Journal of Electronic Testing (2019) 35:741–759
https://doi.org/10.1007/s10836-019-05824-w

Identification of Random/Clustered TSV Defects in 3D IC During Pre-Bond Testing

Dilip Kumar Maity[1] · **Surajit Kumar Roy**[2] · **Chandan Giri**[2]

Received: 19 March 2019 / Accepted: 5 September 2019 / Published online: 11 November 2019

Abstract

Three-dimensional Integrated Circuits (3D ICs) based on Through-Silicon Vias (TSVs) provide many benefits, such as high density, high bandwidth and low-power consumption. However, defects in TSV due to complex fabrication steps decrease the yield and reliability of 3D ICs. Therefore each die should be tested before it is stacked through pre-bond test. Pre-bond test and defect identification of TSVs are extremely important to screen out defective TSVs early in the manufacturing flow. Also, test cost minimization is one of the key issues of the testing process. The existing test time minimization solutions for pre-bond test consider random TSV defects. However, in practice clustered TSV faults are quite common. In this paper, we propose a novel test time minimization technique to address both random and clustered defect distributions. The proposed solutions are based on recursive bi-partitioning and padding of test sessions to minimize the number of required test sessions as well as test time. Simulation results show that the proposed method can achieve more than 50% reduction in test time for a 20-TSV network with four faulty TSVs compared to serial testing approach. The proposed algorithm also pinpoints the defective TSVs in a TSV network with a reduced test time compared to prior works.

Keywords 3D IC · Pre-bond testing · Random defects · Clustered defects · TSV test

1 Introduction

The idea of 3D integrated circuit (3D IC) has been considered as an alternative to overcome the challenges of current two-dimentioal integrated circuit (2D IC). A 3D IC is implemented by stacking multiple dies vertically and interconnecting the dies using through-silicon vias (TSVs). 3D IC provides higher performance, reduced interconnection length, reduced power consumption and smaller footprint than conventional 2D IC [6]. Despite these advantages, there are several challenges related to testing of 3D IC [13]. TSVs provide communication link to upper layer dies of 3D ICs. TSVs support higher density and generate low-capacity interconnects compared to traditional wire-bonds [7]. While TSVs bring many advantages to 3D ICs, one of their major drawbacks is reliability.

Fabrication of TSVs is a complex process and defects may be introduced during the manufacturing of 3D ICs. Different types of TSV faults are presented in [2]. There are many pre-bond defects that can affect the proper functionality [2] of the chip. Faults like pinhole or microvoid are identified by pre-bond testing [1, 5, 37]. Incomplete metal filling or microvoids in the TSV pillar caused by electronic migration increase the TSV resistance. Impurities in the TSV may also increase resistance and interconnect delay. Pinhole within the insulation layer forms a leakage path from TSV to the substrate. This leakage path largely increases the capacitance between the TSV and the substrate. The resistance of a TSV increases due to microvoid whereas the capacitance between TSV and the substrate increases due to pinhole. Hence, capacitance measurement is required for pinhole fault, but resistance measurement is used for microvoid fault [18]. A few defects

Responsible Editor: R. A. Parekhji

✉ Dilip Kumar Maity
dilip.maity@aot.edu.in

Surajit Kumar Roy
suraroy@gmail.com

Chandan Giri
chandangiri@gmail.com

[1] Department of Computer Science and Engineering, Academy of Technology, Adisaptagram, Hooghly, India

[2] Department of Information Technology, Indian Institute of Engineering Science and Technology, Shibpur, India

that evolve due to alignment, bonding or stress can be identified by post-bond testing [10, 14, 20].

A single faulty TSV can paralyze the whole chip. So, testing of TSVs is necessary for proper functionality of the chip. The TSV tests and fault localization can be done before die bonding (pre-bond) or after die bonding (post-bond). Pre-bond TSV test targets the defects that are inherent in the manufacturing of TSV itself, while post-bond testing detects the faults that arise during the assembly process. Pre-bond TSV testing is important as it allows manufacturer to screen out defective dies and avoid any situation where an entire 3D stack needs to be discarded. Moreover, successful pre-bond testing and diagnosis can facilitate in providing known good die (KGD) information prior to bonding.

TSV defect distributions can be broadly classified as two types, namely random defect distribution and clustered defect distribution [26]. For random TSV defect distribution, the failing probability of TSVs is independent of each other. On the other hand, for the cluster defect distribution, faulty TSVs form clusters and the defect probability of each TSV is related to its location. Therefore, the method of defect identification for cluster defect distribution is different from random defect distribution. This paper considers pre-bond TSV testing to facilitate defect localization considering both defect distributions.

Though the prior works [17, 22, 32–34] can reduce the test time significantly, however, there remain challenges. Firstly, instead of identifying all faulty TSVs, a certain number of faulty TSVs are identified within a TSV network. This number depends on the advance information regarding the number of given redundant TSVs for a repair solution. However, this information may not be known in advance at all times. Secondly, prior works consider only random TSV defects while the TSV defects tend to be a cluster. Finally, the test sessions generated by heuristic approaches are not optimal. For example, to identify a single defective TSV at position 4 in a 6-TSV network with upper bound of session size 4, {1, 2, 3, 4}, {1, 2}, {3, 4}, {3}, {4}, {5,6} test sessions are needed for the heuristic based method [7]. However, {1, 2, 3}, {1, 4, 5}, {2, 4, 6}, {3, 5, 6} sessions are sufficient to identify any single fault within this network. Hence, further improvement of test time is possible.

In this paper, we have proposed two fast heuristic methods that identify the defective TSVs efficiently in terms of test session generation and significantly reduces the overall test time for both random and clustered fault. The major contributions of this work are:

(a) There is no need of any constraint regarding maximum number of faulty TSVs to be identified before testing. Also, proposed algorithm is efficient for known number of redundant TSVs.
(b) Proposed method considers test time minimization under random and cluster TSV faults.
(c) Proposed method reduces the test time compared to the prior works. For example, to identify two faulty TSVs in a 20-TSV network, the proposed method reduces the test time in worst case by 35% over the method in [32].
(d) Proposed method runs in linear time.

The rest of this paper is organized as follows: Section 2 provides an overview of related prior works. Section 3 describes the background and preliminaries of this paper. Section 4 presents the problem formulation. Proposed methods for Random defect distribution and Clustered defect distribution are introduced in Section 5 and Section 6 respectively. Section 7 presents the experimental results and finally Section 8 concludes the paper.

2 Prior Work

The pre-bond TSV testing is difficult mainly because TSVs are single-ended in the dies at a pre-bond stage. Recently several research works are being undertaken on the pre-bond TSV testing. A technique of on-chip sense amplification has been proposed in [3] for detecting capacitive TSV faults where each TSV is treated as a dynamic random-access memory cell. Each TSV is charged and discharged and a sense amplifier is tuned such that only a certain range of capacitance on the TSV is acceptable. But this method requires careful calibration and tuning and has significant area overhead. Another novel application of on-chip sense amplification has recently been proposed in the literature [4]. But this application also requires careful calibration and tuning. Furthermore, this method only detects resistive defects of TSVs faults.

Pre-bond method in [31] uses Scan Switch Network (SSN) architecture for testing. In [30], four methods have been presented to detect the pinhole or void problem in a TSV by using leakage-current (test the resistance between ground and TSV). The TSV fault can be measured by measuring the signal strength [5]. A parametric fault is detected in [11] using ring oscillator. A solution for identifying resistive open faults and leakage faults in TSVs using ring oscillators is addressed in [12]. The built-in self-test (BIST) technique for pre-bond TSV testing in 3D SC is proposed in [37].

A pre-bond TSV probing method is presented in [18] where multiple TSVs are shorted together with a probe needle to form a TSV network. The number of TSVs within a network is typically less than 20 and depends on the relative diameter of the probe needle and the pitch of TSVs [17, 34]. The probing method is modeled by adding an active driver in the probe needle and forming a charge

sharing circuit between single (multiple) TSV(s) and the probe needle. The capacitor of the charge-sharing circuit is charged through the connected TSV and the charging time is recorded. It is then compared with a non-faulty TSV. The charging time of the capacitor is considered as the test time of TSV for resistance measurement of TSV. Charging time varies with the number of TSVs tested in parallel [17].

Some techniques have recently been proposed to minimize the identification time of faulty TSVs during pre-bond TSV probing. A heuristic method is presented in [17] where multiple TSVs can be tested simultaneously to speedup the test procedure [18]. The method generates a series of test sessions that can uniquely locate a given number of faulty TSVs within a network in reduced test time compared to that of testing each TSV individually. But only small number of defective TSVs are identified in reduced test time. The authors present heuristic methods to identify defective TSVs in reduced test time in [21, 22]. These two methods improve the results. An integer linear programming (ILP) based method to detect faulty TSVs is proposed in [34]. In this method some sessions are unnecessarily tested during the identification process and time complexity is also higher than [21, 22]. The literatures in [32, 33] also propose other techniques for pre-bond TSV probing but these techniques require prior information about maximum number of faulty TSVs to be identified. In this paper, we address the problem of test time minimization for pre-bond TSV test with focus on minimizing the number of test session and utilizing parallelism.

3 Preliminaries

Pre-bond testing of TSVs faces different challenges due to TSV pitch and density. The diameter of the TSV ranges from 3-20 μm [9, 23] and the pitches of the TSV ranges from 0.4-10 μm [27, 29]. Although TSVs have diameters of 20 μm and pitches of 10 μm or smaller, current probing technology requires a minimum pitch of 35μm [17, 21]. Therefore, current probe technology cannot make contact with individual TSVs. A new technique has been proposed for pre-bond TSV testing that is compatible with current probe technology [18, 19]. This probing method offers many benefits like robustness to process variation, requires less hardware overhead, provides accurate measurement of TSV resistance. In this work, the test technique in [18, 19] is used to identify faulty TSVs within a TSV network. The principle of operation of probing method is summarised below.

According to the TSV probing method in [19], the probe card is shifted up or down only once to contact all TSV matrices. In the first configuration, the probe card is lowered and contact is made between the probe needles on the probe card and some group of TSV networks. Probe card contains a large number of needles, each making contact with a single network. The TSV networks in the first configuration are tested one by one. After all networks in contact are tested and the faulty TSVs in them are identified, the probe card is lifted and shifted to second configuration to contact the previously untested TSVs.

A pre-bond TSV probing method has been proposed in [18, 19] where a big probe needle with system logic is used to short some TSVs together. The set of TSVs contacted by the probe needle is defined as a TSV network. The size of the network depends on the relative sizes of typical test probes and TSVs. To introduce the principle of operation, a circuit model of the test set up for a 4-TSV network is shown in Fig. 1, where 4-TSV network means four TSVs contacted by a needle. A gated scan flip-flop (GSF) drives TSV by the switches B_1 and B_2 [18, 19]. A charge sharing circuit is constructed between GSF and C_{charge} through its corresponding TSV by configuring the switches of a GSF. The probe needle has a known resistance R_p and contact resistance R_c with each TSV. Each TSV i is modeled as a wire with both a resistance (R_i) and capacitance (C_i). The defects in TSVs are identified by resistance and capacitance measurement.

For capacitance measurement, the TSV network is discharged by loading a zero into all gated scan-flops and at the same time capacitor C_{charge} is charged to a known voltage V by closing the switch S_2 and opening the switch S_1. Then all the gated scan flip-flop are closed to discharge the capacitor C_{charge} through the TSV network by closing the switch S_1 and opening the Switch S2. The rate of change of V_1 is monitored through the volt meter until it falls below a certain level i. e., 1% of its maximum charge. And the value of C_{net} is determined from the charge sharing equation.

For resistance measurement, the process is begun by discharging (Switch S_2 is open and switch S_1 is closed) the capacitor C_{charge} as well and loading all of the gated scan-flops with one. Then one of the gated scan-flops is opened to charge the capacitor C_{charge} through its connected TSV. When V_1 reaches the predetermined voltage, the time to charge C_{charge} is recorded. It is then compared to a calibrated charge curve for a non faulty TSV.

According to the proposal [18], multiple TSV networks are tested in parallel and also several TSVs in each network are tested in parallel. C_{charge} is charged faster for parallel TSV test which can be conducted by configuring multiple GSFs at a time. Therefore, the measurement process terminates quickly. However, the number of TSVs tested in parallel is limited due to minimum measurement resolution constraint (p) [18, 19]. If we increase the number of TSVs tested in parallel then there is a reduction in charge time

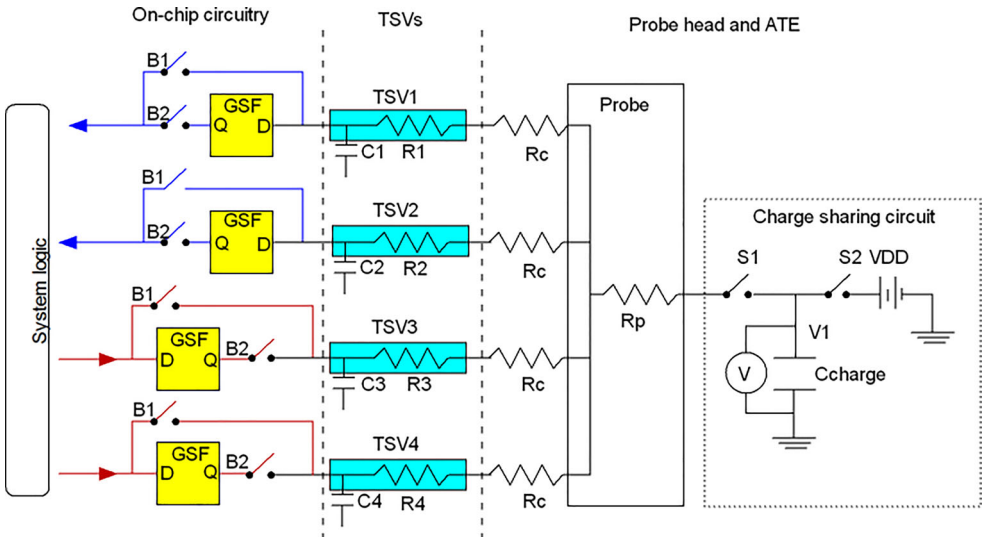

Fig. 1 Circuit model for pre-bond TSV probing [18]

for fault-free and faulty TSV networks. The difference in charge time between fault-free and faulty TSVs decreases as more TSVs are tested in parallel, which affects the resolution. The set of TSVs within the same TSV network tested in parallel is defined as *test session*. And the number of TSVs within a session is defined as *session size*(q). The capacitor charging time depends on the *session size*(q). The capacitor charging time as in [17] with respect to the different parallel active TSVs is presented in Table 1.

4 Problem Formulation

In general, most of the pre-bond TSV faults are resistive in nature [2]. So, proper resistance measurement is required to detect faulty TSVs. In this paper we consider resistive TSV fault as in [17, 21, 34]. Pre-bond TSV testing helps to identify defective dies early before bonding. Also, pre-bond TSV testing ensures that dies to be stacked include only good TSV interconnects which is required for known good die (KGD) information.

Objective of the proposed method is to identify the defective TSVs within a TSV network. A TSV network is formed by all the TSVs that are shorted together to the single probe needle. Identification of defective TSVs can be done by testing one TSV at a time and that requires longer test time. Test time can be reduced if the TSVs are tested in parallel. The proposed pre-bond TSV test method generates sequence of test sessions to detect faulty TSVs in reduced test time. A test session is formed by TSVs that are tested simultaneously and number of TSVs within a session indicates the session size (q). The difference in capacitor charging time between faulty and non-faulty TSVs decreases when size of the session increases which affect the resolution [18]. So the size of the session cannot be increased beyond a certain limit. Resolution constraint p indicates the upper bound of the number of TSVs in a session.

Table 1 Capacitor charging time of parallel TSV test [17]

Number of TSVs tested in parallel (q)	Capacitor charging time $t(q)(\mu s)$
1	0.80
2	0.53
3	0.42
4	0.38

Now, formally the problem can be stated as follows: *Given n identical TSVs within a n-TSV network, resolution constraint p and the test time* $t(q)$, $1 \leq q \leq p$, *for the test session containing q number of TSVs, determine the sequence of test sessions to identify faulty TSVs of the n-TSV network such that testing can be done in reduced test time.*

The overall test time for the TSVs of a 3D IC can be estimated by accumulating the test times of all networks. Therefore, the minimization of overall test time can be achieved when the test time for each network is minimized. If the 3D IC has N number of TSVs, then there are $\lceil \frac{N}{n} \rceil$ number of n-TSV networks. We test each n-TSV network individually and overall test time T_{total} is obtained as $T_{total} = \sum_{i=1}^{\lceil \frac{N}{n} \rceil} T_i$, where T_i is the test time for i^{th} n-TSV network. In this work, our main focus is to minimize the test time T_i of an n-TSV network in terms of number of sessions minimization.

We have proposed two heuristic based algorithms (a) *Heuristic for Random Defects (HRD)* and (b) *Heuristic for*

Clustered Defects (HCD) to solve the above mentioned problem in the context of Random defect distribution (Section 5) and Clustered defect distribution (Section 6).

5 Random Defects and Test Time Minimization Strategies

During manufacturing of TSV based integrated circuits, thousands of TSVs are produced simultaneously in a series of very complex processing steps. The 3D integrated circuits themselves typically contain thousands of such devices. For random defect distribution the probability of a fault occurring on a chip is completely independent of the number of faults already on that chip.

5.1 Heuristic for Random Defects (HRD)

Parallel TSV testing reduces the test time compared to sequential TSV testing. Sequential testing is advantageous when there is a large number of defective TSVs within a TSV network. But it is impractical because each TSV network has limited number of redundant TSVs (r) to repair the network. The TSV network is not repairable if the number of defective TSVs is greater than r. Otherwise, the redundant TSVs are sufficient to replace the defective TSVs. Finding $r + 1$ number of defective TSVs implies that the network is not repairable; so further test is useless. The proposed algorithm can identify m number of defective TSVs, where $0 \leq m \leq r+1$. Also the test process can be done without knowing any prior information about the number of faults or number of redundant TSVs.

The proposed algorithm tries to generate a sequence of test sessions for identifying faulty TSVs uniquely in a TSV network. The test session generation technique is described in Algorithm 1. This algorithm generates a session by picking p number of TSVs from the set of n untested TSVs. The last session may have less than p number of TSVs. All the TSVs of a session can be tested simultaneously. Each generated session is tested and the corresponding test time is recorded. Here, we are considering only the charging time as the test time.

If a session is identified as fault-free then all the TSVs of the session are considered as good TSV. But if the session is faulty, then Algorithm 2 is invoked to find the exact position of the first occurrence of the defective TSV within the current session. Algorithm 2 divides a faulty session until a single faulty TSV is identified. The faulty session of length p (number of TSVs of the session) is bi-partitioned into two smaller sessions (as left half and right half). The length of each half will be $\frac{p}{2}$ if p is even or $\lceil \frac{p}{2} \rceil$ for left half and $\lfloor \frac{p}{2} \rfloor$ for right half if p is odd. Now, the left half is tested. If the left half is found to be faulty, subsequent testing is done on this half itself. On the other hand, if the left half is found as fault free, the right half is bi-partitioned and tested. In this fashion a faulty session is bi-partitioned until the length of session is one. Finally the position of the first defective TSV is identified by the algorithm. When the left half is identified as faulty, we have no concrete information about the right half. In a session p TSVs are tested in parallel and more than one faults can be present within the p TSVs. So after bi-partitioning each partition may contain the fault. Therefore, this right half is considered as untested TSVs. During the test process, non-faulty tested TSVs are padded with smaller sessions to utilize the maximum resolution constraint for reducing the test time.

Algorithm 1 TEST-SESSION-GENERATION.

Input: number of TSVs (n), resolution constraint (p).
Output: Set of test sessions, and total test time.

```
1  if count = r + 1 || n = 0 then
2      return;
3  end
4  if n > 0 then
5      if (n ≥ p) then
6          Create a session s by taking p number of TSVs;
7      else
8          Create a session s by taking all TSVs;
               // length of s is less than
               p
9          Modify the session s by taking non-faulty
               tested TSV(s); // padding with
               non-faulty tested TSVs
10     end
       // test time accumulation
11     total_test_time = total_test_time+t(length(s));
12     if session s is tested as being faulty then
13         j = DEFECTIVE-TSV-
           IDENTIFICATION(length(s), p);

           // defective TSV is identified
               at location j
14         n = n − length(s) + (j + 1);
15         count = count + 1;
16         TEST-SESSION-GENERATION(n, p);
17     else
           // the session s is fault free
18         n = n − length(s);
19         TEST-SESSION-GENERATION(n, p);
20     end
21 end
```

Algorithm 2 DEFECTIVE-TSV-IDENTIFICATION.

Input: The faulty session (s), resolution constraint (p).
Output: Position of defective TSV.

```
1  if length(s) = 1 then
       // defective TSV is identified
2      return the position of the defective TSV;
3  else
4      Bi-partition the session s into two new sessions s1
       and s2;
5      Modify the session s1 and s2;        // padding
       non-faulty tested TSVs
       // test time accumulation
6      total_test_time =
       total_test_time + t(length(s1));
7      if session s1 is tested as being faulty then
8          DEFECTIVE-TSV-
           IDENTIFICATION(length(s1), p);
9      else
           // session s1 is fault free so
              session s2 must be faulty
10         DEFECTIVE-TSV-
           IDENTIFICATION(length(s2), p);
11     end
12 end
```

5.2 Illustrative Example

In this section the proposed method has been elaborated with an example. Let us consider a problem instance where number of TSVs (n) is 16 and resolution constraint (p) is 4. TSVs are represented with the numbers 1, 2...16 and initially all are untested (root of the tree of Fig. 2). Assume that the defective TSVs are at position 6, 7 and 11. If the length of any session is less than the value of the resolution constraint, non-faulty tested TSVs are randomly appended to such sessions to further reduce the test time.

Step 1: Pick the TSVs at position 1, 2, 3, 4 for the first session {1, 2, 3, 4}. Now the session is tested to check whether there is any faulty TSV or not. From Fig. 2 it is seen that if there is no fault, then the next session will be formed from the remaining untested TSVs.

Step 2: Now the next session {5, 6, 7, 8} is created from the remaining untested TSVs (5 to 16) and is identified as faulty. This faulty session is bi-partitioned and two new sessions {5, 6}, and {7, 8} are generated. The session {5, 6} is tested and detected as faulty. So it is decomposed as {5} and {6}. After testing {5}, it is detected as fault-free. The TSV at position 6 must be faulty. We cannot make any decision about whether TSVs at position 7 and 8 are faulty or not i.e. they are considered as untested. So, they are included in the list of untested TSVs and the next session to be tested is {7, 8, 9, 10}.

Step 3: The next session {7, 8, 9, 10} is tested and identified as faulty. TSV at position 7 is detected as faulty by following the process described in Step 2.

Step 4: Similarly, session {8, 9, 10, 11} is identified as faulty and split as {8, 9} and {10, 11}. The session {8, 9} is detected as non-faulty and TSVs at position 8 and 9 are non-faulty. So the fault is within the session {10, 11} which is to be bi-partitioned next. Subsequently, the session {10} is tested and identified as fault-free. Hence, TSV at position 11 is defective as other TSVs of this session are non-faulty.

Step 5: Again, the next session {12, 13, 14, 15} is generated and detected as fault-free.

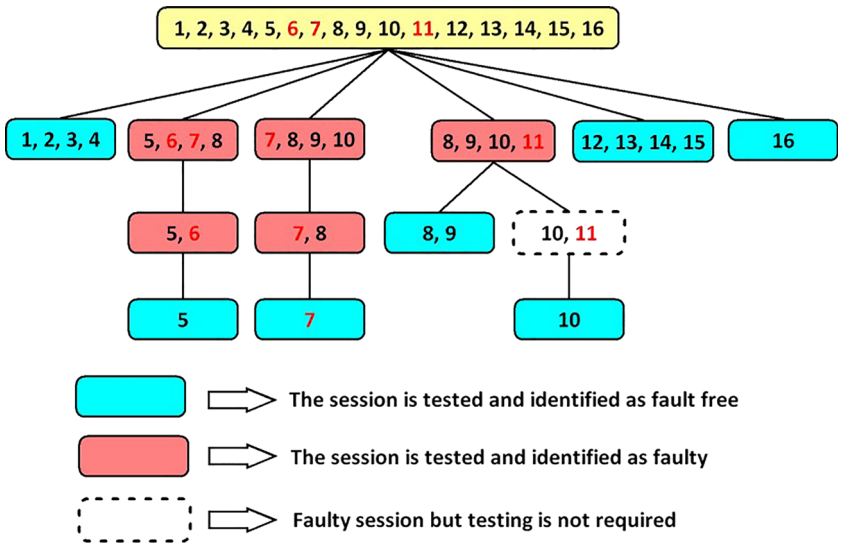

Fig. 2 Illustration of example for 16-TSV Network

Step 6: The last session {16} is generated and detected as fault-free.

5.3 Analysis of HRD

This section presents a detailed analysis of the number of sessions in worst case. To analyze the test session, we first investigate the test session through a decision based algorithm.

Lemma 1 *Given a set of n identical TSVs having multiple faulty TSVs. In worst case, if $T(n)$ be the minimum number of test sessions required to find single defective TSV using any decision based algorithm, then $T(n) \geq \lceil \log_2 n \rceil + 1$.*

Proof We can describe any search algorithm that satisfies the restrictions of the decision tree (Fig. 3) represented by a binary tree. Consider the case in which the n TSVs $L[1:n]$ to be tested are identical. Any test of $L[i:j]$ must result in one of two possibilities: either the test will fail i.e. defective TSV is in $L[i:j]$ or all TSVs are good. So, the decision tree is a binary tree in which each internal node is labeled by the pair $L[i:j]$ which represents the test of $L[i:j]$.

The algorithm proceeds down the left branch of the tree for a failed test; otherwise it proceeds down the right branch. The external nodes represent termination of the algorithm. Every path from the root to an external node is a unique possible solution. The decision tree must have n external nodes. As there are n different possibilities of n TSVs, any one of these might legitimately be the only correct answer on a given instance.

Let $T(n)$ be the minimum number of tests that are sufficient to find the defective TSVs in the worst case. If all internal nodes of a binary tree are at height less than or equal to $h-1$, then there are at most 2^{h-1} external nodes (one more than the number of internal nodes).

Thus $n \leq 2^{h-1}$ and height of the binary tree with n external nodes $h \geq \lceil \log_2 n \rceil + 1$.

Therefore, we get the lower bound of test sessions $T(n) \geq \lceil \log_2 n \rceil + 1$. □

5.3.1 Test Session in Worst Case

Now that we have successfully established the Lemma, let us look at the actual analysis of the worst case test session. The number of sessions $T_s(n)$ depends on the number of TSVs n, number of faults m, and resolution constraint p. Algorithm 2 recursively finds out the position of the defective TSVs from a session of TSVs of length p. The algorithm divides the session of size p into two smaller sessions of size approximately $\frac{p}{2}$ and detect the faulty TSV from one of the sessions depending on the test result. The number of test sessions $S_s(p)$ required to identify the exact position of defective TSVs by Algorithm 2 is represented as

$$S_s(p) = \begin{cases} S_s\left(\frac{p}{2}\right) + 1 & \text{if } p > 1 \\ 0 & \text{if } p = 1 \end{cases} \quad (1)$$

Solving the equation, we get, $S_s(p) = \lceil \log_2 p \rceil$. So, number of tests required to identify the position of a defective TSV within a particular session is $\lceil \log_2 p \rceil$. Let the first fault is identified at the i^{th} position. So, total number of test sessions generated by the proposed method is calculated as

$$T_s(n) = \begin{cases} \left\lceil \frac{i}{p} \right\rceil + \lceil \log_2 p \rceil + T_s(n-i) & \text{if } m > 0 \\ 0 & \text{if } n = 0 \\ \left\lceil \frac{n}{p} \right\rceil & \text{if } m = 0 \end{cases} \quad (2)$$

The first term $\left\lceil \frac{i}{p} \right\rceil$ indicates the number of sessions tested upto the session containing the i^{th} TSV. The second term $\lceil \log_2 p \rceil$ indicates the number of tests within a session to identify the i^{th} TSV. The last term $T_S(n-i)$ indicates the number of sessions tested for $(n-i)$ TSV network.

For our proposed algorithm, the worst case scenario occurs for two reasons. Firstly, it may occur if the defective TSVs are located consecutively and a number of faulty

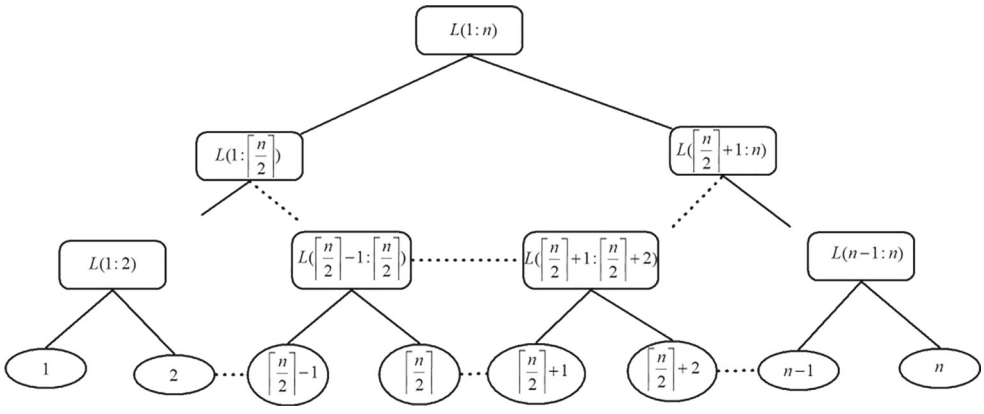

Fig. 3 Decision tree model

TSVs are present in the test session. In such a scenario, consecutive test sessions have a large number of TSVs from previously tested sessions and same TSVs are tested in different sessions repeatedly. Therefore, the number of test sessions increases and hence test time increases. Secondly, our proposed algorithm recursively bi-partitions the faulty session to identify the defective TSVs. There is a possibility that singleton TSV is tested due to insufficient known good TSVs. This situation occurs if faulty TSVs are identified at the starting of the process. Therefore, without loss of generality we assume that the first defective TSV be located at position 1(i.e. $i = 1$).

$$T_s(n) = 1 + \lceil \log_2 p \rceil + T_s(n-1) \tag{3}$$

Now we consider that the TSV network has m number of defective TSVs. These m defective TSVs are concentrated at the initial test sessions. So the number of test sessions required to identify m defective TSVs uniquely,

$$T_s(n) = m(1 + \lceil \log_2 p \rceil) + (n-m)/p \tag{4}$$

5.3.2 Test Time in Worst Case

The test time depends on the resolution constraint used for parallel testing and the number of test sessions required to identify the exact location of faulty TSVs. The total number of sessions generally decreases for larger resolution constraint because larger number of TSVs can be tested. Furthermore, if faulty TSVs are identified at the start of the process, then there is a possibility that sessions of length less than p are tested due to insufficient number of known good TSVs. This leads to smaller length session, which subsequently increases test time.

Minimum number of sessions required to identify one defective TSV within a faulty session is $\lceil \log_2 p \rceil + 1$ (Lemma 1). In our proposed methodology, if a session is faulty then it is bi-partitioned until we get a session of size one. Therefore, the session size is varied from p to 1 and at level k the session size is evaluated as $\frac{p}{2^{k-1}}$, where $1 \le k \le \lceil \log_2 p \rceil + 1$.

Total test time for an n-TSV network ($T(n)$) is represented as $T(n) = T_1 + T_2 + T_3$. Here, T_1 is the test time for p length sessions when $k = 1$, T_2 is the test time for the last session when $k = 1$ and $n - m$ is not power of p and T_3 is the test time for m number of sessions for $k = 2$ to $\lceil \log_2 p \rceil + 1$ due to bi-partitioning the faulty sessions. Thus, T_3 is calculated as $T_3 = m \sum_{k=2}^{\lceil \log_2 p \rceil + 1} t\left(\frac{p}{2^{k-1}}\right)$.

Let $t(q)$ be the capacitor charging time of parallel TSV test (Table 1). Test time for p length sessions:

$$T_1 = \left(m + \left\lfloor \frac{(n-m)}{p} \right\rfloor\right) t(p) \tag{5}$$

Test time for last session:

$$T_2 = t((n-m) \mod p) \tag{6}$$

Test time to find the position of m defective TSVs:

$$T_3 = m \sum_{k=2}^{\lceil \log_2 p \rceil + 1} t\left(\frac{p}{2^{k-1}}\right) \tag{7}$$

5.3.3 Time Complexity of the Algorithm

The major task of the proposed algorithm is parallel testing which depends on the number of test sessions required to identify m number of defective TSVs. From the analysis we have seen that the number of test sessions required (from equation 4) is:

$$T(n) = T_s(n) = m(1 + \lceil \log_2 p \rceil) + (n-m)/p \tag{8}$$

The time complexity for the proposed algorithm is $O(\max(m \log_2 p, n))$. If m is very large ($m \to n$) then the complexity of the algorithm is $O(n \log_2 p)$. If m is small ($m \to 0$) then the complexity of the algorithm is $O(n)$. In real application, for an n-TSV network, the number of defective TSVs is very small compared to n; so running time for our proposed algorithm is $O(n)$.

6 Clustered Defects and Test Time Minimization Strategies

The fabrication process for TSVs is complex and goes through different steps in a manufacturing line [8]. The process complexity results in an increased likelihood that something will go wrong in the line and fault occurring on a TSV may be correlated with other faulty TSVs. Thus, if a TSV is faulty then it is more likely that neighbor TSVs are faulty. This is considered as clustered TSV faults. Therefore, cluster model can be established with the assumption that the probability of a fault occurring on a TSV depends on the number of faulty TSVs already on the die.

6.1 Defect Probability Model with Spatial Correlation

In this clustered model, the presence of a single defect increases the likelihood of more defects in close vicinity [15, 25]. Therefore, test modeling scenario has to take TSV locations into account. Due to clustering effect, failing probability [28] of TSV_i ($P(TSV_i)$) is inversely

proportional to the distance from the existing defect node j that is expressed as:

$$P(TSV_i) \propto \left(\frac{1}{d_{ij}}\right)^{\alpha} \tag{9}$$

Where d_{ij} indicates the distance between node i and defective node j, and α is the clustering coefficient, a larger value for α implies less clustering. Considering a defect cluster center, clustering effect is formulated in [15], where all defective TSV tend to exist around the center. And the failing rate of TSV_i is inversely proportional to the distance from the existing cluster center, which can be expressed as:

$$P(TSV_i) = f\left(1 + \left(\frac{1}{d_{ic}}\right)^{\alpha}\right) \tag{10}$$

Here f represents the single TSV failure rate and d_{ic} is the distance between TSV_i and cluster center. By taking clustering effect into consideration, the distribution of defective TSVs tends to cluster around a cluster center and this becomes random defect distribution when $\alpha \to \infty$.

6.2 Heuristic for Cluster Defects (HCD)

Cluster defects mean the defect probability of a TSV increases due to existing defects. Therefore, exploration of cluster defect concept needs to take the TSV location information into account. The total number of TSVs in a die is denoted by N. We assume that each TSV is located on the integral coordinate of $\lceil\sqrt{N}\rceil \times \lfloor\sqrt{N}\rfloor$ grid with x and y coordinates ranging within $1 \le x \le \lfloor\sqrt{N}\rfloor$ and $1 \le y \le \lceil\sqrt{N}\rceil$ respectively. The minimum distance between two TSVs should be greater than the pitch of the TSV. Hence, the size of the grid depends on the ratio of the pitch of the current probe needle and the pitch of realistic TSVs [19, 24]. In this work, we define any two TSVs with their grid coordinates (x_i, y_i) and (x_j, y_j) as adjacent (neighbour) TSVs, either (i) $|x_i - x_j| = 1$ and $y_i = y_j$ or (ii) $|y_i - y_j| = 1$ and $x_i = x_j$ (Fig. 4a)

According to the TSV probing method, the number of TSVs within a network is typically less than or equal to 20 [33], $\lceil\sqrt{N}\rceil \times \lfloor\sqrt{N}\rfloor$ TSV grid needs to be partitioned into some smaller size TSV blocks. Each block is considered as a single n-TSV network where $n \le 20$. We assume that each block has only one defect cluster as in [35]. Therefore, clusters within different blocks do not affect each other and each block is independent.

Algorithm 3 describes the basic method of the proposed heuristics for cluster defects. HCD is proposed to pinpoint all defective TSVs uniquely in each TSV network such that the test time and test session are minimized as much as possible. Each TSV_i^j placed in the $network_j$ has location information (x_i^j, y_i^j), where x_i^j and y_i^j are the horizontal and vertical coordinates respectively of i^{th} TSV in the $network_j$.

This algorithm generates a session by picking at most p number of TSVs from the set of n untested TSVs within a network. If the generated session is identified as faulty, then DEFECTIVE-TSV-IDENTIFICATION method is invoked to find the exact position of the first occurrence of the defective TSV within the current session. Before localizing all defective TSVs, it is difficult to find the cluster center. Therefore each identified defective TSV is assumed as a cluster center.

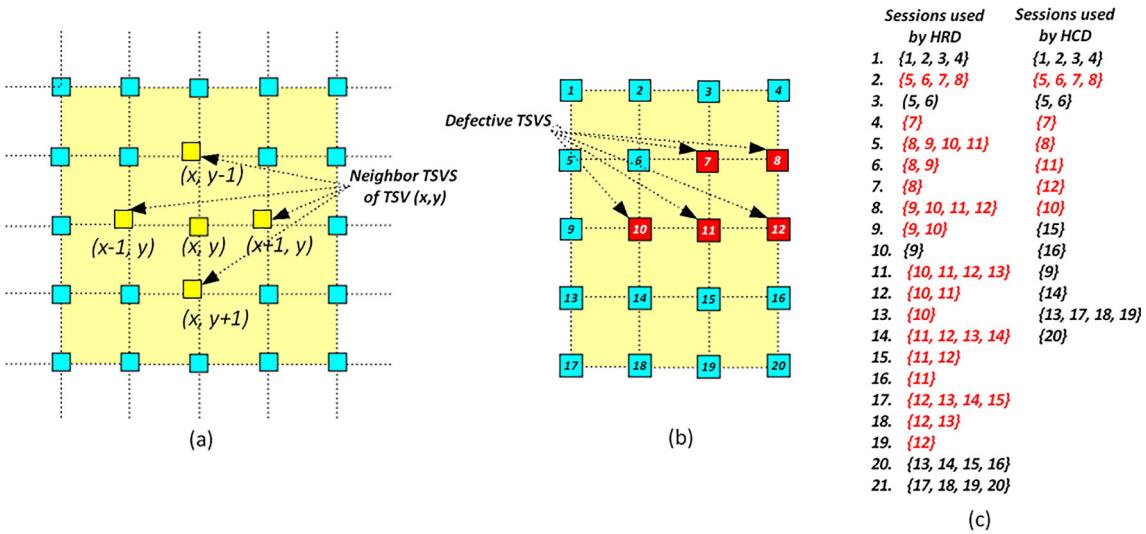

Fig. 4 Illustration of (**a**) Adjacent TSVs (**b**) Example for 20-TSV network with clustered defects (**c**) Required number of test sessions for HRD and HCD

Algorithm 3 CLUSTERED-DEFECT-IDENTIFICATION.

Input: number of TSVs (N) resolution constraint (p), fault rate (f).
Output: Set of test sessions, and total test time.

Place N number of TSVs into integral coordinate of the grid $\lceil\sqrt{N}\rceil \times \lfloor\sqrt{N}\rfloor$ and each $network_j$ has unique properties $TSV_i^j(x_i^j, y_i^j, network_j)$;
$test\_time = 0$;
for *each* $network_j$ **do**
  // $network_j$ is a set of TSVs of j^{th} network
  $untested\_TSV = network_j$;
  $good\_TSV = \Phi$;
  $Q = \Phi$;
  while $untested\_TSV \neq \Phi$ **do**
    Generate a new session s_1 by picking at most p number of TSVs from $untested\_TSV$;
    if $Length(s_1) < p$ **then**
      // padding non-faulty tested TSVs for last session
      Modify the session s_1;
    end
    if $Test(s_1, test\_time)$ **then**
      // session s_1 is tested and detected as faulty
      $u =$ DEFECTIVE-TSV-IDENTIFICATION$(Length(s_1), p)$;
      Q.ENQUEUE(u);
    end
    else
      $good\_TSV = s_1 \cup good\_TSV$;
    end
    while $Q \neq \Phi$ **do**
      $u =$ Q.DEQUEUE();
      for *each* $v \in$ *nearest neighbor of* u **do**
        // v is tested individually and detected as faulty
        Generate new session s_2 containing TSV v;
        Modify the session s_2 ; // padding non-faulty tested TSVs
        if $Test(s_2, test\_time)$ **then**
          Q.ENQUEUE(v);
        end
        else
          // v is detected as fault free
          $good\_TSV = \{v\} \cup good\_TSV$;
        end
      end
    end
  end
end

Next step is to find the untested nearest neighbor TSVs of the already detected defective TSV. These neighbor TSVs are tested individually. If any TSV among these neighbors is detected as faulty, then the process is continued until all neighbors are tested and identified as good. Otherwise, the next session is generated from the untested TSVs within the network.

After all TSVs in a network are tested and the faulty TSVs in them are identified, TSVs in the next network is selected and tested. During the test process, to reduce the test time, non-faulty tested TSVs are randomly appended to the session when the length of any session is less than the value of the resolution constraint.

6.3 Illustrative Example

Let us consider a network where the number of TSVs is 20 and resolution constraint (p) is 4. We consider the size of the block is either 5×4 or 4×5 which meets the requirement of number of TSVs within a network. TSVs are represented with the numbers 1, 2...20 and assigned in a grid-based structure of size 5×4 (Fig. 4b). Assume that the defective TSVs are at position 7, 8, 10, 11 and 12 and form a small cluster within 20-TSV network as shown in Fig. 4b.

Step 1: Pick the TSVs at position 1, 2, 3, 4 for the first session {1, 2, 3, 4}. This session is tested and identified as fault-free. Therefore next session {5, 6, 7, 8} is created from the remaining untested TSVs and is identified as faulty. The exact position (TSV at position 7) of the first occurrence of the defective TSV within the current session is identified by DEFECTIVE-TSV-IDENTIFICATION function.

Step 2: Now the neighbor TSVs of already identified defective TSV at position 7 are at position 3, 6, 8, 11. But the TSVs at position 3 and 6 have been already tested, so TSVs at position 8 and 11 are tested individually and detected as faulty. Subsequently, the TSV at position 12, which is untested neighbor TSV of 8, is tested and identified as faulty. Continuing this process, TSVs at position 9, 14, 15 and 16 are tested individually and are identified as fault-free (Fig. 4c).

Step 3: Now the next session {13, 17, 18, 19} is created from the remaining untested TSVs and is identified as fault-free. And then the last session 20 is created and identified as fault-free.

For the defective pattern shown in Fig. 4c the number of test sessions required for HRD (twenty-one) is larger compared to HCD (fourteen). Since the TSVs at position 1, 2, 3, 4 are identified earlier as good TSVs, to reduce the test time, parallelism is efficiently utilized by both methods.

However, the percentage reduction of test time of HCD over HRD is 33%.

6.4 Worst Case Time Complexity of the Algorithm

For our proposed HCD, the worst-case scenario occurs, if the defective TSVs are distributed randomly then non-faulty adjacent TSVs of a defective TSV are tested individually. And there is a possibility that almost all TSVs within an n-TSV network may be tested as singleton TSVs due to insufficient known good TSVs. Therefore, in worst case number of sessions used by the algorithm is, $S(n) = c_1 n + c_2$, where c_1 ($c_1 > 0$) and c_2 ($c_2 \geq 0$) are constants. Thus proposed method will take $S(n) = O(n)$ number of sessions for the n-TSV network to identify faulty TSVs uniquely.

7 Experimental Results

In this section, the experimental results of the proposed algorithm have been analyzed and compared with [22, 32, 34] and [33]. The proposed algorithm is coded and compiled in gcc compiler and executed on an Intel Core i5 processor with 3GB RAM. HSPICE simulations are used to find the test time for different test sessions. The resistance and capacitance of each TSV are considered as 1Ω and 20 fF respectively as in [18].

We propose two heuristic methods to speed up the pre-bond probing method. Before presenting the results in detail, let us clarify assumptions related to the metrics. The test time in this work considers only the time to charge the capacitor and does not account for the time to move the probe card as in [17]. Maximum resolution constraint p is considered as 4 and results are shown by varying the resolution constraint 2 to 4. The number of TSVs in each die can be quite different and depends on the applications and network styles. Thousands of TSVs may be required in each die for many-core processors or NoC-based designs, whereas hundreds of TSVs may be sufficient for smaller IP-based designs [9, 35]. In this work, the number of TSVs chosen in a die ranges from 1000 to 10000.

The TSV failure rate is not exactly known and recent publications have chosen various failure rates from 10^{-4} [9, 35] to 0.05 [16] and in this work it is considered up to 0.05. According to pre-bond probing test [18], the number of TSVs within a TSV network is not more than 20. So, the number of TSV networks depends on the number of TSVs in a die. Since the fault rate is very small, the average number of defective TSVs within a network is small. But due to the cluster effect, there is a possibility that a large number of networks may have all good TSVs and at the same time, few networks may have a large number of defective TSVs. This assumption roughly meets the probability of occurrence of such a defective pattern. Though in real application TSV networks are rejected if a certain number of faulty TSVs are detected, for the sake of a complete exploration of the proposed method a larger set of test cases are considered.

7.1 Results for Random Faults

Figures 5a and b show the variation of reduction in test time of proposed HRD with the number of faulty TSVs identified for 8-TSV and 20-TSV network respectively. It is seen from the figures that for a given value of test pins the reduction in test time decreases with an increasing number of faulty TSVs. 13 faults can be identified with reduced test time for the 20-TSV network, whereas 7 faults can be identified with reduced test time for the 8-TSV network. The reduction of test time will be large for higher resolution constraint. Because the number of test sessions will be less for a large value of p. There is 15% reduction of test time than serial testing for the 20-TSV network such that 12 faulty TSVs can be detected uniquely. From the Fig. 5a and b it is also observed that, for small m, the differences of

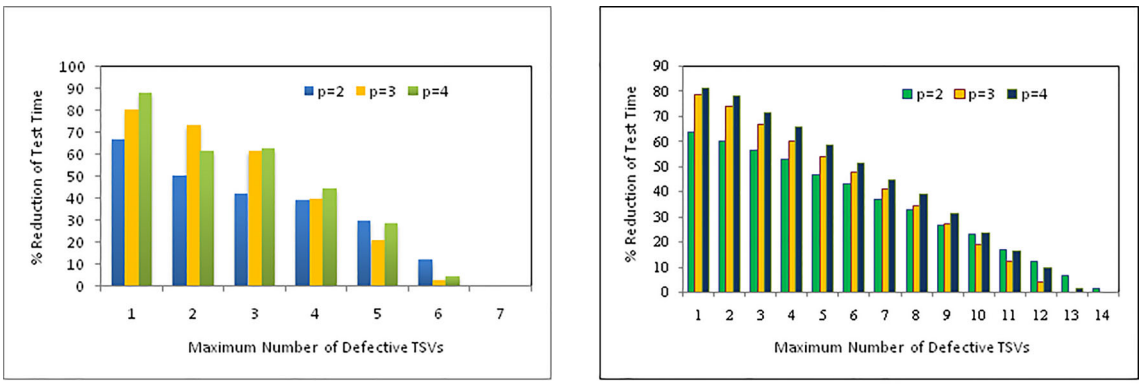

Fig. 5 Percentage Reduction of Test Time of HRD over Sequential Testing for (**a**) 8-TSV Network (**b**) 20-TSV Network

reduction in test time with $p = 2$, $p = 3$ and $p = 4$ are significant, but for large value of m, the test time reduction is quite small. This observation also indicates that for large m, sequential testing is better. It is seen from Fig. 5a that if the number of defects is greater than 6, then test time may not be reduced significantly compared to the serial testing time when considering 8-TSV network. But four defects can be identified up to 40% and 70% reduction in test time for 8-TSV and 20-TSV network respectively when $p = 4$.

Figure 6a shows the comparative study of the relative test time ratio of proposed HRD for both [17] and [22] considering various defective TSVs m and $p = 4$ in a 8-TSV network. From Fig. 6a, it is seen that for small or large value of m, the performance of our proposed algorithm is better than both [17] and [22]. In Fig. 6b, for 20-TSV network and resolution constraint $p = 3$, our experimental result is compared with [34] and [22]. It is observed that the proposed HRD can identify four faults with a reduced test time of 60% and above. It can also identify eight faults with a reduced test time of 35% and above. Figure 6b also shows that to detect seven or more faults, test time used by HRD is slightly higher than [22]. Hence it can be concluded that the proposed heuristic approach for random defect distribution provides an efficient solution for identifying defective TSVs for 3D ICs for larger TSV network.

Table 2 shows the comparisons between the dynamically optimized test in [32] and the proposed method for various TSV networks. Column 1 indicates the parameters n (network size), r (the redundant TSVs in the network) and p (resolution constraint). Here, we consider all possible faults for a given r. So, the average number of test sessions and average test time are considered for a given number of faulty TSVs within a TSV network. The parameter m in Column 2 shows the exact number of defective TSVs that have been identified within the network. Column 3 and Column 4 indicate the total number of sessions and total test time (in μs) used by the method in [32] for average and worst cases respectively. Column 5 and Column 6 indicate the number of tested sessions and test time to identify m defective TSVs for average and worst cases respectively. Column 7 shows the relative reduction in column 5 over column 3. Column 8 indicates the relative reduction in column 6 over column 4. From Table 2, the following observations can be made. First, it is seen from the table that obtained results are better than [32] in almost all cases. For example, to identify two faulty TSVs in a 20-TSV network, the proposed method reduces the test time in the worst case by 35% than the method in [32] and the percentage reduction reaches 45% for single faulty TSV. Interestingly for input parameter (8, 3, 3) (as seen in Table 2) our proposed heuristic approach takes slightly more time compared to the dynamically optimized test [32]. In this case, it is possible that due to the small number of tested TSVs, a single faulty TSV is tested several times in different sessions. So, the number of sessions may be increased. However, this single anomaly can be overlooked when compared to the reduction of time achieved in all other cases. Second, as expected, the number tested session increases as number of fault increases. Third, as m increases the average percentage reduction decreases. This is expected as a large number of sessions is tested to identify more defective TSVs within a TSV network.

7.2 Results for Cluster Faults

Figure 7a shows the total number of test sessions used by HRD and HCD and Fig. 7b shows the percentage reduction of test time of both HRD and HCD considering a fixed resolution constraint 4 and different values of m in a single 20-TSV network.

A smaller number of sessions is used by both HRD and HCD for a smaller number of faults. However, compared to HCD, HRD uses a smaller number of sessions for the small number of defective TSVs within the network. An identified faulty TSV increases the expectation that the neighbor TSVs may be faulty and consequently these TSVs are tested individually. This greedy nature of HCD increases

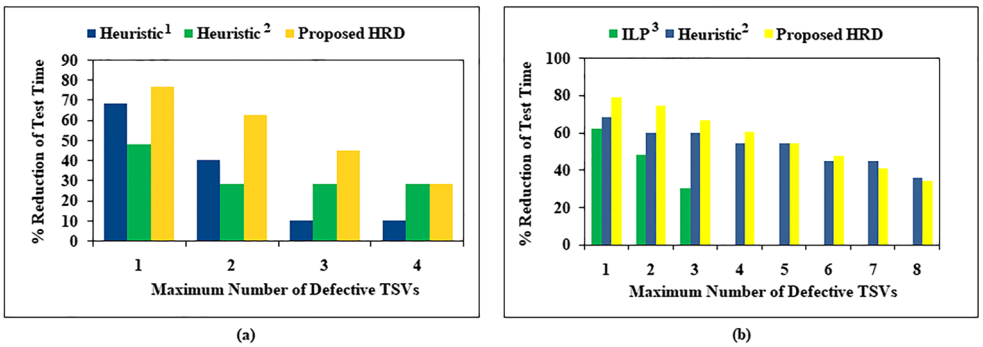

Fig. 6 Comparison in Test Time of Heuristic [17], Heuristic [22], ILP [34] and proposed HRD for (**a**) 8-TSV Network for P=4 (**b**) 20-TSV Network for p=3

Table 2 Comparative study of test sessions and test times constructed by [32] and proposed HRD

(Number of TSVs, Number of redundant TSVs, Resolution constraint)	Number of defective TSVs identified (m)	Dynamically optimized test in [32]		HRD			
		Average case (Number of test sessions, time in μs)	Worst case (Number of test sessions, time in μs)	Average case (Number of test sessions, time in μs)	Worst case (Number of test sessions, time in μs)	Average case reduction (sessions, time)	Worst case reduction (sessions, time)
(8, 3, 3)	0	(5.0, 2.10)	(5, 2.10)	(3.0, 1.26)	(3, 1.26)	(40.00%, 40.00%)	(40.00%, 40.00%)
	1	(5.3, 2.25)	(6, 2.52)	(4.0,1.72)	(6, 2.77)	(24.53%, 23.78%)	(0.00%, –9.92%)
	2	(6.4, 2.71)	(8, 3.36)	(5.4, 2.55)	(8,3.77)	(15.63%, 5.90%)	(0.00%, –12.20%)
	3	(7.5, 3.17)	(8, 8.36)	(6.71, 3.09)	(10, 4.20)	(10.53%, 2.52%)	(–25.00%, –25.00%)
(12, 4, 3)	0	(7.0, 2.94)	(7, 2.94)	(4.0, 1.68)	(4, 1.68)	(42.86%, 42.86%)	(42.86%, 42.86%)
	1	(7.5, 3.14)	(9, 3.78)	(5.0, 2.12)	(7, 3.05)	(33.33%, 32.52%)	(22.22%, 19.31%)
	2	(8.7, 3.65)	(12, 5.04)	(7.7, 3.38)	(10, 4.48)	(11.49%, 7.51%)	(16.67%, 11.11%)
	3	(10.3, 4.32)	(14, 5.88)	(9.3,4.19)	(12, 5.44)	(9.71%, 3.01%)	(14.29%, 7.48%)
	4	(11.8, 4.97)	(16, 6.72)	(10.45, 4.75)	(14, 6.41)	(11.44%, 4.43%)	(12.50%, 4.61%)
(15, 5, 3)	0	(8.0, 3.36)	(8, 3.36)	(5.0, 2.10)	(5, 2.10)	(37.50%, 37.50%)	(37.50%, 37.50%)
	1	(9.6, 4.03)	(14, 5.88)	(6.0, 2.55)	(8, 3.46)	(37.50%, 36.77%)	(42.86%, 41.16%)
	2	(11.1, 4.68)	(17, 7.14)	(8.7, 3.67)	(11, 4.92)	(21.62%, 21.52%)	(35.29%, 31.09%)
	3	(12.6, 5.33)	(20, 8.40)	(10.2,4.53)	(13, 5.79)	(19.05%, 15.08%)	(35.00%, 31.07%)
	4	(14.3, 6.03)	(23, 9.66)	(12.5, 5.53)	(14, 6.4)	(12.59%, 8.29%)	(39.13%, 34.16%)
	5	(15.8, 6.66)	(25,10.50)	(14.1, 6.32)	(16, 7.21)	(10.70%, 5.11%)	(36.00%, 31.33%)
(20, 5, 4)	0	(9.0, 3.42)	(9, 3.42)	(5.0, 1.90)	(5, 1.90)	(44.44%, 44.44%)	(44.44%, 44.44%)
	1	(10.8, 4.10)	(15, 5.69)	(7.8, 3.02)	(8,3.13)	(27.78%, 26.34%)	(46.67%, 44.99%)
	2	(12.3, 4.68)	(18, 6.83)	(8.6, 3.37)	(11, 4.44)	(29.92%, 27.93%)	(38.89%, 34.99%)
	3	(13.9, 5.31)	(21, 7.97)	(11.6, 4.64)	(14, 5.8)	(16.55%, 12.67%)	(33.33%, 27.23%)
	4	(15.1, 5.76)	(24, 9.11)	(13.5, 5.52)	(16, 6.52)	(10.60%, 4.17%)	(33.33%, 28.43%)
	5	(18.0, 6.85)	(25,9.49)	(14.95, 6.11)	(18, 7.31)	(16.94%, 10.80%)	(28.00%, 22.97%)

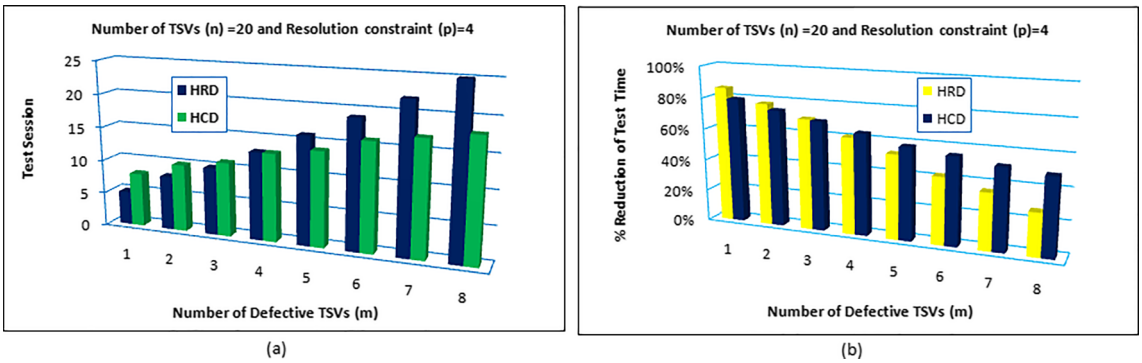

Fig. 7 **a** Number of sessions and **b** test time used by HRD and HCD for 20-TSV Network and resolution constraint 4

the number of sessions. For small m, there is a possibility that the neighbor TSVs may be non-faulty but still they are tested individually. Hence, some sessions are unnecessarily tested and HRD shows a better result than HCD. On the other hand, for large m due to cluster effect, the failing probability of neighbor TSVs of a faulty TSV is high. Therefore, individual TSV testing pinpoints a defective TSV with the cost of a single session. And HCD shows a better result than HRD. From Fig. 7b it is seen that the percentage reduction of test time for both methods increases as m increases since pinpointing a larger number of defective TSVs requires more sessions and longer test time. Also, note that HCD reduces the test time compared to HRD and the improvement generally increases as m increases.

Figure 8a shows the percentage reduction of test time of HCD over HRD considering different fault rates, varying resolution constraint and a fixed number of TSVs (1000). From the figure, it is seen that the percentage reduction of test time increases as the fault rate increases since a larger number of defective TSVs leads to more clusters or a large cluster. As a result, HRD requires more sessions and takes longer time than HCD. For same fault rate, the percentage reduction of test time increases for higher resolution constraint because for small resolution constraint both methods use a large number of test sessions.

Figure 8b shows the percentage reduction of test time of HCD over HRD considering different fault rates, different numbers of TSVs, and resolution constraint 4. From the figure, it is seen that the test time for both methods increases as the fault rate increases since pinpointing a larger number of defective TSVs requires more sessions and takes longer test time. Note that, HCD always helps to reduce the test time for fault rate 1% to 5% compared to the HRD and the improvement generally increases as the fault rate increases. This demonstrates that for large fault rate the probability of cluster formation is high; this leads to the better improvement of test time. One interesting observation is that the reduction of test time is almost same for same fault rate and a varying number of TSVs. This is expected because when we consider the fault rate, the mean value of Poison Distribution function is almost the same.

Subsequently, we analyze the impact of TSV yield on test session and test time for HRD and HCD. In a real application, although the failure rate of one TSV is very small, the probability of a large amount of fault-free TSVs might be low because of the large number of TSVs between two dies in 3D-IC. To enhance the yield of the

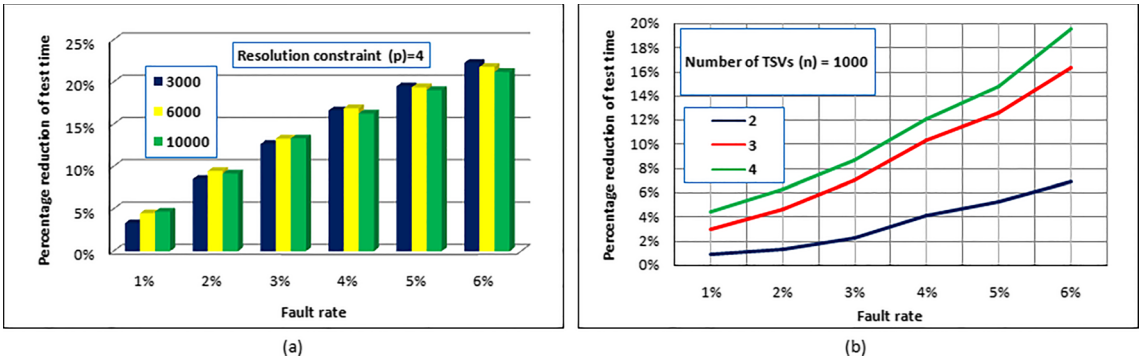

Fig. 8 Percentage reduction of test time of HCD over HRD for (**a**) varying TSV network and resolution constraint 4 (**b**) 1000 TSVs and varying resolution constraint

TSVs, TSV redundancy has been widely acknowledged as an effective approach [9, 35]. If Y_{TSV} be the yield of a TSV then the overall yield Y for n number of TSV can be expressed as $Y = (Y_{TSV})^n$. If r number of redundant TSVs are inserted to improve the yield, then the overall yield Y_r of the TSV with redundancy can be expressed as: $Y_r = \sum_{i=0}^{r} P(x = r)$, where $P(x = r) = {}^{n+r}C_r \times (1 - Y_{TSV})^r \times (Y_{TSV})^{n-r}$. TSV yield is expected to be higher, so the probability $P(m)$ of m faults within a TSV network decreases as m increases. Consequently, the expectation of test session and test time should be reduced for higher TSV yield. If $\xi(m)$ be the test time to identify m number of defective TSVs, $\psi(m)$ be the number of sessions for identifying m defective TSVs and $P(m)$ be the probability of m defective TSVs within an n-TSV network, then the expectation of test time ($E(T)$) and number of test sessions ($E(S)$) are obtained using the equations of literature in [33] as follows.

$$E(T) = \begin{cases} \sum_{m<2} \xi(m) P(m) + T_t \sum_{m\geq 2} P(m) & \text{if } r = 1 \\ \sum_{m\leq 2} \xi(m) P(m) + T_t \sum_{m\geq 3} P(m) & \text{if } r \geq 2 \end{cases} \tag{11}$$

$$E(S) = \begin{cases} \sum_{m<2} \psi(m) P(m) + T_t \sum_{m\geq 2} P(m) & \text{if } r = 1 \\ \sum_{m\leq 2} \psi(m) P(m) + T_s \sum_{m\geq 3} P(m) & \text{if } r \geq 2 \end{cases} \tag{12}$$

Where T_t and T_s represents the total test time and tested sessions respectively to find $m = r + 1$ defective TSVs.

Table 3 shows the expectation of number of tested sessions and tested time for 3-Step test time Optimization Simulator (SOS3) in [33], HRD and HCD. The result is shown for two sets of TSV yield 98.0% and 99.5% respectively with defect clustering coefficient $\alpha = 1$ [33]. The first column of Table 3 indicates the parameter network size, redundant TSVs in the network and resolution constraint. The second column indicates the parameters like total number of tested sessions and percentage reduction of the tested session of HRD and HCD over SOS3. The third column is used to represent the same parameters with respect to the expectation of test time ($10^{-7}s$). From the Table 3 the following observation can be made: a) the expectation of tested sessions and expectation of tested time for HRD decrease for higher TSV yield (99.5%). Because the probability of having a larger number of defective TSVs decreases as TSV yield increases. For $m = 1$, HRD reduces the test time 14.9% and 18.2% for TSV network is eight and eleven respectively. b) On the other hand for smaller TSV yield the probability of m faults within a TSV network increases as m increases. And for more faults (m) HCD minimizes the test time significantly due to the cluster effect. Thus reduction of expected test time and test session are better than HRD when TSV yield is 98%. For TSV network with $n = 20$, $m = 4$, $r = 4$, the reduction of test time expectation of HCD is 37.8%, considering 98% TSV yield. c) The obtained results in the proposed method are better than SOS3 [33] in all cases. Proposed HRD and HCD has an average reduction in expectation of test session and expectation of test time by 20% compared to [33]. This demonstrates that HRD and HCD greatly speed up the pre-bond TSV identification process.

The experimental results of all preceding pre-bond testing of TSVs are based on the assumption that the probability distribution function of fault formation is either random or clustered. It may so happen that some of the faulty TSVs follow random fault distribution and the rest of them follow cluster fault distribution. In this experiment, we analyze the effect of mixed defect distribution i.e. random defect distribution as well as cluster defect distribution using HRD and HCD. The percentage reduction of the test session and test time of HCD over HRD is analyzed by varying the number of regular TSVs and fault rates for different fault distribution ratios (Random: Clustered). These fault distribution ratios are chosen for illustration purposes. The results are shown in Table 4 for 1000 to 10,000 TSVs with single TSV failure rate of 0.01 to 0.05. The first and second columns of Table 4 indicate the number of TSVs and number of faults to be identified in a die respectively. The third column indicates the parameters like percentage reduction of test sessions and percentage reduction of test time with defect distribution ratios (Random: Clustered).

We made the following observations from Table 4: Firstly, the reduction of an average number of test sessions and average test time is negative for the fault distribution ratio 9:1 and 3:1. Because for more randomly distributed faults, HRD performs better than the HCD. As randomness decreases the percentage reduction of the test session and test time increases. For fault distribution ratio 1:3 and 1:9, HCD shows better results compared to HRD, even for small fault rate the percentage of reduction is significant. Second, as the number of faults increases the average percentage reduction of the test session and test time increases. When fault rate ranges from 0.01 to 0.05, the average reduction increases from 4% to 19% as shown in Table 4. Because pinpointing a larger number of faulty TSVs within a TSV network generally requires more sessions to be tested and costs more time. At the same time, the probability of such a large number of faulty TSVs within a small localized silicon area tends more clusters formation or large cluster formation. Thus, for both these cases, the percentage reduction of test time is quite significant. Third, in all cases, the percentage reduction of test time and test session is almost the same for a similar fault rate. This scenario occurs because the average TSV failing probability is the same when the fault rate is the same. And finally, when the faults

Table 3 Comparative study of expectation of test sessions and test time (10^{-7} sec) constructed by [33], proposed HRD and HCD

(Number of TSVs, Number of redundant TSVs, Resolution constraint)	Expected number of tested sessions Defect clustering coefficient $\alpha = 1$			Expectation of test time (10^{-7} sec) Defect clustering coefficient $\alpha = 1$		
	Sessions for SOS3 [33]	(Sessions for HRD, reduction over SOS3)	(Sessions for HCD, reduction over SOS3)	Test time for SOS3 [33]	(Test time for HRD, reduction over SOS3)	(Test time for HCD, reduction over SOS3)
TSV Yield=99.5%						
(8, 1, 2)	4.0	(3.9, 2.5%)	(3.9, 1.9%)	21.5	(21, 2.3%)	(21.0, 2.5%)
(8, 2, 3)	4.0	(3.1, 22.5%)	(3.0, 24.0%)	16.9	(13.2, 21.9%)	(13.8, 18.5%)
(11, 1, 2)	6.0	(5.8, 3.3%)	(5.8, 2.7%)	32.1	(31.0, 3.4%)	(32.0, 0.3%)
(11, 2, 3)	4.1	(4.1, 0.0%)	(4.0, 2.4%)	17.3	(17.3, 0.0%)	(17.1, 1.2%
(15, 2, 2)	8.1	(8.0, 1.2%)	(7.7, 4.8%)	43.0	(42.7, 0.7%)	(43.0, 0.0%
(15, 3, 3)	6.1	(5.1, 16.4%)	(4.9, 20.4%)	25.8	(21.8, 15.5%)	(20.7, 19.8%)
(16, 3, 4)	5.2	(4.2, 19.2%)	(3.9, 24.5%)	19.8	(16.1, 18.7%)	(15.2, 23.0%)
(16, 4, 4)	4.3	(4.2, 2.3%)	(3.9, 8.7%)	16.3	(16.2, 0.6%)	(15.3, 6.4%)
(20, 3, 4)	6.3	(5.2, 17.5%)	(4.8, 23.6%)	23.9	(19.9, 16.7%)	(18.8, 21.5%)
(20, 4, 4)	6.3	(5.2, 17.5%)	(4.8, 23.6%)	24.0	(20.0, 16.7%)	(18.8, 21.8%)
TSV Yield=98.0%						
(8, 1, 2)	4.3	(3.6, 16.3%)	(3.7, 13.7%)	22.8	(19.4, 14.9%)	(20.4, 10.7%)
(8, 2, 3)	4.2	(3.4, 19.0%)	(3.2, 25.0%)	17.7	(14.7, 16.9%)	(14.0, 20.7%)
(11, 1, 2)	6.4	(5.3, 17.2%)	(5.4, 15.3%)	34.1	(27.9, 18.2%)	(31.0, 9.0%)
(11, 2, 3)	4.6	(4.4, 4.3%)	(4.0, 12.6%)	19.3	(18.7, 3.1%)	(17.6, 8.9%)
(15, 2, 2)	8.7	(8.0, 8.0%)	(7.0, 19.6%)	46.5	(42.4, 8.8%)	(38.4, 17.4%)
(15, 3, 3)	6.9	(5.8, 15.9%)	(4.5, 34.6%)	29.0	(24.4, 15.9%)	(20.0, 30.9%
(16, 3, 4)	6.1	(4.9, 19.7%)	(3.8, 38.2%)	23.2	(19.0, 18.1%)	(15.7, 32.4%)
(16, 4, 4)	5.4	(5.2, 3.7%)	(3.8, 30.1%)	20.8	(20.3, 2.4%)	(15.7, 24.4%
(20, 3, 4)	7.5	(5.9, 21.3%)	(4.4, 41.2%)	28.7	(26.6, 21.3%)	(18.4, 35.8%)
(20, 4, 4)	7.8	(6.3, 19.2%)	(4.4, 43.4%)	29.7	(24.1, 18.9%)	(18.5, 37.8%)

Table 4 Comparative study of percentage reduction of test sessions and test time of proposed HCD over HRD for different distribution ratios (Random:Clustered)

Number of TSVs (N)	TSV failure rate (f)	Percentage reduction of test session and test time of HCD over HRD for mixed defect distribution (Random:Clustered) Resolution constraint $p = 4$ and Clustering coefficient $\alpha = 0.5$				
		Reduction over HRD (Test session, Test time)				
		Defect distribution (9:1)	Defect distribution (3:1)	Defect distribution (1:1)	Defect distribution (1:3)	Defect distribution (1:9)
2000	0.01	(−1.99%,−2.51%)	(−1.09%,−1.32%)	(0.73%,0.86%)	(2.91%,2.87%)	(2.91%,2.76%)
	0.02	(−3.31%,−3.68%)	(−1.32%,−1.75%)	(0.50%,0.01%)	(3.49%,3.12%)	(7.67%,7.82%)
	0.03	(4.43%,−5.01%)	(−1.22%,−1.38%)	(0.92%,0.46%)	(6.13%,6.12%)	(11.86%,11.82%)
	0.04	(−5.40%, 5.98%)	(−3.27%,−4.04%)	(2.84%,2.23%)	(9.39%,9.30%)	(15.86%,15.81%)
	0.05	(−6.12%,−6.80%)	(−3.45%,−4.33%)	(3.32%,2.79%)	(11.02,10.81%)	(19.09%,18.59%)
3000	0.01	(–2.05%,−2.34%)	(−1.09%,−1.14%)	(1.09%,1.01%)	(1.33%,1.36%)	(3.16%,2.96%)
	0.02	(−3.43%,−4.06%)	(−1.11%,−1.50%)	(0.55%,0.46%)	(4.44%,4.38%)	(8.46%,8.39%)
	0.03	(−4.18%,−4.63%)	(−2.76%,−3.47%)	(2.04%,1.86%)	(7.27%,7.22%)	(12.64%,12.69%)
	0.04	(−4.93%,−5.34%)	(−3.04%,−3.64%)	(3.52%,3.57%)	(9.62%,9.83%)	(16.62%,16.97%)
	0.05	(−5.78%,−6.27%)	(−2.84%,−3.31%)	(4.53%,4.34%)	(12.00,12.02%)	(19.61%,19.54%)
5000	0.01	(−1.89%,−2.10%)	(−0.87%,−1.07%)	(0.22%,0.27%)	(1.97%,1.85%)	(3.94%,3.93%)
	0.02	(−3.26%,−3.60%)	(−1.86%,−2.35%)	(1.47%,1.23%)	(5.15%,5.34%)	(9.05%,9.19%)
	0.03	(−4.00%,−4.46%)	(−2.28%,−2.76%)	(2.78%,2.54%)	(8.15%,8.49%)	(13.14%,13.58%)
	0.04	(−5.68%,−6.35%)	(−2.06%,−2.49%)	(4.02%,3.76%)	(10.29,10.86%)	(16.70%,16.81%)
	0.05	(−6.34%,−7.09%)	(−2.09%,−2.61%)	(5.04%,5.02%)	(12.24,12.23%)	(19.76%,19.54%)
8000	0.01	(−1.64%,−1.88%)	(−1.14%,−1.31%)	(0.55%,0.43%)	(2.60%,2.57%)	(4.48%,4.63%)
	0.02	(−2.89%,−3.25%)	(−1.47%,−1.85%)	(2.14%,2.07%)	(5.42%,5.51%)	(9.07%,9.32%)
	0.03	(−4.38%,−4.89%)	(−1.59%,−1.94%)	(3.22%,3.22%)	(8.10%,8.38%)	(13.09%,13.56%)
	0.04	(−5.28%,−5.89%)	(−1.66%,−2.13%)	(4.23%,4.26%)	(10.40%,10.68%)	(16.46%,16.45%)
	0.05	(−5.48%,−6.25%)	(−1.69%,−2.05%)	(5.18%,5.14%)	(12.43%,12.61%)	(19.46%,19.38%)
10000	0.01	(−1.68%,−1.91%)	(−0.99%,−1.19%)	(0.73%,0.63%)	(2.53%,2.52%)	(4.44%,4.55%)
	0.02	(−3.22%,−3.66%)	(−1.48%,−1.83%)	(2.22%,2.30%)	(5.33%,5.51%)	(9.00%,9.38%)
	0.03	(−4.39%,−4.92%)	(−1.56%,−1.90%)	(3.21%,3.18%)	(7.98%,8.10%)	(12.81%,13.01%)
	0.04	(−5.03%,−5.59%)	(−1.66%,−2.06%)	(4.25%,4.21%)	(10.04%,10.28%)	(16.12%,16.10%)
	0.05	(−5.59%,−6.24%)	(−1.69%,−1.99%)	(5.13%,5.18%)	(11.81%,11.85%)	(19.02%,18.61%)

Table 5 Number of test sessions and test time for different TSV yield used by proposed HCD for 3D bechmark circuits [36] considering $\alpha = 1$, $p = 4$ and 20-TSV network

Benchmark circuits [36]	N	Number of tested sessions and test time (in μs)			
		TSV Yield = 99.50% (#Session, Test time)	TSV Yield = 99.00% (#Session, Test time)	TSV Yield = 98.50% (#Session, Test time)	TSV Yield = 98.00% (#Session, Test time)
aes_core	1362	(352, 134.90)	(357, 138.52)	(379, 146.80)	(385, 150.48)
ethernet	3782	(967, 371.92)	(1002, 386.74)	(1017, 396.62)	(1031, 408.11)
des_perf	3678	(942, 362.20)	(975, 376.61)	(989, 386.15)	(1006, 397.95)
vga_lcd	7356	(1893, 724.28)	(1926, 745.75)	(1957, 767.60)	(1986, 788.90)
netcard	9112	(2341, 896.85)	(2377, 920.49)	(2418, 946.50)	(2458, 973.80)

are distributed in an equal ratio (1:1), HCD still reduces the test time and test session more than 5% for fault rate 0.05.

Table 5 shows the results of HCD in terms of the number of sessions and test time (in μs) using five benchmark circuits [36]. The clustering coefficient α and resolution constraint are considered as 1 and 4 respectively. Column 1–2 present the benchmark circuits and the corresponding number of functional TSVs respectively. Column 3–6 indicate the number of tested sessions and test time when TSV yield is considered as 99.5%, 99%, 98.5%, and 98% respectively. From the table, it is seen that the number of tested sessions and test time increase for lower TSV yield. This is because the probability of the number of defective TSVs increases when TSV yield decreases. Consequently, a large number of defective TSVs requires more sessions and longer test time.

8 Conclusion

In this paper, we have proposed two heuristics to uniquely identify the defective TSVs under random and clustered defects. The proposed method has some advantages over previous researches in the identification of random defects. First, extensive experimental results for various TSV networks demonstrate the benefit of the proposed heuristic on reducing test time compared to [22, 32–34] for random defect distribution. Besides this, the proposed model was able to identify more than 50% defective TSVs in the reduced test time. Second, the test time reduction remains consistent for various TSV networks. Third, time-saving can occur even in worst-case scenarios compared to [32]. Fourth, the running time complexity is linear which is better than the prior works. Also, we propose a modelling mechanism to speed up the TSV pre-bond probing method for clustering defects on TSVs. Extensive experiments demonstrate that the test time minimization method for clustered defect distribution is effective to reduce pre-bond TSV identification time. We also show how the test time is influenced by different assumptions for the probability distribution function of fault formation. Thus as a framework, proposed heuristics are expected to greatly reduce pre-bond test time for both random and clustered defect distributions.

References

1. Chen P-Y, Wu C-W, Kwai D-M (2009) On-chip tsv testing for 3d ic before bonding using sense amplification. In: 2009 Asian test symposium. IEEE, pp 450–455
2. Chen H, Shih J, Li S, Lin H, Wang M, Peng C (2010) Electrical tests for three-dimensional ics (3dics) with tsvs. In: International test conference 3D-test workshop, pp 1–6
3. Chen P-Y, Wu C-W, Kwai D-M (2010) On-chip testing of blind and open-sleeve tsvs for 3d ic before bonding. In: 2010 28th VLSI test symposium (VTS). IEEE, pp 263–268
4. Cho M, Liu C, Kim DH, Lim SK, Mukhopadhyay S (2010) Design method and test structure to characterize and repair tsv defect induced signal degradation in 3d system. In: 2010 IEEE/ACM international conference on computer-aided design (ICCAD). IEEE, pp 694–697
5. Cho M, Liu C, Kim DH, Lim SK, Mukhopadhyay S (2011) Pre-bond and post-bond test and signal recovery structure to characterize and repair tsv defect induced signal degradation in 3-d system. IEEE Trans Comp Packag Manuf Technol 1(11):1718–1727
6. Davis WR, Wilson J, Mick S, Xu J, Hua H, Mineo C, Sule AM, Steer M, Franzon P (2005) Demystifying 3d ics: the pros and cons of going vertical. IEEE Des Test Comput 22(6):498–510
7. Deutsch S, Chakrabarty K (2014) Contactless pre-bond tsv test and diagnosis using ring oscillators and multiple voltage levels. IEEE Trans Comput-Aided Des Integr Circ Syst 33(5):774–785
8. Dukovic J, Ramaswami S, Pamarthy S, Yalamanchili R, Rajagopalan N, Sapre K, Cao Z, Ritzdorf T, Wang Y, Eaton B, Ding R, Hernandez M, Naik M, Mao D, Tseng J, Cui D, Mori G, Fulmer P, Sirajuddin K, Hua J, Xia S, Erickson D, Beica R, Young E, Kusler P, Kulzer R, Oemardani S, Dai H, Xu X, Okazaki M, Dotan K, Yu C, Lazik C, Tran J, Luo L (2010) Through-silicon-via technology for 3D integration. In: IEEE International memory workshop, pp 1–2
9. Hsieh A-C, Hwang T (2011) Tsv redundancy: architecture and design issues in 3-d ic. IEEE Trans Very Large Scale Integr (VLSI) Syst 20(4):711–722
10. Huang Y-J, Li J-F, Chen J-J, Kwai D-M, Chou Y-F, Wu C-W (2011) A built-in self-test scheme for the post-bond test of tsvs in 3d ics. In: 29th VLSI test symposium. IEEE, pp 20–25

11. Huang S-Y, Lin Y-H, Tsai K-HH, Cheng W-T, Sunter S, Chou Y-F, Kwai D-M (2012) Small delay testing for tsvs in 3-d ics. In: Proceedings of the 49th annual design automation conference. ACM, pp 1031–1036
12. Huang L-R, Huang S-Y, Sunter S, Tsai K-H, Cheng W-T (2013) Oscillation-based prebond tsv test. IEEE Trans Comput-Aided Des Integr Circ Syst 32(9):1440–1444
13. Lee H-HS, Chakrabarty K (2009) Test challenges for 3d integrated circuits. IEEE Des Test Comput 26(5):26–35
14. Marinissen EJ, Chi C-C, Konijnenburg M, Verbree J (2012) A dft architecture for 3d-sics based on a standardizable die wrapper. J Electron Test 28(1):73–92
15. Meyer FJ, Pradhan DK (1989) Modeling defect spatial distribution. IEEE Trans Comput 38(4):538–546
16. Nain RK, Pinge S, Chrzanowska-Jeske M (2010) Yield improvement of 3d ics in the presence of defects in through signal vias. In: 2010 11th International symposium on quality electronic design (ISQED). IEEE, pp 598–605
17. Noia B, Chakrabarty K (2011) Identification of defective tsvs in pre-bond testing of 3d ics. In: 2011 Asian test symposium. IEEE, pp 187–194
18. Noia B, Chakrabarty K (2011) Pre-bond probing of tsvs in 3d stacked ics. In: 2011 IEEE international test conference. IEEE, pp 1–10
19. Noia B, Chakrabarty K (2014) Design-for-test and test optimization techniques for TSV-based 3D stacked ICs. Springer
20. Pasca V, Anghel L, Benabdenbi M (2011) Configurable thru-silicon-via interconnect built-in self-test and diagnosis. In: 2011 12th Latin American test workshop (LATW). IEEE, pp 1–6
21. Roy SK, Chatterjee S, Giri C (2012) Identifying faulty tsvs in 3d stacked ic during pre-bond testing. In: 2012 International symposium on electronic system design (ISED). IEEE, pp 162–166
22. Roy SK, Chatterjee S, Giri C, Rahaman H (2013) Faulty tsvs identification and recovery in 3d stacked ics during pre-bond testing. In: 2013 IEEE international 3D systems integration conference (3DIC). IEEE, pp 1–6
23. Schaper LW, Burkett SL, Spiesshoefer S, Vangara GV, Rahman Z, Polamreddy S (2005) Architectural implications and process development of 3-d vlsi z-axis interconnects using through silicon vias. IEEE Trans Adv Packag 28(3):356–366
24. Smith K, Hanaway P, Jolley M, Gleason R, Strid E, Daenen T, Dupas L, Knuts B, Marinissen EJ, Van Dievel M (2011) Evaluation of tsv and micro-bump probing for wide i/o testing. In: 2011 IEEE international test conference. IEEE, pp 1–10
25. Stapper CH (1986) On yield, fault distributions, and clustering of particles. IBM J Res Dev 30(3):326–338
26. Stapper CH, Armstrong FM, Saji K (1983) Integrated circuit yield statistics. Proc IEEE 71(4):453–470
27. Swinnen B, Ruythooren W, De Moor P, Bogaerts L, Carbonell L, De Munck K, Eyckens B, Stoukatch S, Tezcan DS (2006) Z. Tokei others, 3d integration by cu-cu thermo-compression bonding of extremely thinned bulk-si die containing 10 μm pitch through-si vias. In: 2006 International electron devices meeting. IEEE, pp 1–4
28. Tahoori MB (2005) Defects, yield, and design in sublithographic nano-electronics. In: 20th IEEE international symposium on defect and fault tolerance in VLSI systems (DFT'05). IEEE, pp 3–11
29. Topol AW, La Tulipe D, Shi L, Alam S, Frank D, Steen S, Vichiconti J, Posillico D, Cobb M, Medd S et al (2005) Enabling soi-based assembly technology for three-dimensional (3d) integrated circuits (ics). In: IEEE International electron devices meeting, 2005 IEDM technical digest. IEEE, pp 352–355
30. Tsai M, Klooz A, Leonard A, Appel J, Franzon P (2009) Through silicon via (tsv) defect/pinhole self test circuit for 3d-ic. In: 2009 IEEE International conference on 3D system integration. IEEE, pp 1–8
31. Wang C, Zhou J, Zhao B, Liu X, Royannez P, Je M (2012) Self-test methodology and structures for pre-bond tsv testing in 3d-ic system. In: 2012 IEEE Asian solid state circuits conference (A-SSCC). IEEE, pp 393–396
32. Zhang B, Agrawal VD (2014) An optimal probing method of pre-bond tsv fault identification in 3d stacked ics. In: 2014 SOI-3D-subthreshold microelectronics technology unified conference (S3S). IEEE, pp 1–3
33. Zhang B, Agrawal VD (2014) An optimized diagnostic procedure for pre-bond tsv defects. In: 2014 IEEE 32nd international conference on computer design (ICCD). IEEE, pp 189–194
34. Zhang B, Agrawal VD (2015) Diagnostic tests for pre-bond tsv defects. In: 2015 28th International conference on VLSI design. IEEE, pp 387–392
35. Zhao Y, Khursheed S, Al-Hashimi BM (2011) Cost-effective tsv grouping for yield improvement of 3d-ics. In: 2011 Asian test symposium. IEEE, pp 201–206
36. Zhao Y, Khursheed S, Al-Hashimi BM (2014) Online fault tolerance technique for tsv-based 3-d-ic. IEEE Trans Very Large Scale Integr (VLSI) Syst 23(8):1567–1571
37. Zimouche H, Di Natale G, Flottes M-l, Rouzeyre B (2013) A bist method for tsvs pre-bond test. In: 2013 8th IEEE design and test symposium. IEEE, pp 1–6

Dilip Kumar Maity is an honours graduate in Physics from University of Calcutta, India. He has also a BTech and an MTech degrees in Computer Science and Engineering from University College of Science and Technology, Calcutta, India in 2002 and 2004. He is currently pursuing PhD degree from Department of Information Technology at Indian Institute of Engineering Science and Technology (IIEST), Shibpur, India. His current research interests include VLSI testing, fault-tolerant computing, and yield and reliability enhancement techniques in 3-D integrated circuits.

Surajit Kumar Roy received the BSc (Hons. in Physics) from Calcutta University, India. He also received Bachelor of Technology in computer science and engineering and subsequently Master of Technology in computer science and engineering from Calcutta University, India in 2002 and 2004. He was awarded PhD degree from Indian Institute of Engineering Science and Technology (IIEST), Shibpur in 2016. Currently he is working at Indian Institute of Engineering Science and Technology, India as Associate Professor, in the department of Information Technology. His research interest includes VLSI testing, embedded Systems, hardware security.

Chandan Giri has been at Indian Institute of Engineering Science and Technology, Shibpur since 2008 as Associte Professor, Dept. of Information Technology. His current research is focused on testing and design-for-testability of integrated circuits (especially 3D and multicore chips) and Wireless Sensor Network. His research project has included 3D multi-core IC testing. Research support is provided by the University Grant Commission, Govt. of India. C. Giri received his Bachelor of Technology (B.Tech) in Computer Science & Engineering from Calcutta University, India in 2000 and subsequently Master of Engineering (ME) in Computer Science & Engineering from Jadavpur University, Kolkata, India in 2002. He was awarded PhD degree from Dept. of Electronics and Electrical Communication Engineering of Indian Institute of Technology, Kharagpur in 2008. He also presented his research papers in several International Conferences. He is a member of IEEE and ACM.