IET Computers & Digital Techniques

Research Article

Scalable pseudo-exhaustive methodology for testing and diagnosis in flow-based microfluidic biochips

ISSN 1751-8601
Received on 5th April 2018
Revised 20th November 2019
Accepted on 11th February 2020
E-First on 31st March 2020
doi: 10.1049/iet-cdt.2018.5029
www.ietdl.org

Gokulkrishnan Vadakkeveedu[1] ✉, Kamakoti Veezhinathan[1], Nitin Chandrachoodan[1], Seetal Potluri[2]

[1]Indian Institute of Technology Madras, Chennai, India
[2]School of Engineering Sciences Technische Universität Dresden, Copenhagen, Germany
✉ *E-mail: gokulvadakke@gmail.com*

Abstract: Microfluidics is an upcoming field of science that is going to be used widely in many safety-critical applications including healthcare, medical research and defence. Hence, technologies for fault testing and fault diagnosis of these chips are of extreme importance. In this study, the authors propose a scalable pseudo-exhaustive testing and diagnosis methodology for flow-based microfluidic biochips. The proposed approach employs a divide-and-conquer based technique wherein, large architectures are split into smaller sub-architectures and each of these are tested and diagnosed independently.

1 Introduction

Microfluidic biochips miniaturise bio-chemical laboratories onto chips that are implemented at micro-scales. These chips can manipulate fluids at sub-micro levels and have been demonstrated to be able to execute bio-chemical reactions with better efficiency at lesser costs. These devices are intended to be used in many safety-critical applications including point-of-care health-care, drug discovery, bio-hazard detection and DNA sequencing [1]. Microfluidic devices are broadly classified as *flow-based* devices and *droplet-based* devices [2]. The droplet-based devices work by manipulating individual droplets on electrodes using the principle of electro-wetting on dielectric. The droplet-based devices are reconfigurable by design like the field-programmable gate arrays [3] and can hence be reused for different applications. Conventional flow-based devices on the other hand contain etched micro-channels and valves, and are not reconfigurable. However, fully programmable valve arrays were later proposed [4, 5] and resulted in more flexible and highly-reconfigurable flow-based devices. Each microfluidic component in these devices is replaced by valve arrays which can be reconfigured as per the application. Hence, these devices contain even larger number of valves, have stronger constraints on channel and valve designs [6, 7] and hence end up being costlier than a conventional flow-based biochip. Application-specific flow-based devices are significantly cheaper and more efficient in execution (higher throughput – number of chemical reactions completed in unit time) than droplet-based devices [2]. Ho [8] reiterates the relevance of flow-based devices in current times and in the future. In this paper, we propose a novel

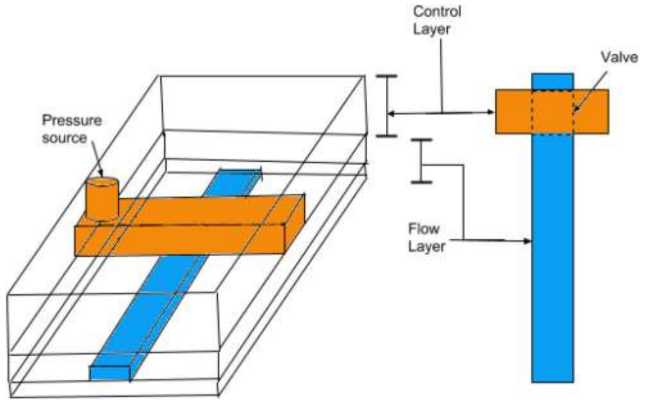

Fig. 1 *Basic structure of a flow-based microfluidic biochip valve [9]*

pseudo-exhaustive testing and diagnosis technique for application-specific flow-based microfluidic biochips. As shown in Fig. 1, a flow-based microfluidic biochip is fabricated in two layers – a *flow* layer and a *control* layer. Each of the different fluids required for the experiment flows through the various etched micro-channels in the flow layer. The control layer controls the sequence and timing of flow of these fluids, thereby realising the desired sequence of operations (chemical reactions) on them. The control layer is fabricated above the flow layer and has pressure valves embedded in its intersections with the flow layer. These valves are activated and deactivated by pumping air though the control layer. If the air pressure in the control layer reaches a valve, it bulges the latter and thus stops the fluid flow through the flow layer. A valve that is stopping the fluid flow is said to be in its *activated* or *closed* state. In its normal state, the valve is said to be in its *deactivated* or *open* state. The inlets and outlets for air in the control layer are collectively called as the *control ports* of the chip. The inlets and outlets for the fluid in the flow layer are collectively called as *flow ports* of the chip.

There are two kinds of faults that can possibly ensue on a biochip – *block* and *leak*. These faults can happen either on the valves or channels, in the flow-layer or in the control layer. These faults are explained in detail in the next section. Testing of these faults is done as follows. One of the flow ports is connected to a pressure source. The remaining flow ports are connected to pressure sensors. Now, valves are activated (closed state) and deactivated (open state) to achieve the different test scenarios. A particular assignment of states to the valves is called a *test vector*. A test vector is usually represented as a string of 1 and 0 s, with a 1 indicating an open valve state and a 0 indicating a closed valve state. Given a flow-based biochip and the state of its valves, the output (pressure) at the pressure sensors in the fault free biochip in response to the input (pressure) applied to one of the flow ports can be estimated using a simple simulation (this is the golden reference value). At the time of post-fabrication testing, the input (pressure) is applied to the same flow port as in the golden reference and pressures at different pressure sensors are sensed and compared with that of the golden reference to detect any faults in the biochip.

2 Faults in flow-based biochip

It is interesting to note that all faults on a flow-based biochip are equivalent to either:

- A block fault on a valve; or,
- A leak fault on a valve; or,

- A *qualitative fault* that causes pressure values on the pressure sensors to be different from those observed in golden reference [10]. This happens when a channel or valve is partially blocked. The pressure in this case is not fully transmitted from the pressure source to the destination valve.

The different faults are explained in detail below:

- Block in flow layer: Block in the flow layer leads to the obstruction of fluid flow through the channel. This can happen either due to a block in the channel or a block in the valve. However, both of these have the same effect and hence block in the channel is equivalent to a block fault in the valve to which it is connected to. Using the terminology of single stuck-at-faults in digital IC testing, these faults translate to *stuck-at-0* valves.
- Leak in the flow layer: Similar to a block in flow layer, a leak can happen either on a channel or on a valve. A lone leak in the channel can either be partial leak or complete leak. A complete leak leads to no fluid reaching the next valve. This is analogous to a stuck-at-0 fault on that valve. However, if it is a partial leak, it translates to a qualitative fault. Multiple leaks in nearby channels can lead to the fluid seeping outside the channels through which they flow. This leads to the possibility of fluid in one channel seeping into other. This is analogous to an OR bridging fault in logic circuits and can be modelled as *stuck-at-1* (leak) valves on each of the leaking channels just before the leaking spot.
- Block in the control layer: A block in the control layer leads to the air pressure not being able to activate a valve. This can lead to a state wherein, a microfluidic valve cannot be activated (closed). This leads to a stuck-at-1 valve.
- Leak in the control layer: A leak in the control layer leads to the air seeping out of the air channel through which it is flowing. Similar to the lone leak in the flow layer, a lone leak in the control layer can be either partial or complete. A complete leak leads to a state wherein, a valve cannot be closed at all. This is a stuck-at-1 valve. A lone partial leak in the control layer causes partial activation/deactivation of valves, causing qualitative faults in the flow layer. When there are leaks in the control layer in multiple channels, air pressure from one channel can seep into other, causing activation of unintended valve(s). This is analogous to a AND bridging fault in logic circuits and can be modelled as stuck-at-0 (block) valves on each of the leaking channels just after the leaking spot.

Thus, all faults in a flow-based microfluidic biochip are *functionally equivalent* to faults in valves. Hence, testing of all the valves in an architecture will ensure testing of the entire chip [11].

3 Previous work

As the size of biochip increases, generation of optimal number of test vectors to ensure maximum possible fault coverage becomes a prohibitively time-consuming task. A testing methodology for flow-based biochips has been proposed in [9], wherein the chip architecture is converted into a Boolean circuit and traditional ATPG algorithms [12] are used to generate test vectors. An example microfluidic architecture consisting of a mixer and a branch, and its corresponding mapped Boolean circuit is shown in Fig. 2. Here, inlet flow port $n1$ is used as the source for pumping air and other flow ports, namely $O1$, $O2$ and $O3$, are connected to pressure sensors to measure the pressure of air flow reaching them. The nodes $a, b, c, d, e, f, g, h, i, j$ and k are the valves. The mapping of architecture to the Boolean circuit is done as follows: every valve maps onto a primary input. Every pressure sensor node is a primary output. Every primary output O_p is realised as a Boolean function B_{O_p} of primary inputs as follows: all primary inputs corresponding to valves in any path from the pressure source ($n1$) to any of the pressure sensors O_i, will be a product term B_{O_i}, $1 \le i \le 3$. Note that there are two paths between $n1$ and $O1$ namely, a - b - c - d and a - g - h - f - e - d. Thus, as in Fig. 2*b*, $B_{O_1} = (abcd + aghfed)$. It is interesting to note that generating a test for detecting a block fault in valve g in Fig. 2*a* is equivalent to generating a test vector to detect line g stuck-at-0 in Fig. 2. Note that $(a, b, c, d, e, f, g, h) = (1, 0, X, 1, 1, 1, 1, 1)$ is a test vector for line g stuck-at-0 in Fig. 2*b*. This implies that by closing valve b, and keeping the valves a, g, h, f, e and d open, if the air reaches $O1$, then g is not blocked. It is also interesting to note that valve c can be either closed or open (do not care). For large biochip architectures, generating test patterns in this manner will take prohibitively huge time as the number of paths between the ports will be much higher [13]. Hence, the methodology proposed in [9] is not scalable. In addition, automatic test pattern generators (ATPGs) for digital circuit use the single stuck-at fault model [14]. Using them for test generation in this scenario restricts the fault model of the underlying biochip under test. This paper proposes a more robust fault model than the single stuck-at-fault model.

The generated test vectors are applied onto the biochip one by one and pressure values reported at the pressure sensors are noted. A difference in pressure sensed in the golden reference and the pressure sensed at the pressure sensors implies that there is a fault in the chip. As the testing is done using air pressure, there are no residues left after the testing process. Thus, the above is a *clean* testing methodology.

When a chip under test fails, the faulty chip has to undergo a diagnosis process to identify the exact location and physical cause of the fault. A fault diagnosis technique for flow-based biochips has been suggested in [15]. This technique needs a significant amount of pre-computation, which includes the following:

- Maintaining a syndrome list that contains the syndromes corresponding to each possible fault for each test vector; and,
- Maintaining a path dictionary consisting of all paths from the pressure source to every flow port.

The syndrome list and path dictionary are fed as inputs to the diagnosis algorithm. The diagnosis algorithm in turn, is modelled as an instance of the hitting-set problem [16]. It is known that the hitting-set problem is NP-complete [16], and only heuristic solutions are possible. With futuristic microfluidic chips containing thousands of valves [17], this solution can be prohibitively time consuming and also lead to poor quality of diagnosis due to degraded diagnostic resolution.

3.1 Contributions of this paper

This paper proposes a pseudo-exhaustive fault testing and diagnosis approach for flow-based biochips. This approach employs a divide-and-conquer based technique to split any large architecture into small sub-graphs. Testing and diagnosis of faults can be performed independently on these sub-graphs. This approach helps to sieve out many of the valves from the possible solution space for the diagnosis problem. This paper initially proposes a technique to split the larger architecture to smaller sub-graphs and later explains how fault testing and diagnosis may be performed on these sub-graphs. These sub-graphs can also be stored to reuse. The existing testing and diagnosis approaches [9, 11, 15, 18] does not avail the idea of localising the possible problem space to a cone of influence of the outlet ports. The testing and diagnosis technique presented in this paper uses this novel method. Experimenting the proposed approach on both real and synthetic benchmarks shows that it is more scalable and better in terms of time complexity, in comparison to the best existing diagnosis approach [15].

4 Motivational example

In Fig. 2*a*, assume that the pressure source has been connected to $n1$ and the pressure sensors are connected to flow ports $O1$, $O2$ and $O3$. Consider the case when all the valves are open and pressure is being sent in through the pressure source. In this case, if the pressure sensor at $O1$ does not detect any pressure, the architecture is said to have a *block* fault. If we assume that a logical 1 denotes a free-flow and a logical 0 denotes a stop-flow, this state of $O1$ can be represented as 1/0 (free-flow in the absence of fault and stop-

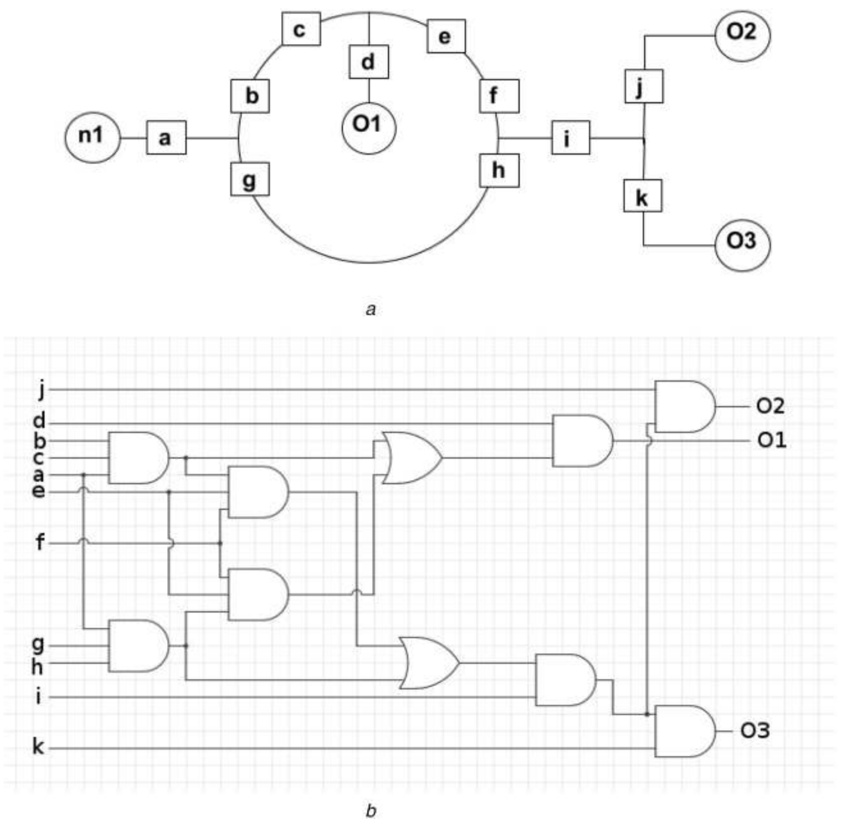

Fig. 2 *Simple microfluidic chip layout and corresponding AND-OR structure [9]*
(a) Example of microfluidic architecture consisting of a mixer and a branch, ***(b)*** AND-OR structure corresponding to Fig. 2*a*

flow in the presence of fault). Similarly, the case of a stop-flow in the absence of fault and free-flow in the presence of fault is represented as a 0/1 and is called a *leak*.

Let V be the set of all valves in the architecture. The *root cause set* $S_F^{I,U}$ of a fault F detected at a flow port U, when the pressure source was connected to flow port I is the set wherein every element of $S_F^{I,U}$ is a *minimal* subset E of V satisfying the following property: if $E = \{v_1, v_2, v_3 \ldots v_k\} \epsilon S_F^{I,U}$, then the dysfunctioning of all valves in E shall result in the fault F at U, and the correct functioning of at least one of the valves in E will not cause the fault. The root cause set for a blockage fault at $O1$, when the pressure source is $n1$, in Fig. 2*a*, is denoted by $S_{\text{block}}^{n1,O1} = \{\{a\}, \{d\}, \{c, e\}, \{c, f\}, \{c, h\}, \{c, g\}, \{b, g\}, \{b, h\}, \{b, f\}, \{b, e\}\}$. The root cause set for a leakage fault at $O1$ in Fig. 2*a* is denoted by $S_{\text{leak}}^{n1,O1} = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}, \{h\}\}$.

Thus, the testing and diagnosis search for a fault observed at a flow port can be carried out after confining the architecture to the corresponding root cause set. For example, in Fig. 2*a*, given that O1 is blocked (1/0), one need not look for faults in the valves i, j and k, as they are not part of $S_{\text{block}}^{n1,O1}$. This allows us to come up with an outlet-specific testing and diagnosis technique. It also helps in significantly pruning the search space for fault diagnosis in most cases [19]. Mathematically, we can arrive at the root cause sets by considering the following property: given a pressure source $n1$ and outlet $O1$, let P denote the set of paths between $n1$ and $O1$. For $p \epsilon P$, let V_p denote the valves in p. Any test to detect block faults, at outlet $O1$ by applying air pressure at source $n1$ should ensure that there exists at least one path $p \epsilon P$ between $n1$ and $O1$ such that all the valves in V_p are open. This statement can be mathematically stated as follows:

Property 1: $\exists p \epsilon P, \forall v \epsilon V_p$, v is open.
The test fails if Property 1 is violated resulting in Property 1′.

Property 1′: $\forall p \epsilon P, \exists v \epsilon V_p$, v is closed or blocked.

The construction of $S_{\text{block}}^{n1,O1}$ ensures that at least one of its elements is faulty (blocking) *if and only if* Property 1 is contradicted while applying the test.

Similarly, any test to detect a leak fault, at outlet $O1$ by applying air pressure at source $n1$ should ensure that in every path $p \epsilon P$, at least one node of V_p is closed. This statement can be mathematically stated as follows:

Property 2: $\forall p \epsilon P, \exists v \epsilon V_p$, v is closed.
The test fails if Property 2 is violated resulting in Property 2′.

Property 2′: $\exists p \epsilon P, \forall v \epsilon V_p$, v is open or leaking.

The construction of $S_{\text{leak}}^{n1,O1}$ ensures that at least one of its elements is faulty (leaking) *if and only if* Property 2 is contradicted while applying a test.

Example: A test to detect leak in the path between $n1$ and $O1$ in Fig. 2*a* is to keep valves a, c, d, e, f and h open, and close b and g. If this test fails, then either b or g or both should leak. Note that both the valves b and g belong to $S_{\text{leak}}^{n1,O1}$.

The root cause sets can be computed independent of the testing and diagnosis process. For the sake of completeness, the following graph theory fundamentals are stated. A bi-connected graph is a graph that remains connected, even if one of its vertices is removed. A *bi-connected component* of a graph G is defined as a maximal bi-connected sub-graph of G. In Fig. 3, $\{A\}$, $\{F\}$, $\{K\}$, $\{B, C, E, D\}$ and $\{G, H, J, I\}$ are the bi-connected components. *Articulation points* of a graph are those vertices, which when removed disconnects it and thus increases the number of connected components in the graph. Vertices B, E, F, G and J are articulation points of the graph in Fig. 3. Given a source S and sink K, a *cut* of a graph is defined as a partition of vertices of the graph such that S and K are in different connected components. A *minimal cut* (mincut) is defined as a cut in which the removal of any of its members results in the set not being a cut. In Fig. 3, assuming A is the source and K is the sink, each of the articulation vertices are by themselves a minimal cut of the graph whose removal separates A and K into two connected components. In addition, $\{C, D\}$ and

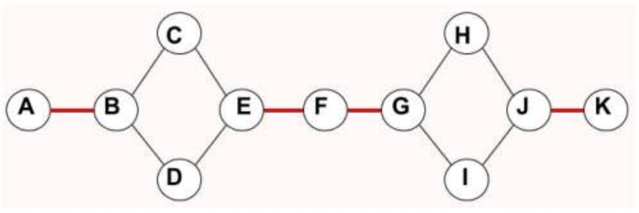

Fig. 3 *Example graph for illustration of bi-connectivity*

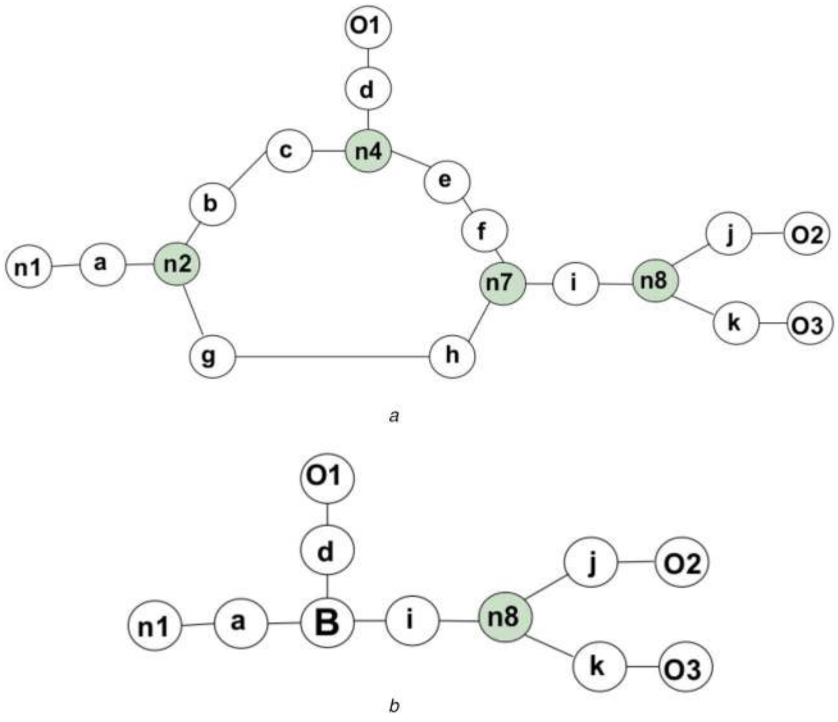

Fig. 4 *Graph representation and corresponding block cut graph of the layout in Fig. 4a*
(a) Graph representation corresponding to the layout in Fig. 2*a*, ***(b)*** Block cut graph of the graph in Fig. 4*a*

$\{H, I\}$ represent two other minimal cuts. Note that all vertices in a given minimal cut are present in the same bi-connected component.

5 Fault model for flow-based biochips

As was explained in Section 2, all faults on a flow-based biochip translate as faults on valves. Further, as [15] mentions, it is more important to identify the faulty sites exactly rather than the fault type. This is because, once the faulty site has been correctly identified, the fault type can be easily verified using a microscope. The techniques proposed in this paper work based on the *single fault type assumption*. Note that this fault model is more robust than the single stuck-at fault model employed by ATPG for digital circuits, wherein, *exactly one* gate is assumed to be faulty in the latter while multiple faults are possible in the former, however of the same type. According to this assumption, either all faults are block faults, or all faults are leak faults. Thus, there can exist no two faulty valves such that one of them has a block (stuck-at-0) and the other has a leak (stuck-at-1). This assumption can be safely followed considering the following facts:

- There is very less probability of the same physical fault leading to a block fault on one valve and a leak fault on other; and,
- Flow layer and control layer are fabricated separately, and the flow layer is mounted over the control layer. Hence, there is no possibility of the same physical cause affecting both the flow layer and control layer simultaneously.

Thus, the single-fault assumption can be wrong only when there are multiple physical causes leading to different fault types and failures on different valves. However, empirically this is a very rare condition [15].

6 Graph representation of biochip architecture

Before testing and subsequent diagnosis of flow-based microfluidic architectures, we start by modelling these architectures as graphs. The flow ports on the chip are represented by vertices labelled as source and sinks(s). Valves are the internal vertices. Channel junctions in the original architecture are represented using dummy vertices. These vertices can be ignored during the testing and diagnosis process. The graph representation of the architecture in Fig. 2*a* is shown in Fig. 4*a*.

7 Outlet specific sub-graph identification

Let $G = (V, E)$ denote the graph corresponding to a given microfluidic architecture. Using the bi-connected components of G, the *block-cut graph*, BC_G is constructed, where every node corresponds to a bi-connected component in G. There exists an edge between two vertices of BC_G, if and only if there exists at least one edge between the vertices of corresponding bi-connected components of G. BC_G will always be a tree [20]. The block-cut tree of the graph in Fig. 4*a* is shown in Fig. 4*b*, where B denotes the bi-connected component with vertices $\{n2, b, c, n4, e, f, n7, h, g\}$. BC_G being a tree, there exists a unique path between any two of its vertices. G is called the *expanded* graph of BC_G ($G = Ex(BC_G)$). The notion of expanded graph can be extended to sub-trees of BC_G and corresponding sub-graphs of G. For example, $Ex(B)$ is the bi-connected component as mentioned above.

Given any two vertices in a block-cut-graph (BC_G) corresponding to two flow ports I and O, such that I is connected to a pressure source and O is connected to a pressure sensor, two cases arise:

- I and O lie inside the same bi-connected component. In this case, there exists a sub-graph $R^{I,O}$ of G that maps on to a single node in BC_G representing the bi-connected component that contains all the paths between I and O in $Ex(BC_G)$.
- The pressure source and the pressure sensor do not lie on the same bi-connected component. In this case, all paths between I and O in $Ex(BC_G)$ are contained in the unique path in the sub-graph $R^{I,O}$ of BC_G, starting from the vertex representing the bi-connected component containing I and ending at the vertex representing the bi-connected component containing O. $Ex(R^{I,O})$ is the expanded graph of all bi-connected components in $R^{I,O}$.

Hence, an outlet-specific sub-graph is either a single node in the BC_G or a path in the BC_G.

Once an outlet-specific sub-graph $R^{I,O}$ corresponding to an outlet O has been identified, $S^{I,O}_{\text{leak}}$ and $S^{I,O}_{\text{block}}$ can be calculated as follows. $S^{I,O}_{\text{leak}}$ is the set of all valves in $Ex(R^{I,O})$. For example, $S^{n1,O1}_{\text{leak}}$ in Fig. 4*b* includes valves $\{a\}, \{d\}$ and all valves contained in B, namely, $\{b\}$, $\{c\}$, $\{e\}$, $\{f\}$, $\{g\}$ and $\{h\}$. $S^{I,O}_{\text{block}}$ is computed as follows. If I and O belong to the same bi-connected component B of G, then $S^{I,O}_{\text{block}}$ is the set of minimal cuts of B that separates I and O. If I and O are not in the same bi- connected component of G, then $R^{I,O}$ is a path that will contain articulation points and vertices representing bi-connected components that are on the path from I to O in G. The valves corresponding to articulation points in $Ex(R^{I,O})$ are members of $S^{I,O}_{\text{block}}$. For every vertex B in $R^{I,O}$ representing a bi-connected component, there exists two unique vertices u and v in $Ex(B)$ that connects it to the path in $Ex(R^{I,O})$. For example, in $R^{n1,O1}$, for the vertex B in Fig. 4*a*, the nodes $n2$ and $n4$ connect it to the rest of the path in $Ex(R^{n1,O1})$. The set of valves corresponding to minimal cuts in B that separates nodes u and v, are members of $S^{n1,O1}_{\text{block}}$. Note that $S^{I,O}_{\text{block}}$ and $S^{I,O}_{\text{leak}}$ can be pre-computed independent of the testing and diagnosis process.

An example of outlet specific sub-graph identification in the case of ChIP (Chromatin Immuno Precipitation) biochip is shown in Fig. 5. Fig. 5*a* shows the ChIP architecture. Here, the sub-layout corresponding to flow ports $P2$ (connected to pressure source) and $P10$ (connected to pressure sensor) is shown in Fig. 5*b*. Thus, while testing and subsequently diagnosing for a fault observed at flow port $P10$, we can confine our search to valves in Fig. 5*b*.

Note that outlet-specific sub-graphs can be pre-computed in advance and used multiple times during the testing and diagnosis process.

8 Pseudo-exhaustive testing

Unlike the currently reported testing process ([9, 11]) where the entire biochip architecture is converted to an AND-OR structure, after which ATPG tools are used to generate the test vectors, we propose a novel scalable pseudo-exhaustive [21] testing technique

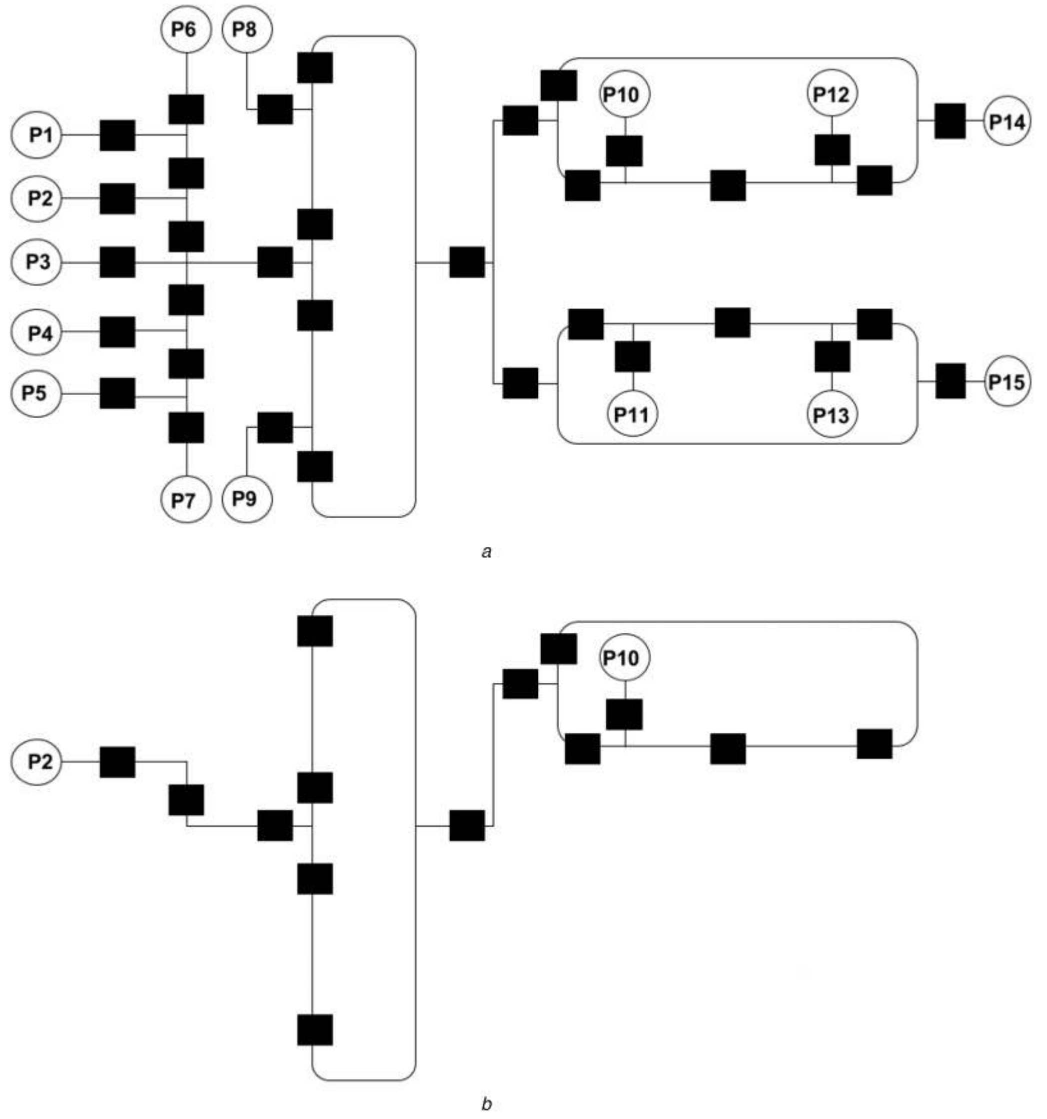

Fig. 5 *Illustration of sub-graph identification on ChIP (Chromatin Immuno Precipitation)*
(a) Original ChIP architecture layout. Circles labelled P1-P5 represent the inlets, P6-P15 are the outlets and the black rectangles represent the valves [15], ***(b)*** Sub-layout of ChIP architecture corresponding to inlet P2 and outlet P10, after the stage I minimisation

```
Input: $I$, $O$, $R^{I,O}$ and $G$
Output: Test vectors and the pressure table for $Ex(R^{I,O})$

/* $G$ is the graph representation of the microfluidic
  architecture*/
/* $I$ is the inlet flow port connected to pressure source*/
/* $O$ is a outlet flow port connected to pressure sensor*/
/*$V_{Block}$ maintains the set of valves already tested for block
  faults */
/*$V_{Leak}$ maintains the set of valves already tested for leak
  faults */

1. $V_{Block} = V_{Leak} = \phi$;
2. Compute $S_G = Ex(R^{I,O})$;
3. Compute Root-cause set $S^{I,O}_{Block}$;
4. /*Check for block faults in valves that are articulation points*/
4.1. Assign 1 to all valves $v \in G$;
4.2. Note pressure at $O$ as $P$;
4.3. Add all articulation points in $S_G$ to $V_{Block}$;
4.4. Output this test vector (values assigned to valves) along with
$P$;
5. /*Check for block faults in remaining valves*/
for each valve $v \in S_G$ such that $v \notin V_{Block}$ do
    5.1. Select a set $C \in S^{I,O}_{Block}$ such that $v \in C$;
    5.2. Assign 1 to $v$ and 0 to valves $C$-$\{v\}$;
    5.3. Assign 1 to valves in ($G$-$C$);
    5.4. Note pressure sensed at $O$ as $P$;
    5.5. $V_{Block} = V_{Block} \cup v$;
    5.6. $V_{Block} = V_{Block} \cup u, \forall\, u \in S_G$ such that $u \notin C$
    and $\exists\, C' \in S^{I,O}_{Block}$ such that $u \in C'$ and $C$-$\{v\}$ = $C'$-$\{u\}$;
    5.7. Output this test vector (values assigned to valves) along
    with $P$;
end
6. /*Check for leak faults in valves*/
for each valve $v \in S_G$ such that $v \notin V_{Leak}$ do
    6.1. Select a set $C \in S^{I,O}_{Block}$ such that $v \in C$;
    6.2. Assign 0 to all valves in $C$;
    6.3. Assign 1 to all valves in ($G$-$C$);
    6.4. Note pressure sensed at $O$ as $P$;
    6.5. $V_{Leak} = V_{Leak} \cup u, \forall\, u \in C$;
    6.6. Output this test vector (values assigned to valves) along
    with $P$;
end
```

Fig. 6 *Algorithm 1: pseudo-exhaustive test vector and pressure table generation*

that uses *only the relevant* parts of the circuit structure to carry out the testing process and not the full one.

Section 7 shows that, given a graph G and source port I through which air is pumped in, and an outlet port O where the pressure is sensed and found to be a faulty response (different than what is expected from the golden reference simulation), there exists a sub-graph $Ex(R^{I,O})$ of G, corresponding to the relevant parts of the circuit that contains all the possible root cause sets for the fault observed at O. Thus, the testing and diagnosis process followed is specific to the outlet port O. Such a pseudo-exhaustive *outlet-specific testing* approach gives the following advantages:

- Smaller sub-graph sizes makes it easier to target 100% fault coverage;
- Simplified test vector generation;
- Flexibility of independently testing intended portions of the chip; and,
- Possibility of parallel test execution;

The proposed testing and diagnosis procedure is explained in three phases, namely, test generation, test application and fault diagnosis. For a given chip under test with a specified input flow port through which air is to be pumped and output flow ports connected with pressure sensors, the test generation phase computes:

- The test vectors each specifying the state of different valves; and,
- A pressure table, which lists the expected responses (pressure values) on each of the pressure sensors. This is computed by executing each of the test vectors in the golden reference simulation platform.

Algorithm 1 (see Fig. 6) takes as input the sub-graph containing the source flow port (connected to the pressure source) and an outlet flow port (connected to the pressure sensor), and generates test vectors that shall detect *block* and *leak* faults in valves in the sub-graph. Algorithm 1 *is executed repeatedly for each such pair of source and outlet ports*.

We shall proceed to explain the different steps of Algorithm 1 in detail. Note that, every articulation point in a sub-graph is a min-cut of the sub-graph, and corresponds to an element in the root cause set for block faults. Step 4 of Algorithm 1 computes a test vector that keeps all valves open and hence, checks for block faults at all the articulation points in the sub-graph.

Steps 5 of Algorithm 1 computes test vectors that check for block faults on valves other than the articulation points. The details of this step are as follows: given a node $n1$ that is connected to pressure source, and a node $O1$ that is connected to a pressure sensor, let P denote the set of paths between $n1$ and $O1$. Note that the elements of $S^{n1,O1}_{\text{block}}$ correspond to the mincuts of $Ex(R^{n1,O1})$. Further, any min-cut disconnects all paths between source and sink, in this case $n1$ and $O1$, respectively. Every valve in a given element of the root cause set for block faults shall belong to a unique path in P. In other words, given a $C \in S^{n1,O1}_{\text{block}}$, for any $u, v \in C$, there is a path between $n1$ and $O1$ that contains u and not v, and vice-versa. From Property 1 in Section 4, any test for block faults in P has to ensure that there exists at least one path $p \in P$ such that all valves in p are open. Hence, choosing a $C \in S^{n1,O1}_{\text{block}}$ and keeping one of the valves $v \in C$ open and all valves in $C - \{v\}$ closed, is a test vector ***TV*** that detects block fault in v. Step 5.1 to Step 5.5 of Algorithm 1 implement the above detail. Let C' be another element of $S^{n1,O1}_{\text{block}}$ different from C. Now, if there exists $u \in C$ and $v \in C'$, such that $C - \{u\} = C' - \{v\}$, we can conclude the following:

- C and C' are part of the same bi-connected component in $Ex(R^{n1,O1})$. This follows from the fact that all the elements of a mincut have to be in the same biconnected component; and,
- u and v are on the same path between $n1$ and $O1$. As explained above, for any mincut, for every path between $n1$ and $O1$, there will be exactly one valve from the path that is in the mincut. Else, this shall contradict the property of minimality of the mincut. If P_v is the set of paths in P that gets blocked by closing valve v and P_u is the set of paths in P that gets blocked by closing valve u, then $P_u = P_v$. This follows from the following facts:

  i. C and C' are mincuts that differ only with elements u and v.
  ii. Elements of $C - \{v\}$ will block all paths except the ones mapped to v. This statement is true for $C - \{u\}$ and u as well.
  iii. Given that $C - \{u\} = C - \{v\}$, the paths not blocked by $C - \{v\}$ and $C - \{u\}$ must be the same.

Thus, the test vector that detects a block fault in valve v can also detect a block fault in valve u. By this, we are actually finding the set of *equivalent valves*, on which the test vector can detect the block faults and removing them from further consideration. This is implemented in Step 5.6 of Algorithm 1.

Step 6 of Algorithm 1 computes the test vectors that check for leak faults on valves. By Property 2 in Section 4, any test for leak faults has to ensure that all paths $p \in P$ are blocked by some valve. Hence, choosing a $C \in S^{n1,O1}_{\text{block}}$ and keeping all valves in C closed, is a test vector that detects leak faults in any valve $v \in C$.

As seen above, a test vector may be able to detect faults on multiple valves. This property can be used to minimise the number

```
**Input**: $G, V_C, V_M, I, F, TV$
**Output**: Valves that are fault suspects

/*$G$ is the graph representation of the biochip architecture
$V_M$ is the set of outlet flow ports connected to pressure sensors
  that reported errors
$V_C$ is the set of all the outlet flow ports
$I$ is the inlet flow port connected to pressure source
$F$ is the fault type identified from the test results
$TV is the set of test vectors that did not report errors$
$TV_{out}$ denotes the test vector for outlet flow port $out$ that did
  not report error at the pressure sensor connected to $out$*/

1. Mark state of all valves in $G$ as *unknown*;
2. /* initialize all valves in sub-graphs of faulty outlets as faulty*/
for every outlet out $\epsilon$ $V_M$ do
  begin
  Mark all valves that are part of $S_F^{I,out}$ as faulty;
end
3. /*These are outlets connected to pressure sensors that did not
report error */
for every outlet out $\epsilon$ $V_C$-$V_M$ do
  begin
  /*Since the outlet $out$ did not report error, all valves in
  $S_F^{I,out}$ should be working correctly. Therefore, mark all
  valves in $S_F^{I,out}$ as non-faulty/*
  for every element $V_i$ $\epsilon$ $S_F^{I,out}$ do
    Mark all valves in $V_i$ as non-faulty;
  end
end
4. /* Test vectors that did not report errors */
for every test vector $TV_O \in TV$ that did not report error do
  if $F$ is a block then
    if $TV$ assigned 1 to all valves then
      Mark all valves that are articulation points, as
      non-faulty;
    end
    else
      Let $X$ be the set of valves assigned 0 by $TV$;
      Select a $C \in S_{block}^{I,out}$ that contains all elements of
      $X$;
      Let $v = C$-$X$;
      Mark $v$ as non-faulty;
      $\forall$ $u$ such that $u \notin C$ and $\exists$ $C' \in S_{Block}^{I,O}$ such that
      $u \in C'$ and $C$-$\{v\}$ = $C'$-$\{u\}$, mark $u$ as non-faulty;
    end
  end
  if $F$ is a leak then
    Let $X$ be the set of valves assigned 0 by $TV$;
    Mark all valves in $X$ as non-faulty.
  end
end
5. Output all valves that are marked as faulty.
```

Fig. 7 *Algorithm 2: pseudo-exhaustive fault diagnosis*

of test vectors generated. Step 5.6 and Step 6.5 of Algorithm 1 achieve this *test vector minimisation*. The iterative loops in Step 5 and Step 6 of Algorithm 1 handle this using two sets V_{Block} and V_{Leak} that maintain the valves for which test vectors have already been generated to detect block and leak faults, respectively.

It is interesting to note that, in Algorithm 1, only the root cause sets for block faults ($S_{\text{Block}}^{I,O}$) has been used to compute the test vectors for both block faults and leak faults. Thus, single-fault-type model allows the detection of multiple faulty valves by a single test, unlike the single-stuck fault model used in conventional ATPGs.

The test application phase involves the following steps:

- Applying the different test vectors – this involves opening and closing of different valves as specified in the test vector, using the control layer, followed by pumping of air through the specified input flow port.
- Recording the response on the pressure sensor for each test vector; and,
- Reporting *failures* (mismatch of the observed pressure on an output flow port with that given in the pressure table computed by Algorithm 1).

Further, this pseudo-exhaustive test application phase opens up the possibility of parallel testing for sub-graphs that contain different set of valves including the input and output ports. Thus, for two sub-graphs S_1 and S_2, if their corresponding valve sets, input port and output ports V_{S_1} and V_{S_2} are such that $V_{S_1} \cap V_{S_2} = \phi$, then the testing of the two sub-graphs can be done in parallel. Note that the set of sub-graphs that can be tested in parallel can be pre-computed once. These sub-graphs can be reused multiple times during the testing phase.

The proposed diagnosis technique is explained in the next section. In the diagnosis phase, every mismatch reported by the test application phase is input to the pseudo-exhaustive diagnosis algorithm (Algorithm 2 (see Fig. 7)), that in turn identifies the most probable list of faulty valves. This immensely aids in the diagnosis process by *localising* the faulty area in the chip which can further be investigated using microscopes to find the actual *physical* fault.

9 Pseudo-exhaustive diagnosis

As mentioned in Section 3, the currently reported testing process uses ATPG tools to generate the test vectors. Note that such a test vector generation only ensures that fault on a valve v is reflected at some outlet flow port O. The fault may not be reflected at some output port out such that v is a valve in $Ex(R^{I,\text{out}})$. This is illustrated by the following example: consider the architecture given in Fig. 2*a*. Let $n1$ be connected to the pressure source. Let $O1$, $O2$ and $O3$ be connected to the pressure sensors. Now, consider that there is a block fault in valve c. Note that one of the tests to detect this fault is to keep valve e closed. Such a test reports a faulty pressure at the pressure sensor connected to $O1$. However, it does not report any error at pressure sensors connected to $O2$ and $O3$. Note that, an ATPG tool might use such a generated test vector to detect this fault. The tool only tries to ensure that the fault in a valve v translates to a faulty pressure reported at any of the pressure sensors. However, in the proposed test vector generation, for a block fault on valve b, there will be three test vectors $\boldsymbol{TV}_{O1}$, $\boldsymbol{TV}_{O2}$ and $\boldsymbol{TV}_{O3}$, each of which ensures that the block fault on valve v is reflected at the corresponding outlets $O1$, $O2$ and $O3$, respectively. This ensures the two cases listed below:

- If a pressure sensor connected to an outlet flow port out does not report an error, all elements of $S_{I,\text{out}}^{F}$ has to be non-faulty; and,
- If there is a fault on a valve v, for any outlet flow port out such that $v \in Ex(R^{I,\text{out}})$, erroneous pressure would be reported at the pressure sensor connected to out.

However, such a guarantee cannot be given for the currently reported fault diagnosis techniques, as explained above. In this paper, we propose a much simpler fault diagnosis than the currently reported techniques ([15, 19]) using these guarantees.

The testing process is intended to be a pass/fail test. There may be multiple possible faults that would have been responsible for causing the same faulty responses. Hence, there is the need for a diagnosis process to more accurately identify the physical nature of faults. In addition, the diagnosis process is important to give inputs for modification of fabrication processes so that the observed physical faults can be avoided in the future [15]. This can also be used to facilitate use of partially defective chips.

Once a biochip has been found to be faulty, the graph corresponding to the faulty biochip, the inlet flow port to which the pressure source is connected, the outlet flow ports that reported errors and the test vectors that did not report errors are taken as inputs to our proposed diagnosis technique. As mentioned before, we follow the single fault type assumption. Hence, if the faulty response was observed while testing for block faults, we can assume that the chip has only block faults. On the other hand, if the

Table 1 Test vectors for block faults

Valve(s) being tested	Test vector							
	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
a, *d*	1	1	1	1	1	1	1	1
b, *c*	1	1	1	1	1	1	0	1
g, *h*, *e*, *f*	1	0	1	1	1	1	1	1

faulty response was observed while testing for leak faults, we can assume that the chip has only leak faults. In a digital circuit, there exists the possibility of faults getting masked by each other [22]. However, in the case of biochips, this is not possible by the simple fact that a block fault can never become a leak fault because of another block fault. From Property 1 and Property 2, note that the test vectors generated are such that the syndrome of a block fault is *nil* pressure reported at the pressure sensor, when the pressure table would have reported a non-zero pressure value for the same test vector. Similarly, the test vectors generated for leak faults are such that the syndrome of a leak fault is a non-zero pressure value reported at the pressure sensor, when the pressure table would have reported a *nil* pressure for the same test vector. In the remaining part of this section, let $F \in \{\text{block}, \text{leak}\}$ denote the fault type identified from the test results.

Like the proposed test methodology which is a pseudo-exhaustive approach, the subsequent diagnosis process can also be carried out as a pseudo-exhaustive algorithm. Algorithm 2 details the proposed diagnosis technique wherein, initially, it sieves out valves from the list of fault suspects using the outlets that did not report errors during the test. This is achieved in Step 3 of Algorithm 2. Later, it sieves out more valves from the list of fault suspects using the test vectors that did not report errors. This is achieved in Step 4 of Algorithm 2. We will proceed to explain this step in more detail.

Given an inlet flow port I connected to the pressure source, let $\boldsymbol{TV}_{\text{out}}$ denote a test vector that did not report any error for the test for F in $Ex(R^{I,\text{out}})$. From the test vector generation proposed in Section 8, if F is a block, we can conclude the following:

- If $\boldsymbol{TV}_{\text{out}}$ assigned 1 to all valves in $Ex(R^{I,\text{out}})$, from Section 8 and Step 4 of Algorithm 1, the test vector was designed to test for block at articulation points. As $\boldsymbol{TV}_{\text{out}}$ did not report an error at the pressure sensor connected to out, all valves that are articulation points in $Ex(R^{I,\text{out}})$ can be removed from the list of fault suspects.
- If $\boldsymbol{TV}_{\text{out}}$ assigned 0 to some valves in $Ex(R^{I,\text{out}})$, from Section 8 and steps 5.1–5.3 of Algorithm 1, the test vector was designed to test for block on a valve(s) other than articulation points in a path from I to out in $Ex(R^{I,\text{out}})$. Note that a test vector for a block fault on valve v is computed by choosing a cut C such that $v \in C$ and keeping v deactivated and all other valves in C activated. As explained in step 5.6 of Algorithm 1 in Section 8, such a test vector also tests for block faults on equivalent valves of v. The equivalent set of valves for which $\boldsymbol{TV}_{\text{out}}$ is designed can be computed as follows: If X is the set of valves that were assigned a 0 (activated) by $\boldsymbol{TV}_{\text{out}}$, it can be understood that there exists a $C \in S^{I,O}_{\text{Block}}$ such that all elements of X are also elements of C and that $\boldsymbol{TV}_{\text{out}}$ is designed to test for valve v such that $v = C - X$ and equivalent valves of v. As $\boldsymbol{TV}_{\text{out}}$ did not report an error at the pressure sensor connected to out, v and all valves equivalent to it, can be removed from the list of fault suspects.

Similarly, if F is a leak, we can conclude the following:

- From Property 2 in Section 4, a test vector for leak on valve v is computed by choosing a $C \in S^{I,O}_{\text{block}}$ such that $v \in C$ and keeping all valves in C blocked. Such a test vector tests for block faults in all valves in C. Hence, if $\boldsymbol{TV}_{\text{out}}$ did not report an error, all valves that have been assigned a 0 by $\boldsymbol{TV}_{\text{out}}$ can be removed from the list of fault suspects.

Once the minimal set of fault suspects are computed from Algorithm 2, the exact physical location and cause of the fault can be verified using a microscope.

Thus, the test methodology proposed in this paper is not only simple and scalable compared to existing techniques, but also makes the task of fault diagnosis much easier than the techniques reported in the literature.

10 Pre-computation and complexity

It is interesting to note that, for a given biochip and an inlet flow port, the sub-graphs and the root cause sets can be pre-computed even before venturing on to the testing and diagnosis process. For the sub-graph identification, bi-connected components of a given graph $G = (V, E)$ can be computed in $O(|V| + |E|)$ time [23]. Given that flow-based networks are planar in nature, mincuts that will be used during the computation of root cause sets, outlet-specific sub-graph identification, test vector generation and fault diagnosis are computed using [24], which is a $O(|V|\text{loglog}|V|)$ algorithm for computing mincuts in an undirected planar graph with V vertices and a source and a sink. Note that this computation is limited to the small bi-connected components and not the complete graph.

The time required for computation of root cause sets $S^{I,O}_{\text{Block}}$ and $S^{I,O}_{\text{Leak}}$ and the time required for diagnosis that involves sieving out the faulty valves depend on the lengths of the paths between the inlet pressure source and the outlets. Assuming that all valves are part of paths from the inlet pressure source to the outlets, this shall take $O(|V| \cdot |V_C|)$ time. However, this is an extremely worse scenario. In practice, $|V_C| << |V|$, and in most cases, $|V_C|$ is a constant, leading to a $O(|V|)$ time complexity in the worst case.

11 Analysis and results

The outlet-specific sub-graph identification helps in confining the testing and diagnosis process to smaller parts of the graph. The test vectors generated for the sub-graph *n*1-*a*-*B*-*d*-*O*1 in Fig. 4*b* to test for block faults are shown in Table 1. The test vectors for the same sub-graph, to test for leak faults are shown in Table 2. Note that the test vectors are able to achieve 100% fault coverage. A 0 corresponding to a valve column indicates that the valve needs to be closed for the test. A 1 indicates that it has to be left deactivated. During a test, the observed responses at the pressure sensors can be compared to the pressure table to come up with the test results. Once the mis-matching responses have been identified, the diagnosis process on the corresponding sub-graphs can be started.

In this paper, we have shown the results of the diagnosis process on three benchmarks, ChIP (Chromatin Immuno Precipitation) [15], PCR (Polymerase Chain Reaction) [25] and SA-I (Synthetic Benchmark). SA-1 is a commonly used benchmark for biochip simulations [25]. The characteristics of these benchmarks are given in Table 3. Table 4 shows the extent of reduction achieved from the sub-graph identification. Here, N denotes the number of possible sub-graphs for the benchmark. $S_{\max}$ is the vertex cardinality of the maximum sized sub-graph. $S_{\min}$ is the vertex cardinality of the minimum sized sub-graph. N_b is the number of sub-graphs that have vertices less than that of the average. S_{avg} is the average number of vertices present in the sub-graph. As can be observed from Table 4, the sizes of the sub-graphs are significantly smaller than the complete graphs.

Table 5 shows the average running times for the proposed fault diagnosis, for the three benchmarks. According to [17], the valve density of the flow-based biochips being manufactured is growing

Table 2 Test vectors for leak faults

	Test vector							
Valve(s) being tested	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*
a	0	1	1	1	1	1	1	1
b, *g*	1	0	1	1	1	1	0	1
b, *h*	1	0	1	1	1	1	1	0
b, *e*	1	0	1	1	0	1	1	1
b, *f*	1	0	1	1	1	0	1	1
c, *e*	1	1	0	1	0	1	1	1
d	1	1	1	0	1	1	1	1

Table 3 Benchmark characteristics

Name	Vertex count	Edge count	Inlets	Outlets
ChIP	68	70	5	10
PCR	100	107	8	1
SA-I	145	155	8	1

Table 4 Minimisation characteristics of sub-graph identification

Name	Vertex count	N	$S_{\max}$	$S_{\min}$	S_{avg}	N_b
ChIP	68	14	29	5	19.14	8
PCR	100	8	53	5	39.125	4
SA-I	145	8	60	23	50.125	4

Table 5 Running time for fault diagnosis

Name	Avg. running time, ms
ChIP	2.42
PCR	3.27
SA-I	3.33

Table 6 Estimated sizes of the syndrome lists (N_{SL})

Valve count	Flow port count	N_{SL}
1000	50	601,667
10,000	100	65,340,000
25,000	250	408,375,000
50,000	500	1,633,500,000

at very high rates. The rates are even higher than the growth of transistor density according to Moore's law in digital IC's. Chips containing thousands of microfluidic valves are expected soon. Table 6 shows the size of the syndrome list (N_{SL}) for these expected sizes of biochips. The number of test vectors has been assumed to be one third of the number of valves on the chip. The syndrome analysis and succeeding hitting-set formulation as described in [15] can become heavily inefficient in these cases. The currently reported algorithm for the diagnosis [15] is of time complexity $O((|E| + |V|)pn)$, where p is the number of flow ports and n is the number of test vectors.

As mentioned before, the currently existing testing technique works by converting the biochip architecture into an AND-OR structure and then generating the test vectors using conventional ATPG tools like TetraMAX (tool from Synopsys). Our proposed approach for testing and diagnosis helps to provide an abstraction from the size of these chips. The test vector generation is very simple and complete fault coverage is easily achieved. The testing and diagnosis can be done in terms of smaller sub-graphs that can be pre-computed and reused in spite of original sizes of these chips. The root cause sets can also be pre-computed. There is also possibility of parallelising these approaches, wherein testing and successively diagnosis may be performed independently on the sub-graphs.

12 Conclusion

This paper proposes a pseudo-exhaustive approach for testing and diagnosis of faults on flow-based microfluidic biochips, that is based on the fact that the biochip architecture can be considered as a set of independent sub-graphs on which the testing and diagnosis may be performed separately. These sub-graphs are often very small and simplify the whole process. With the increasing trend in valve densities of these chips, the conventional testing and diagnosis process can become inefficient and slow, which may lead to inefficient and sub-optimal results. Hence, existing methods will be non-scalable. However, the proposed methodology helps to break down any large architecture into smaller sub-architectures, on which testing and subsequently diagnosis may be performed independently. Thus, the proposed approach is both simpler and scalable. The approach has been verified on both real-life and synthetic benchmarks.

13 References

[1] Kerkhoff, H.G.: 'Testing microelectronic biofluidic systems', *IEEE Design Test Comput.*, 2007, **24**, (1), pp. 72–82

[2] Mark, D., Haeberle, S., Roth, G., *et al.*: 'Microfluidic lab-on-a-chip platforms: requirements, characteristics and applications', *Chem. Soc. Rev.*, 2010, **39**, pp. 1153–1182

[3] Sadrozinski, H.F.-W., Wu, J.: '*Applications of field-programmable gate arrays in scientific research*' (Taylor & Francis, Inc., Bristol, PA, USA, 2010, 1st edn.)

[4] Fidalgo, L.M., Maerkl, S.J.: 'A software-programmable microfluidic device for automated biology', *Lab Chip*, 2011, **11**, pp. 1612–1619

[5] Liu, C., Li, B., Bhattacharya, B.B., *et al.*: 'Testing microfluidic fully programmable valve arrays (FPVAS)'. Design, Automation Test in Europe Conf. Exhibition (DATE), Lausanne, Switzerland, March 2017, pp. 91–96

[6] Lai, G., Lin, C., Ho, T.: 'Pump-aware flow routing algorithm for programmable microfluidic devices'. 2018 Design, Automation Test in Europe Conf. Exhibition (DATE), Dresden, Germany, March 2018, pp. 1405–1410

[7] Grimmer, A., Klepic, B., Ho, T., *et al.*: 'Sound valve-control for programmable microfluidic devices'. 2018 23rd Asia and South Pacific Design Automation Conf. (ASP-DAC), Jeju, Republic of Korea, January 2018, pp. 40–45

[8] Ho, T.: 'Design automation and test for flow-based biochips: past successes and future challenges'. 2018 IEEE Computer Society Annual Symp. on VLSI (ISVLSI), Hong Kong, People's Republic of China, July 2018, pp. 650–654

[9] Hu, K., Ho, T.Y., Chakrabarty, K.: 'Testing of flow-based microfluidic biochips'. 2013 IEEE 31st VLSI Test Symp. (VTS), Berkeley, CA, USA, April 2013, pp. 1–6

[10] Cheung, P., Toda-Peters, K., Shen, A.Q.: 'In situ pressure measurement within deformable rectangular polydimethylsiloxane microfluidic devices', *Biomicrofluidics*, 2012, **6**, (2), pp. 026501–026501-12. 023202BMF[PII]

[11] Hu, K., Ho, T.Y., Chakrabarty, K.: 'Test generation and design-for-testability for flow-based mvlsi microfluidic biochips'. 2014 IEEE 32nd VLSI Test Symp. (VTS), Napa, CA, USA, April 2014, pp. 1–6

[12] Kirkland, T., Mercer, M.R.: ‘Algorithms for automatic test-pattern generation’, *IEEE Design Test Comput.*, 1988, **5**, (3), pp. 43–55

[13] Pop, P., Minhass, W.H., Madsen, J.: ‘*Microfluidic very large scale integration (VLSI): modeling, simulation, testing, compilation and physical synthesis*’ (Springer, Switzerland, 2016)

[14] Abramovici, M., Breuer, M.A., Friedman, A.D.: ‘*Fault modeling*’ (Wiley-IEEE Press, New York, USA, 1990), p. 672

[15] Hu, K., Bhattacharya, B.B., Chakrabarty, K.: ‘Fault diagnosis for leakage and blockage defects in flow-based microfluidic biochips’, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2016, **35**, (7), pp. 1179–1191

[16] Niedermeier, R., Rossmanith, P.: ‘An efficient fixed-parameter algorithm for 3-hitting set’, *J. Discret. Algorithms*, 2003, **1**, (1), pp. 89–102. Combinatorial Algorithms

[17] Eskesen, M.C., Pop, P., Potluri, S.: ‘Architecture synthesis for cost-constrained fault-tolerant flow-based biochips’. 2016 Design, Automation Test in Europe Conf. Exhibition (DATE), Dresden, Germany, March 2016, pp. 618–623

[18] Hassanin, H., Mohammadkhani, A., Jiang, K.: ‘Fabrication of hybrid nanostructured arrays using a pdms/pdms replication process’, *Lab Chip*, 2012, **12**, pp. 4160–4167

[19] Gokulkrishnan, V., Kamakoti, V., Chandrachoodan, N.*, et al.*: ‘A scalable pseudo-exhaustive search for fault diagnosis in microfluidic biochips’. 2017 IEEE Int. Symp. on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Cambridge, UK, October 2017, pp. 1–4

[20] Tutte, W.T.: ‘*Graph theory*’ (Cambridge Mathematical Library. Cambridge University Press, Cambridge, UK, 2001)

[21] Wunderlich, H.J., Hellebrand, S.: ‘The pseudoexhaustive test of sequential circuits’. Test Conf., 1989. Proc. Meeting the Tests of Time., Int., Washington, DC, USA, August 1989, pp. 19–27

[22] Dias, F.J.O.: ‘Fault masking in combinational logic circuits’, *IEEE Trans. Comput.*, 1975, **C-24**, (5), pp. 476–482

[23] Hopcroft, J., Tarjan, R.: ‘Algorithm 447: efficient algorithms for graph manipulation’, *Commun. ACM*, 1973, **16**, (6), pp. 372–378

[24] Italiano, G.F., Nussbaum, Y., Sankowski, P.*, et al.*: ‘Improved algorithms for min cut and max flow in undirected planar graphs’. Proc. of the Forty-third Annual ACM Symp. on Theory of Computing, STOC ‘11, New York, NY, USA, 2011, pp. 313–322

[25] DTU. Biochip simulator. https://sites.google.com/site/biochipsimulator/evaluationtest-cases/