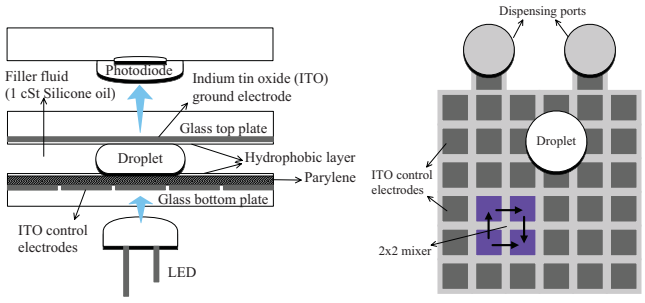
Contamination-Aware Routing Flow for Both Functional and Washing Droplets in Digital Microfluidic Biochips*

Qin Wang[1], Yiren Shen[1]†, Hailong Yao[1], Tsung-Yi Ho[2], and Yici Cai[1]
1. Tsinghua University 2. National Chiao Tung University
woodythu@163.com, shenyiren36@gmail.com, hailongyao@tsinghua.edu.cn
ho.tsungyi@gmail.com, caiyc@mail.tsinghua.edu.cn

Abstract— A major issue in digital microfluidic biochips is cross-contamination caused by different biomolecule droplets crossing the same sites, where washing operations are necessary to avoid wrong assay results. Existing works either assume unrealistic infinite washing capacity, or ignore execution-time constraint and/or routing conflicts between functional and washing droplets. This paper presents the first practical droplet routing flow considering both realistic washing capacity constraint and routing conflicts between washing and functional droplets. Experimental results are promising.

I. Introduction

Digital microfluidic biochips (DMFB) are emerging as a revolutionary technique for the automation of laboratory in biochemistry, where nanolitre-sized droplets are manipulated on an open surface of a 2-D array of electrodes using the electrowetting technology [1, 2]. Compared with the traditional laboratory procedures, DMFB greatly reduces analysis time and sample and reagent consumption. DMFBs have many promising applications in biochemical analysis including enzymatic assays, DNA sequencing, cell-based assays and immunoassays [2, 3, 4, 5].



(a) Cross-sectional view. (b) Top view.

Fig. 1. Schematic of a digital microfluidic biochip [1, 2].

Figure 1 shows the schematic of a DMFB. DMFB is based on the electrowetting technology [2], which controls the wetting behavior of a polarizable or conductive liquid droplet by an electric field, and thus controls the movement of the droplet. Figure 1(a) shows the cross-sectional view of the DMFB. By applying a series of voltages to adjacent control electrodes, the droplets between the top and bottom plates will move to different cells as expected. Here, a *cell* refers to the square room of a control electrode. Utilizing the electrowetting technology, automatic biochemical experiments can be performed. Different sample or reagent droplets can be transported to the same cell for mixing and then transported to another cell for detection. A typical method for detection is to use LED and photodiode detector. Figure 1(b) shows the 2-D electrode array. The dispensing ports are used to input/output the droplets. As shown in Figure 1(b), the size of the droplet is large enough such that two droplets sitting on adjacent electrodes will automatically mix together. Therefore, droplets are not allowed to be adjacent in the DMFB unless they are planned to mix together. A mixer is needed for completely mixing two droplets. Figure 1(b) shows a 2×2 mixer. There are other modules in DMFB for the biochemical experiments [1], such as the mixers of other sizes, the storage cell, etc. For different types of sequential experiments, the modules can be dynamically reconfigured to different places. As a result, the whole droplet routing problem is decomposed into a series of *sub-problems*, where in each sub-problem the modules are located at prespecified positions. As the size of DMFB becomes larger and the bioassay becomes more complicated, computer-aided design (CAD) methods are becoming necessary for the automated droplet path computation and scheduling.

In the past decade, noticeable advances have been made in CAD methodology for digital microfluidic biochips, including resource binding, operation scheduling, module placement, and droplet routing [6, 7, 8, 9, 10, 11]. Among different stages in the DMFB design flow, droplet routing is a most important stage, which determines the final routing paths for the droplets between the reservoirs/dispensing ports, optical detectors, etc., and thus determines the correctness and performance (execution time) for the assays. Previous droplet routing methods mostly focus on two basic routing constraints [6, 8, 9, 10, 11]: (1) fluidic constraint to avoid unexpected mixing of two droplets during their transportation, and (2) timing constraint to satisfy the maximum allowed transportation time of a droplet. A typical objective is to minimize the number of cells used for droplet routing, such that the number of driv-

*The work of H. Yao was supported in part by the National Natural Science Foundation of China (61106104), Doctoral Fund of Ministry of Education of China (20111011328), and Tsinghua University Initiative Scientific Research Program (20121087997). The work of T.-Y. Ho was supported in part by the Taiwan Ministry of Science and Technology under grant no. MOST 102-2221-E-009-194-MY3, 103-2220-E-009-029, and 103-2923-E-009-006-MY3. The work of Y. Cai was supported by the National Natural Science Foundation of China (61274031).

†This work was done during Mr. Yiren Shen's internship at Tsinghua University.

ing electrodes can be minimized for power and interconnection savings. However, the above basic constraints do not consider the cross-contamination issue, which occurs between sequential droplet routes on their intersection sites. As functional (i.e., sample or reagent) droplets often leave residues along the cells (electrodes) on their paths, cross-contamination will occur when the routing paths have intersections. Such cross-contamination will cause wrong assay results.

Therefore, routing paths of different *nets*[1] should ideally be disjoint from each other to avoid the number of cross-contamination sites. When no disjoint routing paths are available, *washing droplets* will be introduced for cleaning the prior droplets' residue before the latter droplet passes through the intersection site [12]. Several droplet-routing methods have been presented to consider the cross-contamination issue [13, 14, 15, 16]. However, the above works have unrealistic assumptions that the washing droplets have unlimited washing capacity. In reality, the washing capacity of a washing droplet will decrease as more and more residues are washed away from the electrodes. Thus, the capacity constraint for a washing droplet needs to be considered [17].

Mitra et al. propose a residue removal algorithm, which considers the finite capability for washing droplets [17]. However, the algorithm has oversimplified assumptions, where functional (sample/reagent) droplet routing (*functional routing* in short) is not considered at all. In their algorithm, a washing droplet is allowed to cross any pathways of a functional droplet before reaching the target cross-contamination site. As a result, routing conflicts between functional and washing droplets occur, and the functional droplet's paths will contaminate the washing droplet before it performs the cleaning task. Besides, synchronization between washing droplet routing (*washing routing* in short) and functional routing is ignored.

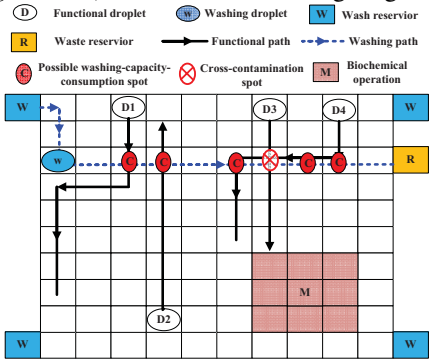


Fig. 2. Practical issues in washing operation with realistic capacity constraint.

Figure 2 illustrates an example showing the practical issues in the washing operation with realistic capacity constraint. From Figure 2, a washing droplet w is dispensed from the wash reservior on the top left corner. In the experiments, we adopt the same configuration from [15] that there are four wash reserviors at the four corners of the DMFB. In Figure 2, the washing droplet w needs to wash the cross-contamination site of functional droplets D_3 and D_4. However, w's path intersects with the functional droplets D_1 and D_2, respectively. Thus, it is possible for the residue of D_1 and D_2 to consume w's washing capacity, even if the routing paths are carefully synchronized. In this paper, we call the above issue as *routing conflicts* between functional and washing droplets. If we want to keep the washing droplet clean on its way, then we have to make D_1 and D_2 wait until w passes the possible washing-capacity-consumption sites. That may result in timing constraint violations on functional droplets D_1 or D_2. The same issue happens for D_3 and D_4. As the washing capacity of w is limited, we need to avoid the possible washing-capacity-consumption sites as much as possible, so as to wash more cross-contamination sites. Moreover, w needs to reach the cross-contamination site after the first functional droplet passes the site and before the second functional droplet reaches it. Only in this way can the washing operation be meaningful. This paper addresses both the above important washing issues.

This paper presents the first practical droplet routing flow considering the realistic washing capacity constraint. More importantly, the routing conflicts between washing and functional droplets are resolved. Major contributions of the paper are as follows.

- The first practical droplet routing flow is presented considering realistic washing capacity constraints.
- Functional droplet paths are considered in washing routing and scheduling to minimize washing capacity consumption.
- Effective routing algorithm is presented for finding satisfactory droplet paths with minimized path crossings.
- Efficient routing compaction algorithm is presented to schedule functional and washing droplets considering the timing constraint.

The rest of the paper is organized as follows. Section II presents the problem formulation. Section III presents the overview of the whole flow. Section IV presents the functional routing method. Section V presents the washing routing method. Section VI presents the experimental results. Finally, conclusion is drawn in Section VII.

II. Problem Formulation

There are three constraints in contamination-aware droplet routing: (1) the fluidic constraint, (2) the timing constraint, and (3) the contamination constraint. However, in real applications the washing droplet gets dirty after several washing operations. Thus, the washing capacity limit needs to be considered as an additional constraint.

The fluidic constraint is used to prevent unexpected mixing between two droplets of different nets during droplet transportation. We use D_i at position (x_i^t, y_i^t) and D_j at position (x_j^t, y_j^t) to represent two individual droplets at time t. Then the static and dynamic fluidic constraints [15] are as follows:

$$|x_i^t - x_j^t| > 1 \quad or \quad |y_i^t - y_j^t| > 1 \tag{1}$$

$$\begin{aligned} &|x_i^{t+1} - x_j^t| > 1 \quad or \quad |y_i^{t+1} - y_j^t| > 1 \\ OR \quad &|x_i^t - x_j^{t+1}| > 1 \quad or \quad |y_i^t - y_j^{t+1}| > 1 \end{aligned} \tag{2}$$

Another important constraint is the timing constraint, which is mainly used to ensure the bioassay execution time. Typically, the shorter paths for the droplets, the faster bioassay execution and the better reliability. The contamination constraint is used to prevent cross-contamination between functional

[1] In the droplet routing context, a *net* refers to a set of cells to be connected, among which there may be more than one source cells and a single target cell.

droplets. The liquid residue left by the first droplet should be washed away before the second droplet passes through the intersection site. Furthermore, during the washing process, the realistic washing capacity constraint should be considered for each washing droplet.

The practical contamination-aware droplet routing problem can be formulated as follows:

Input: A list of nets, a set of washing droplets, a set of routing blockages, a set of reserviors, the timing constraint, and the capacity constraint of the washing droplet.

Objective: Compute the feasible routing and scheduling solution for all of the nets without violating any constraints, while minimizing weighted sum of the execution time, the number of cross-contamination spots (sites), and the number of used cells for routing.

Constraint: Fluidic constraint, timing constraint, contamination constraint, and the washing capacity constraint.

III. Algorithm Overview

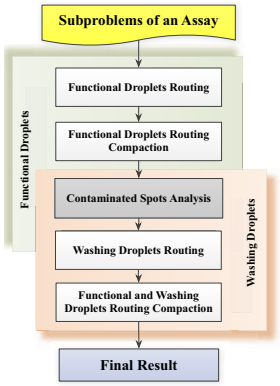


Fig. 3. Design flow of our approach.

Figure 3 shows the overall flow of the presented approach, which consists of five major steps: (1) functional routing, (2) functional droplets routing compaction, (3) cross-contamination spots analysis, (4) washing routing, and (5) functional and washing droplets routing compaction. In functional routing stage, the nets are connected from their source cells to target cells while minimizing the length of path and the number of intersections. During functional droplet routing compaction, a simultaneous compaction algorithm is proposed to optimize the overall execution time. The cross-contamination spots analysis step obtains the coordinates and the desired washing time-interval of each cross-contamination spot. Then, during washing routing, we use the information of cross-contamination spots to determine the washing order and the routing paths of the washing droplets: First, a washing duration relaxation is applied to expand the life time of the cross-contamination spots without violating the specified timing constraint; Second, the washing order decision technique is used to construct the routing paths for washing droplets while considering the washing capacity constraint; Finally, a routing compaction is used to check if all the constraints are satisfied and the completion time of bioassay is minimized.

IV. Functional Droplet Routing

The purpose of functional routing is to connect the list of nets in each subproblem. The routes must satisfy fluidic constraint and timing constraint. During functional routing, if we take the number of intersections into consideration, then washing routing will have less pressure for the cleaning task. Thus, the objective of functional routing is to find a result with minimized lengths and number of intersections.

A. Routing for Objective Function

We developed our routing method based on the classic A* searching algorithm. In order to optimize the objective, the cost function of A* algorithm includes an additional variable used to record whether the current cell has been used by other routes. Routing paths that intersect with other finished routes have higher routing cost. However, it is not always better for a route to choose a longer path for avoiding the intersections, because a longer path may cause violation in timing constraint and block other routes. In the experiments, we set an experimentally determined cost of 3 for choosing a used cell, i.e., when the path has to detour more than 3 cells, it will prefer to choose a used cell.

B. Routing for Constraints

- Fluidic constraint: During droplets moving, the spacing between droplets must be large enough to prevent unexpected mixing error. To obtain an available solution, our routing method sets the surrounding cells of finished routes as used. In this way, the following routes will choose unused cells first and the routing result will not be very congested.
- Timing constraint: The timing constraint gives an upper limit on droplets' moving time along their paths. This constraint is used to ensure the execution time of an assay. Those paths that violate the timing constraint are pruned away in the routing process.

C. Functional Droplet Routing Compaction

The compaction process obtains a scheduling solution for the movement of each droplet. During the movements of the droplets, unexpected droplet mixing must be avoided and the timing constraint must be satisfied. Furthermore, in order to make the bioassay execution fast, the execution time should be minimized. To achieve the above goals, we purpose a simultaneous compaction method. Compared with previous compaction approach [15], a new feature of our method is that the conflicts are resolved in a global manner. In our method, all the paths are compacted simultaneously. Besides, each path has a variable M, which records its margin of compaction. M is updated for each step of the droplet movement. Functional path P_i's margin at current position, M_i, is computed as follows:

$$M_i = T_c - clock - Dis_i \tag{3}$$

where T_c represents the timing constraint, $clock$ represents current clock cycle, and Dis_i represents the routing distance from current position to the target along P_i. Here, we assume droplets move one-cell distance in each clock cycle. M_i is used to determine the routing ordering to resolve routing conflicts.

Algorithm 1 shows the functional routing compaction algorithm, which simultaneously checks the fluidic constraint and timing constraint for each step of droplets' movement. If conflict occurs between two functional droplets, the one with higher M_i value will fall back and wait (Lines 13-18). When a droplet has spent too much time on its way, or has violated the timing constraint, its M_i value will be decreased with higher priority to finish its routing.

Input: Set of functional paths $\mathcal{P} = \{P_1, P_2, \cdots, P_n\}$ and timing constraint T_c
Output: The compacted paths $\mathcal{P}$ with minimized execution time

```
for clock = 0 to T_c do
    if all the paths in P have finished compaction then
        break;
    Move each path in P forward by one step;
    for i = 0 to n do
        if P_i violates the timing constraint then
            Decrease P_i's margin M_i;
            Restart compaction;
        for j = i + 1 to n do
            Calculate Dis_i and Dis_j for P_i and P_j, respectively;
            if P_i and P_j violate fluidic constraint then
                Calculate margins M_i and M_j for P_i and P_j, respectively;
                if M_i > M_j then
                    Path P_i fall back and wait;
                    Restart compaction;
                else
                    Path P_j fall back and wait;
                    Restart compaction;
```

Algorithm 1: Functional Droplet Routing Compaction.

Now we analyze the time complexity of the proposed algorithm. The outer loop counts i from zero to T_c. The inner loops are used to check the fluidic constraint for each pair of droplets. Therefore, the time complexity of inner loops is $O(n^2)$, where n denotes the number of paths in a subproblem. Furthermore, when we solve one conflict of two paths, the algorithm will be restarted. Assume that the number of restarts is n_r. Then the overall time complexity of the algorithm is $O(n_r \times T_c \times n^2)$.

V. Conflict and Capacity Aware Washing Droplet Routing

A. Washing Duration Relaxation

Cross-contamination occurs when two or more functional droplets arrive at the same cell sequentially. To successfully clean the cell at the cross-contamination spot, a washing droplet should arrive at the spot within the time interval between the two sequentially arriving functional droplets. We call this time interval as *washing duration* for each cross-contamination spot, which represents the allowed time interval for the washing operation. Assume all the functional and washing droplets start to move at the initial time 0. Then the initial washing duration after functional routing compaction can be described as follows:

$$T_{washing} = [T_{early}, T_{late}] \tag{4}$$

where T_{early} represents the arrival time of the first functional droplet (e.g., D_1 in Figure 4), and T_{late} represents the arrival time of the second functional droplet (e.g., D_2 in Figure 4). However, the washing droplets may not be able to finish the cleaning task within the designated washing duration. One possible reason is that the cross-contamination spot is too far away from the washing reservior. To solve this problem, the algorithm in [14] seeks to relax the washing duration by delaying the arrival time of the latter functional droplet. However, the delayed functional droplet may violate the timing constraint. In this paper, we present a washing duration relaxation method to guarantee the timing constraint. Let T_{used} be the time used for transferring the second functional droplet from its source cell to target cell. As mentioned above, we assume the droplets move one-cell distance in each clock cycle. Thus, T_{used} equals to the length of the second droplet's routing path. Let T_c be the timing constraint. Then the maximum allowed relaxation time of the cross-contamination spot is $T_{relax} = T_c - T_{used}$. The relaxed washing duration for a cross-contamination spot is:

$$T^{relax}_{washing} = [T_{early}, T_{late} + T_{relax}] \tag{5}$$

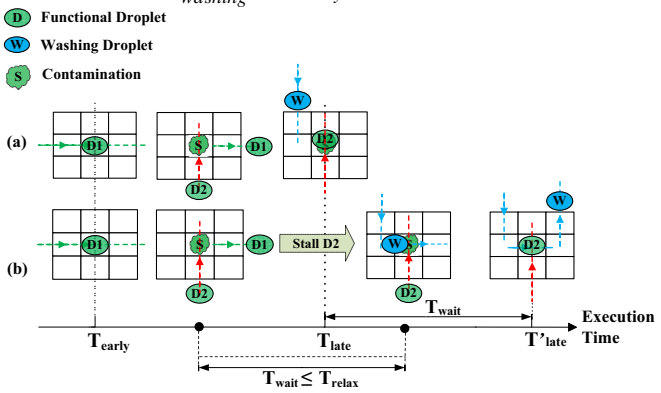


Fig. 4. Two functional droplets cross the same cell, forming cross-contamination spot S: (a) washing droplet fails to clean the cross-contamination spot on time, and (b) by stalling droplet D_2, the residue is successfully washed by W without droplet routing conflicts.

Figure 4 illustrates the washing duration relaxation process. In Figure 4(a), functional droplets D_1 and D_2 arrive at cross-contamination spot S on time T_{early} and T_{late}, respectively. And washing droplet W fails to reach S in the washing duration $T_{washing}$ to finish its cleaning task. So we need to stall D_2 before W arrives at S and let it wash away the residue first. In Figure 4(b), T_{wait} is the time for stalling D_2, which should not exceed T_{relax}. After the adjustment, the new arrival time of D_2 is $T'_{late} > T_{late}$. Moreover, T'_{late} does not exceed $T_{late} + T_{relax}$ because $T_{wait} \leq T_{relax}$, which ensures the timing constraint. Thus, the relaxed washing duration for each cross-contamination spot facilitates the scheduling of functional and washing droplets and avoids the timing violation.

B. Washing Order Decision

As mentioned in Section I, after cleaning a certain number of cross-contamination spots, the washing droplets will become saturated and cannot wash anymore. Such washing capacity constraint requires that washing droplets choose the right washing order to clean the cross-contamination spots in time and utilize their washing capacity as much as possible. This is one of the major challenges of the realistic washing capacity constraint. We propose a washing order decision method to tackle this problem.

Figure 5 shows the washing droplet routing process. One washing droplet is dispensed from the wash reservior. Then the feasible cross-contamination spots are searched in several neighboring columns (e.g., 3) of the biochip array (Figure 5(a)). Here, feasible cross-contamination spots refer to the spots with feasible relaxed washing durations that the washing droplet can reach in time. In Figure 5(b), we obtain two feasible cross-contamination spots as candidate routing destinations. These candidates are chosen according to:

$$Cost_{spot} = \alpha \cdot \frac{L}{L_c} + \beta \cdot \frac{T_{early}}{T_c} \tag{6}$$

where L represents the length of the routing path from the washing droplet's current position to the cross-contamination spot[2], T_{early} means the arriving time of the first functional droplet as defined above, T_c means the timing constraint as defined above, and L_c means the designated length constraint. As

[2] A* routing algorithm is used to compute the routing paths of the washing droplet from its current position to the candidate cross-contamination spots.

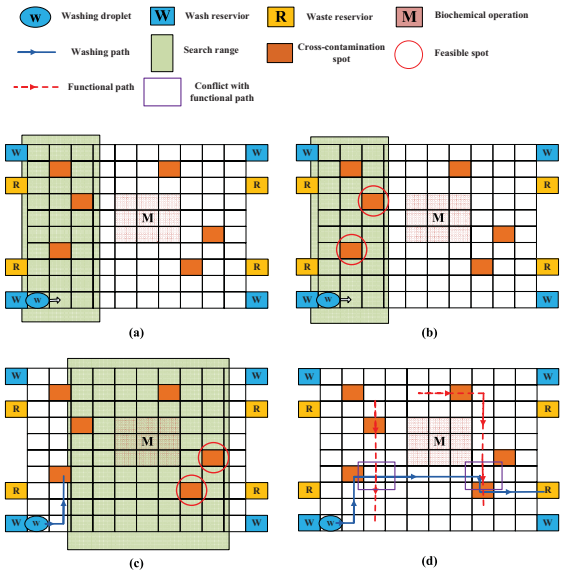


Fig. 5. Washing order decision method for washing droplet routing: (a) washing droplet starts from the source with the searching range initialized, (b) two feasible cross-contamination spots are found satisfying the washing duration, (c) washing droplet moves to the best destination chosen from the feasible cross-contamination spots and a new searching operation starts, and (d) finish the washing path construction when the washing capacity constraint is met until reaching the biochip boundary.

we assume the droplets move one cell at each clock cycle, we set L_c equal to T_c to consider timing optimization. The cross-contamination spot with the minimum cost $Cost_{spot}$ is chosen as the intermediate routing target (see Figure 5(c)). The intrinsic idea of Equation (6) is to choose the cross-contamination spot which is both close to the current washing droplet's position and early in time. It is easy to understand that a close spot helps reduce the path length. Moreover, the smaller T_{early} is, the earlier the contamination happens, and thus the washing droplet does not need to wait for a long time to wash the early generated residues. In this case, after washing the spot, more time is left for the washing droplet to clean the remaining cross-contamination spots. α and β are user-defined parameters. In the experiments, α and β are set to be 2 and 0.5, respectively.

As shown in Figure 5(c), after one cross-contamination spot is determined along with the routing path, a new searching area (denoted as large shaded rectangle) is constructed to find the next set of feasible cross-contamination spots. This time the searching area becomes larger than that in Figure 5(b) because we need to find adequate candidates (e.g., at least 2 candidates). Besides, the conflicts/crossings between the washing path and the existing functional paths are recorded. Such conflicts result in washing capacity consumption for the washing droplet. Thus, we need to subtract the consumption from the washing capacity. The searching and recording process is repeated until the DMFB boundary is reached or the washing capacity is exhausted. Figure 5(d) shows an example of a complete washing path from wash reservior to waste reservior. It has two routing conflicts with existing functional paths, where each conflict possibly consumes one washing capacity. Then a new washing droplet is dispensed from another wash reservior in clockwise order to clean the remaining cross-contamination spots. The process is repeated until all the cross-contamination spots are finished.

Input: List of cross-contamination spots with position information and relaxed washing duration: *ccS pots*
Output: List of routing paths of the washing droplets: *washPaths*

```
1  while ccS pots is not empty do
2      nextS pot = NULL;
3      Set currentS pot to be the next dispensed washing droplet;
4      washedS pots = 0;
5      for ccS pot belongs to next closest searching columns do
6          Add ccS pot to CheckS et;
7          if The number of ccSpot Candidates in CheckS et < 2 then
8              continue;
9          for i = 0 to CheckS et.size do
10             nextS pot = CheckS et[i];
11             Start A* routing to find the paths of the washing droplet;
12             Perform routing path compaction to avoid fluidic constraint
               violation;
13             if The routing result satisfies nextS pot's relaxed washing duration
               then
14                 Add nextS pot to the set of feasibleS pots;
15         if feasibleS pots is empty then
16             continue;
17         nextS pot = Choose the best one from feasibleS pots;
18         Append the routing path from currentS pot to nextS pot to the end of p;
19         Compute the number of consumed washing capacity and accumulate it
           to washedS pots;
20         if washedS pots ≥ wash capacity constraint then
21             break;
22         else
23             Add p to washPaths;
24             currentS pot = nextS pot;
25             ccS pots.delete(nextS pot);
```

Algorithm 2: Washing Droplet Routing Algorithm.

The washing droplet routing algorithm is summarized in Algorithm 2. First, we initialize the washing droplet and prepare for recording its routing path (Lines 2-4). Second, we check the relaxed washing duration constraint of the cross-contamination spots in the searching area, and record the spots that satisfy the constraint along the routing path of the washing droplet (Lines 6-14). In Line 12, a compaction process schedules the washing path with existing functional paths to avoid fluidic constraint violation. Then we choose the best destination from the feasible cross-contamination spots based on Equation (6), record the washing path, append it to the end of washing path list, and delete the chosen cross-contamination spot (Lines 17-25). At the same time, we monitor the washing capacity consumption of the washing droplet, and stop the washing routing when the washing droplet meets the washing capacity constraint (Lines 19-21).

Now we analyze the time complexity of Algorithm 2. We first sort the cross-contamination spots according their column indices. Therefore, to find the feasible cross-contamination spots, we only need to scan the columns sequentially in the designated searching area. Let w and h denote the width and height of the biochip array, respectively. And let D represent the number of cross-contamination spots. The time complexity of sorting and searching for feasible cross-contamination spots is O(D) using bucket sort. The routing paths of the washing droplet are computed using A* routing, where the worst-case time complexity is O($k \times w \times h$). Here k represents the average number of routing paths obtained in each searching area. In the worst case, each washing droplet can only clean one cross-contamination spot in its washing path. I.e., the algorithm will be finished in D rounds. Therefore, the overall time complexity for one subproblem is O($D \times k \times w \times h$).

VI. Experimental Results

We have implemented our practical droplet routing flow in C++ on a 2.40GHz 16-core Intel Xeon Linux workstation with 40GB memory. Only a single thread is used for the experiments. Four commonly used bioassays are tested to verify our approach. Table I shows the details of the benchmarks, where "Size" represents the size of DMFB array, "#Sub" gives the number of subproblems, "#Net" gives the number of nets, "#D_{max}" records the maximum number of droplets within one subproblem, and "#W" denotes the number of wash reservoirs. In the experiments, the washing capacity constraint for each washing droplet is set to be 4.

TABLE I
Statistics of the routing benchmarks.

Circuit	Size	#Sub	#Net	#D_{max}	#W
in-vitro_1	16×16	11	28	5	4
in-vitro_2	14×14	15	34	6	4
protein_1	21×21	64	181	6	4
protein_2	13×13	68	137	6	4

TABLE II
With vs. without washing capacity limit.

Bioassay	#Cont.	Without Capacity Limit					With Capacity Limit				
		#UC	T_{exe} (*ck*)	CPU (s)	#VS	#ER	#UC	T_{exe} (*ck*)	CPU (s)	#VS	#ER
in-vitro_1	8	476	255	0.02	3	37%	491	267	0.03	0	0%
in-vitro_2	11	466	240	0.03	5	46%	512	243	0.03	0	0%
protein_1	38	3068	1509	0.25	5	13%	3054	1507	0.26	0	0%
protein_2	26	1327	900	0.11	15	58%	1376	876	0.12	0	0%
Total	83	5337	2904	0.41	28	34%	5433	2893	0.44	0	0%

In the first experiment, we compute the number of cross-contami- nation spots washed by washing droplets violating the capacity constraint. This experiment verifies the importance of washing capacity constraint and the effectiveness of our method. Table II shows the comparison result of routing with the washing capacity constraint vs. without the capacity constraint. "#Cont." gives the number of cross-contamination spots, "#UC" gives the number of used cells for routing, "T_{exe}" gives the execution time for bioassays (i.e., the number of clock cycles, denoted by *ck*), "CPU" gives the CPU time in seconds (s), "#VS" gives the number of cross-contamination spots washed by washing droplets exceeding the capacity limit, and "#ER" gives the error washing rate, which is calculated by "#VS/#Cont.". The results show that our work is effective with significant improvement, which reduces all the error washing rates to 0. Without the capacity constraint, overall there are 34% invalid washing operations. Using our algorithm, all the washing operations are valid within the capacity limit.

TABLE III
Comparison result between [15] and our algorithm.

Bioassay	[15]				Our Method			
	#Cont.	#UC	T_{exe} (*ck*)	CPU (s)	#Cont.	#UC	T_{exe} (*ck*)	CPU (s)
in-vitro_1	21	351	193	0.58	8	491	267	0.03
in-vitro_2	5	281	191	0.39	11	512	243	0.03
protein_1	82	2213	1394	2.58	38	3054	1507	0.26
protein_2	61	1362	1108	1.49	26	1376	876	0.12
Total	169	4207	2886	5.04	83	5433	2893	0.44

In the second experiment (Table III), we compare our approach with state-of-the-art contamination-aware droplets routing method in [15], which does not consider the washing capacity constraints. Because our approach has an additional washing capacity constraint, the overhead (i.e., number of used cells and the execution time) is reasonable. Besides, the runtime of our method is much faster with about 10× speedup.

In the third experiment, we compare two approaches of constructing the washing droplet paths. The first method finds the washing paths by diagonal searching. That is, the next destination is found for washing droplets in the diagonal direction in the 2-D Microfluidic array. The second method (our presented method) finds the washing paths by horizontal searching. That is, the next destination is found in the horizontal direction. The results in Table IV shows that the horizontal searching has a better performance in CPU time than diagonal direction. This is because the horizontal searching has a smaller searching range in each step and thus is more efficient than the diagonal one. The statistics of the used cells and the execution time of two methods are similar.

TABLE IV
Diagonal searching vs. horizontal searching.

Bioassay	#Cont.	Our Method (Diagonal)					Our Method (Horizontal)				
		#UC	T_{exe} (*ck*)	CPU (s)	#VS	#ER	#UC	T_{exe} (*ck*)	CPU (s)	#VS	#ER
in-vitro_1	8	478	254	0.07	0	0%	491	267	0.03	0	0%
in-vitro_2	11	515	252	0.07	0	0%	512	243	0.03	0	0%
protein_1	38	2925	1451	0.75	0	0%	3054	1507	0.26	0	0%
protein_2	26	1435	1005	0.15	0	0%	1376	876	0.12	0	0%
Total	83	5353	2962	1.04	0	0%	5433	2893	0.44	0	0%

VII. Conclusion

Existing works in droplet routing have oversimplified assumptions on the washing operations and constraints. And simple solutions, such as following each functional droplet immediately with a washing droplet, consume much washing resource and increase the execution time. This paper presents the first practical droplet routing flow, which not only considers the realistic washing capacity constraint, but also resolves the routing conflicts between washing and functional droplets. Effectiveness and efficiency of the presented flow are validated by real-life biochemical applications.

References

[1] K. Chakrabarty and F. Su, "Digital Microfluidic Biochips", *CRC Press*, 2006.
[2] R. B. Fair, A. Khlystov, T. D. Tailor et al., "Chemical and Biological Applications of Digital-Microfluidic Devices", *IEEE Design and Test of Computers*, vol. 24, no. 1, pp. 10-24, 2007.
[3] V. Srinivasan, V. K. Pamula, and R. B. Fair, "An Integrated Digital Microfluidic Lab-on-a-Chip for Clinical Diagnostics on Human Physiological Fluids", *Lab Chip*, pp. 310-315, 2004.
[4] I. Barbulovic-Nad, H. Yang, P. S. Park et al., "Digital Microfluidics for Cell-based Assays", *Lab Chip*, pp. 519-526, 2008.
[5] V. Srinivasan, V. K. Pamula, P. Paik et al., "Protein Stamping for MALDI Mass Spectrometry Using an Electrowetting-based Microfluidic Platform", *Optics East*, 2004, vol. 5591, pp. 26-32.
[6] M. Cho and D. Z. Pan, "A High-performance Droplet Router for Digital Microfluidic Biochips", in *Proc. of ISPD*, 2008, pp. 200-206.
[7] F. Su and K. Chakrabarty, "Unified High-level Synthesis and Module Placement for Defect-tolerant Microfluidic Biochips", in *Proc. of DAC*, 2005, pp. 825-830.
[8] F. Su, W. Hwang, and K. Chakrabarty, "Droplet Routing in the Synthesis of Digital Microfluidic Biochips", in *Proc. of DATE*, 2006, pp. 1-6.
[9] T. Xu and K. Chakrabarty, "Integrated Droplet Routing in the Synthesis of Microfluidic Biochips", in *Proc. of DAC*, 2007, pp. 948-953.
[10] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang, "BioRoute: A Network-Flow-Based Routing Algorithm for the Synthesis of Digital Microfluidic Biochips", *IEEE Trans. on CAD*, vol. 27, no. 11, pp. 1928-1941, 2008.
[11] P.-H. Yuh, S. S. Sapatnekar, C.-L. Yang et al., "A Progressive-ILP-Based Routing Algorithm for the Synthesis of Cross-Referencing Biochips," *IEEE Trans. on CAD*, vol. 28, no. 9, pp. 1295-1306, 2009.
[12] M. Campàs and I. Katakis, "DNA Biochip Arraying, Detection and Amplification Strategies", *TrAC Trends in Analytical Chemistry*, vol. 23, no. 1, pp. 49-62, 2004.
[13] Y. Zhao and K. Chakrabarty, "Cross-contamination Avoidance for Droplet Routing in Digital Microfluidic Biochips," in *Proc. of DATE*, 2009, pp. 1290-1295.
[14] Y. Zhao and K. Chakrabarty, "Synchronization of Washing Operations with Droplet Routing for Cross-contamination Avoidance in Digital Microfluidic Biochips," in *Proc. of DAC*, 2010, pp. 635-640.
[15] T.-W. Huang, C.-H. Lin, and T.-Y. Ho, "A Contamination Aware Droplet Routing Algorithm for the Synthesis of Digital Microfluidic Biochips," *IEEE Trans. on CAD*, vol. 29, no. 11, pp. 1682-1695, 2010.
[16] C. C. Y. Lin and Y.-W. Chang, "Cross-Contamination Aware Design Methodology for Pin-Constrained Digital Microfluidic Biochips," *IEEE Trans. on CAD*, vol. 30, no. 6, pp. 817-828, 2011.
[17] D. Mitra, S. Ghoshal, H. Rahaman et al., "On Residue Removal in Digital Microfluidic Biochips," in *Proc. of Great Lakes Symposium on VLSI*, pp. 1-4, 2011.