A High-Performance Droplet Router for Digital Microfluidic Biochips

Minsik Cho and David Z. Pan
ECE Dept. Univ. of Texas at Austin, Austin, TX 78712
thyeros@cerc.utexas.edu, dpan@ece.utexas.edu

ABSTRACT

In this paper, we propose a high-performance droplet router for digital microfluidic biochip (DMFB) design. Due to recent advancements in bio-MEMS, the design complexity and the scale of a DMFB are expected to explode in near future, thus requiring strong support from CAD as in conventional VLSI design. Among multiple design stages of a DMFB, droplet routing which schedules the movement of each droplet in a time-multiplexed manner is a critical challenge due to high complexity as well as large impacts on performance. Our algorithm first routes a droplet with higher bypassibility which less likely blocks the movement of the others. When multiple droplets form a deadlock, our algorithm resolves it by backing off some droplets for concession. A final compaction step further enhances timing as well as fault-tolerance by tuning each droplet movement greedily. Experimental results on hard benchmarks show that our algorithm achieves over 35x and 20x better routability with comparable timing and fault-tolerance than the popular prioritized A* search [2] and the state-of-the-art network-flow based algorithm [18], respectively.

Categories and Subject Descriptors

B.7.2 [**Hardware, Integrated Circuit**]: Design Aids

General Terms

Algorithms, Design, Performance

Keywords

Synthesis, Routing, Microfluidics, Biochip

1. INTRODUCTION

One of the most advanced technologies to build a biochip is based on microfluidics where micro/nano-liter droplets are controlled or manipulated to perform intended biochemical operations on a *miniatured lab*, so called *lab-on-a-chip* (LOC) [9]. The old generation of microfluidic biochip consists of several micrometer scale components including channels, valves, actuators, sensors, pumps, and so on. Even though this generation shows successful applications like DNA probing, it is unsuitable to build a large and complex biochip, because it uses *continuous* liquid flows, as like continues voltages in analog VLSI design (See Section 2.1 for more details). The new generation of microfluidic biochip has been proposed based on a recent technology breakthrough where the continuous liquid flow is sliced or *digitized* into droplets. Such droplets are manipulated independently by an electric field. This new generation is referred to as a *digital microfluidic biochip* (DMFB).

Due to such a digital nature of a DMFB, any operation on droplets can be accomplished with a set of library operations like VLSI standard library, controlling a droplet by applying a sequence of preprogrammed electric signals. Therefore, a hierarchical cell-based design methodology can be applied to a DMFB. Under this circumstance, we can easily envision that a large scale complex DMFB can be designed as done in VLSI, once strong CAD frameworks are ready.

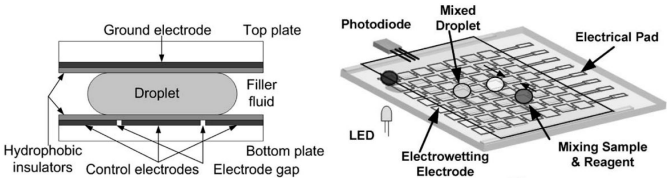
However, CAD research for DMFB design has started very recently. In [12], the first top down methodology for a DMFB is proposed, which mainly consists of architecture-level synthesis and geometry-level synthesis. Geometry-level synthesis can be further divided into module placement and droplet routing. During module placement, the location and time interval of each module are determined to minimize chip area or response time. Since different modules can be on the same spot during different time intervals based on reconfigurability (See Section 2.1), module placement is equivalent to a 3D packing problem [14, 17]. Meanwhile, in droplet routing, the path of each droplet is found to transport it without any unexpected mixture under design requirements. Similarly to module placement, a spot can be used to transport different droplets during different time intervals (simply in a time-multiplexed manner), which increases the complexity of routing. The most critical goal of droplet routing is routability as in VLSI [2], while satisfying timing constraint and maximizing fault-tolerance. More discussion on prior papers to achieve this goal is in Section 2.2.

In this paper, we propose a high-performance droplet router for a digital microfluidic biochip (DMFB). Our approach is mainly based on two ideas, *bypassibility* and *concession*. These two ideas provide higher quality solutions than [2,18]. The major contributions of this paper include the following.

- We propose a simple yet effective metric, bypassibility to estimate the degradation of routability after a droplet is routed. This maximizes the number of routed droplets, and narrows down the problem size until multiple droplets under a deadlock are identified.

- We introduce the concept of a concession zone where some droplet may migrate to break a deadlock between

ISPD'08, April 13–16, 2008, Portland, Oregon, USA.



(a) EWOD-based basic unit cell (b) top view of microfluidic array

Figure 1: The schematic view of digital microfluidic biochips for colorimetric assays [2].

droplets. We route earlier a droplet with longer distance to any of concession zones, as it is harder to be routed in a later stage of routing.

- We propose 2D routing for the droplet chosen by bypassibility analysis to reduce runtime. If only one droplet chosen by bypassibility is routed while the others are frozen, this can be solved in a compact 2D plane rather than in a huge 3D plane where the third axis represents time.
- We further propose routing compaction which is performed to meet the multi-objectives like routability, timing, and fault-tolerance.

The rest of this paper is organized as follows. Section 2 presents preliminaries. Especially, routing problems in a DMFB and a VLSI circuit are compared in Section 2.2 to help readers with VLSI background. The droplet routing in a DMFB is defined in Section 3, and Section 4 presents our proposed algorithm for DMFB routing. Experimental results are discussed in Section 5, followed by the conclusion in Section 6.

2. PRELIMINARIES

2.1 Digital Microfluidic Biochip

The first generation of biochips is based on a continuous-flow system where liquid flows through microfabricated channels continuously using electrokinetic-based micro-actuators. Although a continuous-flow biochip is widely used for simple yet well-defined biochemical operations, it is inherently unsuitable for large scale complex biochip design due to the following reasons: (a) permanently microfabricated channels limit the reconfigurability for both applications and fault-tolerance, (b) inevitable shear flow around micro-actuators and diffusion on channels increase the possibility of sample contamination.

To overcome the above drawbacks, a digital microfluidic biochip (DMFB) is devised where liquid is discretized or *digitized* into independently controllable droplets ($\ll 1\mu l$), and each droplet is moved or manipulated on a substrate according to a preprogrammed schedule. Such digitization and programmability enable to design a large scale and complex DMFB by allowing a hierarchical and cell-based design methodology as in modern VLSI design. They also provide reconfigurability for various biochemical applications and enhanced fault-tolerance. Although there are multiple technologies to control a droplet [5, 7, 8, 11], we mainly consider a electrowetting-on-dielectric (EWOD)-based DMFB [4] in this paper.

Fig. 1 shows the schematic view of a EWOD-based DMFB. As shown in Fig. 1 (a), a unit cell consists of two parallel glass plates which sandwich biochemical droplets. While the top glass plate has a ground electronode only, the bottom has a regularly patterned array of individually controllable electronodes. The EWOD effect to drive the droplet occurs when a control voltage is applied to the controllable electronodes. Therefore, by controlling a voltage to each electronode in the bottom glass plate with VLSI circuitries, we can have a fine control over droplet movement. Fig. 1 (b) illustrates the overview of a DMFB. Due to individual controllability of each electronode (thus, each droplet), we can manipulate multiple droplets simultaneously and move them parallelly to anywhere in the chip to perform preprogrammed biochemical operations. Therefore, any operation on droplets can happen anywhere in the chip which provides the reconfigurability of a DMFB. For example, when multiple droplets perform operations like mixing, they need some real estate of the chip for fixed amount of time. After the operation time elapses, these droplets can go to somewhere else for their next scheduled operations, after releasing the taken area for the other droplets to perform different operations such as diluting.

This reconfigurability raises two important physical challenges: (a) where and when to perform which biochemical operations, (b) how to move droplets avoiding undesired mixtures and blockages. The first problem is DMFB placement which is essentially 3D packing [13, 17], and the second problem is droplet routing [2, 15, 16] which will be further discussed in Section 2.2.

2.2 Routing for Digital Microfluidic Biochip

The goal of droplet routing in a DMFB is to find an efficient schedule for each droplet which transports it from its source to target locations, while satisfying all constraints. This sounds similar to VLSI routing where wires need to be connected under design rules, but the reconfigurability of a DMFB makes fundamental differences from VLSI routing in the following aspects:

- DMFB routing allows multiple droplets to share the same spot during different time intervals [2, 15, 18] like time division multiplexing, while VLSI routing makes one single wire permanently and exclusively occupy the routing area.
- DMFB routing allows a droplet to stall/stand-by at a spot, if needed.
- VLSI routing requires 2D spacing by design rules, but DMFB routing needs 3D spacing by dynamic/static fluidic constraints.

A highly equivalent problem to DMFB droplet routing has been extensively studied in robotics as mobile robot motion planning, and solved by prioritized A* search [2]. In [1, 10], the mobile robot motion planning is shown to be NP-hard, and an integer linear programming approach is proposed. Recent research efforts in DMFB design from VLSI community attack the problem using various heuristics such as internet routing protocol (Open Shortest Path First) or pattern selection [6, 15]. However, these approaches suffer from initialization overhead to build either routing tables or to discover a set of feasible routing patterns. Also, as a DMFB keeps reconfigured, this overhead occurs repeatedly, involving large storage overhead. In [18], a novel network-flow based algorithm with negotiation is proposed for DMFB droplet routing, showing better performance than [2, 15]. However, the network-flow formulation is significantly bottlenecked by the distribution of blockages. If a width of

Table 1: The notations in this paper.

d_i	droplet i
S_i	source location of $d_i = (x_i^s, y_i^s)$
T_i	target location of $d_i = (x_i^t, y_i^t)$
AT_i	arrival time of d_i at T_i
R_i^t	shadowed region of d_i at (x_i^t, y_i^t) at time t_i^l $= \{(x,y,t) \mid \lvert x_i^l - x\rvert \leq 1, \lvert y_i^l - y\rvert \leq 1, \lvert t_i^l - t\rvert \leq 1\}$

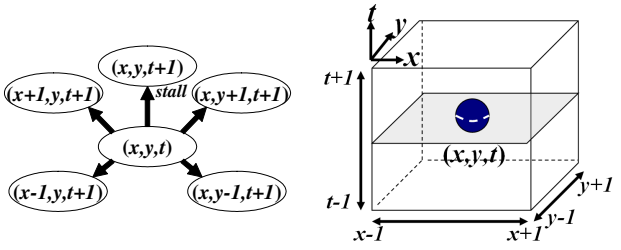
channel between blockages is less than 3 unit cells, the channel will not be utilized in the network-flow formulation.

3. PROBLEM FORMULATION

In this section, we show a routing model and constraints, and propose a problem formulation. Since the problem can be abstracted as transporting each droplet from its source to target, we cast droplet routing problem into graph search as done in VLSI routing. In a DMFB, a droplet moves on a 2D plane toward its target. Hence, finding paths on 2D graph may be appropriate. However, as resource sharing in a time-multiplexed fashion is allowed in a DMFB, we can model it as a 3D graph where z axis is for time, which enables to optimize geometric paths and temporal schedules simultaneously. Fig. 2 (a) shows our graph where a droplet at (x, y, t) can move to one of five nodes at $t+1$. This graph is not only directed, but also acyclic due to the causality of time multiplexing, differently from the graph in VLSI routing [3].

Since all the droplets are moving in parallel, there can be unwanted mixtures if a keep-off distance/spacing is not observed. This imposes static and dynamic fluidic constraints in Fig. 2 (b) which requires that there should not be any other droplets in a cube centered by one droplet. Additionally, defective or reserved unit cells can be blockages for routing [14]. Sometimes, droplets may have a required arrival time to prevent spoilage, which becomes a timing constraint. Lastly, it is desirable to minimize the number of unit cells which are used at least once by droplets. Since a unit cell of a DMFB can be defective due to manufacturing or environmental issues, using less number of nodes (each node corresponds to one unit cell) can be beneficial for robustness. Finally, we can define the problem as follows using the notations in Table 1:

Given a maximum arrival time RT, for each droplet $d_i \in \{d_1, d_2, ..., d_n\}$, transport d_i from S_i to T_i through an acyclic graph $G = (V, E)$ with blockages such that



(a) our graph for droplet routing models geometric paths as well as temporal schedules simultaneously.

(b) dynamic and static fluidic constraints are to prevent unexpected mixtures of droplets during movement.

Figure 2: Graph model and fluidic constraints for digital microfluidic biochip design.

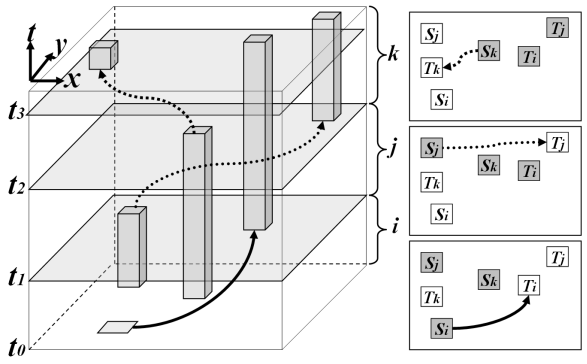


Figure 3: Each droplet is routed during different time intervals to reduce A* search complexity.

d_i is the only one in R_i^t $(t \geq 0)$ for any droplet routing, while satisfying $AT_i \leq RT$ and minimizing the total number of unit cells in use.

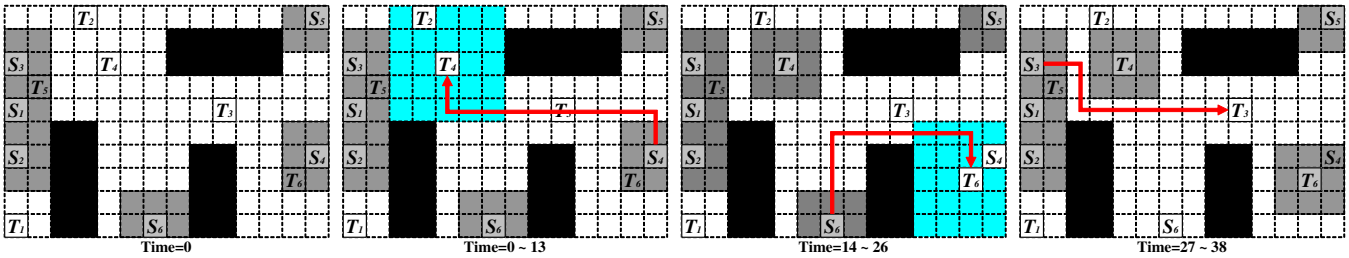
As an efficient solution to this NP-hard problem, we propose a strategy inspired by *k-coloring* [3], where all the nodes in a graph should be colored differently from their connected nodes using k colors. They first take off a node with less than k edges from the graph, as it is guaranteed to be colored differently from its neighbors (at most $k-1$ colors will be used for the neighbor nodes). By removing such nodes repeatedly, eventually the graph is reduced to the level where no node can be removed, which implies a hard part of the problem is identified. Then, a complex approach can be applied to attack the hard part which is significantly smaller than the original graph. We use bypassibility analysis to reduce the problem size, and concession to solve a hard part of the problem as in Section 4.

4. ALGORITHM

In this section, we propose our algorithm for droplet routing in a DMFB. The key ideas behind our approach are:

- If T_i happens to be in a highly sparse region, it may not be hard for the unrouted droplets to bypass the blockages induced by routing d_i, implying high bypassibility of d_i. This motivates us to route d_i first.
- In case more than two droplets are in a deadlock, we need to back some droplets off to provide other droplets with free paths. This is done based on the distances to concession zones which will be explained in Section 4.2.
- We route each droplet chosen by bypassibility during different time intervals to improve runtime, which effectively converts 3D routing into 2D routing. As a result, this approach reduces runtime overhead.

Our overall algorithm is presented in Algorithm 1. First, we repeat picking a routable droplet with the maximum bypassibility and making it routed in line 2, which continuously narrows down the problem size as in Section 4.1. When no droplet can be routed as in line 3, it means there is a deadlock between droplets and we find a hard part of the problem. Hence, we apply an algorithm with concession to resolve the deadlock in line 4, which is in Section 4.2. Then, we continue to route based on bypassibility in line 2. As a



(a) an example routing problem with $d1$–d_6 with blockages. (b) d_4 is routed due to full bypassibility. (c) after T_6 is freed up, d_6 has the most bypassibility. (d) d_3 is the only routable one, despite no bypassibility.

(e) d_2 is routed due to the longest distance to the concession zone. (f) d_1 migrates to the concession zone first to avoid d_2. (g) d_5 is the only unrouted with half routability. (h) the timing requirement (20) is met after compaction.

blockage **source of net *i*** **target of net *i*** **shadowed cell** **impacted cell** **concession cell** **droplet stalling** **droplet moving**

Figure 4: This example describes the proposed droplet routing algorithm. After the first three routings, (b)-(d) are done by Algorithm 2 (Routing-Bypassibility). Then, no droplet can be routed in a 2D plane due to a deadlock between d_1 and d_2. Thus, as in Algorithm 1, (e) and (f) are done in a 3D plane by Algorithm 3 (Routing-Concession) to resolve the deadlock. After the resolution, (g) is done in 2D again by Algorithm 2, followed by the compaction in (h) using Algorithm 4.

Algorithm 1 Overall Algorithm

Require: A set of all droplets D, a routing graph G, a timing constraint RT

Ensure: $D_u \leftarrow D, T_b \leftarrow 0, T_c \leftarrow 0$

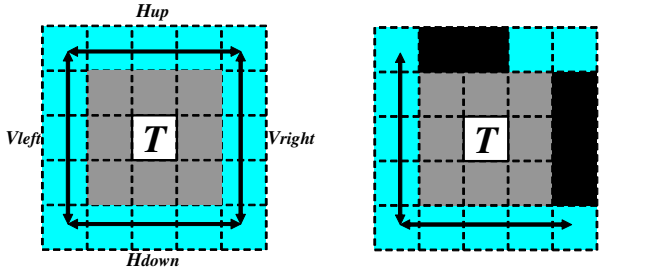
1: **repeat**
2: T_b = Routing-Bypassibility$(D_u, G, max(T_b, T_c))$
3: **if** T_b is not increased **then**
4: $T_c = max$(Routing-Concession$(D_u, G, T_b), T_c)$
5: **end if**
6: **until** No droplet routed
7: Routing-Compaction(D_u, D, G, RT)

final step in line 7, we compact the routing solution greedily to enhance multiple design objectives as in Section 4.3.

While routing based on bypassibility, we move only one droplet while freezing the others, which can be done in a 2D plane rather than a 3D plane. Fig. 3 shows an example of routing three droplets d_i, d_j, and d_k. Until routing d_i is completed (until t_1), d_j and d_k are frozen at S_j and S_k respectively. And, from t_1, T_i becomes a blockage for d_j and d_k. In the same fashion, d_j is routed while d_k is frozen. In this way, we can find a path in a 2D plane, then map the path to a 3D plane as shown in Fig. 3. For this, we need to keep track of the last time when a droplet routing is completed such as t_1, t_2, and t_3 in Fig. 3 using T_b and T_c in Algorithm 1.

4.1 Routing by Bypassibility

Once a droplet d_i is routed (moved to T_i), it stays at T_i, permanently blocking shadowed regions $\{R_i^t \mid t \geq AT_i\}$.



(a) A 5x5 window is considered to evaluate the bypassibility. Four bypasses are shown right out of the shadowed regions. (b) This example has full bypassibility, as there exist at least one vertical and one horizontal bypasses.

Figure 5: The bypassibility is based on whether there exist bypasses for the unrouted droplets.

Therefore, if T_i happens to be in a highly congested region, it can have negative impacts on the rest of unrouted droplets in terms of bypassibility. In this subsection, we propose a way to quantify the bypassibility of d_i, which depends on whether there will be any bypass for the unrouted droplets after d_i is routed. Fig. 5 (a) shows four possible bypasses right out of the shadowed region (which is to keep fluidic constraints), $H_{up}, H_{down}, V_{left}$, and V_{right} within a 5X5 window centered by the target location T. Then, depending on whether these bypasses are blocked or not, we can divide all the possibilities into three classes based on Table 2:

- **Full bypassibility:** This allows both horizontal and vertical bypasses.

- **Half bypassibility:** This allows only either horizontal or vertical bypass.
- **No bypassibility:** This do not allow any bypass.

Table 2: Bypassibility analysis table.

Direction		Full								Half						No
H	H_{up}	o	x	o	o	o	o	o	x	x	o	x	o	x	x	x	x
	H_{down}	o	o	x	o	o	x	x	o	o	o	x	x	o	x	x	x
V	V_{left}	o	o	o	x	o	x	o	x	o	x	o	x	x	o	x	x
	V_{right}	o	o	o	o	x	o	x	o	x	x	o	x	x	x	o	x

Note that it is *not* required to have both H_{up} and H_{down} unblocked to have horizontal bypassibility, as either bypass can be shared by multiple droplets in a time-multiplexed manner (also the same for the vertical case). The example in Fig. 5 (b) has full bypassibility as (a), in spite of blocked or shadowed regions (H_{up} and V_{right} are blocked), as it still has one vertical and one horizontal bypasses. Therefore, if a droplet with full bypassibility is routed first, it will not affect overall chip routability, because the other droplets can bypass vertically or horizontally in a time-multiplexed manner.

As shown in Algorithm 2, we find a routable droplet d_i with the best bypassibility and route it, then update the routing base time (T_b) by returning $AT_i + 1$. The next droplet will stall until T_b to accomplish fast 2D routing. If there is a tie in terms of bypassibility, we route a shorter one first. After d_i is routed, we need to dynamically update the bypassibilities of all the unrouted droplets, as the shadowed region (which works as blockages) around S_i disappears but new blockages appear around T_i. Note that bypassibility update can be done incrementally using a bucket list.

Consider the example in Fig. 4 where $D = \{d_1, d_2, ..., d_6\}$ are to be routed. While T_1, T_5, and T_6 are inaccessible due blockages or shadows by droplets, T_2, T_3, and T_4 are accessible. To decide the droplet to be routed first, we measure bypassibilities as in Fig. 6 which shows that T_4 has full bypassibility. After d_4 is routed from S_4 to T_4 as in Fig. 4 (b), we need to update bypassiblities of all the unrouted droplets. Then, as T_6 becomes accessible (S_4 is released), d_6 turns out to have full bypassibility. Thus, d_6 is routed after waiting at S_6 until $t = 14$. In the same fashion, routing d_3 follows as shown in Fig. 4 (d).

4.2 Routing with Concession

For a complex DMFB, a naive sequential routing of droplets can cause failure due to a deadlock between droplets. Consider the situation in Fig. 4 (e) where d_1, d_2, and d_5 remain

Algorithm 2 Routing-Bypassibility

Require: A set of unrouted droplets D_u, a routing graph G, a routing base time T_b
1: $S \leftarrow$ sort D_u in desc. order of bypassibility
2: **for** each $d_i \in S$ **do**
3: A path $P \leftarrow$ 2D min-cost path for d_i after T_b stalling
4: **if** $P \neq \emptyset$ **then**
5: Make d_i routed with P
6: $D_u \leftarrow D_u \setminus \{d_i\}$
7: **return** $AT_i + 1$
8: **end if**
9: **end for**
10: **return** T_b

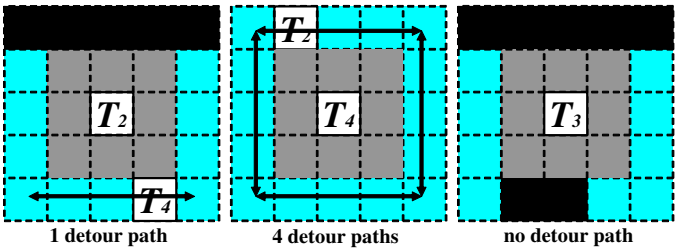


Figure 6: This example shows bypassibility analysis of Fig. 4 (a) where d_4, d_2, and d_3 have half (horizontal), full, and no bypassibility, respectively.

Algorithm 3 Routing-Concession

Require: A set of unrouted droplets D_n, a routing graph G, a routing base time T_b
1: $S \leftarrow$ sort D_u in desc. order of dist. to concession zone
2: **for** each $d_i \in S$ **do**
3: A path $P \leftarrow$ 3D min-cost path for d_i after $T_b + \alpha_i$ stalling
4: **if** $P \neq \emptyset$ **then**
5: Make d_i routed with P
6: $D_u \leftarrow D_u \setminus \{d_i\}$
7: **return** $AT_i + 1$
8: **end if**
9: **end for**
10: **return** T_b

unrouted. Since d_1 and d_2 block the ways to T_2 and T_1 respectively, they form a deadlock. Hence, 2D routing by Algorithm 2 or A* search [2] is ended up with failure, and 3D routing may fail too for complex cases. According to our experiments on Fig. 4 (e), routing either d_1 or d_2 in a 2D or a 3D plane without special consideration (which will be our concession) will cause failure eventually. Therefore, it would be desirable to move d_1 and d_2 simultaneously, but any parallel routing approach will increase computational complexity significantly.

An only sequential solution for Fig. 4 (e) is to make d_1 back off and wait in some empty space, so called *concession zone* for sufficient amount of time until d_2 passes by. The concession zone is defined by any unoccupied continuous space in the chip which is larger than a 3x1 window. Hence, we first identify all the concession zones, and compute the shortest distances from all the unrouted droplets to any nearby concession zones. Then, we route a droplet with the longest distance before the others, as it is harder for such a droplet to migrate and wait in a concession zone. Regarding the example in Fig. 4 (e) and (f), we route d_2 before d_1, as d_1 can migrate to a concession zone easily and wait there until the path taken by d_2 becomes available. To make such interaction between two droplets feasible, we stall the departure of a droplet like d_2 by some additional amount of time, α_i in Algorithm 3, which can be computed as follows:

$$\alpha_i = \sum_{j \in B_i \bigcap D_u} |x_j^s - x_j^t| + |y_j^s - y_j^t|$$

where B_i is a set of droplets whose source locations are inside the bounding box of d_i. Assume the case $\alpha_2 = 0$ for Fig. 4 (e) and (f), then at $t = 41$, d_2 is one grid above S_2 toward T_2, and d_1 is one grid right of S_1, which violates fluidic

Algorithm 4 Routing-Compaction

Require: A set of unrouted droplets D_u, A set of all droplets D, a routing graph G, a timing constraint RT

```
 1: for each d_i ∈ D_n do
 2:    AT_i ← ∞
 3: end for
 4: repeat
 5:    S ← sort D in desc. order of AT_*
 6:    for each d_i ∈ S do
 7:       if RT < max {AT_i | ∀i} then
 8:          A path P ← 3D min-cost path for d_i for timing
 9:          if P ≠ ∅ and AT_i will improve then
10:             Make d_i routed with P
11:          end if
12:       else
13:          A path P ← 3D min-cost path for d_i for fault-
             tolerance
14:          if AT_i will be ≤ RT then
15:             Make d_i routed with P
16:          end if
17:       end if
18:    end for
19: until no improvement or maximum iteration
```

constraints. If we set $\alpha_2 = 5$ due to $B_2 \bigcap D_u = \{d_1\}$, d_2 first stalls for 5 cycles which is enough for d_1 to escape from the shadowed region by d_2 and reach the concession zone safely. After d_1 waits until d_2 passes by, it returns to S_1 to head for T_1. Note that this is the only available path for d_1 to go to T_1 at this moment, thus any min-cost path algorithm should be able to find this path including stalling in the concession zone. As in Algorithm 1, d_1 and d_2 start moving at $t = 39$ when the last successful routing based on bypassibility analysis (Routing-Bypassibility) occurred. As soon as d_1 is routed, the path from S_5 to T_5 becomes available. Thus, d_5 can be routed by Routing-Bypassibility from $max(\ AT_1{+}1,\ AT_2{+}1\ ) = 56$.

4.3 Solution Compaction

After the procedures in Section 4.1 and 4.2, all the droplets including any unrouted one are rerouted greedily to compact the solution. As the procedure in Section 4.1 allows only one droplet routing during a certain time interval, and the one in Section 4.2 intentionally stalls the departure of a droplet to enhance routability, the routing resources are under low utilization, creating a large number of timing violations. Therefore, by rerouting each droplet in a greedy manner, we can increase the resource utilization, satisfy timing constraints, and improve fault-tolerance without hurting routability. Fig. 4 (h) shows that the routing solution after the compaction is completed with timing constraint 20. The latest arrival time is reduced from 72 to 19, as the routing path for each droplet is optimized to meet timing. During this compaction, a droplet d_i with larger AT_i is rerouted first. This compaction is repeated until there is no improvement or maximum iteration is reached as in Algorithm 4.

In detail, Algorithm 4 shows two different phases, the first for timing (from lines 7 to 11) and the second for fault-tolerance (from lines 13 to 16). Until a timing constraint is satisfied, we find a min-cost path where a cost is purely the distance. Once the timing constraint is met, we utilize the slack of each droplet to enhance fault-tolerance by finding a different min-cost path where passing a unit cell already in use by others is encouraged. Therefore, fault-tolerance will be pursued only if the timing constraint is satisfied. Compare the path of d_5 in Fig. 4 (g) with the one in Fig. 4 (h). In Fig. 4 (h), d_5 passes by the center of the design (around T_3) to minimize the number of unit cells in use to increase fault-tolerance at a cost of larger AT_5 (still ≤ 20).

5. EXPERIMENTAL RESULTS

We implement the proposed droplet routing algorithm for digital microfluidic biochips in C++, and perform all the experiments on an Intel 2.6 GHz 32bit Linux machine with 4GB RAM. Since the benchmark suite for droplet routing in [15, 18] has only two fairly small/simple cases, we randomly generate 30 hard test designs with various potions of blockages to demonstrate the performance of our algorithm. In detail, for a given design size, the number of droplets are the same as the length of the longer side of the design. Then, multiple blockages are randomly generated and placed until the total area of blockages exceeds the given threshold. A source of each droplet is randomly placed on the boundary, while its target is randomly located at any place in the design. To prevent any trivially short case, the Manhattan distance in a 2D plane between the source and target is forced to be longer than 50% of the length of the longer side of the design. We set a timing constraint of all the test designs as 100 time unit.

For comparison purpose, we implement the widely used prioritized A* search [2].We also obtain the simulation results on our test designs from the author of the network-flow based algorithm [18] which is shown to be superior to the prioritized A* search and the two-stage algorithm [15]. We make the same assumptions as in [15,18] for fair comparison.

Table 3 shows the overall comparison results. First, our approach shows significantly better routability by completing 27 test cases out of 30 (90.0%), while the priority A* search and the network-flow approach complete 8 (26.7%) and 12 (40%), respectively. In terms of the number of failures, our approach shows 35x and 20x better routability. This result is consistent with that in [18] in a sense that the network-flow based algorithm is superior to the prioritized A* search. Overall, our algorithm yields stronger routability on harder/larger test designs.

Table 3 also reveals the effectiveness of the proposed bypassibility analysis. We find that 752 out of 864 droplets (87%) can be routed by compaction and bypassibility analysis only (no concession), which is shown to be as powerful as the sophisticated network-flow based algorithm for some cases. Regarding test17, the number of droplets routed by simply bypassibility analysis is more than that by the network-flow based algorithm. Our bypassibility-only based routing works as well as the network-flow based algorithm for about 40% of test designs (these test designs are in bold).

6. CONCLUSION

Digital microfluidic biochip design is expected to be in larger scale with higher complexity shortly due to its various applications and high efficiency. In order to cope with droplet routing automation, one of the key steps in digital microfluidic biochip design, we propose a high-performance droplet router with timing and fault-tolerance taken into account. Experiments demonstrate that our algorithm works significantly better than the widely used prioritized A* search and the state-of-the-art network-flow based algorithm.

Table 3: Comparison between prioritized A* search, network-flow based algorithm, and our algorithm.

test designs				Prioritized A* [2]			Network-flow [18]			Our algorithm			
name	droplets	size	blockage area	failure[a]	la.time[b]	u.cell[c]	failure[a]	la.time[b]	u.cell[c]	failure[a]	la.time[b]	u.celll[c]	d.bypass[d]
test 1	12	12x12	8 (5.6%)	0	37	66	2	n/a	n/a	0	100	67	7
test 2	12	12x12	9 (6.2%)	4	n/a	n/a	7	n/a	n/a	1	n/a	n/a	**8**
test 3	12	12x12	11 (7.6%)	4	n/a	n/a	6	n/a	n/a	1	n/a	n/a	3
test 4	12	12x12	11 (7.6%)	3	n/a	n/a	5	n/a	n/a	0	70	64	2
test 5	16	16x16	17 (6.6%)	0	28	108	2	n/a	n/a	0	78	118	**14**
test 6	16	16x16	14 (5.5%)	0	42	116	0	44	132	0	55	119	14
test 7	16	16x16	27 (10.5%)	0	33	104	3	n/a	n/a	0	89	113	9
test 8	16	16x16	26 (10.2%)	2	n/a	n/a	0	47	129	0	41	94	15
test 9	16	16x16	39 (15.2%)	4	n/a	n/a	3	n/a	n/a	1	n/a	n/a	9
test10	16	16x16	39 (15.2%)	4	n/a	n/a	2	n/a	n/a	0	77	110	9
test11	24	24x24	64 (11.1%)	0	62	252	0	100	264	0	47	249	**24**
test12	24	24x24	58 (10.1%)	3	n/a	n/a	0	80	242	0	52	219	22
test13	24	24x24	89 (15.5%)	0	60	241	2	n/a	n/a	0	52	247	19
test14	24	24x24	91 (15.8%)	3	n/a	n/a	2	n/a	n/a	0	57	234	19
test15	24	24x24	119 (20.7%)	0	63	246	0	74	233	0	83	230	17
test16	24	24x24	117 (20.3%)	4	n/a	n/a	3	n/a	n/a	0	63	223	19
test17	32	32x32	205 (20.0%)	9	n/a	n/a	2	n/a	n/a	0	68	394	**31**
test18	32	32x32	205 (20.0%)	4	n/a	n/a	0	88	408	0	91	403	**32**
test19	32	32x32	260 (25.4%)	0	70	402	2	n/a	n/a	0	90	371	**32**
test20	32	32x32	259 (25.3%)	3	n/a	n/a	0	91	382	0	99	393	24
test21	32	32x32	257 (25.1%)	8	n/a	n/a	2	n/a	n/a	0	76	389	22
test22	32	32x32	269 (26.3%)	5	n/a	n/a	4	n/a	n/a	0	85	393	27
test23	48	48x48	499 (21.7%)	6	n/a	n/a	0	100	681	0	78	738	**48**
test24	48	48x48	492 (21.4%)	8	n/a	n/a	0	99	737	0	94	807	**48**
test25	48	48x48	601 (26.1%)	5	n/a	n/a	0	100	729	0	91	792	**48**
test26	48	48x48	604 (26.2%)	3	n/a	n/a	0	99	709	0	88	798	**48**
test27	48	48x48	698 (30.3%)	4	n/a	n/a	0	100	770	0	99	762	47
test28	48	48x48	692 (30.0%)	5	n/a	n/a	4	n/a	n/a	0	99	808	**48**
test29	48	48x48	816 (35.4%)	7	n/a	n/a	6	n/a	n/a	0	98	733	**46**
test30	48	48x48	824 (35.8%)	8	n/a	n/a	4	n/a	n/a	0	88	751	41
total	864			106			61			3			752

[a] the number of failed droplets (unable to find a valid routing path or satisfy timing constraint).
[b] latest arrival time of droplets.
[c] total number of unit cells used for routing.
[d] the number of droplet routed based bypassibility and compaction using Algorithm 2 and 4 only.

7. ACKNOWLEDGMENT

The authors would like to thank Ping-Hung Yuh, Prof. Chia-Lin Yang, and Prof. Yao-Wen Chang from National Taiwan University for providing experimental results of the network-flow based algorithm on our test designs.

8. REFERENCES

[1] S. Akella and S. Hutchinson. Coordinating the motions of multiple robots with specified trajectories. In *Proc. Int. Conf. Robotics and Automation*, 2002.

[2] K. F. *Böhringer*. Modeling and Controlling Parallel Tasks in Droplet-based Microfluidic Systems. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 25:329–339, Feb 2006.

[3] G. Chaitin. Register allocation and spilling via graph coloring. *SIGPLAN Note.*, 39(4):66–74, 2004.

[4] S. K. Cho, S. K. Fan, H. Moon, and C. J. Kim. Toward digital microfluidic circuits: Creating, transporting, cutting and merging liquid droplets by electrowetting-based actuation. In *Proc. Micro Electro Mechanical Systems (MEMS) Conf.*, Jan 2002.

[5] B. S. Gallardo, V. K. Gupta, F. D. Eagerton, L. I. Jong, V. S. Craig, R. R. Shah, and N. L. Abbott. Electrochemical principles for active control of liquids on submillimeter scales. *Science*, 283:57–60, Jan 1999.

[6] E. J. Griffith, S. Akella, and M. K. Goldberg. Performance characterization of a reconfigurable planar array digital microfluidic system. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 25(2):340–352, Feb. 2006.

[7] K. Ichimura, S. Oh, and M. Nakagawa. Light-driven motion of liquids on a photoresponsive surface. *Science*, 288:1624–1626, Jun 2000.

[8] T. B. Jones, M. Gunji, M. Washizu, and M. J. Feldman. Dielectrophoretic liquid actuation and nanodroplet formation. *Journal of Applied Physics*, 89:1441–1448, Jan 2001.

[9] T. Mukherjee. Design Automation Issues for Biofluidic Microchips. In *Proc. Int. Conf. on Computer Aided Design*, Nov 2005.

[10] J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. In *Proc. Workshop Algorithmic Foundations Robotics*, 2002.

[11] T. S. Sammarco and M. A. Burns. Thermocapillary pumping of discrete droplets in microfabricated analysis devices. *AIChe Journal*, 45:350–366, 1999.

[12] F. Su and K. Chakrabarty. Architectural-level synthesis of digital microfluidics-based biochips. In *Proc. Int. Conf. on Computer Aided Design*, Nov 2004.

[13] F. Su and K. Chakrabarty. Module Placement for Fault-Tolerant Microfluidics-Based Biochips. *ACM Trans. on Design Automation of Electronics Systems*, 11:682–710, 2006.

[14] F. Su, K. Chakrabarty, and R. B. Fair. Microfluidics-Based Biochips: Technology Issues, Implementation Platforms, and Design-Automation Challenges. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 25:211–223, Feb 2006.

[15] F. Su, W. Hwang, and K. Chakrabarty. Droplet Routing in the Synthesis of Digital Microfluidic Biochips. In *Proc. Design, Automation and Test in Eurpoe*, 2006.

[16] T. Xu and K. Chakrabarty. Integrated Droplet Routing in the Synthesis of Microfluidic Biochips. In *Proc. Design Automation Conf.*, Jun 2007.

[17] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang. Placement of Digital Microfluidic Biochips using the T-tree Formulation. In *Proc. Design Automation Conf.*, Jul 2006.

[18] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang. BioRoute: A Network Flow Based Routing Algorithm for Digital Microfluidic Biochips (to appear). In *Proc. Int. Conf. on Computer Aided Design*, 2007.