Design Automation for Dilution of a Fluid Using Programmable Microfluidic Device–Based Biochips

ANKUR GUPTA, Computing and Design Automation (CoDA) Laboratory, Dept. of CSE, IIT Roorkee, India
JUINN-DAR HUANG, Dept. of Electronics Engineering, National Chiao Tung University, Taiwan
SHIGERU YAMASHITA, College of Information Science and Engineering, Ritsumeikan University, Japan
SUDIP ROY, Computing and Design Automation (CoDA) Laboratory, Dept. of CSE, IIT Roorkee, India

Microfluidic lab-on-a-chip has emerged as a new technology for implementing biochemical protocols on small-sized portable devices targeting low-cost medical diagnostics. Among various efforts of fabrication of such chips, a relatively new technology is a programmable microfluidic device (PMD) for implementation of flow-based lab-on-a-chip. A PMD chip is suitable for automation due to its symmetric nature. In order to implement a bioprotocol on such a reconfigurable device, it is crucial to automate a sample preparation on-chip as well. In this article, we propose a dilution PMD algorithm (namely *DPMD*) and its architectural mapping scheme (namely generalized architectural mapping algorithm (*GAMA*)) for addressing fluidic cells of such a device to perform dilution of a reagent fluid on-chip. We used an optimization function that first minimizes the number of mixing steps and then reduces the waste generation and further reagent requirement. Simulation results show that the proposed *DPMD* scheme is comparative to the existing state-of-the-art dilution algorithm. The proposed design automation using the architectural mapping scheme reduces the required chip area and, hence, minimizes the valve switching that, in turn, increases the life span of the PMD-chip.

CCS Concepts: • **Hardware → Electronic design automation**; **Emerging interfaces**; *Integrated circuits*;

Additional Key Words and Phrases: Microfluidics, biochip, continuous-flow, programmable microfluidic device

ACM Reference format:
Ankur Gupta, Juinn-Dar Huang, Shigeru Yamashita, and Sudip Roy. 2019. Design Automation for Dilution of a Fluid Using Programmable Microfluidic Device–Based Biochips. *ACM Trans. Des. Autom. Electron. Syst.* 24, 2, Article 21 (February 2019), 24 pages.
https://doi.org/10.1145/3306492

This work by S. Roy was supported partially by the early career research award (grant #: ECR/2016/001921) sponsored by SERB, Govt. of India and the Indo-Japan joint research project (grant #: DST/INT/JSPS/P-256/2017) sponsored by DST, Govt. of India. This work by S. Yamashita was supported partially by the Bilateral Joint Research Project sponsored by JSPS, Govt. of Japan.

Authors' addresses: A. Gupta, Department of Computer Science and Engineering, IIT Roorkee, Roorkee, Uttarakhand, 247667, INDIA; email: ankursynon@gmail.com; J.-D. Huang, Dept. of Electronics Engineering, National Chiao Tung University, TAIWAN; email: jdhuang@mail.nctu.edu.tw; S. Yamashita, College of Information Science and Engineering, Ritsumeikan University, Shiga, JAPAN; email: ger@cs.ritsumei.ac.jp; S. Roy, Computing and Design Automation (CoDA) Laboratory, Dept. of CSE, IIT Roorkee, Roorkee, INDIA; email: sudiproy.fcs@iitr.ac.in.

1084-4309/2019/02-ART21 $15.00
https://doi.org/10.1145/3306492

1 INTRODUCTION

Microfluidic biochip (also called lab-on-a-chip) is an emerging technology for implementing biochemical laboratory protocols (widely known as bioprotocols) on a miniaturized, portable, and integrated system targeting low-cost medical diagnostics. It automates repetitive laboratory tasks and consumes a little amount (i.e., order of nano/pico/femto liter of volume) of fluids. Several biochemical applications such as drug discovery, protein analysis, and point-of-care disease diagnosis have been demonstrated on microfluidic biochips [1–3]. Though the biochip research is still in its infancy, the continuous-flow–based microfluidics (CMF) [4] and digital microfluidics (DMF) [5, 6] are popular lab-on-a-chip architectures. A CMF biochip is composed of microchannels, microvalves, and micropumps. The continuous liquid flow in the CMF biochip is controlled using micropumps and microvalves. The microchannels are used for transportation of fluid. Whereas, in a DMF biochip, fluids are dispensed as minute discrete units, those are handled by controlling voltage applied on electrodes based on the principle of electrowetting-on-dielectric (EWOD). In 2011, Wang et al. [7] proposed an advanced version of DMF biochips, known as micro-electrode dot array–based digital microfluidic biochips (MEDA-DMF). Unlike DMF, MEDA-DMF allows manipulation of variable size droplets and has variability in the movement of droplets.

In 2011, Fidalgo et al. [8] proposed a new and advanced technology-based on CMF, known as programmable microfluidic device (PMD), which is software-programmable to execute bioprotocols without modifying the hardware. These kind of chips (of size $\mathcal{W} \times \mathcal{H}$) allows the mixer size ($w \times h$), where $w \leq \mathcal{W}$, $h \leq \mathcal{H}$, and $w, h, \mathcal{W}, \mathcal{H} \in \mathbb{Z}^+$. A schematic view of the PMD chip layout with 16 cells is shown in Figure 1(a). The architecture contains a set of intersecting channels to form a patterned and symmetric grid of cells controlled by an array of microvalves. In Figure 1(a), the green square box indicates the location of a virtual mixer (of size 2×2) configured with four cells. Each square-shaped cell in PMD has four microvalves (or junction valves) at the four boundaries as shown in Figure 1(b), and these valves can be switched on or off in order to decide the direction of fluid movement. The route of fluid-flow from the input port to the output port can be determined by switching the microvalves coordinated with micropumps. The input and output port is connected to the top-right cell (i.e., $(4, 4)$) and bottom-left cell (i.e., $(1, 1)$), respectively, as shown in Figure 1(a).

Sample preparation (dilution and mixing) is a crucial preprocessing step of any bioprotocol. Dilution is mixing of input fluid with buffer solution in different proportions iteratively to get the desired target concentration factor (CF). It is the ratio of volume of the input fluid to the final volume. Dilution tree is a binary tree with leaf nodes representing input reactant fluid, while internal nodes and roots denote the intermediate and target CF, respectively. Mixing is the general case of dilution where more than two reactant fluids are mixed in some proportion iteratively to get the desired mixing ratio. In order to implement a bioprotocol on PMD, it is crucial to automate a sample preparation as well.

Design automation of the sample preparation over PMD-chip needs automation of fluidic operations on-chip. Fluidic operations performed on PMD chip are the loading and/or washing of a fluid in a PMD cell, transportation of fluid from input port to output port or from one PMD cell to another PMD cell, and mixing of fluids stored in PMD cells. The pictorial representation of on-chip fluidic operations is given in Figure 2. A PMD chip of size 3×3 containing nine PMD cells with cell ID $p1, p2, \ldots, p9$ and 14 junction valves with valve ID $v1, v2, \ldots, v14$ is taken as shown in Figure 2(a). Initially, all microvalves are on (i.e., closed). The corresponding actuation sequence for microvalves is shown below to every operation in Figure 2. The transportation of fluid is handled by creating pressure for selected microvalves using pressure pumps. The initial situation of fluid storage at Time Cycle 1 is given in Figure 2(b) in which microvalves $v7$, $v8$, $v9$, $v11$, $v12$, and $v13$ are off and the fluid needs to be transported to cell $p1$. Therefore, at Time Cycle 2, we close the

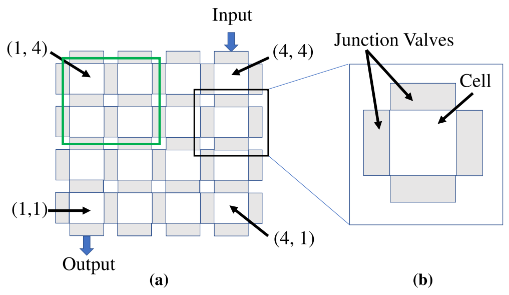


Fig. 1. (a) Schematic view of a PMD layout of size 4×4. (b) Top view of a PMD cell and connected junction valves in four orthogonal directions.

valves $v11$, $v12$, and $v13$ and open the valve $v4$ simultaneously to create flow in the direction of cell $p1$. Eventually, at Time Cycle 3, we close the valve $v4$ and the fluid gets stored in cell $p1$ (shown in red). PMDs are designed in such a way that few of the cells may be incorporated with a detection system for online analysis of results as shown in Figure 2(c) in green. Loading of fluid in a PMD cell takes two time cycles. At first, we need to transport sample fluid from the fluid inlet to the fluid outlet via the PMD cell that needs to be loaded. In Figure 2(d), a flow from the fluid inlet to the fluid outlet is created to load cell $p5$. The flow sequence is $\langle v14, p9, v13, p8, v10, p5, v7, p4, v4, p1, v1\rangle$ at Time Cycle 1. This flow is created by keeping the valves $v14$, $v13$, $v10$, $v7$, $v4$, and $v1$ off and remaining valves on at Time Cycle 1. It can be observed from Figure 2(d) at Time Cycle 1 that several other unintended cells also get loaded by the same sample fluid; therefore, we need another flow (i.e., wash flow) that can wash this sample fluid from these unintended cells. A flow sequence $\langle v14, p9, v13, p8, v12, p7, v9, p4, v4, p1, v1\rangle$ of wash fluid at Time Cycle 2 wash cells other than $p5$ loaded with sample fluid. Now, if this cell $p5$, filled with sample fluid, need to be washed after completion of some specified operation as shown in Figure 2(e), then the same flow sequence for washing of the cell $p5$ can be used but with the wash fluid as shown in Figure 2(e) at Time Cycle 2 in yellow. Mixing of fluids is a crucial part of execution of any bioprotocol. A closed loop is created to mix the fluids loaded in PMD cells. For example, we need to mix two units of buffer and two units of sample stored in cells $p2$, $p6$, and $p3$, $p5$, respectively. To mix, the valves $v8$, $v6$, $v3$, and $v5$ need to keep off and on sequentially and periodically as shown in Figure 2(f) at Time Cycle 2, 3, 4, and 5 until the fluids get mixed properly. The number of mixing cycles required to mix them homogeneously depends upon the chemical properties (such as viscosity, cohesiveness, adhesiveness, and surface tension) of the fluid. After n time cycles of fluid mixing, the mixed fluid is stored in all four cells of PMD as shown in Figure 2(f) at Time Cycle n.

During the last few decades, researchers have done significant work in the sample preparation and design automation of DMF biochips. Several dilution algorithms came into existence with consideration of design constraints for DMF biochips. The dilution algorithms are categorized as single-demand single-target (*SDST*), single-demand multiple-target (*SDMT*), multiple-demand single-target (*MDST*), and multiple-demand multiple-target (*MDMT*) dilution algorithms. The *SDST* dilution algorithms are TwoWayMix (*TWM*) [4] proposed by Thies et al., Dilution and Mixing with Reduced Wastage (*DMRW*) [9] and Improved Dilution and Mixing Algorithm (*IDMA*) [10] proposed by Roy et al., Reactant Minimization Algorithm (*REMIA*) [11] proposed by Huang et al., and Graph-based Optimal Reactant Minimization Algorithm (*GORMA*) [12] proposed by Chiang et al. to name a few. The *SDMT* dilution algorithms are Waste Recycling Algorithm (*WARA*) [13] proposed by Huang et al., Multiple Target Concentration (*MTC*) [14] proposed by Mitra et al., and Generalized Dilution Algorithm (*GDA*) [15] proposed by Roy et al. to name a few. Dinh et al. proposed the network-flow–based optimal sample preparation *MDMT* dilution algorithm, Network-Flow based

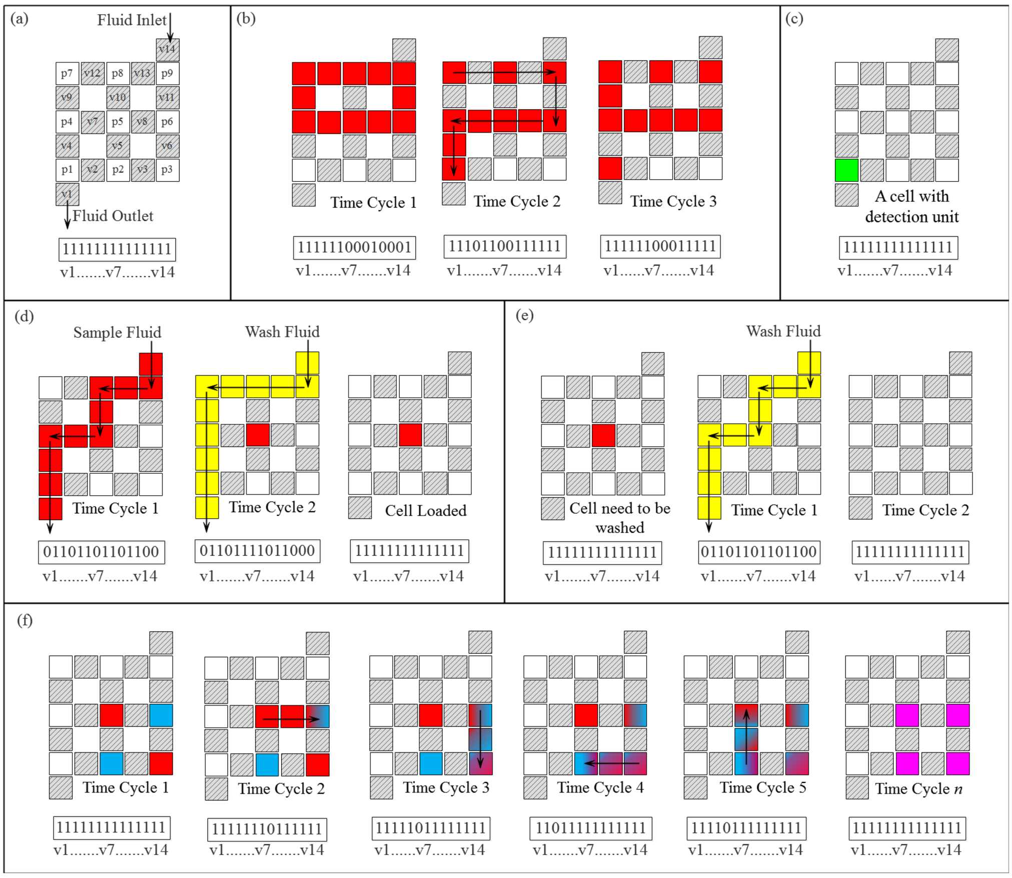


Fig. 2. (a) PMD layout with nine cells and 14 microvalves. (b) Transportation of unit volume fluid from one location to another. (c) Depiction of detection unit at cell location $p1$ shown in green. (d) Loading of fluid in unit cell $p5$. (e) Washing of fluid from unit cell $p5$. (f) Mixing of fluids loaded in four cells $p2$, $p3$, $p5$, and $p6$.

Sample Preparation (*NFSP*) [16]. Several algorithms have been proposed in recent years to automate mixing in DMF biochips, namely *MinMix* [4] proposed by Thies et al., Reagent-Saving Mixing (*RSM*) [17] proposed by Hsieh et al., Mixing Tree with Common Subtrees (*MTCS*) [18] proposed by Kumar et al., Common Dilution Operation Sharing (*CoDOS*) [19] proposed by Liu et al., and Ratio-ed Mixing Algorithm (*RMA*) [20] and Generalized Mixing Algorithm (*GMA*) [21] proposed by Roy et al. to name a few.

Although, several works have been reported in synthesis of CMF biochips, sample preparation got much less attention from researchers. Huang et al. proposed Volume-Oriented Sample Preparation Algorithm (*VOSPA*) [22], which reduces the reactant usage by utilizing the intermediate units and achieves target *CF* by filling up segments one-by-one. Liu et al. introduced the Tree Pruning and Grafting (*TPG*) [11] algorithm, which prunes the $(1 : 1)$ dilution tree created by *TWM* [4] or *REMIA* [23] algorithm and uses grafting on the pruned tree in a bottom-up manner to give the blended tree with minimum reactant usage. Bhattacharjee et al. proposed a state-of-the-art dilution algorithm, Flow-based Sample Preparation Algorithm for dilution (*FloSPA-D*) [24], with the objective of minimizing the number of mixing steps and sample units. The algorithm is designed on the principle of satisfiability theory and ensures optimality of solution considering all possible mixing models. To date, only one state-of-the-art mixing algorithm for CMF biochips, namely Flow-based Sample Preparation Algorithm for mixing (*FloSPA-M*) [24], is proposed. This algorithm also follows the principle of satisfiability theory and ensures optimality of solution but incurs exponential time complexity as well.

The PMD is an advance architecture based on CMF developed by Fidalgo et al. [8]. Though, there has been increasing research concerning PMDs, researchers have done work in synthesis of PMD chips, but the sample preparation and design automation at this architecture has not been started yet. Tseng et al. [25] proposed a reliability-aware synthesis for dynamically mapping the resources over the flow-based microfluidic biochips. Yao et al. [26] proposed a method for flow-control design for CMF biochips. Su et al. [27] has proposed an efficient flow-based routing algorithm targeting PMDs. Liu et al. [28] proposed a testing method for fully programmable valve arrays (FPVAs). Wang et al. [29] proposed a valve switching method based on hamming-distance calculation for control-layer multiplexing in flow-based microfluidic biochips.

Main Contribution: The main contribution of the work is as follows:

— We have proposed a dilution algorithm (*DPMD*) that constructs a dilution tree for target *CF* C_t. This dilution tree is used to generate a dilution sequence ($\mathcal{D}$) in the form of *sbx*-sequences that represents the participation of sample (s), buffer (b), and immediately preceding intermediate *CF*(x) levelwise. This $\mathcal{D}$ is used for placement of resources over the PMD-chip.
— Next, we have proposed a resource placement algorithm, the generalized architectural mapping algorithm (*GAMA*) that places the resources on-chip in a serpentile manner in order to optimally utilize the chip area. Moreover, fluid assignment algorithm (*FAAP*) loads the intended fluid (sample, buffer, reagent, etc.) to the cells specified by the *GAMA*.
— At last, the fluid loading algorithm (*FLAP*) is proposed that loads multiple PMD cells of the same fluid in one load cycle. Thus, reducing the total number of load cycles reduces the valve-switching of PMD and increases the life span of PMD.

The rest of the article is organized as follows. A motivation for the proposed work is given in Section 2. The proposed workflow is presented in Section 3. The mixing models along with its *sbx*-sequence representation is discussed in Section 4. The proposed algorithm for dilution in PMD is presented in Section 5. The resource placement for a given sequencing graph of dilution in PMD is presented in Section 6. The fluid assignment and loading procedures are presented in Section 6.3 and Section 7, respectively. Simulation results and analysis are presented in Section 8. Finally, the article is concluded in Section 9.

2 MOTIVATION

Traditional CMF biochip architectures [4] have fixed and dedicated locations for resources (e.g., ring-like structure for on-chip mixing of fluids and n-ary tree-like channels for storing fluids). The geometry of PMD architecture supports the Manhattan movement of fluid in all four possible directions. Each PMD cell can be used for the mixing as well as storing of fluids. As discussed in Section 1, sample preparation and its design automation (i.e., mapping of sequencing graph to the PMD chip layout) need to be done. The performance metric for sample preparation (*PMSP*) is $\langle T_{ms}, W, S, B\rangle$, where T_{ms}, W, S, and B represent the number of mixing steps, sample, buffer, and waste units, respectively. Another performance metric for chip-design (*PMCD*), i.e., $\langle T, P, A\rangle$, is equally as important as *PMSP* for automating a real-life bio-protocol on-chip, where T, P, and A represent the time, power, and area, respectively. The execution time (T) is the cumulative sum of the loading (T_ℓ), washing (T_w), mixing (T_m), and routing (T_r) time of fluids (i.e., $T = T_\ell + T_w + T_m + T_r$). The mixing time is larger than the loading, washing, and routing time. These time cycles (loading, washing, and routing) collectively participate significant time in comparison to mixing time (i.e., $T_m \simeq T_\ell + T_w + T_r$). In PMD, every load cycle is followed by a wash cycle to remove the unintended filling of cells during the load cycle (say, load-wash pair). Sometimes, a few extra wash cycles (say, T'_w) are required during online bioassay execution. Moreover, routing overhead also incurs to transport fluids from one cell to another cell. These overheads are calculated as given in Equation (1).

$$\Delta = T'_w + T_r \tag{1}$$

Since our solution procedure places resources adjacent to previous ones so it does not require the routing of fluids, that implicitly minimizes the overhead T_r along with the objective of minimizing the *PMCD*, $\langle T, P, A\rangle$. Here, power P, represents the total number of valve actuations done to execute a bioprotocol on-chip. Apparently, minimizing the valve actuations reduces the power consumption and increases the life span of the chip as well.

On the other side, the existing dilution algorithms for the DMF biochip uses a $(1:1)$ or $(k:k)$ mixing model (e.g., *TWM* [4] and K-array Mixer Scheduling (*KMS*) [30]). The only dilution algorithm for the MEDA-DMF biochip utilizes the general mixing model (i.e., Weighted Sample Preparation Method (*WSPM*) [31, 32]). The state-of-the-art dilution algorithm for the CMF biochip (i.e., *FloSPA-D* [24]) exploits all possible mixing models but is computationally exponential in nature. In either case, no mapping algorithm exists to map the sequencing graph in PMD. We have proposed the first architectural mapping algorithm to map the sequencing graph in PMD considering both *PMSP* and *PMCD*. Apparently, it is not always necessary to minimize the sample requirement because few of them are costly but not all. Moreover, if the execution steps of a bioprotocol are lengthy, then its sequencing graph has a large number of mixing steps that eventually requires larger chip area. In this case, implicitly, the generated waste is higher that would also need to be minimized along with chip area for a smooth and error-free (because of cross-contamination) operation. However, there is always a tradeoff between a few pairs of parameters. For example, $S + B = W + C_t$ always holds, which maintains an equilibrium between the fluid input and generated output. Therefore, considering both metrices, we are presenting a motivational example with pictorial representation in Figure 3. Here, mixer of size (2×2) is considered as it takes a minimum number of reactant fluid units and generates minimum waste. The proof is given in Section 5.3.

Two algorithms, *TWM* developed under the design constraints of DMF biochips and *FloSPA-D* designed for CMF biochips, are considered along with our proposed algorithm *DPMD*. The mixing trees for *FloSPA-D*, *TWM*, and *DPMD* and their design automation in PMD chip are shown in Figure 3(a), (b), and (c), respectively. Here, nodes with sample, buffer, and target *CF* are represented in red, blue, and green, respectively. All nodes other than these three colors are intermediate *CF*s. The magenta circles show the number of waste units in that mixing step. A placement cycle P_i loads the reactant fluids for mixing step m_i. Mixing cycle M_i mixes the fluids loaded by P_i, though both P_i and M_i may need multiple time cycles to load and mix the input fluids. The virtual mixers overlap for consecutive mixing cycles M_{i-1} and M_i as m_i requires few units of immediately preceding intermediate *CF* (say, CF_{i-1}). In Figure 3(a), P_1 loads three sample units and one buffer unit in four different PMD cells for m_1. Then, M_1 mixes these fluids by making a closed loop by microvalve actuations. Next, since m_2 considers one unit of CF_1 from m_1, one sample unit, and two buffer units, therefore, P_2 loads only two buffer units and one sample unit at the intended cells. Then, virtual mixer is created at M_2 as shown in Figure 3(a) at alternate placement-and-mixing cycles. Then, m_3 takes three units of CF_2 from m_2 and mixes it with one buffer unit. In order to create a virtual mixer that can consider three units of CF_2, it can be done by placing the mixer for M_3 at the same cell location as M_2. Therefore, it incurs an extra wash cycle to unload one cell from M_2 and load it with thebuffer as depicted in Figure 3(a) at P_3. Now, M_3 mixes these fluids. Afterward, m_4 takes three units of CF_1 from m_1 and one unit of CF_3 from m_3. Apparently, explicit routing of three units of CF_1 from M_1 (i.e., from cell locations $\langle(3,1),(4,1),(4,2)\rangle$ to cell locations $\langle(1,1),(2,1),(1,2)\rangle$) is

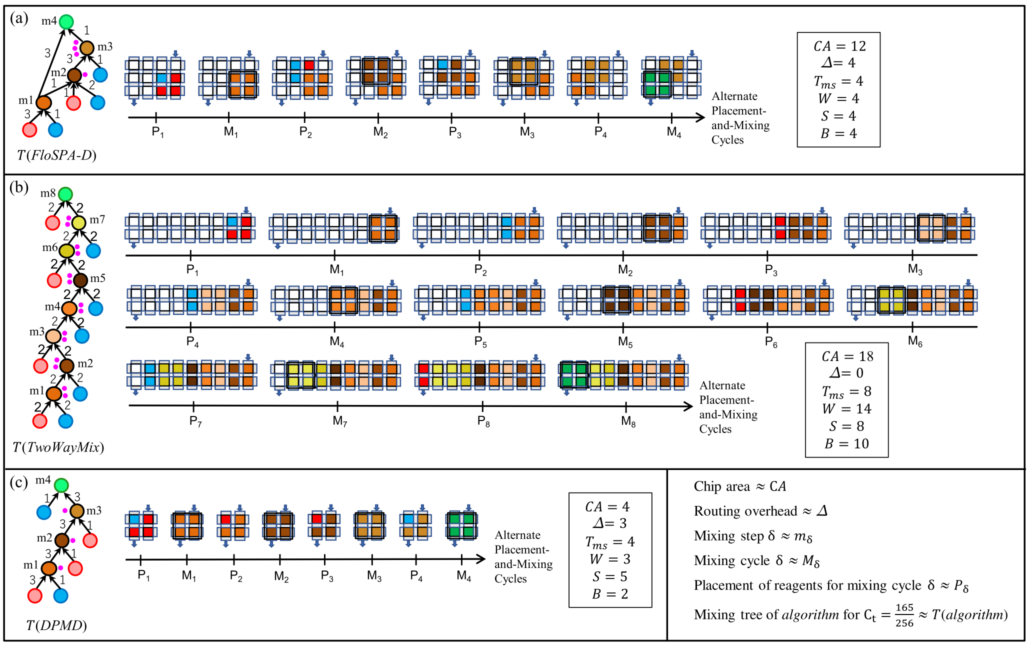


Fig. 3. Comparison of sample preparation and chip-design performance metric $\langle T_{ms}, W, S, B\rangle$ and $\langle T, P, A\rangle$, respectively, for the dilution graph of algorithm (a) *FloSPA-D*, (b) *TWM*, and (c) *DPMD*, for target $CF = \frac{165}{256}$. Here, mixing cycle M_i and placement cycle P_i in alternate placement-and-mixing cycles has one-to-one correspondence with mixing step m_i and m_{i-1} of the dilution graph, respectively.

needed. Eventually, the output is generated in M_4 at cells $\langle(1,1),(2,1),(1,2),(2,2)\rangle$ (following the cell coordinate system explained in Figure 1). The routing overhead is 4 as given by Equation (1). Similarly, *TWM* and *DPMD* design automation is presented in Figure 3(b) and (c), respectively. Here, it can be observed that *TWM* does not incur any routing overhead (i.e., zero) but the required chip area (i.e., 18) is largest because of a large number of mixing steps in the sequencing graph. On the other side, *FloSPA-D* incurs routing overhead (i.e., 4) and a larger chip area (i.e., 12), though *DPMD* needs routing overhead (i.e., 3) and, comparatively, quite a smaller chip area (i.e., 4). However, *PMSP* $\langle T_{ms}, W, S, B\rangle$ for *FloSPA-D*, *TWM*, and *DPMD* is $\langle 4,4,4,4\rangle$, $\langle 8,14,8,10\rangle$, and $\langle 4,3,5,2\rangle$, respectively. It is clear that the metric for *FloSPA-D* and *DPMD* is comparative.

In general, the required chip area depends upon the number of mixing steps in the dilution tree. As it is clear from Figure 3(b), *TwoWayMix* requires eight mixing steps; hence, eight virtual mixers need to be placed on-chip in comparison to *DPMD* that requires four mixing steps, and hence, four virtual mixers need to be placed on-chip. Lower the mixing steps, lower the number of virtual mixers, and that, in turn, lowers the valve switching. Though, another aspect is that *DPMD* and *FloSPA-D* both require an equal number of mixing steps for a given *CF*, and hence, an equal number of virtual mixers and chip area is required; but *FloSPA-D* may exploit any of the previously generated intermediate *CF*s that incur routing overhead. Thus, it requires more chip area in comparison to *DPMD*.

3 THE PROPOSED WORKFLOW

The proposed workflow is presented in Figure 4. The inputs to the *DPMD* algorithm are reactant *CF*s as C_ℓ and C_h, and the target *CF* C_t needs to be achieved with accuracy level q and optimization

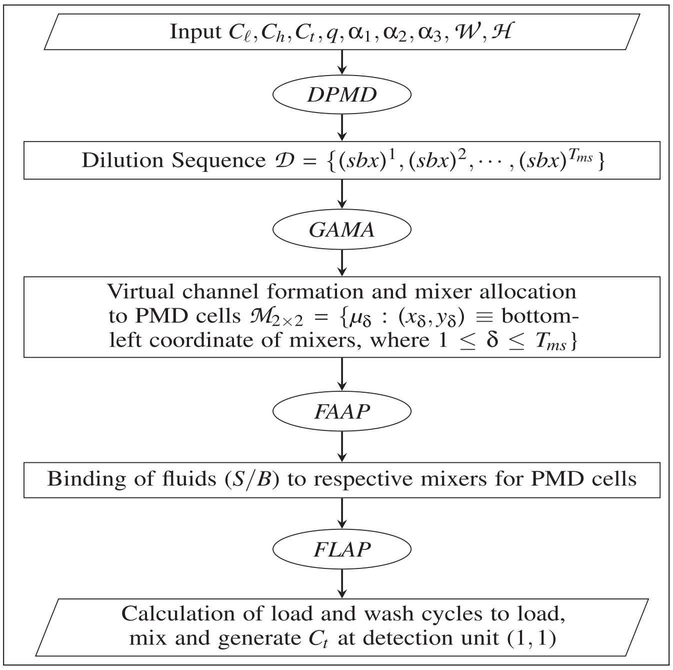

Input $C_\ell, C_h, C_t, q, \alpha_1, \alpha_2, \alpha_3, \mathcal{W}, \mathcal{H}$
DPMD
Dilution Sequence $\mathcal{D} = \{(sbx)^1, (sbx)^2, \cdots, (sbx)^{T_{ms}}\}$
GAMA
Virtual channel formation and mixer allocation to PMD cells $\mathcal{M}_{2\times 2} = \{\mu_\delta : (x_\delta, y_\delta) \equiv$ bottom-left coordinate of mixers, where $1 \le \delta \le T_{ms}\}$
FAAP
Binding of fluids (S/B) to respective mixers for PMD cells
FLAP
Calculation of load and wash cycles to load, mix and generate C_t at detection unit $(1,1)$

Fig. 4. Flowchart for overview of proposed workflow.

weightage for different statistical parameters $\alpha_1, \alpha_2, \alpha_3$. *DPMD* generates the dilution tree along with the dilution sequence $\mathcal{D}$.

The dilution sequence is the *sbx*-sequence that gives information about the proportion of sample, buffer, and immediately preceding intermediate CFs in the generated intermediate *CF* or target *CF*. The dilution sequence is derived from the dilution tree. In a similar way, this dilution sequence is the input for resource placement algorithm *GAMA* and gives the location coordinates for all virtual mixers corresponding to each mixing step in the dilution tree. Next, these coordinates are input to fluid assignment algorithm FAAP that gives a way to assign fluids to the mixers. This assignment actually requires the loading of corresponding fluids (sample, buffer, reagent, etc.) in the PMD cells. At last, the fluid loading algorithm *FLAP* takes the fluid assignment to PMD cells and chip size $\mathcal{W} \times \mathcal{H}$ as input and calculates the load and wash cycles required to load the intended fluid in the cells in order to execute the bioprotocol on-chip. Though these algorithms seem interactive as one's output goes as input to the next algorithm, all are independent and input to any of the algorithms can also be taken from other methods (existing or newly developed).

4 MIXING MODELS

A general mixing model (GMM) is $(k_1 : k_2 : k_3 : \cdots : k_n)$, where $k_1, k_2, k_3, \ldots, k_n \in \mathbb{Z}^+$. The resultant *CF* using a general mixing model is calculated as $C_t = \frac{k_1.c_1 + k_2.c_2 + \cdots + k_n.c_n}{k_1 + k_2 + \cdots + k_n}$, where $c_1, c_2, \ldots, c_n$ are *CF*s of n different input fluids. In case of PMD layout, an array mixer of size $w \times h$ can be configured to implement GMM $(k_1 : k_2 : k_3 : \cdots : k_n)$, where $2(w+h) - 4 = k_1 + k_2 + \cdots + k_n$. For a minimum size mixer, $w = h = 2$; hence, $k_1 + k_2 + \cdots + k_n = 4$, where $n \le 4$. Therefore, the constrained representation of GMM $(k_1 : k_2 : \cdots : k_n)$, implementing minimum size mixer for PMD, is $(k_1 : k_2 : k_3 : k_4)$. The unique mixing models (UMMs) based on $(k_1 : k_2 : k_3 : k_4)$ are (1:1:1:1), (1:1:2), (1:2:1), (2:1:1), (1:3), (3:1), and (2:2). In case of dilution using UMMs, we have at most four different input fluids such as the sample (labeled as 's') with $CF = C_h$, buffer (labeled as 'b') with $CF = C_\ell$, and diluted sample (labeled as 'x') with $CF = C_r$. Since UMM (1:1:1:1) requires four different input fluids (e.g., one sample unit, one buffer unit, and two different intermediate *CF*s) that, in turn, incurs routing overhead as two different intermediate *CF*s cannot be generated in one mixing step. To avoid routing overhead, we consider only the remaining six UMMs. These UMMs can be represented as either (k:ℓ:m) or (k:ℓ). We can extend the (k:ℓ) mixing model as (k:ℓ:0). An assignment $(sbx) \leftarrow \langle k\ell m\rangle$, where $\langle k\ell m\rangle$ is the string obtained from (k:ℓ:m) mixing model, denotes a mixing operation with k units of 's', ℓ units of 'b' and m units of 'x' (i.e., $s = k$, $b = \ell$, $x = m$). There are 12 specific mixing models that can be derived from the (k:ℓ:m) mixing model, where $0 \leq k, \ell, m \leq 3$, such that $k + \ell + m = 4$. These models are named as derived mixing models (DMMs). These DMMs are d_1:$\langle 220\rangle$, d_2:$\langle 130\rangle$, d_3:$\langle 310\rangle$, d_4:$\langle 301\rangle$, d_5:$\langle 031\rangle$, d_6:$\langle 121\rangle$, d_7:$\langle 211\rangle$, d_8:$\langle 022\rangle$, d_9:$\langle 202\rangle$, d_{10}:$\langle 112\rangle$, d_{11}:$\langle 103\rangle$, and d_{12}:$\langle 013\rangle$. Out of these 12 DMMs, d_1, d_2, and d_3 generate first intermediate *CF* (CF_1) of a dilution tree since no intermediate *CF* is available initially. The remaining nine DMMs, d_4 to d_{12}, are used at upper levels of the dilution tree to generate other intermediate and target *CF*s (CF_i, where $1 < i \leq T_{ms}$).

A dilution tree is a sequence of DMMs taken from the sequence $(d_1, d_2, \ldots, d_{12})$ or the sequence of (sbx) with certain constraints (like d_1, d_2, d_3 can only arise at level-1 and d_4 to d_{12} can be at any level other than one). This sequence of (sbx) from level one to T_{ms} is denoted as dilution sequence $\mathcal{D}$. The dilution sequence $\mathcal{D}$ is represented as $\{(sbx)^1, (sbx)^2, \ldots, (sbx)^i, \ldots, (sbx)^{T_{ms}}\}$, where $(sbx)^1 = \{\langle k\ell 0\rangle \,|\, k + \ell = 4, k \neq 0, \ell \neq 0\}$ and $(sbx)^i = \{\langle k\ell m\rangle \,|\, i \neq 1, k + \ell + m = 4, 0 < (k + \ell) < 4\}$. In general, a dilution graph may use a general mixing model $(k_1 : k_2 : k_3 : \cdots : k_n)$ at each mixing step. Since the number of fluids mixed at any level in the mixing tree can be atmost $(q + 1)$ (i.e., one sample fluid, one buffer fluid, and atmost $(q - 1)$ preceding intermediate fluids, if q is the accuracy), then, $n \leq q + 1$. The sequence of (sbx) can also be generalized for these dilution graphs as $(\sigma\tau\chi) \leftarrow \langle k_1k_2k_3 \cdots k_n\rangle$, where $\langle k_1k_2k_3 \cdots k_n\rangle$ is the string obtained from $(k_1 : k_2 : k_3 : \cdots : k_n)$ mixing model and $n = q + 1$. Here, k_1 denotes the units of the sample (labeled as "σ") used, k_2 denotes the units of buffer (labeled as "τ") used and $k_3, k_4, \ldots, k_n$ denote the portion of preceding intermediate *CF*s from level-1 to the immediately preceding level (i.e., $\sigma = k_1$, $\tau = \ell_1$, $\chi = k_3k_4 \cdots k_n$). The dilution sequence π is represented as $\{(\sigma\tau\chi)^1, (\sigma\tau\chi)^2, \ldots, (\sigma\tau\chi)^i, \ldots, (\sigma\tau\chi)^{T_{ms}}\}$, where $(\sigma\tau\chi)^1 = \{\langle k_1k_200 \cdots 0\rangle \,|\, k_1 + k_2 = k_1 + \cdots + k_n, k_1 \neq 0, k_2 \neq 0\}$, $(\sigma\tau\chi)^2 = \{\langle k_1k_2 \cdots k_n\rangle \,|\, k_1 + k_2 + k_3 = k_1 + \cdots + k_n, k_3 \neq 0, k_1 \neq 0 || k_2 \neq 0\}$ and $(\sigma\tau\chi)^i = \{\langle k_1k_2 \cdots k_n\rangle \,|\, i \neq 1, i \neq 2, 0 \leq (k_1 + k_2) < k_1 + \cdots + k_n, 0 < (k_3 + \cdots + k_n) \leq k_1 + \cdots + k_n\}$.

5 PROPOSED DILUTION SCHEME FOR *PMD*

This section presents the proposed routingless dilution algorithm *DPMD*.

5.1 Problem Formulation

The dilution problem targeting PMD architecture is formulated as follows:
Inputs: Given two fluids with *CF* C_ℓ and C_h, the target *CF* C_t with accuracy level q, where $0 \leq C_\ell < C_t < C_h \leq 4^q$, optimization parameters (α_1, α_2, and α_3), and DMMs.
Outputs: Dilution sequence $\mathcal{D}$ to achieve C_t and performance quadruple $\langle T_{ms}, W, S, B\rangle$, where T_{ms} represents the number of mixing steps, and W, S, and B represent the waste, sample, and buffer units, respectively.
Constraints: (1) $T_{ms} \leq q$. (2) $S + B = W + 4$.
Objectives: (1) Minimize T_{ms}. (2) Minimize W.

The proposed dilution scheme *DPMD* takes two input *CF*s C_ℓ and C_h and generates target *CF* C_t with accuracy level q. *DPMD* utilizes the derived mixing models as discussed in Section 4. *DPMD* explores all possible intermediate *CF*s based on the DMMs applicable at each level. In order to prioritize the parameters, *DPMD* assigns weightage to the specified parameters as α_1, α_2, and α_3. On the basis of these mentioned inputs, *DPMD* generates the dilution sequence $\mathcal{D}$, which is the platform independent format for resource placement in PMD. Since the accuracy level is q, the number of mixing steps required to achieve the target *CF* C_t is less than or equal to q, i.e., $T_{ms} \leq q$. In the dilution process, the fluid is preserved, i.e., the input number of fluid units is equal to the output number of fluid units. The input fluids are sample (S) and buffer (B), and the output fluid might be intermediate *CF* or target *CF*. Intermediate fluid that is not used in subsequent mixing steps may go wasted and be considered as waste (W). Therefore, for a dilution tree, $S + B = W + \#C_t$, where $\#C_t$ represents the number of target *CF* units. Since four units of target *CF* would be generated because the mixer size is 2×2. The modified equation for fluid preservation is $S + B = W + 4$. The prime goal of *DPMD* is to minimize the number of mixing steps T_{ms} since it leads to the minimization of chip area required for the design automation of PMD. The secondary goal is to minimize the waste. The number of target *CF* units are fixed (i.e., 4) because the mixer size is fixed (i.e., 2×2); therefore, minimizing the waste will eventually minimize the fluid input requirement (i.e., sample as well as buffer).

5.2 Dilution on Programmable Microfluidic Device

We present an algorithm for Dilution on PMD-chip (*DPMD*) based on the dynamic programming paradigm that intends to produce target *CF* without incuring on-chip routing overhead (T_r) for intermediate *CF*s. The pseudocode is shown as Algorithm 1. The algorithm, *DPMD*, takes two *CF*s as input (i.e., $C_\ell = \frac{a}{4^q}$ and $C_h = \frac{z}{4^q}$) and generates the dilution sequence $\mathcal{D}$ as output to achieve the target *CF* (i.e., $C_t = \frac{c}{4^q}$), where $0 \leq a < c < z \leq 4^q$. Depending upon the DMM used, the unit(s) of C_ℓ, C_h, and $CF_{\delta-1}$ (intermediate *CF* generated at immediately preceding level ($\delta - 1$)) are mixed to obtain four volume units of intermediate or target *CF* (CF_δ, where $\delta \leq q$). The algorithm, *DPMD*, follows a bottom-up approach while exploring the search space and generating the intermediate *CF*s to build the dilution tree. *DPMD* explores all possible *CF*s at each level depending upon the DMM applicable (e.g., d_1, d_2, d_3 at level-1 and d_4 to d_{12} at level-δ, where $1 < \delta \leq q$). The levels will be constructed iteratively until the accuracy (q) reaches.

5.2.1 *Optimization Function "f".*

On-chip implementation of a real-life bioprotocol concerns with different parameters (e.g., input reactant fluids, waste generation, mixing steps, and chip area) depending upon the application-specific implementation. An application may demand bioprotocol execution in minimum possible steps or concerns with minimum input reactant fluid used (e.g., an infant's blood [33] or residues collected from a crime scene [34]) or may be bound to exploit the minimum chip area (e.g., on-chip space is a bottleneck to develop a handheld device). Concerning these parameters, the optimization function can be defined as $f^g = \alpha_1.P_1 + \alpha_2.P_2 + \cdots + \alpha_n.P_n$, where n is the number of parameters needed to be optimized, α_i is the weightage assigned to the parameter P_i, and $\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1$.

A concentration (C_i) may appear multiple times at different levels in an explored search space (*ESS*). Since we are concerned to optimize mixing steps, waste generated, reagent used, and chip area required for PMD layout, the optimization function (f_i^δ) calculates the optimization value of *CF* y_i^δ with node ID i at level δ in *ESS* as presented in Equation (2), where δ, W_i^δ, and S_i^δ represents the current level of *ESS*, waste generated, and sample units used, respectively.

$$f_i^\delta = \alpha_1.\delta + \alpha_2.W_i^\delta + \alpha_3.S_i^\delta \tag{2}$$

It can be noted that f_i^δ does not include chip area since optimizing mixing steps minimizes the chip area implicitly. The node η_i^δ of *CF* C_i^δ at level-δ with parametric values $\langle \delta, S_i^\delta, B_i^\delta, W_i^\delta \rangle$ is explored from the node $\eta_j^{\delta-1}$ at level ($\delta - 1$) with *CF* $C_j^{\delta-1}$ and parametric values $\langle \delta\text{-}1, S_j^{\delta-1}, B_j^{\delta-1}, W_j^{\delta-1} \rangle$.

ALGORITHM 1: *DPMD*(C_ℓ, C_h, C_t, q)

```
begin
    Values of $\alpha_1, \alpha_2, \alpha_3$ are set based on optimization criteria;
    $a = C_\ell.4^q; z = C_h.4^q; c = C_t.4^q$;
    Create a start node; $\delta = 1$;
    for $(p_n | n = 1, 2, 3)$ do
        $(sbx)^1_n \leftarrow \langle k\ell 0\rangle$ for $p_n$;
        $C^1_n = \frac{s.z + b.a}{4}$;
        Create a child node for $CF = C^1_n$;
    $\delta = \delta + 1$;
    while $(\delta \leq q)$ do
        $j = 1; i = 1$;
        for (each node j created at level $(\delta - 1)$) do
            for $(p_n | n = 4, 5, \ldots, 12)$ do
                $(sbx)^\delta_i \leftarrow \langle k\ell m\rangle$ for $p_n$;
                $C^\delta_i = \frac{s.z + b.a + x.C^{\delta-1}_j}{4}$;
                Calculate quadruple $\{W^\delta_i, S^\delta_i, B^\delta_i, \delta\}$ for $C^\delta_i$;
                Calculate $f^\delta_i = \alpha_1.\delta + \alpha_2.W^\delta_i + \alpha_3.S^\delta_i$;
                for (all created nodes $C^{\delta'}_{i'} | 1 \leq \delta' \leq \delta$) do
                    if $(C^\delta_i == C^{\delta'}_{i'})$ then
                        if $(f^\delta_i < f^{\delta'}_{i'})$ then Create a child node for $CF = C^\delta_i$; Delete all previously
                        created nodes with $CF$ value $C^\delta_i$ and their explored subtrees;
                        if $(f^\delta_i == f^{\delta'}_{i'})$ then Create a child node for $CF = C^\delta_i$;
                $i = i + 1$;
            $j = j + 1$;
        $\delta = \delta + 1$;
    Generate solution set $\Sigma$ = {node(s) with $C^\delta_i = c$ of minimum $f^\delta_i$};
    for (each node in $\Sigma$) do
        Calculate $T_{ms} = |\mathcal{D}|$;
        Apply tie-break by considering $T_{ms}, W, S$ in the respective order of preference and obtain
        optimal node opt;
    Backtrack from node opt till start node to obtain the sequence of $(sbx)$ and reverse the sequence to
    get $\mathcal{D}$;
    return $\mathcal{D} = \{(sbx)^1, (sbx)^2, \ldots, (sbx)^{T_{ms}}\}$;
```

Assume that the (sbx)-sequence at level δ is $\langle k\ell m\rangle$. The parametric values for node η^δ_i are calculated as given in Equations (3–6).

$$C^\delta_i = \begin{cases} k.4^{q-1} & \text{if } \delta = 1 \\ k.4^{q-1} + \frac{m}{4}.C^{\delta-1}_j & \text{if } \delta > 1 \end{cases} \tag{3}$$

$$S^\delta_i = \begin{cases} k & \text{if } \delta = 1 \\ S^{\delta-1}_j + k & \text{if } \delta > 1 \end{cases} \tag{4}$$

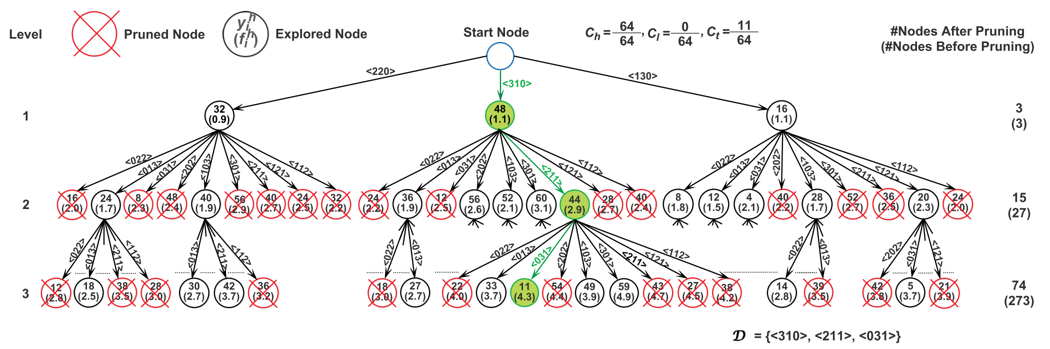


Fig. 5. Dilution sequence $\mathcal{D}$ obtained after search space exploration with pruning by *DPMD* for $C_t = \frac{11}{64}$.

$$B_i^\delta = \begin{cases} \ell & \text{if } \delta = 1 \\ B_j^{\delta-1} + \ell & \text{if } \delta > 1 \end{cases} \tag{5}$$

$$W_i^\delta = \begin{cases} 0 & \text{if } \delta = 1 \\ W_j^{\delta-1} + (4-m) & \text{if } \delta > 1 \end{cases} \tag{6}$$

5.2.2 *Pruning.* *DPMD* explores all possible *CF*s at each level by applying DMMs. The number of intermediate *CF*s generated at level-δ is $3^{2\delta-1}$, where $1 \le \delta \le q$. The space as well as time complexity of the algorithm is $O(9^q)$, where q is the depth of *ESS*. Since the complexity is exponential in nature, pruning of subtrees helps in restricting the *ESS* and reduces the complexity. Pruning is done by calculating f_i^δ-value for every node of *ESS*. If multiple nodes have the same *CF*, then a node with minimum f_i^δ-value is explored at the next depth. Nodes having the same *CF* and f_i^δ-value are considered in respective order of mixing steps, waste units generated, and sample units for breaking the tie.

An Illustrative Example: The *ESS* for target $CF = \frac{11}{64}$ is explored in Figure 5. At first, DMMs $\{d_1, d_2, d_3\} \equiv \{\langle 220\rangle, \langle 310\rangle, \langle 130\rangle\}$ are considered with sample and buffer, i.e., $C_h = \frac{64}{64}$ and $C_\ell = \frac{0}{64}$, respectively. The node set (η^1) at level-1 of *ESS* contains three intermediate *CF*s, corresponding to each DMM, i.e., $\{\eta_1^1, \eta_2^1, \eta_3^1\} \equiv \{\frac{32}{64}, \frac{48}{64}, \frac{16}{64}\}$. Now, from level-2 onward, the remaining nine DMMs $\{d_4, d_5, d_6, d_7, d_8, d_9, d_{10}, d_{11}, d_{12}\} \equiv \{\langle 301\rangle, \langle 031\rangle, \langle 121\rangle, \langle 211\rangle, \langle 022\rangle, \langle 202\rangle, \langle 112\rangle, \langle 103\rangle, \langle 013\rangle\}$ are considered with each node of (η^1) and add these $3 \times 9 = 27$ new intermediate *CF*s to η^2. Similarly, at level-3, DMMs are applied and add $27 \times 9 = 243$ new intermediate *CF*s to η^3. In *ESS* for target $CF = \frac{11}{64}$, explored nodes are $3 + 27 + 243 = 273$. For higher accuracies ($q \ge 5$), η is extremely large and the space complexity is high. The optimization function considered is $f = 0.5 \times \delta + 0.4 \times W + 0.1 \times S$. The pruned nodes are shown with a red cross mark over the red circle. Here, it can be observed that the target *CF* $C_t = \frac{11}{64}$ is found at level-3. The path from start node to target node is shown in green and the corresponding dilution sequence is $\mathcal{D} = \{\langle 310\rangle, \langle 211\rangle, \langle 031\rangle\}$. The dilution tree is built by scanning the dilution sequence $\mathcal{D}$ as shown in Figure 6. The *PMSP* for C_t is $\langle 3, 6, 5, 5\rangle$.

5.3 Theoretical Results

THEOREM 5.1. *For a given PMD chip of size $\mathcal{W} \times \mathcal{H}$, the mixer of size 2×2 is more efficient in comparison to other mixer variants of size $w \times h$ (where $w, h \ge 2$ and $w + h > 4$) as it requires a minimum amount of input fluids and generates a minimum amount of waste fluid in sample preparation.*

PROOF. A mixer $(\mathcal{M})$ of size 2×2 has four PMD cells to be filled with input fluids. So, the number (F) of PMD cells to be filled with input fluids before a mixing step is four, i.e., $F = 4$. Whereas a

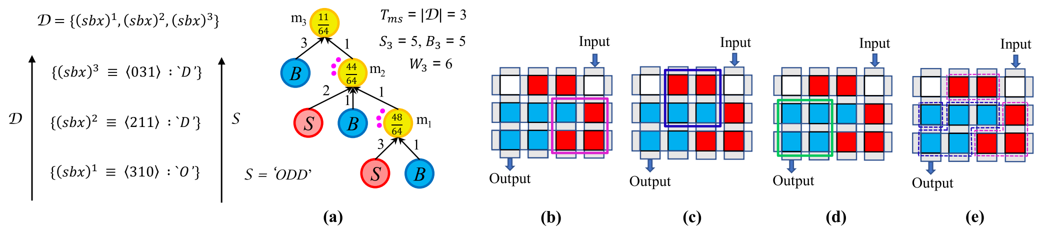

$\mathcal{D} = \{(sbx)^1, (sbx)^2, (sbx)^3\}$
$\{(sbx)^3 \equiv \langle 031 \rangle : `D'\}$
$\{(sbx)^2 \equiv \langle 211 \rangle : `D'\}$
$\{(sbx)^1 \equiv \langle 310 \rangle : `O'\}$
$\mathcal{D}$
$\mathcal{S}$
S = 'ODD'
$T_{ms} = |\mathcal{D}| = 3$
$S_3 = 5, B_3 = 5$
$W_3 = 6$
m3
$\frac{11}{64}$
m2
$\frac{44}{64}$
m1
$\frac{48}{64}$
B
S
Input
Output
(a)
(b)
(c)
(d)
(e)

Fig. 6. (a) Dilution tree for target $CF\ C_t = \frac{11}{64}$ and its scanning sequence $\mathcal{S}$ derived from $\mathcal{D}$ obtained in Figure 5. (b) Placement of mixer $\mathcal{M}_1$ for mapping of mixing step m_1 on-chip ($\mathcal{D}_1 = (sbx)^1 \equiv \langle 310 \rangle, \mathcal{S}_1 =$ 'O'), where the red and blue cells represent the sample and buffer, respectively, and the mixer is shown in magenta. (c) Placement of mixer $\mathcal{M}_2$ (shown in blue) for mapping of mixing step m_2 on-chip ($\mathcal{D}_2 = (sbx)^2 \equiv \langle 211 \rangle$, $\mathcal{S}_2 =$ 'D'). (d) Placement of mixer $\mathcal{M}_3$ (shown in green) for mapping of the mixing step m_3 on-chip ($\mathcal{D}_3 = (sbx)^3 \equiv \langle 031 \rangle$, $\mathcal{S}_3 =$ 'D'). (e) Grouping of PMD cells with the same type of fluid that can be loaded in one cycle. Four cycles are required to load all cells (two cycles for loading the sample as shown in magenta and two cycles for the buffer as shown in blue).

mixer variant ($\mathcal{M}'$) of size $w \times h$ (where $w, h \geq 2$ and $w + h > 4$), in general, has $2w + 2(h - 2)$, i.e., $2(w + h) - 4$ PMD cells (considering only the boundary cells of a $w \times h$ grid) as the closed loop. These $2(w + h) - 4$ cells are to be filled with input fluids before a mixing step, i.e., $F' = 2(w + h) - 4$. Hence, in case of any mixer variant ($\mathcal{M}'$) with $w + h > 4$, $F' > 4$ always.

In a sequence of mixing steps, the total input volume is preserved at the output. Hence, the sum of amounts (volumetric units) of output and waste is equal to the sum of amounts (volumetric units) of all input fluids taken in the sequence. In case of dilution, which follows a sequence of mixing steps, $S + B = W + \#C_t$, where $\#C_t$ is the volumetric units of fluid with target $CF = C_t$. In order to produce an equal amount of fluid with target $CF = C_t$, the amount of waste fluid generated by a 2×2 mixer is always less than that by any other mixer variant $\mathcal{M}'$. The similar argument is independent of the sequence of mixing steps, and the same is true for any bioprotocol other than this simple case of dilution.

Hence, a mixer of size 2×2 can generate a target $CF\ C_t$ with minimum possible usage of input fluids and can generate a minimum possible waste fluid in comparison to other mixer variants. □

6 ARCHITECTURAL MAPPING OF SKEWED DILUTION TREES ON PMD

This section is divided into two subsections: one is *GAMA* and another is *FAAP*. The mapping algorithm, *GAMA*, allocates mixers on-chip to map the mixing steps of the skewed dilution tree. The algorithm, *FAAP*, assigns the intended fluid in the allocated mixers for mixing. The mixing tree is built by using different mixing models at each level. Every mixing model is implemented differently on PMD as it may contain a variable amount of intermediate *CF* from immediately preceding level of mixing tree. PMD cells in allocated virtual mixers for consecutive levels are either overlapped and/or incur routing overhead. Here, we derive a scanning sequence based on mixing models used at each level that represents relative positioning of mixers on-chip.

6.1 Scanning Sequence

A *scanning sequence* is derived to map the mixing steps of a dilution tree on PMD to achieve the target *CF* based on the amount of intermediate *CF*s used from preceding levels. The scanning sequence $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_i, \ldots, \mathcal{S}_{T_{ms}}\}$ for GMM $(k_1 : k_2 : \cdots : k_n)$ is derived from the dilution

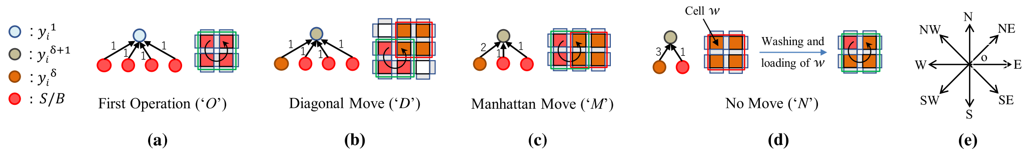


Fig. 7. Mixer placement for dilution operation (a) '*O*', (b) '*D*', (c) '*M*', and (d) '*N*', and its architectural mapping on a PMD layout. (e) The representation of direction for diagonal movements in PMD cells, where 'NW', 'SW', 'NE', and 'SE' represent diagonal moves. 'N', 'S', 'E', and 'W' represent the Manhattan moves on PMD layout.

sequence $\pi = \{(\sigma\tau\chi)^1, (\sigma\tau\chi)^2, \ldots, (\sigma\tau\chi)^i, \ldots, (\sigma\tau\chi)^{T_{ms}}\}$ as depicted in Equation (7).

$$\mathcal{S}_\delta = \begin{cases} \text{`}O\text{'} & \text{if } k_1 + k_2 = k_1 + \cdots + k_n, k_1 \neq 0, k_2 \neq 0 \\ \text{`}D\text{'} & \text{if } k_{\delta+1} = 1 \\ \text{`}M\text{'} & \text{if } k_{\delta+1} = 2 \\ \text{`}N\text{'} & \text{if } k_{\delta+1} = 3 \end{cases} \tag{7}$$

Since *DPMD* utilizes DMMs, then the scanning sequence $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_i, \ldots, \mathcal{S}_{T_{ms}}\}$ for DMM $(k : \ell : m)$, where $k + \ell + m = 4$, is derived from dilution sequence $\mathcal{D} = \{(sbx)^1, (sbx)^2, \ldots, (sbx)^i, \ldots, (sbx)^{T_{ms}}\}$, as depicted in Equation (8).

$$\mathcal{S}_\delta = \begin{cases} \text{`}O\text{'} & \text{if } (sbx)^\delta = \langle k\ell 0\rangle \\ \text{`}D\text{'} & \text{if } (sbx)^\delta = \langle k\ell 1\rangle \\ \text{`}M\text{'} & \text{if } (sbx)^\delta = \langle k\ell 2\rangle \\ \text{`}N\text{'} & \text{if } (sbx)^\delta = \langle k\ell 3\rangle \end{cases} \tag{8}$$

In PMD, the bottom-left cell (cell coordinate $(x, y) = (1, 1)$ as given in Figure 1(a)) is considered as the detection unit where C_t is generated. Mixer coordinates for each mixing step are calculated in such a way that C_t is generated at (1, 1). Since the $(\delta + 1)^{th}$ mixing step $(m_{\delta+1})$ in a dilution tree has a portion of intermediate *CF* (i.e., CF_δ) from the δ^{th} mixing step (m_δ), then the mixer for $m_{\delta+1}$ is overlapped on cell(s) from the mixer for m_δ. The scanning sequence $\mathcal{S}$ gives an idea about the relative mixer placement based on the intermediate *CF* utilized from the immediately preceding level. Figure 7 represents the mixer placement, $M_{\delta+1}$, for level $(\delta + 1)$ on the basis of scanning sequence element $\mathcal{S}_{\delta+1}$, which gives relative placement of the mixer comparative to M_δ placed for level δ.

The dilution tree is scanned in bottom-up fashion. On the basis of mixing model used and the dilution sequence $(\mathcal{D})$, the scanning sequence $\mathcal{S}$ is calculated as depicted in Figure 6. The relative positioning of mixers (or scanning sequence element) is either horizontal or vertical (named as Manhattan move, '*M*'), diagonal move (named as '*D*'), without movement of the mixer (named as no move, '*N*'), or the first operation that is absolute (named as '*O*'). The term *move* represents the relative placement of the mixer (shown in the green square box) for level $(\delta + 1)$ with respect to the mixer placed (shown in the red square box) for level δ. The Manhattan move ('*M*') is possible in four directions in PMD chip; horizontally, either left (W) or right (E), vertically, either up (N) or down (S), and are denoted as M_W, M_E, M_N, and M_S, respectively. The diagonal move ('*D*') is also possible in four directions; northeast (NE), southeast (SE), southwest (SW), northwest (NW), and are denoted as D_{NE}, D_{SE}, D_{SW}, and D_{NW}, respectively, as shown in Figure 7(e). The scanning sequence element '*O*' represents the first mixer placement where only sample and buffer units (nodes in the graph and cells in the PMD layout shown in the sunset-orange color in Figure 7(a)) are mixed. The element '*D*' places mixer $M_{\delta+1}$ diagonal to mixer M_δ because it utilizes only one cell unit intermediate *CF* y_i^δ (nodes in the graph and cells in the PMD layout shown in the orange-red color in Figure 7(b)), k and ℓ cell units of sample and buffer, respectively, where $k + \ell = 3$, as shown in Figure 7(b). Here, $\mathcal{S}_{\delta+1} = D_{SW}$, because $M_{\delta+1}$ is placed southwest to M_δ. The element

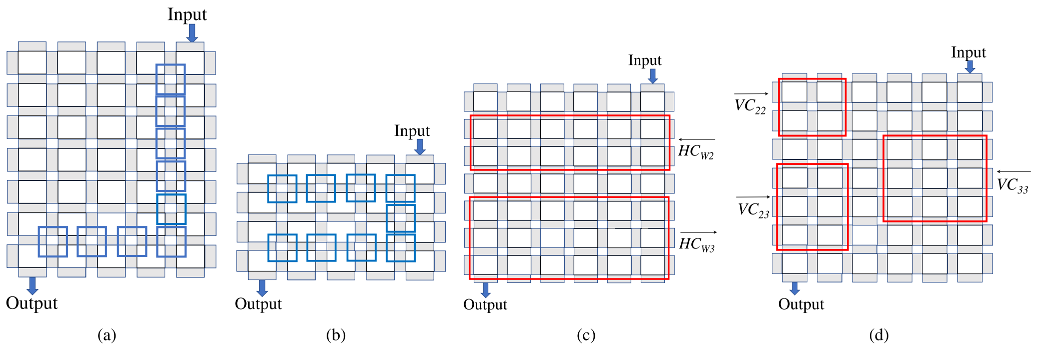


Fig. 8. (a) Mixer (square box in "blue") placement representation over the PMD chip in the Manhattan manner. (b) Mixer placement representation over the PMD chip in the serpentile manner. (c–d) Schematic view of varying size virtual directional horizontal and vertical channels assignment in PMD layout.

'M' represents the Manhattan (either horizontal or vertical) move of $M_{\delta+1}$ with respect to M_δ as depicted in Figure 7(c). Two cell units of intermediate CF y_i^δ (nodes in the graph and cells in the PMD layout shown in the orange-red color in Figure 7(c)) are mixed with k and ℓ cell units of sample and buffer units, respectively, where $k + \ell = 2$, and four cell units of intermediate CF $y_i^{\delta+1}$ (node in the graph shown in the sisal color) is generated. Here, $\mathcal{S}_{\delta+1} = N_W$, because $M_{\delta+1}$ is placed horizontally left to M_δ. The element 'N', corresponding to DMM $\langle k\ell 3\rangle$, exploits three cell units of intermediate CF y_i^δ. Since there is no way to place $M_{\delta+1}$ to utilize three cells of CF_δ, then the same mixer, M_δ, is used for $m_{\delta+1}$. It needs to wash one cell filled with y_i^δ that, in turn, leads to extra time cycles and routing overhead as shown in Figure 7(d).

An Illustrative Example: The dilution sequence is $\mathcal{D} = \{\langle 310\rangle, \langle 211\rangle, \langle 031\rangle\}$ for $C_t = \frac{11}{64}$ as depicted in Figure 5. Here, the $(sbx)^1 = \langle 310\rangle \equiv \langle k\ell 0\rangle$ that derives $\mathcal{S}_1$ = 'O' from Equation (8). Next, $(sbx)^2 = \langle 211\rangle \equiv \langle k\ell 1\rangle$ that derives $\mathcal{S}_2$ = 'D'. At last, $(sbx)^3 = \langle 031\rangle \equiv \langle k\ell 1\rangle$ that again derives $\mathcal{S}_3$ = 'D'. Therefore, $\mathcal{S} = \mathcal{S}_1\mathcal{S}_2\mathcal{S}_3 \equiv$ 'ODD' as depicted in Figure 6.

6.2 *GAMA*: Generalized Architectural Mapping Algorithm

Resource placement is a well-known NP-hard problem. The scheduled DAG of a given bioprotocol is mapped to PMD for execution of operations. The mapping can be done in several ways, but the serpentile way of resource placement, especially in the case of routing-free placement for skewed dilution trees, is the optimum way of utilizing the available chip area as depicted in Figure 8. It can be observed that any random or diagonal placement (as shown in Figure 8(a)) will occupy a larger chip area as compared to mixer placement in the serpentile manner for maximal utilization of chip area (as shown in Figure 8(b)). This is our motivation for serpentile mixer placement for the mapping of mixing steps of the dilution tree to the mixing cycles in PMD.

6.2.1 Problem Formulation. The mapping problem for PMD is formulated as follows:
Inputs: Given dilution sequence $\mathcal{D}$, chip area $\mathcal{W} \times \mathcal{H}$.
Outputs: Mixer set $\mathcal{M}$ with mixer cell coordinate pair μ_δ for each $\mathcal{M}_\delta$, where $1 \le \delta \le T_{ms}$.
Objectives: (1) Maximize the utilization of chip area $\mathcal{W} \times \mathcal{H}$. (2) Target CF generation at (1,1), i.e., $\mathcal{M}_{T_{ms}} = (1, 1)$.

The mapping algorithm, named as *GAMA* with given chip area $(\mathcal{W} \times \mathcal{H})$, is designed to map the mixing steps of a mixing tree by allocating mixers on PMD. The pseudocode is presented in Algorithm 2. The serpentile placement of mixers is implemented as alternate horizontal and vertical channels. Since mixer placement for a dilution tree is time-dependent, hence, mixer placement

ALGORITHM 2: *GAMA*$(\mathcal{D}, \mathcal{W}, \mathcal{H})$

```
begin
1    $\mathcal{S} = \phi; \delta = length(\mathcal{D})$;
2    foreach $(sbx)^{\delta} \in \mathcal{D}$ do
3        if $x == 0$ then $\mathcal{S} = O\mathcal{S}$;
4        if $x == 1$ then $\mathcal{S} = \mathcal{S}D$;
5        if $x == 2$ then $\mathcal{S} = \mathcal{S}M$;
6        if $x == 3$ then $\mathcal{S} = \mathcal{S}N$;
7        $\delta = \delta - 1$;
8    $Channel\_Formation(\mathcal{S}, \mathcal{W}, \mathcal{H})$;
9    Calculate sum of height of horizontal channels ($\overrightarrow{HC}$ and $\overleftarrow{HC}$) and non-overlapped vertical channels
     ($\overrightarrow{VC}$ and $\overleftarrow{VC}$) as $H$;
10   if $(H \leq \mathcal{H})$ then $Mixer\_Assignment(\mathcal{S}, N_m)$;
11   else "Routing overhead is required to mix within the available space (chip area).";
```

is directional in terms of space allocation. Apparently, the virtual channel formation is also directional. There are four virtual channels: left-to-right horizontal channel ($\overrightarrow{HC}$), right-to-left horizontal channel ($\overleftarrow{HC}$), left-to-right vertical channel ($\overrightarrow{VC}$), and right-to-left vertical channel ($\overleftarrow{VC}$). The relative positioning of mixers decides the size of these virtual channels. The horizontal directional channels may be of size $\mathcal{W} \times 3$ or $\mathcal{W} \times 2$ denoted as ($\overrightarrow{HC}_{\mathcal{W}3}$), ($\overrightarrow{HC}_{\mathcal{W}2}$), ($\overleftarrow{HC}_{\mathcal{W}3}$), and ($\overleftarrow{HC}_{\mathcal{W}2}$), respectively, as shown in Figure 8(c). Similarly, the vertical directional channels may be of size 2×2, 2×3, or 3×3 denoted as ($\overrightarrow{VC}_{22}$), ($\overrightarrow{VC}_{23}$), ($\overrightarrow{VC}_{33}$), ($\overleftarrow{VC}_{22}$), ($\overleftarrow{VC}_{23}$), and ($\overleftarrow{VC}_{33}$), respectively, as shown in Figure 8(d).

The algorithm, *GAMA*, takes dilution sequence ($\mathcal{D}$) and the chip area ($\mathcal{W} \times \mathcal{H}$) as input. Initially, *GAMA* generates scanning sequence ($\mathcal{S}$) and assigns mixers to PMD cells starting from the bottom left cell coordinate $(x, y) = (1, 1)$ for ease of operation. The scanning sequence is $\mathcal{S} = \mathcal{S}_1\mathcal{S}_2 \cdots \mathcal{S}_{\lambda-1}\mathcal{S}_{\lambda}$, where $\mathcal{S}_1$ is the first operation 'O'. At the next step, *GAMA*, constructs horizontal and vertical directional channels as $\langle \overrightarrow{HC}, \overleftarrow{VC}, \overleftarrow{HC}, \overrightarrow{VC} \rangle$, repeatedly, until all mixing steps are mapped. The left-to-right (right-to-left) directional channels $\overrightarrow{HC}$ and $\overrightarrow{VC}$ ($\overleftarrow{HC}$ and $\overleftarrow{VC}$) place $\mathcal{M}_{\delta}$ over $\mathcal{M}_{\delta-1}$ in left to right (right to left) manner. However, overlapping of mixers $\mathcal{M}_{\delta}$ and $\mathcal{M}_{\delta-\tau}$, where $\tau < \delta$, incurs washing overhead (T'_w). Wash cycles are time cycles to wash the unintended PMD cells of mixer $\mathcal{M}_{\delta-\tau}$, so that new fluid(s) can be loaded in newly vacant PMD cells of $\mathcal{M}_{\delta-\tau}$ to construct $\mathcal{M}_{\delta}$. These wash cycles increase the execution time of a bioprotocol. If channel formation is not possible in this way, then dilution can be done in same PMD chip area by incuring the routing overhead. This routing overhead also leads to extra time cycles for routing the intermediate *CF* along with the washing overhead (T'_w) of a few previously used cells as well, hence, increasing the assay completion time. This overhead (Δ) increases valve switchings that, in turn, reduce the life span of PMD. At last, *GAMA* calls procedure *Mixer_Assignment* to bind PMD cells of virtual channels to mixers.

Channel Formation: The procedure *Channel_Formation* reorders scanning sequence ($\mathcal{S}$) and divides it into subsequences considering size of virtual directional channels. The pseudocode for this is given in Algorithm 3. For ease of operation, 'N' elements are eliminated from $\mathcal{S}$ and a new sequence, $\overline{\mathcal{S}}$, is generated because element 'N' does not affect the mixer allocation. The binding of $\mathcal{M}_{\delta}$ is the same as for $\mathcal{M}_{\delta-1}$ in case of 'N' element as shown in Figure 7(d). After channel formation, these 'N' elements are embedded into the sequence ($\overline{\mathcal{S}}$) back to get the reordered scanning

ALGORITHM 3: *Channel_Formation*($\mathcal{S}, \mathcal{W}, \mathcal{H}$)

```
begin
    Generate a new sequence $\overline{\mathcal{S}}$ from $\mathcal{S}$ by removing 'N' elements;
    $\tau = length(\overline{\mathcal{S}}); N_m[1] = \mathcal{W} - 1; i = 2; cnt = \mathcal{W} - 1;$
    while $(cnt < \tau)$ do
        if ($i$ is odd) then
            if $(N_m[i-1] == 0 \,||\, n_D[i-1] == 0$ or $2)$ then
                if $(\overline{\mathcal{S}}_{cnt+1} ==$ 'D') then $N_m[i] = \mathcal{W} - 2;$
                else $N_m[i] = \mathcal{W} - 1;$
            if $(N_m[i-1] > 0)$ then
                if $(n_D[i-1] == 1)$ then $N_m[i] = \mathcal{W} - 1;$
                else $N_m[i] = \mathcal{W} - 2;$
        else
            if $(n_D[i-1] == 0$ or odd$)$ then $N_m[i] = 1;$
            else $N_m[i] = 2;$
        $n_D[i]$ = no of 'D' elements in subsequence $\overline{\mathcal{S}}^i$;
        $cnt = cnt + N_m[i]; i = i + 1;$
    if $(cnt \neq \tau)$ then $N_m[i-1] = N_m[i-1] - (cnt - \tau);$
    Embed elements 'N' and get sequence $\mathcal{S}$ and update $N_m$ accordingly;
    return $N_m$;
```

sequence, $\mathcal{S}$. The length of these subsequences, stored in N_m array, is equivalent to the width of the channels (horizontal and vertical, alternatively). This alternate placement of horizontal and vertical channels (starting from (1,1)) creates a serpentile tunnel that utilizes the chip area efficiently without the routing of intermediate *CF*s since $\mathcal{M}_\delta$ exploits cells from $\mathcal{M}_{\delta-1}$ only to utilize $CF_{\delta-1}$.

Mixer Assignment: The procedure *Mixer_Assignment* scans the sequence $\mathcal{S}$ and assigns mixer $\mathcal{M}_\delta$ to each scanning sequence element $\mathcal{S}_\delta$. The pseudocode is presented in Algorithm 4. The location (x_δ, y_δ) is the coordinate of the bottom-left cell of mixer $\mathcal{M}_\delta$ of size 2×2.

An Illustrative Example: The algorithm, *GAMA*, derives scanning sequence $\mathcal{S}$ from $\mathcal{D}$ for target $CFC_t = \frac{11}{64}$. The scanning sequence $\mathcal{S}$ is '*ODD*'. At next step, *GAMA* calls procedure *Channel_Formation* to construct virtual directional channels. The procedure *Channel_Formation* returns $N_m = \langle 3 \rangle$ and $N_d = \langle 2 \rangle$. The cell coordinate (x_1, y_1) for element '*O*' is $(1,1)$ as given by line 2 in Algorithm 4. The cell coordinate $(1,1)$ represents the bottom-left cell of $\mathcal{M}_1$. The other coordinates of $\mathcal{M}_1$ are $(1,2)$, $(2,1)$, and $(2,2)$. The mixer $\mathcal{M}$ is $\{(1,1)\}$. The $\mathcal{S}_2$ is 'D_{NE}', so by line 14, the coordinates are calculated as $x_2 = x_1 + 1 = 1 + 1 = 2$ and $y_2 = y_1 + 1 = 1 + 1 = 2$. The coordinates of mixer $\mathcal{M}_2 = \{(2,2)\}$ and $\mathcal{M} = \{(1,1),(2,2)\}$. Since $\mathcal{S}_3 \equiv$ 'D_{SE}', by line 13, the coordinates are calculated as $x_3 = x_2 + 1 = 2 + 1 = 3$ and $y_3 = y_2 - 1 = 2 - 1 = 1$. Apparently, $\mathcal{M}_3 = \{(3,1)\}$ and $\mathcal{M} = \{(1,1),(2,2),(3,1)\}$. After reversing the mixer sequence, we get $\mathcal{M} = \{(3,1),(2,2),(1,1)\}$. The mixers $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3\} = \{(3,1),(2,2),(1,1)\}$ are represented in Figure 6(b), (c), and (d) in magenta, blue, and green, respectively.

6.3 Fluid Assignment to PMD Cells

This section presents an algorithm, namely *FAAP*, that returns a fluid assignment map for PMD cells in the form of fluid-to-cell binding pattern $\mathbb{P}$. The pattern, $\mathbb{P}$, a model of PMD chip of $\mathcal{W} \times \mathcal{H}$

Table 1. Fluid Assignment Table

		022	202	112	103	013	301	031	121	211	220	130	310
O		-	-	-	-	-	-	-	-	-	$\langle S, S, B, B\rangle$	$\langle B, S, B, B\rangle$	$\langle S, B, S, S\rangle$
M	M_E	$\langle *, *, B, B\rangle$	$\langle *, *, S, S\rangle$	$\langle *, *, S, B\rangle$	-	-	-	-	-	-	-	-	-
	M_W	$\langle B, B, *, *\rangle$	$\langle S, S, *, *\rangle$	$\langle S, B, *, *\rangle$	-	-	-	-	-	-	-	-	-
	M_S	$\langle B, *, B, *\rangle$	$\langle S, *, S, *\rangle$	$\langle S, *, B, *\rangle$	-	-	-	-	-	-	-	-	-
N		-	-	-	$\langle *, S, *, *\rangle$	$\langle *, B, *, *\rangle$	-	-	-	-	-	-	-
D	D_{NE}	-	-	-	-	-	$\langle *, S, S, S\rangle$	$\langle *, B, B, B\rangle$	$\langle *, B, S, B\rangle$	$\langle *, S, B, S\rangle$	-	-	-
	D_{NW}	-	-	-	-	-	$\langle S, S, *, S\rangle$	$\langle B, B, *, B\rangle$	$\langle S, B, *, B\rangle$	$\langle B, S, *, S\rangle$	-	-	-
	D_{SW}	-	-	-	-	-	$\langle S, S, S, *\rangle$	$\langle B, B, B, *\rangle$	$\langle B, S, B, *\rangle$	$\langle S, B, S, *\rangle$	-	-	-
	D_{SE}	-	-	-	-	-	$\langle S, *, S, S\rangle$	$\langle B, *, B, B\rangle$	$\langle B, *, B, S\rangle$	$\langle S, *, S, B\rangle$	-	-	-

cells, is represented by the $2D$ matrix where each element represents a PMD cell. An element p_{xy} ($\in \mathbb{P}$) is defined by quadruple $\langle \delta, x, y, \mathcal{F} \rangle$. Here, δ is level, x and y represent the cell coordinates with constraints $1 \leq x \leq \mathcal{W}$ and $1 \leq y \leq \mathcal{H}$, and $\mathcal{F}$ denotes the fluid type (e.g., sample, reagent, buffer) assigned to cell p_{xy}. A lookup table, named as *Fluid_Assignment_Table*, is constructed for the mapping of mixing models $\langle k\ell m\rangle(\equiv (sbx)^{\delta} \cong \mathcal{D}_{\delta} \in \mathcal{D})$ to scanning sequence elements (i.e., $O/M/N/D$). The *Fluid_Assignment_Table* is presented in Table 1. This table is static in nature and utilizes DMMs of *DPMD* targeting routing-free mixer placement in PMD chip. It can also be modified/extended for a general mixing model for online fluid assignment.

The algorithm, *FAAP*, takes dilution sequence ($\mathcal{D}$), scanning sequence ($\mathcal{S}$), and mixer coordinate set ($\mathcal{M}$) as input and searches for an entry ($\mathcal{D}_{\delta}, \mathcal{S}_{\delta}$), where $1 \leq \delta \leq T_{ms}$, in *Fluid_Assignment_Table* and returns the metric $\langle \mathcal{F}_0^{\delta}, \mathcal{F}_1^{\delta}, \mathcal{F}_2^{\delta}, \mathcal{F}_3^{\delta} \rangle$. The metric $\langle \mathcal{F}_0^{\delta}, \mathcal{F}_1^{\delta}, \mathcal{F}_2^{\delta}, \mathcal{F}_3^{\delta} \rangle$ is assigned to the mixer $\mathcal{M}_{\delta}$. The element $\mathcal{F}^{\delta}$ may be either S, B, or $*$ that represents sample, buffer, or $CF_{\delta-1}$, respectively. The metric $\langle \mathcal{F}_0^{\delta}, \mathcal{F}_1^{\delta}, \mathcal{F}_2^{\delta}, \mathcal{F}_3^{\delta} \rangle$ is mapped to coordinates $\langle x_{\delta}, y_{\delta} \rangle, \langle x_{\delta}, y_{\delta} + 1 \rangle, \langle x_{\delta} + 1, y_{\delta} \rangle$, and $\langle x_{\delta} + 1, y_{\delta} + 1 \rangle$, respectively, where $\langle x_{\delta}, y_{\delta} \rangle$ is the bottom-left coordinate of M_{δ}. The pseudocode is presented in Algorithm 5.

An Illustrative Example: The input is $\mathcal{D} = \{\langle 310\rangle, \langle 211\rangle, \langle 031\rangle\}$, $\mathcal{S} =$ 'ODD', and $\mathcal{M} = \{(3,1), (2,2), (1,1)\}$ as depicted in Figure 6. For m_1, three sample units are mixed with one buffer unit. The location of each fluid in the mixer is returned from *Fluid_Assignment_Table* for $(\mathcal{D}_1, \mathcal{S}_1) \equiv$ ($\langle 310\rangle$, 'O') as $\langle S, B, S, S\rangle$. Therefore, mixer $\mathcal{M}_1$ with coordinates $\{(3,1), (3,2), (4,1), (4,2)\}$ is loaded with $\langle S, B, S, S\rangle$ and fluid-to-cell binding pattern is $\mathbb{P} = \{\langle 1,3,1,S\rangle, \langle 1,3,2,B\rangle, \langle 1,4,1,S\rangle, \langle 1,4,2,S\rangle\}$. Next, for $(\mathcal{D}_2, \mathcal{S}_2) \equiv$ ($\langle 211\rangle$, 'D_{NW}'), the entry is $\langle B, S, *, S\rangle$. Here, $\mathcal{M}_2$ is in the northwest location to $\mathcal{M}_1$; therefore, entry for 'D_{NW}' is returned. $\mathcal{M}_2$ with coordinates $\{(2,2), (2,3), (3,2), (3,3)\}$ is loaded with $\langle B, S, *, S\rangle$ and $\mathbb{P}$ is $\{\langle 1,3,1,S\rangle, \langle 1,3,2,B\rangle, \langle 1,4,1,S\rangle, \langle 1,4,2,S\rangle, \langle 2,2,2,B\rangle, \langle 2,2,3,S\rangle, \langle 2,3,2,*\rangle, \langle 2,3,3,S\rangle\}$. Here, $*$ implies presence of intended fluid CF_1 present at location (3,2) so it does not need to be loaded. Similarly, for $(\mathcal{D}_3, \mathcal{S}_3) \equiv$ ($\langle 031\rangle$, 'D_{SW}'), the entry is $\langle B, B, B, *\rangle$. Therefore, $\mathcal{M}_3$ with coordinates $\{(1,1), (1,2), (2,1), (2,2)\}$ is loaded with $\langle B, B, B, *\rangle$ and $\mathbb{P}$ is $\{\langle 1,3,1,S\rangle, \langle 1,3,2,B\rangle, \langle 1,4,1,S\rangle, \langle 1,4,2,S\rangle, \langle 2,2,2,B\rangle, \langle 2,2,3,S\rangle, \langle 2,3,2,*\rangle, \langle 2,3,3,S\rangle, \langle 3,1,1,B\rangle, \langle 3,1,2,B\rangle, \langle 3,2,1,B\rangle, \langle 3,2,2,*\rangle\}$.

ALGORITHM 4: *Mixer_Assignment*$(\mathcal{S}, N_m)$

```
begin
    $\delta = 1, \mathcal{M} = \phi$;
    $x_1 = 1; y_1 = 1$;
    while ($\delta \le length(\mathcal{S})$) do
        if ($\mathcal{S}_\delta$ == 'N') then $x_\delta = x_{\delta-1}; y_\delta = y_{\delta-1}$;
        else if ($\mathcal{S}_\delta$ == 'M') then
            if ($\delta \in \overrightarrow{VC}||\overleftarrow{VC}||$ first element of $\overrightarrow{HC}$ or $\overleftarrow{HC}$) then
                $x_\delta = x_{\delta-1}; y_\delta = y_{\delta-1} + 1$;
            else if ($\delta \in \overleftarrow{HC}$) then $x_\delta = x_{\delta-1} - 1; y_\delta = y_{\delta-1}$;
            else if ($\delta \in \overrightarrow{HC}$) then $x_\delta = x_{\delta-1} + 1; y_\delta = y_{\delta-1}$;
        else if ($\mathcal{S}_\delta$ == 'D') then
            if ($D_{SW}$) then $x_\delta = x_{\delta-1} - 1; y_\delta = y_{\delta-1} - 1$;
            else if ($D_{NW}$) then $x_\delta = x_{\delta-1} - 1; y_\delta = y_{\delta-1} + 1$;
            else if ($D_{SE}$) then $x_\delta = x_{\delta-1} + 1; y_\delta = y_{\delta-1} - 1$;
            else if ($D_{NE}$) then $x_\delta = x_{\delta-1} + 1; y_\delta = y_{\delta-1} + 1$;
        $\mathcal{M}_\delta = \{\mu_\delta : \langle x_\delta, y_\delta \rangle\}; \mathcal{M} = \mathcal{M} \cup \mathcal{M}_\delta$;
        $\delta = \delta + 1$;
    Reverse the cell coordinates pair in $\mathcal{M}$;
    return $\mathcal{M}$;
```

ALGORITHM 5: *FAAP*$(\mathcal{D}, \mathcal{S}, \mathcal{M})$

```
begin
    $\mathcal{D} = \{(sbx)^1, (sbx)^2, \ldots, (sbx)^{T_{ms}}\}; \mathcal{S} = S_1 S_2 \cdots S_{T_{ms}}; \mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_{T_{ms}}\}$;
    $\delta = 1$;
    while ($\delta \le T_{ms}$) do
        Given $\mathcal{D}_\delta$ and $\mathcal{S}_\delta$, map entry in Fluid_Assignment_Table and get the quadruple
        $\langle \mathcal{F}_0^\delta, \mathcal{F}_1^\delta, \mathcal{F}_2^\delta, \mathcal{F}_3^\delta \rangle$;
        For selected Mixer $\mathcal{M}_\delta$ assign binding of fluids as:
        $\mathbb{P} \leftarrow \langle \delta, x_\delta, y_\delta, \mathcal{F}_0^\delta \rangle, \mathbb{P} \leftarrow \langle \delta, x_\delta, y_\delta + 1, \mathcal{F}_1^\delta \rangle, \mathbb{P} \leftarrow \langle \delta, x_\delta + 1, y_\delta, \mathcal{F}_2^\delta \rangle$,
        $\mathbb{P} \leftarrow \langle \delta, x_\delta + 1, y_\delta + 1, \mathcal{F}_3^\delta \rangle$;
        $\delta = \delta + 1$;
    return fluid-to-cell binding pattern $\mathbb{P}$;
```

7 FLUID LOADING TO PMD CELLS

Fluid-to-cell loading pattern $\mathbb{P}$ derived from Section 6.3 assigns fluids to cells of virtual mixers. In the absence of the automation of the fluid loading mechanism, the valve switching is calculated manually and it requires load-wash cycles equivalent to the number of PMD cells to be loaded. For example, presented in Figure 6, 10 PMD cells (five cells with sample and five cells with buffer), out of 12 are loaded. It takes 10 load-wash cycles to fill in these PMD cells without automation. The automation of fluid loading can load multiple cells of the same fluids in one load-wash cycle, which reduces the number of load-wash cycles. This, in turn, also minimizes the valve actuations and increases the life span of the PMD-chip. This section presents an algorithm named as *FLAP* that minimizes the number of load-wash cycles for fluid loading in PMD cells.

7.1 Problem Formulation

The fluid loading problem for PMD architecture is formulated as follows:
Inputs: Given fluid binding pattern $\mathbb{P}$, chip area $\mathcal{W} \times \mathcal{H}$.
Outputs: #Load cycles, #wash cycles.
Objectives: To minimize the number of load-wash cycles to load fluids into the PMD chip.

7.2 FLAP: Fluid Loading Algorithm for PMD

FLAP takes fluid-to-cell binding pattern $\mathbb{P}$ as input and groups the connected cells that need to be filled with the same fluid. These cells are loaded in one load cycle because these cells are in the fluid flow path from fluid inlet to the fluid outlet. To identify the cells that have already been part of a load cycle, cells are marked as loaded cells; otherwise, they are marked as unloaded cells.

Step 1: Find the cell at the most lower-right place among all unloaded cells.
Step 2: Find the largest connected group of cells filled with fluid of the same type adjacent to the cell found at Step 1 that can be found in the path from fluid inlet to the fluid outlet in a manhattan way. If there are no such cells, select just one cell.
Step 3: If there is still an unloaded cell, go back to Step 1. Otherwise, finish.

An Illustrative Example: The fluid-to-cell loading pattern $\mathbb{P} = \{\langle 1,3,1,S\rangle, \langle 1,3,2,B\rangle, \langle 1,4,1,S\rangle, \langle 1,4,2,S\rangle, \langle 2,2,2,B\rangle, \langle 2,2,3,S\rangle, \langle 2,3,2,*\rangle, \langle 2,3,3,S\rangle, \langle 3,1,1,B\rangle, \langle 3,1,2,B\rangle, \langle 3,2,1,B\rangle, \langle 3,2,2,*\rangle\}$ is input.

There are four such groups formed as shown in magenta (for cells with sample fluid) and blue (for cells with buffer fluid) as depicted in Figure 6(e). Therefore, four load-wash cycles (since, in this case, no extra washing overhead incurs) can load these fluids for generating the target *CF* $C_t = \frac{11}{64}$.

8 SIMULATION RESULTS AND ANALYSIS

We have implemented *TWM* [4], *FloSPA-D* [24], and the proposed algorithm *DPMD*, using C programming language and execute the programs in a Linux machine with Intel *i*7 3.40*GHz* CPU and 8*GB* memory. We have compared the *DPMD* with *FloSPA-D* and *TWM*. *FloSPA-D* is the state-of-the-art dilution algorithm for CMF-based biochips that exploits all possible mixing models ($k_1 : k_2 : \cdots : k_n$). *TWM* is a naive dilution algorithm for skewed dilution trees that works on a (1 : 1) mixing model. *TWM* is directly mapped in PMD chip using the proposed design automation technique. The proposed dilution scheme *DPMD* utilizes only DMMs as discussed in Section 4.

It is observed from the simulation results that the performance of *DPMD* is better than the *TWM* and comparative to *FloSPA-D*. The mixing steps taken by *DPMD* are equivalent to the number of mixing steps taken by *FloSPA-D* as depicted in Figure 9(a). Though, *TWM* requires almost double the number of mixing steps comparative to *FloSPA-D* and *DPMD* as it works on the principle of the (1 : 1) mixing model. The objective of *FloSPA-D* is to minimize the mixing steps as well as the sample units. In order to find out the optimal solution, *FloSPA-D* explores all possible solutions using all possible mixing models. Therefore, it is inefficient in terms of computation time as given in Table 2. Since there is a tradeoff between time and space, *DPMD* considers DMMs and restricts the *ESS*, hence, it terminates in polynomial time. The comparative performance of waste, sample, and buffer units are shown in Figure 9(b), (c), and (d), respectively. Table 2 represents the comparative CPU time for *DPMD*, *FloSPA-D*, and *TWM* for accuracies 1–6 including total execution time, maximum execution (ME) time for a given accuracy over the range, and includes search space

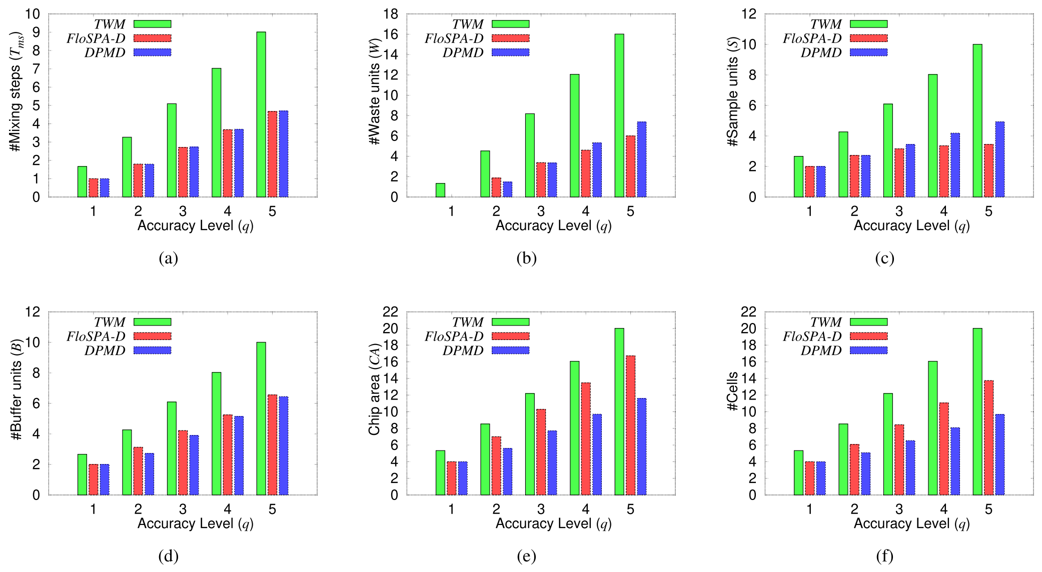


Fig. 9. Comparison of sample preparation parameters (a) mixing steps, (b) waste, (c) sample, and (d) buffer with varying accuracies from 1 to 5. (e) Chip area and (f) the number of cells used are compared for all three algorithms with varying accuracies from 1 to 5.

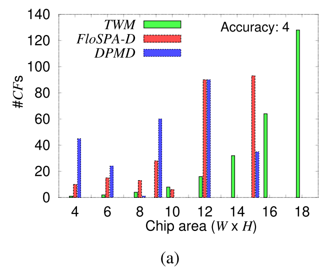
Table 2. CPU Time (in milliseconds) Comparison of *DPMD*, *FloSPA-D* [24], and *TWM* [4] with Varying Accuracies from 1 to 6, where SSE Time, Total Time, and ME Time Represents Search Space Exploration Time, Total Execution Time for All *CF*s for Selected Accuracy, and Maximum Execution Time Taken for a Specific *CF* over the Range

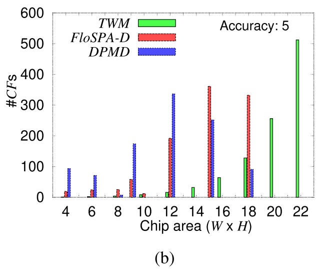
q	*TWM* [4]		*FloSPA-D* [24]		*DPMD*		
	Total Time	ME Time	Total Time	ME Time	SSE Time	Total Time	ME Time
1	1.1	0.4	27.3	10.2	0.2	0.4	0.2
2	5.6	0.5	223.8	21.1	0.4	0.7	0.4
3	43.9	4.2	1,969.5	50.7	2.2	2.7	2.2
4	81	8.7	41,608	687.8	6.1	9.3	6.2
5	140.1	58	2,080,141.2	21,432.7	104.9	158.8	105.2
6	423.9	70.1	215,998,215.5	2,855,287.9	1,730.6	2,534.8	1,731.8

exploration (SSE) time in case of *DPMD*. Once the search space is explored, *DPMD* searches for intended *CF* only within the *ESS*. The CPU time indicated that *FloSPA-D* is exponential while *TWM* and *DPMD* are polynomial in nature.

Although the *PMSP* of *DPMD* is better than *TWM* and comparative with *FloSPA-D*, the *PMCD* is improved for *DPMD* as compared to *TWM* and *FloSPA-D* as well. All three dilution algorithms are mapped using *GAMA*. The simulation results show that *DPMD* is better in chip area utilization as compared to *TWM* and *FloSPA-D* as depicted in Figure 9(e) and (f). The number of cells is also crucial as it relates to the number of valve actuations over the given chip area.

The comparative histograms for thechip area and the number of *CF*s over the range for accuracies 4, 5, and 6 are presented in Figure 10(a), (b), and (c). The analysis justifies that *TWM* requires double the mixing steps, hence, a larger chip area is required for most of the *CF*s. *DPMD* is better in minimizing the chip area, and, at max, 15, 18, and 21 are required for accuracies 4, 5, and 6. On the





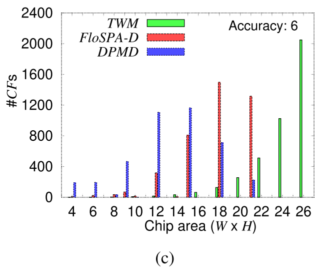
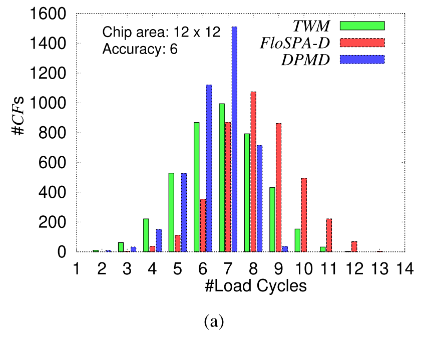


Fig. 10. Comparative histograms for chip area and number of *CFs* for accuracy (a) 4, (b) 5, and (c) 6 for all three algorithms.

Table 3. Non-doable *CF*s for Varying Chip Area and Accuracies from 1–6 for Dilution Algorithms: *FloSPA-D* [24], *TWM* [4], *DPMD*

Chip area	*TWM* [4]						*FloSPA-D* [24]						*DPMD*					
	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6
8 × 8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5 × 5	-	-	-	-	-	3,072	-	-	-	-	-	-	-	-	-	-	-	-
3 × 6	-	-	-	-	768	3,840	-	-	-	-	-	1,099	-	-	-	-	-	207
3 × 5	-	-	-	192	960	4,032	-	-	-	-	252	2,599	-	-	-	-	82	856
3 × 4	-	-	32	224	992	4,064	-	-	-	82	641	3,537	-	-	-	32	309	1,937

- : Represents that all *CF*s for selected accuracy can be executed in selected chip area.



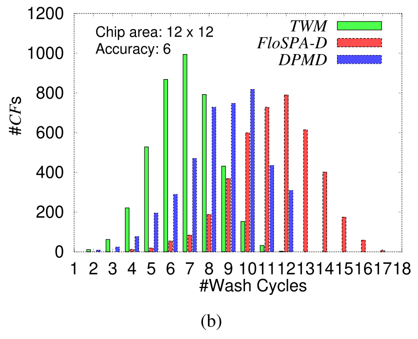


Fig. 11. Comparative histograms for (a) load cycles and number of *CF*s and (b) wash cycles and number of *CF*s for chip area 12 × 12 with accuracy 6 for all three algorithms.

other side, if the chip area is given, then the number of doable accuracies over the range for given accuracy is also crucial. Table 3 shows the non-doable accuracies while varying chip area and accuracy. For example, *TWM*, *FloSPA-D*, and *DPMD* do not execute 4,064, 3,537, and 1,937 number of *CF*s out of 4,096 for accuracy-6 if the given chip area is 3 × 4 only. Considering the objective of chip area minimization for higher accuracies, it can be observed that *DPMD* outperforms, and, comparatively, there are less number of accuracies that cannot be executed (i.e., non-doable accuracies). The table entry "-" represents that all *CF*s for selected accuracy can be executed for a given chip area.

The comparative histograms for load and wash cycles with number of *CF*s are presented in Figure 11(a) and (b). Here, we presented the result for accuracy 6 with chip area 12 × 12. It is clear from the histogram that *DPMD* outperforms in minimizing the load and wash cycles.

9 CONCLUSIONS AND FUTURE WORK

In this article, we have presented an automated design methodology for PMD for each mixing step of a skewed dilution tree. The proposed dilution algorithm *DPMD* generates a skewed dilution tree with the objective of minimizing the mixing steps, waste generated, and sample requirement. The proposed architectural mapping algorithm *GAMA* maps the mixing steps of this tree into the scanning sequence, which is used for synthesis. A linear time dilution algorithm, *DPMD*, outperforms *TWM* and is comparative in results to the state-of-the-art technology *FloSPA-D*. The resource placement algorithm, *GAMA*, forms virtual channels in a serpentile manner and allocates mixers to them to maximally utilize the chip area. Next, the fluid-to-cell binding algorithm, *FAAP*, assigns intended fluids to PMD cells. Eventually, the fluid loading algorithm, *FLAP*, calculates the load-wash cycles. Simulation results show that *PMSP* is comparative to *FloSPA-D* but outperforms *TWM*. The *PMCD* for *DPMD* outperforms and generates better results from *TWM* and *FloSPA-D*.

As a future research perspective, the proposed work can be extended for a general dilution graph or complete dilution tree. Though the placement is an NP-Hard problem, a heuristic along with an optimal solution can also be proposed for general dilution trees. Similarly, optimization of fluid loading is also part of our future research work.

ACKNOWLEDGEMENTS

The authors wish to thank the Associate Editor Prof. Tsung-Yi Ho, and the anonymous reviewers for their critical comments and valuable suggestions.

REFERENCES

[1] K. Karns and A. E. Herr. 2011. Human tear protein analysis enabled by an alkaline microfluidic homogeneous immunoassay. *Anal. Chem.* 83, 21 (2011), 8115–8122.

[2] S. Einav, D. Gerber, P. D. Bryson, E. H. Sklan, M. Elazar, S. J. Maerkl, J. S. Glenn, and S. R. Quake. 2008. Discovery of a hepatitis C target and its pharmacological inhibitors by microfluidic affinity analysis. *Nat. Biotechnol.* 26, 9 (2008), 1019–1027.

[3] C. D. Chin, T. Laksanasopin, Y. K. Cheung, D. Steinmiller, V. Linder, H. Parsa, J. Wang, H. Moore, R. Rouse, G. Umviligihozo, E. Karita, L. Mwambarangwe, S. L. Braunstein, J. Wijgert, R. Sahabo, J. E. Justman, W. El-Sadr, and S. K. Sia. 2008. Microfluidics-based diagnostics of infectious diseases in the developing world. *Nat. Med.* 17, 8 (2008), 1015–1019.

[4] W. Thies, J. P. Urbanski, T. Thorsen, and S. Amarasinghe. 2008. Abstraction layers for scalable microfluidic biocomputing. *Nat. Comput.* 7, 2 (2008), 255–275.

[5] K. Chakrabarty and T. Xu. 2010. *Digital Microfluidic Biochips: Design and Optimization.* CRC Press.

[6] R. B. Fair. 2007. Digital microfluidics: Is a true lab-on-a-chip possible? *Microfluid. Nanofluid.* 3, 3 (2007), 245–281.

[7] G. Wang, D. Teng, and Shih-Kang Fan. 2011. Digital microfluidic operations on micro-electrode dot array architecture. *IET Nanobiotechnol.* 5, 4 (2011), 152–160.

[8] M. L. Fidalgo and S. J. Maerkl. 2011. A software-programmable microfluidic device for automated biology. *Lab Chip* 11, 1 (2011), 1612–1619.

[9] S. Roy, B. B. Bhattacharya, and K. Chakrabarty. 2010. Optimization of dilution and mixing of biochemical samples using digital microfluidic biochips. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. (TCAD)* 29, 11 (2010), 1696–1708.

[10] S. Roy, P. P. Chakrabarti, K. Chakrabarty, and B. B. Bhattacharya. 2015. Waste-aware single-target dilution of a biochemical fluid using digital microfluidic biochips. *Integr., VLSI J.* 51, 194–207.

[11] C. H. Liu, K. C. Shen, and J. D. Huang. 2015. Reactant minimization for sample preparation on microfluidic biochips with various mixing models. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 34, 12 (2015), 1918–1927.

[12] T. W. Chiang, C. H. Liu, and J. D. Huang. 2013. Graph-based optimal reactant minimization for sample preparation on digital microfluidic biochips. In *Proc. of International Symposium on VLSI Design, Automation, and Test (VLSI-DAT)*. 1–4.

[13] J. D. Huang, C. H. Liu, and H. S. Lin. 2013. Reactant and waste minimization in multitarget sample preparation on digital microfluidic biochips. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 32, 10 (2013), 1484–1494.

[14] D. Mitra, S. Roy, S. Bhattacharjee, K. Chakrabarty, and B. B. Bhattacharya. 2014. On-chip sample preparation for multiple targets using digital microfluidics. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 33, 8 (2014), 1131–1144.

[15] S. Roy, B. B. Bhattacharya, S. Ghoshal, and K. Chakrabarty. 2013. On-chip dilution from multiple concentrations of a sample fluid using digital microfluidics. In *Proc. of the International Symposium on VLSI Design and Test (VDAT)*. 1–9.

[16] T. A. Dinh, S. Yamashita, and T. Y. Ho. 2014. A network-flow-based optimal sample preparation algorithm for digital microfluidic biochips. In *Proc. of the Asia and South Pacific Design Automation Conference (ASP-DAC)*. 225–230.

[17] Y.-L. Hsieh, T.-Y. Ho, and K. Chakrabarty. 2012. A reagent-saving mixing algorithm for preparing multiple-target biochemical samples using digital microfluidics. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 31, 11 (2012), 1656–1669.

[18] S. Kumar, S. Roy, P. P. Chakrabarti, B. B. Bhattacharya, and K. Chakrabarty. 2013. Efficient mixture preparation on digital microfluidic biochips. In *Proc. of the IEEE International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS)*. 205–210.

[19] C.-H. Liu, H.-H. Chang, T.-C. Liang, and J. D. Huang. 2013. Sample preparation for many-reactant bioassay on DMFBs using common dilution operation sharing. In *Proc. of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 615–621.

[20] S. Roy, P. P. Chakrabarti, S. Kuamr, K. Chakrabarty, and B. B. Bhattacharya. 2015. Layout-aware mixture preparation of biochemical fluids on application-specific digital microfluidic biochips. *ACM Trans. Des. Autom. Electron. Syst.* 20, 3 (2015), 45.1–45.34.

[21] S. Roy, B. B. Bhattacharya, S. Ghoshal, and K. Chakrabarty. 2014. Theory and analysis of generalized mixing and dilution of biochemical fluids using digital microfluidic biochips. *ACM J. Emerging Technol. Comput.* 11, 1 (2014), 2.1–2.33.

[22] C. M. Huang, C. H. Liu, and J. D. Huang. 2015. Volume-oriented sample preparation for reactant minimization on flow-based microfluidic biochips with multi-segment mixers. In *Proc. of the Design, Automation Test in Europe Conference Exhibition (DATE)*. 1114–1119.

[23] J. D. Huang, C. H. Liu, and T. W. Chiang. 2012. Reactant minimization during sample preparation on digital microfluidic biochips using skewed mixing trees. In *Proc. of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 377–384.

[24] S. Bhattacharjee, S. Poddar, S. Roy, J. D. Huang, and B. B. Bhattacharya. 2017. Dilution and mixing algorithms for flow-based microfluidic biochips. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 36, 4 (2017), 614–627.

[25] T. M. Tseng, B. Li, T. Y. Ho, and U. Schlichtmann. 2015. Reliability-aware synthesis for flow-based microfluidic biochips by dynamic-device mapping. In *Proc. of the 52nd Annual Design Automation Conference (DAC)*. 141.

[26] H. Yao, Q. Wang, Y. Ru, Y. Cai, and T. Y. Ho. 2015. Integrated flow-control codesign methodology for flow-based microfluidic biochips. *IEEE Des. Test* 32, 6 (2015), 60–68.

[27] Y. S. Su, T. Y. Ho, and D. T. Lee. 2016. A routability-driven flow routing algorithm for programmable microfluidic devices. In *Proc. of the 2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. 605–610.

[28] C. Liu, B. Li, B. B. Bhattacharya, K. Chakrabarty, T. Y. Ho, and U. Schlichtmann. 2017. Testing microfluidic fully programmable valve arrays (FPVAs). In *Proc. of the Design, Automation Test in Europe Conference Exhibition (DATE)*. 91–96.

[29] Q. Wang, S. Zuo, H. Yao, T. Y. Ho, B. Li, U. Schlichtmann, and Y. Cai. 2017. Hamming-distance-based valve-switching optimization for control-layer multiplexing in flow-based microfluidic biochips. In *Proc. of the 2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*. 524–529.

[30] S. Kumar, A. Gupta, S. Roy, and B. B. Bhattacharya. 2016. Design automation of multiple-demand mixture preparation using a k-array rotary mixer on digital microfluidic biochip. In *Proc. of the 34th IEEE International Conference on Computer Design (ICCD)*. 273–280.

[31] Z. Li, K. Y. Lai, K. Chakrabarty, T. Ho, and C. Lee. 2017. Sample preparation on micro-electrode-dot-array digital microfluidic biochips. In *Proc. of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. 146–151.

[32] Z. Li, K. Y. Lai, K. Chakrabarty, T. Ho, and C. Lee. 2017. Droplet size-aware and error-correcting sample preparation using micro-electrode-dot-array digital microfluidic biochips. *IEEE Trans. Biomed. Circuits Syst.* 11, 6 (2017), 1380–1391.

[33] K. A. Kalanick. 2011. *Phlebotomy Technician Specialist, 2nd Edition.* Delmar Cengage Learning.

[34] J. M. Butler. 2009. *Fundamentals of Forensic DNA Typing.* Academic Press.

Received April 2018; revised September 2018; accepted December 2018