graph Model

In order to find K ($K \leq n$) edge-disjoint paths for an f-TSV $f_i \in V_1$, we construct a directed graph $G_s(V_s, E_s)$ from G' by adding an extra sink vertex t and some edges. The vertex set V_s contains two portions, $V_s = V' \cup \{r\}$, and r is the sink vertex. The edge set $E_s = E' \cup \{V_2' \to r\}$.

When finding edge-disjoint paths for a certain TSV $f_i \in V_1$, the edge capacities are defined as follows: the capacity of the edge from f_i to its splitting vertex f_i' equals to K; while the capacities of all the other edges are set to 1. The capacity constraints ensure that we can find up to K edge-disjoint paths from f_i' to s-TSV vertices, which correspond to K independent TSV-replacing chains for the TSV f_i.

For the splitting edges corresponding to f-TSVs, the edge costs are defined as zero while the splitting edges corresponding to s-TSVs are defined as follows:

$$ec_s(w,w') = \begin{cases} 0, & \text{if } (w,w') \in E_1', w \in V_2, \text{ and } w \text{ has been used} \\ C^K, & \text{if } (w,w') \in E_1', w \in V_2, \text{ and } w \text{ has not been used.} \end{cases} \tag{10}$$

C is constant, which represents the costs of introducing a new s-TSV for constructing the fault-tolerance structure. And the edge costs tend to restrict the use of s-TSVs. In the experiment, we set C to 3 by the experimental results shown in Section VII-A.

For the edges in E_2', which correspond to the replaceable relations between TSVs, the edge costs are defined as follows:

$$ec_s(u,v) = \begin{cases} 0, & \text{if } (u,v) \in E_2' \text{ and } (u,v) \text{ corresponds to a TSV connection} \\ C^{tc[v]}, & \text{if } (u,v) \in E_2' \text{ and } (u,v) \text{ does not correspond to a TSV connection.} \end{cases} \tag{11}$$

In the edge cost function (11), $tc[v]$ is defined to be the number of edges that end at v and have been used as TSV connections in the generated partial fault-tolerance structure, that is, the edges that have been traversed by edge-disjoint paths of some other f-TSVs. Therefore, $tc[v]$ corresponds to the input port number of the multiplexer in the input side of the TSV v.

With this edge costs function, first, we tend to make full use of existing TSV connections to build the edge-disjoint paths for the current f-TSV since it will not increase the input ports of the multiplexers.

Second, to minimize the maximum size of multiplexers, the costs of the edges that do not correspond to TSV connections are defined as the exponential function of $tc[v]$.

B. Algorithmic Flow of Heuristic

The algorithmic flow of the proposed heuristic is summarized in Algorithm 1. Because the quality of solution depends on the order of f-TSVs selected, an iterative post-processing stage is used to improve the generated fault-tolerance structures. In the post-processing stage, we randomly select an f-TSV, and define the edge costs based on the TSV paths of all the other f-TSVs. Then we resolve the MCMF model to find edge-disjoint paths for the selected f-TSV. The procedure is repeated until the multiplexer maximum input port number keeps unchanged over a predefined threshold iteration number.

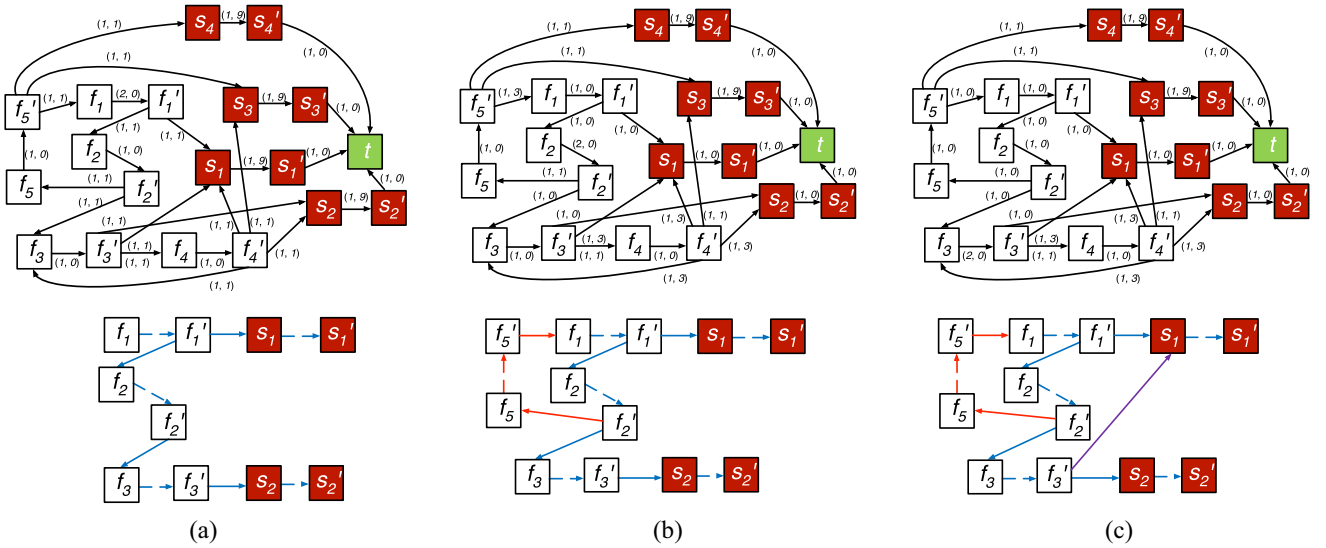


Fig. 4. Label on edges represents (*capacity*, *cost*). (a) MCMF network for f-TSV f_1', where the two edge-disjoint paths for f_1': $\{f_1' \rightarrow s_1 \rightarrow s_1'\}$ and $\{f_1' \rightarrow f_2 \rightarrow f_2' \rightarrow f_3 \rightarrow f_3' \rightarrow s_2 \rightarrow s_2'\}$. (b) After solving f_1', the MCMF network for f-TSV f_2', where the two edge-disjoint paths for f_2': $\{f_2' \rightarrow f_5 \rightarrow f_5' \rightarrow f_1 \rightarrow f_1' \rightarrow s_1 \rightarrow s_1'\}$ and $\{f_2' \rightarrow f_3 \rightarrow f_3' \rightarrow s_2 \rightarrow s_2'\}$. (c) After solving f_1' and f_2', the MCMF network for f-TSV f_3', where the two edge-disjoint paths for f_3': $\{f_3' \rightarrow s_1 \rightarrow s_1'\}$ and $\{f_3' \rightarrow s_2 \rightarrow s_2'\}$.

Algorithm 1 Pseudo Code of Our Heuristic Method

Input: A directed graph $G'(V', E')$, which contains m f-TSVs and n s-TSVs.

Output: A repairable structure including $m \times K$ paths.

1: **for** f-TSV $f_i \leftarrow 1$ to m **do**
2: Construct a directed graph $G_s(V_s, E_s)$ for f_i;
3: ▷ Find K edge-disjoint paths for f_i;
4: Solve the MCMF model for f_i;
5: **end for**
6: ▷ Perturb the repairable structure;
7: **while** no coverage **do**
8: Randomly select an f-TSV f_i;
9: Resolve edge-disjoint paths for f_i by MCMF;
10: Record the maximum number of TSV connections on all TSVs;
11: **end while**

Fig. 4(a)–(c) illustrate the process of the heuristic method. We choose the f-TSV f_1' to start with. The MCMF network for f_1' is shown in Fig. 4(a). All the costs of edges that end at f-TSVs and s-TSVs are initialized at 1 since there are no any other f-TSV paths and for all v, $tc[v] = 0$. By solving the MCMF, 2 edge-disjoint paths, which correspond to two independent TSV replacing chains for f_1, are obtained and the TSV connections (solid edges) in the partial fault-tolerance structure.

With the 2 edge-disjoint paths for f_1, the flow network is updated (edge costs and capacities) for f-TSV f_2' and shown in Fig. 4(b). The edges that are on the edge-disjoint paths of f_1 have zero costs. Considering the vertex s_1, for example, the edge (f_1', s_1) has zero costs since it has been traversed by the TSV path of f_1 while the edges (f_3', s_1) and (f_4', s_1) have a cost of 3 because the both edges are not traversed by any TSV paths

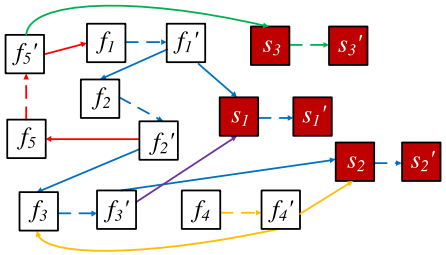


Fig. 5. Generated 2-fault tolerance structure by solving edge-disjoint paths for all f-TSVs, where the TSV connections are shown in solid edges.

of f_1 and $tc[s_1] = 1$. A new TSV connection will be introduced if we use (f_3', s_1) or (f_4', s_1) on the edge-disjoint paths for f_2, which will increase the input ports of multiplexer in the input side of the TSV s_1. With the updated network, we can find two edge-disjoint paths from f_2' to s-TSVs by making use of the existing TSV connections as many as possible, which potentially reduces the TSV connections on individual TSVs and minimizes the maximum number of the input ports of multiplexers. The bottom part of Fig. 4(b) shows the TSV connections in the updated partial fault-tolerance structure.

Repeating the same process until the MCMF model is solved for all f-TSVs, we obtain 2 edge-disjoint paths from each split f-TSV vertex in $V_1', f_1' \cdots f_5'$, to split s-TSV vertices in $V_2', s_1' \cdots s_3'$, as shown in Fig. 5. Here, the solid edges are TSV connections.

VI. Fault Tolerance TSV Planning

In this section, we discuss a top-down fault tolerance TSV planning framework to form f-TSV groups and generate AFTSs. The number of f-TSV groups is greatly reduced as well as the total number of s-TSVs because of AFTSs.

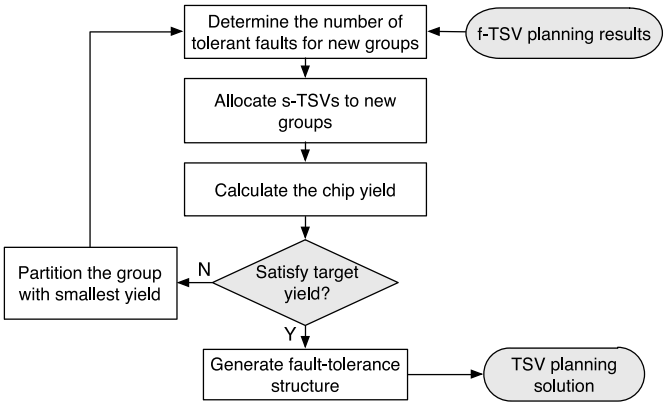


Fig. 6. Flow of the proposed fault tolerance TSV planning.

Given an f-TSV planning result and the floorplan of the blocks, we know the number and positions of all f-TSVs. Then f-TSV groups are first formed using a top-down iterative f-TSV partitioning under the yield constraint and, then, the AFTSs are generated for each group. In each iteration of the f-TSV partitioning stage, the group with the smallest yield will be partitioned into two new f-TSV groups using the min-cut bi-partitioning algorithm and the required s-TSVs are also allocated for evaluating the group yield. The iterative f-TSV partitioning is repeated until the target chip yield is satisfied. Therefore, the number and position of required s-TSVs for each f-TSV group are determined simultaneously in the f-TSV partitioning stage.

The chip yield is the product of group yield, which depends on the maximum number of tolerant faults (K), the number of TSVs, and the defect probability of TSVs as discussed in Section II-A. We construct the replaceable relation graph G, whose vertex set includes the f-TSVs in the group and the corresponding candidate s-TSVs, for computing K and allocating s-TSVs. The maximum number of tolerant faults, K, can be determined in polynomial time by solving a max-flow problem on G, as discussed in Section III. The MCMF-based heuristic in Section V is used to temporarily generate an adaptive K-fault tolerance structure, thus the number of required s-TSVs is determined.

Finally, the ILP-based method in Section IV and the MCMF-based heuristic in Section V can be adopted to generate AFTSs with minimization of both the multiplexer delay overhead and the hardware cost. Fig. 6 illustrates the proposed TSV planning framework.

In [14], a greedy method is used to partition f-TSVs into groups and then an ILP formulation is adopted to allocate s-TSVs for each group. The generation of fault-tolerance structure is not considered since they assume regular structures always exist. In [15], the TSV planning framework includes a top-down partitioning followed by a bottom-up iterative merging (clustering) for reducing the number of f-TSV groups. Then, an MCMF-based method is used to allocate s-TSVs for each group and an ILP model is adopted to generate fault-tolerance structures. The same number of s-TSVs is allocated to all the f-TSV groups in [14] and [15] and, for an f-TSV group, the key point is to ensure enough number of candidate s-TSVs that can be shared by all the f-TSVs in the group. As a result, many small f-TSV groups are formed, which potentially causes an overuse of s-TSVs.

Compared with the above two works, the proposed TSV planning framework includes a similar top-down partitioning stage, but the allocation of s-TSVs during the partitioning is quite different. That is because AFTSs with various number of s-TSVs are built temporarily by solving a series of MCMF problems.

VII. Experimental Results

The proposed algorithms have been implemented in C++ language and tested on a 12-core 2.0 GHz Linux server with 64 GB RAM. The TSV pitch is assumed to be 5 um $\times$ 5 um [3]. LEDA [20] is adopted to solve the max-flow and the MCMF problems. GLPK [21] is used as the ILP solver. hMetis [22] is adopted on f-TSVs partitioning.

A. Effectiveness and Efficiency of Fault-Tolerance Structure Generation Method

We generate several TSV replaceable relation graphs G_{11}–G_{18} by using the proposed TSV planning framework on MCNC and GSRC benchmarks. Each graph contains f-TSVs and the corresponding candidate s-TSVs, which are covered by at least one of the bounding boxes of the f-TSVs. In order to compare the proposed ILP model with the ILP method in [15] on G_{11}–G_{18}, we adapt the ILP formulation in [15] here. To generate the K-fault tolerance structure on a TSV replaceable relation graph G, we select K s-TSVs in all n s-TSVs, and unit flow constraints are defined from all f-TSVs to those chosen K s-TSVs. If the K-fault tolerance structure is still not achieved after solving all K combinations, we think the ILP method in [15] cannot generate the K-fault tolerance structure on this TSV replaceable relation graph G.

In addition, the previous work in [15] deals with a special type of TSV fault-tolerance structure generation. That is, they are under an assumption that a predetermined number of s-TSVs is assigned to each TSV group, and an f-TSV in a group should be replaced by any s-TSV within the group. We also generate some specific TSV replaceable relation graphs G_{21}–G_{28} by using the TSV planning methods in [15] on MCNC and GSRC benchmarks. Since the f-TSVs can be replaced by all n s-TSVs in each graph, the n-fault tolerance structure always exists.

First, we show the effectiveness of the proposed ILP model. Table II shows the experimental results, where "ILP" and "Heuristic" denote results of the proposed ILP model and MCMF-based heuristic method, respectively. Columns "m," "n," "#Edges," and "K" list the number of f-TSVs, the total number of available s-TSVs, the number of edges, and the number of maximumly tolerant faults on each TSV replaceable relation graph. Besides, columns "#Port" and "#us" show the maximum port number of multiplexers and the number of s-TSVs used in the generated fault-tolerance structure. "IWire" shows the sum of incremental half-perimeter wirelength overhead of all f-TSVs incurred by the fault-tolerance structure,

TABLE II
COMPARISON BETWEEN ILP [15] AND OUR METHODS FOR GENERATING AFTS

Graph	m	n	#Edges	K	ILP [15]				ILP				Heuristic			
					#Port	#us	IWire(um) (ratio)	RT(s)	#Port	#us	IWire(um) (ratio)	RT(s)	#Port	#us	IWire(um) (ratio)	RT(s)
G_{11}	9	4	72	3	3	3	32.90 (0.51%)	535.20	3	3	32.90 (0.51%)	301.53	3	4	25.88 (0.40%)	0.008
G_{12}	13	4	129	2	3	2	6.85 (0.18%)	603.68	3	2	9.65 (0.25%)	67.80	3	4	16.79 (0.43%)	0.013
G_{13}	14	4	101	1	NA	NA	NA	>3600	2	4	29.77 (1.77%)	1.09	3	4	28.99 (1.72%)	0.006
G_{14}	15	5	177	2	NA	NA	NA	>3600	3	4	32.50 (0.62%)	96.90	3	4	32.71 (0.62%)	0.009
G_{15}	18	5	215	2	NA	NA	NA	>3600	3	4	65.20 (0.96%)	240.07	4	5	52.35 (0.77%)	0.013
G_{16}	18	6	199	2	NA	NA	NA	>3600	3	6	90.03 (1.76%)	155.74	3	6	98.36 (1.93%)	0.011
G_{17}	21	7	255	2	NA	NA	NA	>3600	NA	NA	NA	>3600	4	6	214.39 (1.83%)	0.017
G_{18}	26	13	529	4	NA	NA	NA	>3600	NA	NA	NA	>3600	4	12	333.60 (1.47%)	0.038
G_{21}	9	5	99	5	4	5	16.34 (0.15%)	100.84	4	5	16.34 (0.15%)	101.10	4	5	16.34 (0.15%)	0.005
G_{22}	12	5	155	5	5	5	49.77 (0.25%)	304.91	5	5	49.77 (0.25%)	306.14	6	5	56.26 (0.28%)	0.007
G_{23}	14	5	197	5	5	5	10.21 (0.06%)	3435.64	5	5	10.21 (0.06%)	3468.93	5	5	11.84 (0.07%)	0.010
G_{24}	16	5	225	5	5	5	108.19 (0.71%)	3519.16	5	5	108.19 (0.71%)	3519.16	7	5	123.18 (0.81%)	0.016
G_{25}	18	5	329	5	NA	NA	NA	>3600	NA	NA	NA	>3600	5	5	72.01 (0.26%)	0.016
G_{26}	23	6	467	6	NA	NA	NA	>3600	NA	NA	NA	>3600	6	6	45.99 (0.12%)	0.027
G_{27}	24	6	550	6	NA	NA	NA	>3600	NA	NA	NA	>3600	6	6	30.65 (0.08%)	0.034
G_{28}	25	7	524	7	NA	NA	NA	>3600	NA	NA	NA	>3600	7	7	24.65 (0.06%)	0.037

TABLE III
EFFECT OF C ON S-TSV NUMBERS AND MAXIMUM PORT NUMBER OF MULTIPLEXERS

Benchmark	C = 2		C = 3	
	#s-TSV	#Port	#s-TSV	#Port
ami33	52	4	46	4
ami49	80	8	66	6
n50	108	7	98	7
n100	181	8	169	7
n200	267	7	250	7
n300	395	8	381	6

and the ratio of IWire to the sum of net wirelength of all f-TSVs is listed in "ratio." "RT" reports the total computational time in seconds. "NA" represents that the K-fault tolerance structure cannot be achieved within the time limit (3600 s). As shown in Table II, the ILP method in [15] generates the fault-tolerance structure only on two smallest graphs. However, the proposed ILP formulation can achieve the fault-tolerance structure on six graphs.

Second, we show the efficiency of the proposed heuristic method. Table II also compares the proposed heuristic method with the proposed ILP method. It can be noticed that, on small graphs G_{11}–G_{16} and G_{21}–G_{24}, the fault-tolerance structure generated by ILP has smaller maximum port number of multiplexers and used less s-TSV numbers than that generated by the heuristic method. Therefore, for small TSV replaceable relation graphs, ILP can achieve an optimal solution, which can be used to verify the accuracy of the solution generated by the heuristic method. But since ILP is an NP-hard problem, its runtime increases dramatically with the size of TSV replaceable relation graphs. As shown in Table II, the ILP method cannot generate the fault-tolerance structure on large graphs G_{17}–G_{18} and G_{25}–G_{28} within the time limit (3600 s). Therefore, for large TSV replaceable relation graphs, the ILP-based method is very time consuming, which can indirectly demonstrate the efficiency of the proposed heuristic method.

In addition, the parameter C in edge cost functions (10) and (11) is also set through experimental results. The experiment is performed on MCNC and GSRC benchmarks. In the experiment, if C is set to 4, some edge cost values are out of bound, which cannot be solved by MCMF-based model. And we also set C to 2 and 3, the number of used s-TSVs and the maximum port number of multiplexers varied with C, which is shown in Table III. Columns "#s-TSV" and #Port list the total number of allocated s-TSVs and the maximum port number of multiplexers among all f-TSV groups. We noticed that compared with $C = 2$, $C = 3$ can achieve a fault tolerance structure with less number of used s-TSVs and smaller maximum port number of multiplexers. Therefore, in the experiment, we set C to 3.

B. Comparison With Previous TSV Fault Tolerance Planning Work

We use simulated annealing-based multilayer floorplanning [23] to generate the block floorplan and the f-TSV planning method in [15] to generate f-TSV planning result as the input to the proposed fault-tolerance TSV planning framework. Based on the same f-TSV planning result, we run the flow in [14] and [15], and the proposed heuristic-based framework, respectively. The experiment is tested on MCNC and GSRC benchmarks, including two MCNC circuits (ami33 and ami49), and four GSRC circuits (n50, n100, n200, and n300). We adopt one more industrial 2-D design, which contains 403 266 cells and 448 514 nets. hMetis [22] is adopted to partition the design into several blocks for floorplanning. Based on different block numbers, two benchmark cases, t337 and t469, are generated. That is, t337 has 337 blocks and 1836 nets, while t469 has 469 blocks and 5479 nets. Since the square has the smallest perimeter among all the rectangles with the same area [24], here the shapes of all the blocks are set to square. The experiment is executed 20 times independently for each benchmark.

In fault-tolerance structures, the multiplexers are used to reroute signals, and the delay of a multiplexer is increased along with the number of input ports. Besides the hardware cost incurred by the fault-tolerance structure is related to the

TABLE IV
COMPARISONS AMONG [14], [15], AND THE PROPOSED AFTS UNDER 3-FAULT TOLERANCE STRUCTURES (TARGET YIELD = 99.7%, $p = 0.001$)

Bench	#f-TSV	[14]			[15]					AFTS (K≤3)					AFTS (maximum K)				
		#s-TSV	#gp	Yield	#s-TSV	#gp	#Port	K	Yield	#s-TSV	#gp	#Port	K	Yield	#s-TSV	#gp	#Port	K	Yield
ami33	55	48	16	100%	48	16	4	3	100%	31	2	3	3	100%	46	2	4	4	100%
ami49	130	72	24	100%	66	22	5	3	100%	54	2	5	3	99.99%	66	2	6	5	100%
n50	386	210	70	99.97%	204	68	7	3	100%	82	5	6	2	99.96%	98	5	7	5	99.98%
n100	592	294	98	99.91%	291	97	7	3	99.94%	136	7	6	3	99.91%	169	7	7	6	99.93%
n200	1127	396	132	99.86%	393	131	6	3	99.86%	179	8	5	3	99.85%	250	8	7	6	99.86%
n300	1232	501	167	99.81%	498	166	6	3	99.83%	246	9	5	3	99.78%	381	7	6	6	99.80%
t337	640	315	105	99.90%	309	103	4	3	99.91%	158	8	5	3	99.88%	214	6	6	6	99.90%
t469	1546	600	200	99.71%	588	196	6	3	99.73%	313	11	6	3	99.71%	412	9	7	7	99.72%
avg.	714	305	102	99.90%	300	100	6	3	99.91%	150	7	5	3	99.89%	205	6	7	6	99.90%
ratio	–	+48.78%	–	–	+46.34%	–	–	–	–	-26.83%	–	–	–	–	1.00	–	–	–	–

TABLE V
COMPARISONS AMONG [14], [15], AND THE PROPOSED AFTS UNDER 3-FAULT TOLERANCE STRUCTURES (TARGET YIELD = 99.5%, $p = 0.01$)

Bench	#f-TSV	[14]			[15]					AFTS (K≤3)					AFTS (maximum K)				
		#s-TSV	#gp	Yield	#s-TSV	#gp	#Port	K	Yield	#s-TSV	#gp	#Port	K	Yield	#s-TSV	#gp	#Port	K	Yield
ami33	54	51	17	100%	51	17	4	3	100%	35	4	3	3	100%	48	4	4	4	100%
ami49	130	87	29	99.96%	81	27	5	3	99.96%	62	5	4	3	99.94%	73	5	5	4	99.95%
n50	388	231	77	99.89%	222	74	6	3	99.92%	102	8	5	3	99.88%	113	8	7	5	99.90%
n100	589	330	110	99.84%	324	108	6	3	99.87%	165	12	5	3	99.84%	194	11	7	6	99.87%
n200	1130	438	146	99.73%	435	145	7	3	99.74%	210	17	6	2	99.72%	280	15	7	6	99.73%
n300	1236	555	185	99.62%	549	183	6	3	99.63%	295	20	5	3	99.60%	426	20	6	5	99.61%
t337	637	342	114	99.82%	330	110	4	3	99.82%	184	13	4	3	99.78%	227	12	7	7	99.81%
t469	1553	645	215	99.55%	633	211	7	3	99.56%	352	25	6	3	99.52%	455	23	7	6	99.55%
avg.	715	335	112	99.80%	329	110	6	3	99.81%	176	13	5	3	99.79%	227	12	7	6	99.80%
ratio	–	+47.58%	–	–	+44.93%	–	–	–	–	-22.47%	–	–	–	–	1.00	–	–	–	–

number of s-TSVs. In this experiment, we compare the number of s-TSVs and the maximum port number of multiplexers of [14] and [15], and the proposed TSV planning framework under 3-fault tolerance structures. The layer number is set to 3. The target chip yield is set to 99.7% and the TSV defect probability p is set to 0.001. The yield results in experiment are accurate to the fourth decimal place. 3 s-TSVs are assigned to each f-TSV group in [14] and citexu2017clustered, that is, the maximum number of tolerant faults K equals to 3.

Table IV lists the statistic results averaged over 20 independent experiments. All results listed in table satisfy the target chip yield. Column "#f-TSV" represents the total number of f-TSVs. Since the three frameworks are run on the same f-TSV planning result, the number of f-TSVs is the same. Columns #s-TSV, "#gp," and "Yield" list the total number of allocated s-TSVs, the number of groups, and the chip yield, respectively. Besides, column #Port provides the maximum port number of multiplexers among all groups, while column K gives the number of tolerant faults in that group, respectively. Since the generation of fault-tolerance structure is not considered in [14], the maximum port number of multiplexers is not listed. As shown in Table IV, the number of f-TSV groups is greatly reduced in the proposed method. Compared with [14] and [15], the proposed fault tolerance TSV planning framework can reduce the number of used s-TSVs by 48.78% and 46.34% on average, respectively. In addition, in the proposed framework, if the maximum K is used for each group, it will cause larger multiplexers. Because the maximum number of tolerant faults (K) in AFTSs is often much greater than that of [15], which is fixed at 3. As a result, the maximum port number of multiplexers is increased accordingly in the generated fault-tolerance structures.

To reduce the size of required multiplexers, we also run the proposed fault tolerance TSV planning framework with $K \leq 3$, that is, we set K to 3 if the maximum number of tolerant faults K in a group is greater than 3. As shown in Table IV, compared with [15], the proposed fault tolerance TSV planning framework with $K \leq 3$ has comparable maximum port number of multiplexers. But the required s-TSVs are surprisingly reduced by 50% on average under the same target yield, as shown in Table IV.

The TSV defect probability p in [13] ranges from 0.001 to 0.01. In order to see the impact of p on performance, we also execute the experiment when p is set to 0.01 under 3-fault tolerance structures. The layer number is set to 3. The target chip yield is set to 99.5%. Table V lists the statistic results averaged over 20 independent experiments. All results listed in table satisfy the target chip yield. Based on the same f-TSV planning result, we run the flow in [14] and [15], and the proposed heuristic-based framework, respectively. Compared with [14] and [15], the proposed fault tolerance TSV planning framework can reduce the number of used s-TSVs by 47.58% and 46.34% on average, respectively. In order to reduce the size of required multiplexers, we also run the proposed fault tolerance TSV planning framework with $K \leq 3$. As shown in Table V, compared with [15], the proposed fault tolerance TSV planning framework with $K \leq 3$ has comparable maximum port number of multiplexers. But the required s-TSVs are reduced by 46.50% on average under the same target yield, as shown in Table V.

Besides, in [7], 1-fault tolerance structures are generated using minimum spanning tree-based method. However, it is difficult to apply the method to the fault-tolerance structure using more than one spare TSVs. In addition, the delay

TABLE VI
COMPARISONS AMONG [7], [14], [15], AND THE PROPOSED AFTS UNDER 1-FAULT TOLERANCE STRUCTURES (TARGET YIELD = 99.5%)

Bench	#f-TSV	[14]			[7]				[15]				AFTS (K=1)			
		#s-TSV	#gp	Yield	#s-TSV	#gp	#Port	Yield	#s-TSV	#gp	#Port	Yield	#s-TSV	#gp	#Port	Yield
ami33	52	16	16	99.99%	16	16	4	99.99%	16	16	3	99.99%	13	2	2	99.99%
ami49	124	28	28	99.95%	25	25	5	99.96%	25	25	4	99.96%	22	3	3	99.95%
n50	383	74	74	99.84%	68	68	8	99.87%	68	68	4	99.87%	53	8	3	99.84%
n100	596	108	108	99.65%	95	95	8	99.68%	95	95	5	99.68%	78	12	4	99.64%
n200	1126	141	141	99.61%	132	132	8	99.64%	132	132	6	99.64%	110	22	5	99.61%
n300	1230	197	197	99.51%	183	183	9	99.53%	183	183	6	99.53%	158	31	5	99.51%
t337	639	124	124	99.65%	113	113	8	99.67%	113	113	6	99.67%	91	16	5	99.64%
t469	1551	252	252	99.50%	236	236	8	99.52%	236	236	6	99.52%	214	40	5	99.50%
avg.	713	118	118	99.71%	109	109	8	99.73%	109	109	5	99.73%	93	17	4	99.71%
ratio	–	+26.88%	–	–	+17.20%	–	–	–	+17.20%	–	–	–	1.00	–	–	–

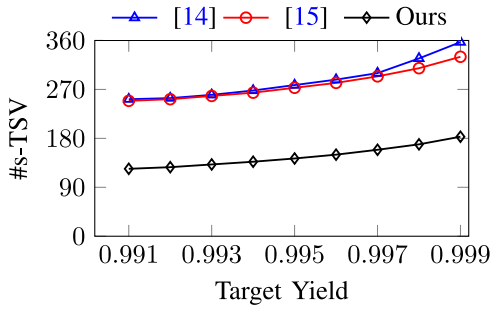


Fig. 7. Number of required s-TSVs under various target yields.

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. In this experiment, we consider 1-fault tolerance structures case, that is, the maximum number of tolerant faults K equals to 1. Since the chip yield is lower under 1-fault tolerance structures, the target chip yield is set to 99.5% and the TSV defect probability p is set to 0.001. We compare [7], [14], [15], with our proposed heuristic-based model under 1-fault tolerance structures. One s-TSV is assigned to each f-TSV group in [14] and [15]. We also set K to 1 in the proposed fault tolerance TSV planning framework, if the maximum number of tolerant faults K in a group is greater than 1. Based on the TSV planning method in [15], we run the minimum spanning tree method in [7]. Therefore, the s-TSV numbers and the chip yield of [7] and [15] are same in the experiment.

Table VI lists the statistic results averaged over 20 independent experiments. As shown in Table VI, compared with [7] and [15], the proposed fault tolerance TSV planning framework can reduce the number of s-TSVs and the maximum port number of multiplexers when generating 1-fault tolerance structures.

Fig. 7 shows the required s-TSV numbers under various target yields, in comparison among [14], [15], and our proposed framework. The experiment is performed on n100 benchmark. Each data point in the figure is an average of 20 independent experiments. It can be observed that the number of required s-TSVs increases along with increasing target yield and is significantly reduced by the proposed framework for all target chip yields.

VIII. CONCLUSION

In this paper, we focus on the generation of adaptive TSV fault-tolerance structure. An ILP-based model and an efficient MCMF-based heuristic method are proposed to generate the AFTSs in minimizing both the multiplexer delay overhead and the used s-TSV number. In the end, a fault-tolerance TSV planning methodology is also proposed to provide yield awareness in TSV planning. Experimental results show that, compared with state-of-the-art, the proposed fault tolerance TSV planning methodology can effectively reduce the number of s-TSVs used for fault tolerance structures.

It should be noted that, the proposed TSV fault tolerance planning is performed in floorplanning stage where we have no accurate timing information. Therefore, we only use the wirelength to reflect the wire delay in floorplanning stage. In future we plan to evaluate the delay more accurately by executing time-consuming routing.

ACKNOWLEDGMENT

The authors would like to thank the Information Science Laboratory Center of USTC for hardware and software services.

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.

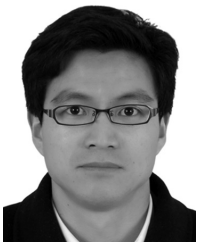
[3] (2008). *ITRS*. [Online]. Available: http://www.itrs.net

[4] T. Lu *et al.*, "TSV-based 3-D ICs: Design methods and tools," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 10, pp. 1593–1619, Oct. 2017.

[5] 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.

[6] B. Yu *et al.*, "Design for manufacturability and reliability in extreme-scaling VLSI," *Sci. China Inf. Sci.*, vol. 59, pp. 1–23, Jun. 2016.

[7] 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.
[8] 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.
[9] H.-H. S. Lee and K. Chakrabarty, "Test challenges for 3D integrated circuits," *IEEE Des. Test. Comput.*, vol. 26, no. 5, pp. 26–35, Sept./Oct. 2009.
[10] 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.
[11] 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)*, Apr. 2010, pp. 104–107.
[12] 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.
[13] 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.
[14] 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.
[15] Q. Xu, S. Chen, X. Xu, and B. Yu, "Clustered fault tolerance TSV planning for 3-D integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 8, pp. 1287–1300, Aug. 2017.
[16] Y. Chen, D. Niu, Y. Xie, and K. Chakrabarty, "Cost-effective integration of three-dimensional (3D) ICs emphasizing testing cost analysis," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2010, pp. 471–476.
[17] B. Noia and K. Chakrabarty, *Design-for-Test and Test Optimization Techniques for TSV-Based 3D Stacked ICs*. Cham, Switzerland: Springer, 2014.
[18] 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.
[19] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, vol. 24. Heidelberg, Germany: Springer, 2002.
[20] K. Mehlhorn and S. Naher, *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge, U.K.: Cambridge Univ. Press, 1999.
[21] A. Makhorin. (2008). *GLPK (GNU Linear Programming Kit)*. [Online]. Available: https://www.gnu.org/software/glpk/
[22] 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.
[23] 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.
[24] 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.



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

He served with the Graduate School of Information, Production and Systems, Waseda University, Tokyo, Japan, as a Research Associate from 2005 to 2009 and as an Assistant Professor from 2009 to 2012. He is currently an Associate Professor with the Department of Electronic Science and Technology, University of Science and Technology of China, Hefei, 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.

Qi Xu (S'17) 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.

Bei Yu (S'11–M'14) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX, USA, 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 Four Best Paper Awards at the International Symposium on Physical Design (ISPD) 2017, the SPIE Advanced Lithography Conference 2016, the International Conference on Computer Aided Design (ICCAD) 2013, and the Asia and South Pacific Design Automation Conference (ASPDAC) 2012, and three additional Best Paper Award nominations at DAC/ICCAD/ASPDAC, and four ICCAD/ISPD contest awards. He has served in the editorial boards of *Integration*, *VLSI Journal* and *IET Cyber-Physical Systems: Theory and Applications*.