2014 IEEE International Conference on Systems, Man, and Cybernetics
October 5-8, 2014, San Diego, CA, USA

A Genetic Algorithm for the Routing of Droplets in DMFB: Preliminary Results

Julio Juárez*, Carlos Brizuela†, Israel Martínez‡, Karen Velázquez§ and Ema Lozano¶
Department of Computer Science,
Center for Scientific Research and Higher Education of Ensenada (CICESE),
Ensenada, Baja California 22860, Mexico
*jjuarez@cicese.edu.mx, †cbrizuel@cicese.mx, ‡israelmp@cicese.mx,
§anavelaz@cicese.edu.mx, ¶elozano@cicese.edu.mx

***Abstract*— Digital Microfluidic Biochips (DMFB) are revolutionizing clinical diagnostics and most of the biochemical processes that take place in a laboratory, mainly because of its capability for automation, low cost, portability, and efficiency. One important challenge in designing such devices is the droplet routing problem, which basically consists in mobilizing each droplet in the DMFB from its source to its target cells so that fluidic and timing constraints are met, while minimizing simultaneously the amount of cells used and the arrival time of the last droplet. In this paper, we propose a novel representation along with the corresponding genetic operators for the NSGA-II to deal with the droplet routing problem, where the number of used cells and the last droplet arrival time are optimized. Experimental results show that our representation and genetic operators lead to competitive results with respect to those obtained by state-of-the-art methods. To the best of the authors knowledge, this is the first time the droplet routing problem is dealt with an evolutionary algorithm approach.**

***Index Terms*— Digital Microfluidic Biochips, Droplet Routing, Genetic Algorithms, NSGAII, Genetic Operators.**

I. INTRODUCTION

A Digital Microfluidic Biochip (DMFB) is a two-dimensional array consisting of electrodes and some peripheral devices (optical detectors, dispensing ports, etc.). The reagents and samples used in these devices are droplet shaped liquids that are controlled by underlying electrodes through electrical pulses via the process known as Electrowetting on Dielectrics (EWOD) [16], [10]. EWOD technique allows manipulating the shape and flow of the fluids through electrical signals alone. EWOD digital microfluidics involves forming individual droplets from a reservoir and its independent handling through an electrode array [12]. Each cell in the array contains two electrodes that act as two parallel plates. The droplets are moved within the array by applying voltage to a targeted electrode which is adjacent to the droplet and simultaneously disabling the electrode under the droplet as well; in this manner, droplets can be moved within any location in the array [24]. Microfluidics-based biochips offer interesting possibilities such as [8]: clinical diagnostics [20], detecting certain abnormalities in blood (abnormal glucose and lactate levels) [19], DNA sequencing [13], detecting environmental toxicity through the analysis of airborne particulate matter [27], or detecting explosives such as TNT [14], among others. One of the major challenges in designing such devices is the droplet routing problem, which basically consists in mobilizing each droplet in the DMFB from its source cell to its target cell so that fluidic and timing constraints are met, while minimizing simultaneously the amount of cells used and the arrival time of the latest droplet. It is worth noting that this problem is different from the ones addressing design [21] and washing operations [26] in the DMFB context.

The remainder of the paper is organized as follows. Section 2 states the problem and reviews the main works in the field. Section 3 explains the proposed approach. Section 4 presents the experimental setup and results. Section 5 states the conclusions and presents some ideas for future research.

This research was partially supported by CONACYT under grants SEP-CONACYT-CB-2010-01-154863 and CB-2010-154737.

II. DROPLET ROUTING FOR DMFB'S

In this section, we state the droplet routing problem and briefly review the main approaches in the field.

A. Problem statement

For convenience, let the symbol $|\cdot|$ be the absolute value for a scalar, length of a sequence, and cardinality of a set.

Let $C^* = \{c_1^*, c_2^*, \cdots, c_{m\times m}^*\}$ be the set of cells of a microfluidic array of size $m \times m$. Let $D = \{d_1, d_2, \cdots, d_n\}$ be the set of n droplets. Let $ST_d = \{(s_{d_1}, t_{d_1}), (s_{d_2}, t_{d_2}), \cdots, (s_{d_n}, t_{d_n})\}$, be the set of ordered pairs of source (s_{d_i}) and target (t_{d_i}) cells for each droplet d_i. Let $B \subseteq C^*$ be the set of restricted cells, called blockages. The set of valid cells is defined as $C = C^* \backslash B = \{c_1, c_2, \cdots, c_{|C^*|-|B|}\}$, which is the set of cells that are not restricted.

Let us denote the pair $(x_{d_i}^k, y_{d_i}^k)$ as the location of a droplet d_i at time k, similarly, $(x_{s_{d_i}}, y_{s_{d_i}})$ and $(x_{t_{d_i}}, y_{t_{d_i}})$, denote the fixed location of its source and target cells. Finally, let us denote the pair (x_{c_i}, y_{c_i}) as the location of a valid cell. Thus, the Manhattan distance between any two locations l_1, l_2 is defined as $d(l_1, l_2) = |x_{l_1} - x_{l_2}| + |y_{l_1} - y_{l_2}|$.

Let us denote the route rd_i of a droplet d_i as a finite sequence of cells. So that $rd_i = \{c_{\sigma_i(1)}, c_{\sigma_i(2)}, \cdots, c_{\sigma_i(m_i)}\}$, where $c_{\sigma_i(1)} = s_{d_i}$, and $c_{\sigma_i(m_i)} = t_{d_i}$. Note that, in a route rd_i, $c_{\sigma_i(k)}$ might be the same as $c_{\sigma_i(l)}$, for $k \neq l$, indicating

978-1-4799-3840-7/14/$31.00 ©2014 IEEE

that a droplet d_i might stay in a given cell, or, might return to it at a different time stamp.

Any droplet might travel from c_i to c_j if and only if $d(c_i, c_j) = 1$, i.e., the droplet travels between adjacent cells (horizontal or vertical), diagonal moves are not permitted.

For the routing of several droplets, fluidic constraints must be introduced to avoid mixtures between droplets, and taint. In order to select an admissible cell for a droplet at times k and $k+1$, fluidic constraints are defined as follows:

- Static constraint
$$|x_{d_i}^k - x_{d_j}^k| > 1 \text{ or } |y_{d_i}^k - y_{d_j}^k| > 1, \quad i \neq j. \tag{1}$$
- Dynamic constraint
$$\begin{aligned} |x_{d_i}^{k+1} - x_{d_j}^k| > 1 \text{ or } |y_{d_i}^{k+1} - y_{d_j}^k| > 1, \text{ and} \\ |x_{d_i}^k - x_{d_j}^{k+1}| > 1 \text{ or } |y_{d_i}^k - y_{d_j}^{k+1}| > 1, \quad i \neq j. \end{aligned} \tag{2}$$

The static constraint indicates the minimum separation between two droplets which is, at least, one cell at any time stamp k, during routing (Fig. 1(a)). The dynamic constraint implies that the activated cell for d_i cannot be adjacent to d_j (Fig. 1(c)) since there cannot be more than one neighboring cell activated for d_j. Otherwise, it is possible to have an unexpected mixture between d_i and d_j (Fig. 1(b) and Fig. 1(d)).

Besides fluidic constraints, a time constraint is also included. The time constraint T_{max} specifies the maximum arrival time of the latest droplet to its target cell.

Let RD be a set of routes for all droplets in D defined as $RD = \{rd_1, rd_2, \cdots, rd_n\}$. Then the *completion time* of RD is defined by the function

$$T_{latest}(RD) = \max_{i \in \{1,\cdots,n\}} \{|rd_i|\}, \tag{3}$$

which indicates the latest arrival time of a droplet.

The *number of cells used* by RD is defined by the function

$$C_{used}(RD) = |\{c_{\sigma_i(k)} : k \in \{1, ..., |rd_i|\}, i \in \{1, ..., n\}\}|, \tag{4}$$

which indicates the cardinality of the set of cells occupied by all droplets along their routes.

Finally, the problem is defined as follows:

$$\text{Given } C^*, B, D, \text{ and } ST_d,$$

$$\text{minimize}_{RD}(T_{latest}(RD), C_{used}(RD))$$

subject to

$$T_{latest}(RD) \leq T_{max}, \tag{5}$$

$$(1), (2).$$

Note that in the droplet routing problem with a single droplet, if T_{latest} is minimized, then C_{used} will also be minimized. However, it is not difficult to see that in a crowded (many droplets) microfluidic array, if every droplet pursuits to minimize T_{latest}, then C_{used} will be increased due to detours so that constraints (1) and (2) are met. Therefore, the problem should be cast as a multi-objective optimization problem.

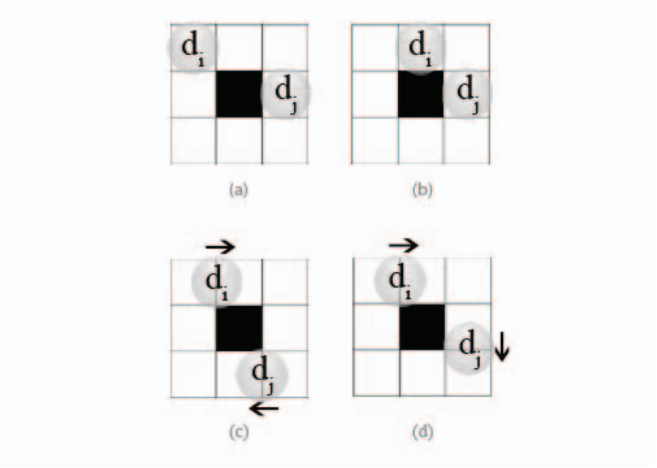

Fig. 1. This example illustrates the static and dynamic constraints in a 3×3 array. (a) Static constraint example. (b) Example of a violation to the static constraint. (c) Dynamic constraint example. (d) Example of a violation to the dynamic constraint.

In [17], it is shown that the problem of coordinating multiple robots is NP-hard. This problem results equivalent to the droplet routing problem if each droplet is seen as an independent robot and if a single objective is considered, *i.e.*, $T_{latest}(RD)$. Moreover, the droplet routing problem, being the workspace naturally divided into cells, is equivalent to a cell decomposition approach of robot motion planning problem [11].

B. Literature review

The droplet routing problem has been addressed by various approaches. Primarily it has been modeled as a robot motion planning problem, as a flow network problem, and as a VLSI routing problem, among others.

Similar to the coupled robot motion planning approach an A* search algorithm was studied in [1], [2] for route generation. Likewise, in order to reduce the search space for the A* search algorithm, an A* priority search was applied in [1], [2]; each droplet's routing priority can be assigned both, randomly or criterion based.

A variation of the *Open Shortest Path First* [3], [23] network protocol has been applied in [9]. A graph was used for this approach, where each node stores information about the shortest routes to its neighboring nodes. Furthermore, the Dijkstra algorithm was used for calculating the shortest paths and thus to build the routing table. Another graph based approach was proposed in [22], inspired in a VLSI routing technique, where the shortest M_i routes are generated (but only one is randomly selected) for each droplet d_i; subsequently, the routes are modified for collision avoidance. In [25]. a network flow based routing algorithm is proposed consisting of a global routing stage and a detailed routing stage.

In [5], a bypassibility metric was proposed, it measures a droplet ability for bypassing another one which has already arrived to its target cell and has become a blockage.

When a dead-lock between two droplets occurs, concession encourages a droplet to retreat, allowing this way the passage of another droplet. An entropy variant based metric has been applied in [10], which measures a droplet congestion throughout its route and not only when it has arrived to its target cell. It also estimates which congestion regions could act as a droplet's buffer for avoiding dead-locks and, thereafter, include it into a route.

In addition to the approaches that have already been presented, there are also other works ([18], [4]) that include additional optimization objectives besides (3) and (4). The first one [18] consists of a graph based heuristic that tackles (3) and (4) along with the minimization of cross-contamination. The second one [4] considers droplet routing as a multi-objective optimization problem and uses a parameterized objective function subject to an upper bound of control pins used for electrode activation to transport a droplet. Regarding metaheuristic approaches we can mention [15] and [16]. In [15], an ant colony optimiation technique is applied, each colony represents a droplet. In [16], a particle swarm optimization method is proposed. Both metaheuristics are aimed at minimizing (3) and (4), however, it is not clear how the objectives are measured and the experimental results are compared over few instances of the benchmark suites reported in literature.

In this work, we propose a multi-objective optimization strategy by using a genetic algorithm, specifically the NSGA-II [6] which is described in the next section.

III. GENETIC ALGORITHM

In this section, a special representation and genetic operators are proposed as the method to solve the problem in the framework of the NSGA-II [6]. Before introducing the individual representation we show how to model the microfluidic array as a graph.

A. Array representation

We represent C^* as a graph $G = (V, E)$. Where, $V = C$, i.e., is the set of valid cells; and E is the set of edges. An edge $e \in E$ is defined as, $e = (c_i, c_j)$, where c_i and c_j are a pair of adjacent valid cells. This way, each cell in B is omitted from the graph, as shown in Fig. 2.

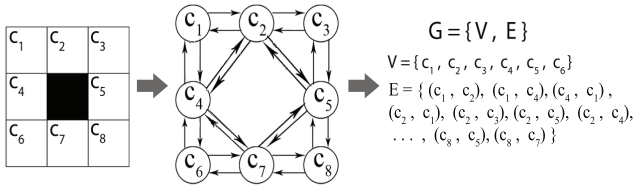

Fig. 2. An example of the convertion of a cell map to a cell graph representation.

This representation allows us not only to route droplets through valid cells, but to verify and prevent a collision at any time.

The computational complexity for building the graph is $O(|C^*|)$.

B. Individual representation

Each individual I represents a feasible solution for the problem. The individual is defined by a set of n routes of droplets $I = \{rd_1, rd_2, \cdots, rd_n\}$. According to the problem statement, each route in turn is comprised of a finite sequence of cells $rd_i = \{c_{\sigma_i(1)}, c_{\sigma_i(2)}, \cdots, c_{\sigma_i(m_i)}\}$. Being the first element in the sequence the source cell s_{d_i} and the last one the target cell t_{d_i}, while the intermediate elements represent the cells sequence that connects them (Fig. 3). Also, each element's index, of a route rd_i, shows the time stamp k in which the droplet is held in the cell $c_{\sigma_i(k)}$.

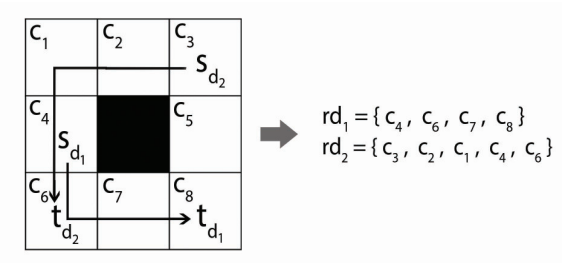

Fig. 3. Individual representation.

C. Initializing the population

Routes are generated in a biased-random manner. We say biased-random since, a droplet d_i, while generating the route, chooses at random the next cell $c_{\sigma_i(k+1)}$ to occupy, as long as the distance to its target decreases or remains the same, i.e., $d(c_{\sigma_i(k+1)}, t_{d_i}) \leq d(c_{\sigma_i(k)}, t_{d_i})$. Therefore, a droplet d_i has, at most, two cells to travel where it could decrease $d(c_{\sigma_i(k+1)}, t_{d_i})$; one of those is chosen randomly. In contrast, there is a probability p of taking a completely random decision, at $k+1$, as long as it satisfies the problem constraints, *i.e.*, a given droplet can move to either adjacent permitted cells as long as (1) and (2) are satisfied. Finally, if the only possible move of d_i is blocked by any other droplet d_j, this will remain in its current cell, *i.e.*, $c_{\sigma_i(k+1)} = c_{\sigma_i(k)}$, waiting for d_j to move towards t_{d_j}, and concede, at time $k + 2$. This may lead to deadlocks, however, adding a deterministic mechanism to prevent them may impact on the quality of the solution, due to the strong bias that would be introduced. Besides, given the time constraint (T_{max}), if the distance between droplet's current and target cells is greater than the remaining time, that is, $d(c_{\sigma_i(k)}, t_{d_i}) + k > T_{max}$, then the current generation of the individual is suspended and starts over. At each time stamp k, each droplet chooses its next cell to be moved to $(c_{\sigma_i(k+1)})$. In order to preserve (2), once a cell is chosen by a droplet, it is bound to that droplet $(rd_i = rd_i \cup \{c_{\sigma_i(k+1)}\})$. Therefore, the chosen cell becomes unavailable, along with its adjacent cells, to the

remaining choosing droplets. A distance $d(c_{\sigma_i(k)}, t_{d_i})$ based priority mechanism is added to let a droplet choose its next cell before the rest, giving the highest priority to the droplet with the longest distance to its target cell; the priority of a droplet may vary at each time k.

D. Fitness of solutions

Based on the objective functions T_{latest} (3) and C_{used} (4), each individual from the pool is assigned a rank according to the nondominated front level in which it is sorted; individuals nondominated by any other individual in the population are assigned rank 1, individuals dominated only by those with rank 1 are assigned rank 2, and so on (see [6]).

E. Parent selection

Two individuals are selected at random from the pool, each pair competes in a binary tournament, the winners are selected as parents. One individual wins the tournament against another if it has a lower nondominated rank or, if both have the same rank, the one with the largest crowding distance wins (see [6]).

F. Genetic operators

Due to hard constraints that must be met ((1), (2), and (5)), it is difficult to generate, most of the time, feasible solutions when applying even a small perturbation to a feasible solution. Therefore, in order to increase the chance to get a feasible solution, three crossover and two mutation operators are proposed. The description of these operators are explained in the following subsections. The notation $I(rd_i)$ must be referred to as, i^{th} route of individual I, i.e., $rd_i \in I$. Also notice that given two individuals $I1$ and $I2$, respective routes of d_i: $I1(rd_i)$, $I2(rd_i)$ are not necessarily the same.

1) Crossover 1: Let P1, P2 be *parent 1* and *parent 2*, respectively, for $i = \{1, 2, ..., n\}$, route $rd_i \in P1$ is swapped with $rd_i \in P2$, two children for each iteration *i* are generated (Fig. 4(a)). This generates an offspring with $2n$ children, of which only those satisfying (1) and (2) are kept. This crossover does not seem to contribute with a significant change, however, swapping a larger number of routes per parent, may easily lead to an invalid child (constraints (1), (2), or (5) unsatisfied), and therefore, to a large number of generations with unsuccessful crossovers. The computational complexity is $O(n)$ for the crossover, and $O(n \cdot T_{max})$ for the constraint satisfaction verification, overall $O(n^2 \cdot T_{max})$, see Algorithm 1.

2) Crossover 2: This is a variant of the crossover known as one point crossover [7]. From both parents' RD, $n/2$ is chosen as the crossover point, offspring is generated swapping each route $rd_i \in P1$ with $rd_i \in P2$, for $i > n/2$ (Fig. 4(b)). Then, both children are verified to make sure that they satisfy the constraints. If any child is invalid, it is rejected. Unlike Crossover 1, it may exist one pair of parents such that swapping a larger number of routes does not lead to an invalid offspring. The computational complexity for this crossover is $O(n)$, and $O(n \cdot T_{max})$ for the constraint satisfaction verification, overall $O(n \cdot T_{max})$, see Algorithm 2.

Algorithm 1: CROSSOVER 1

Input: Two individuals P1, P2
Output: Offspring

```
1 n ← |P1|
2 Offspring ← ∅
3 for i ← 1 to n do
4 |   O1 ← P1
5 |   O2 ← P2
6 |   swap(O1(rd_i), O2(rd_i))
  |   Offspring ← Offspring ∪ {O1, O2}
7 return Offspring
```

Algorithm 2: CROSSOVER 2

Input: Two individuals P1, P2
Output: Offspring

```
1 n ← |P1|
2 Offspring ← ∅
3 O1 ← P1
4 O2 ← P2
5 for i ← (n/2 + 1) to n do
6 |   swap(O1(rd_i), O2(rd_i))
7 Offspring ← Offspring ∪ {O1, O2}
8 return Offspring
```

3) Crossover 3: The third crossover used is a variant of the crossover known as uniform crossover [7]. First, it is assigned "heads" to P1 and "tails" to P2, then, for each droplet d_i, a coin is tossed and the route rd_i from the winner parent is copied to O1 (child 1), while O2 (child 2) is the complement child of O1 (Fig. 4(c)). Note that, the expected number of swaps of routes is $n/2$, similar to Crossover 2, however, the routes of a parent do not necessarily remain together. At last, both children are verified to make sure they satisfy the constraints, if any child is invalid, it is rejected. The computational complexity is similar to that in Crossover 2, see Algorithm 3.

Algorithm 3: CROSSOVER 3

Input: Two individuals P1, P2
Output: Offspring

```
// Replace Crossover 2's lines 5,6
   with below
1 for i ← 1 to n do
2 |   Select at random {0,1}
3 |   if 1 then
4 |   |   swap(O1(rd_i), O2(rd_i))
```

4) Mutation 1: A rd_i route from the n available routes is selected randomly from one individual; all the routes have the same probability of being selected. All the existing cycles in the rd_i route are determined. From those cycles, one is randomly selected; each cycle has the same probability of

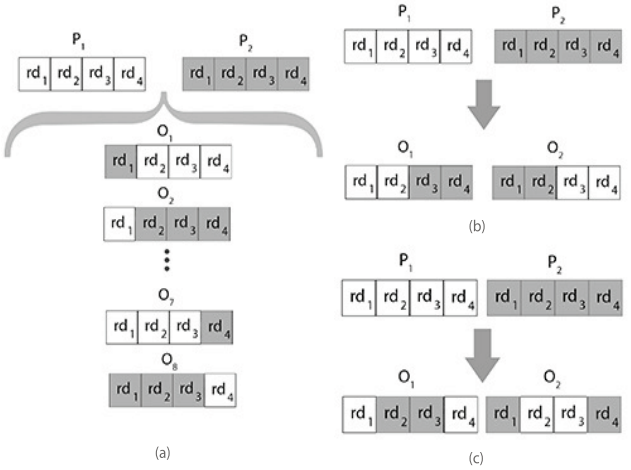

Fig. 4. Example of the crossovers. (a) Crossover 1. (b) Crossover 2. (c) Crossover 3.

being chosen. Then, the intermediate sub-route is compacted (Figure 5(a)). Given the randomness of the pool, an unnecessary cycle for the valid routing of a droplet may exist and its removal implies a reduction in the arrival time of the droplet to its target cell. Finally, it is determined if the individual meets all the constraints or if it must be discarded. If the individual is discarded, then *P* becomes the mutated individual. The cost of computing all the cycles in a route is $O(T_{max})$, the compaction cost is $O(1)$, and $O(n \cdot T_{max})$ to verify its validity, hence, the Mutation 1 computational cost is $O(n \cdot T_{max})$, see Algorithm 4.

Algorithm 4: MUTATION 1

Input: One individual P
Output: Mutated individual
$r \leftarrow$ Select at random $\{1, \cdots, |P|\}$
$route \leftarrow P(rd_r)$
Find each repeated cells in $route$
Select at random one pair of those repeated cells
$route \leftarrow route$ - each cell in between the selected pair of cells
$Offspring \leftarrow P$
$Offspring(rd_r) \leftarrow route$
return $Offspring$

5) Mutation 2: A rd_i route from the n available routes is selected randomly from one individual; each route has the same probability of being selected. Then, from the chosen rd_i route, two cells are randomly selected; each of them has the same probability of being chosen. Subsequently, a new sub-route conecting those two cells is randomly generated. The new sub-route is generated in the same way as the pool was produced (Fig. 5(b)). Finally, it is determined whether the individual meets all the constraints or if it must be discarded. If the individual is discarded, then *P* becomes the mutated individual. The computational cost of generating the new sub-route is $O(T_{max})$, and $O(n \cdot T_{max})$ to verify its validity, hence, the Mutation 2 computational cost is $O(n \cdot T_{max})$, see Algorithm 5.

Algorithm 5: MUTATION 2

Input: One individual P
Output: Mutated individual
$r \leftarrow$ Select at random $\{1, \cdots, |P|\}$
$route \leftarrow P(rd_r)$
$c1 \leftarrow$ Select at random $\{1, \cdots, |route|\}$
$c2 \leftarrow$ Select at random $\{1, \cdots, |route|\}$
if $c2 < c1$ **then**
  $swap(c1, c2)$
From $route$ reroute, as initializing population, segment between cells $c1, c2$
$Offspring \leftarrow P$
$Offspring(rd_r) \leftarrow route$
return $Offspring$

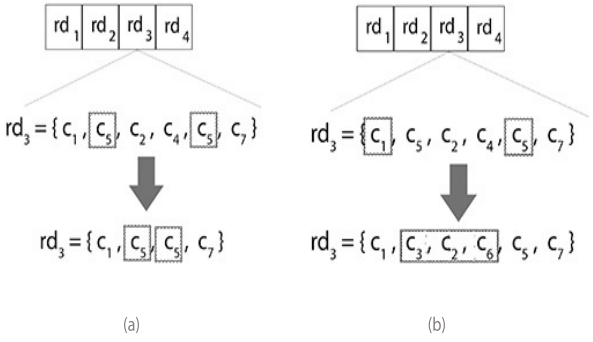

Fig. 5. An example of the used mutations. (a) Mutation 1. (b) Mutation 2.

IV. EXPERIMENTAL SETUP AND RESULTS

The droplet routing algorithm implementation (Java SE 7) and the corresponding tests, were performed in a 2.50GHz Intel(R) Core(TM) i5 CPU-3210M (64-bit), Windows 7 machine with 4GB of memory. The NSGA-II running parameters, shown in Table I, were chosen after a non-exhaustive analysis of six instances with different parameters and ten runs each. Table I shows that the overall crossover probability is 1.0 while the mutation reaches 0.31. This means that each time one of the crossover operators is to be chosen with the probabilities given in Table I. The same applies for the mutation operators.

Due to the high computation cost to generate the initial population, the algorithm's performance was measured over the first 16 instances of Benchmark Suite I [5] which consists of 30 instances. Table II shows a general comparison with previous works: prioritized A* search [1], network-flow-based-algorithm [25], high-performance routing algorithm [5], and fast routability driven droplet algorithm [10]. Our

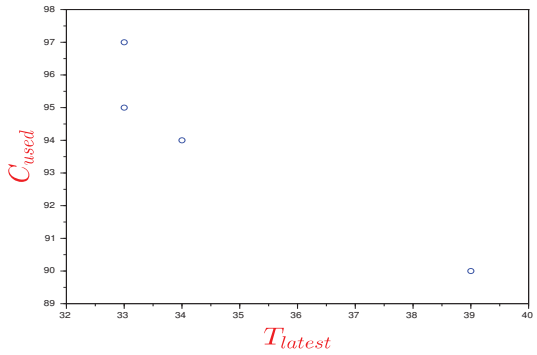

Fig. 6. Nondominated individuals found in ten runs for Test 3 of Benchmark Suite I (Table 2).

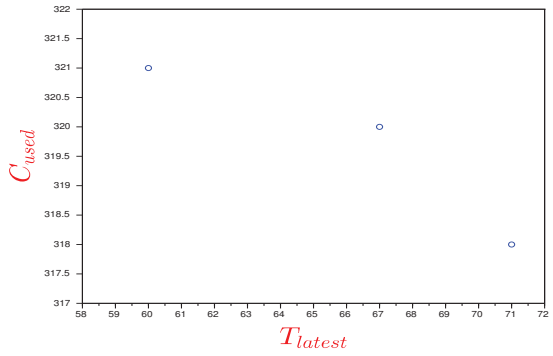

Fig. 7. Nondominated individuals found in ten runs for Test 14 of Benchmark Suite I (Table 2).

algorithm achieves 100% of routing completion over all the analyzed instances, while the previous four works completed 43.75%, 31.25%, 81.25%, and 100%, respectively. We can also see in this table that our approach obtains 10 out of 16 (62.5%) nondominated solutions while the remaining 6 are dominated by solutions of some of the previous approaches. Our is only outperformed by the fast routability [10] method which produces 13 out of 16 nondominated solutions. In Table II, nondominated solutions are highlighted in boldface characters.

Additionally, algorithm's performance is also measured over three instances of Benchmark Suite II reported in [10]. Results from our proposed approach are shown in Table III along with the ones reported in [10]. We can see in this table that fast routability [10] and our approach have comparable performance. The high performance algorithm [5] was not able to find a single solution on these instances. Although the work in [10] presented more instances, only three of them could be retrieved.

Some resulting nondominated fronts, from the instances shown in tables II and III, converge to one or two individuals only. We can see in Fig. 6 and Fig. 7 that the nondominated fronts contain only few points; four for Benchmark Suite I: Test 3, and three for for Benchmark Suite I: Test 14. It is

TABLE I
NSGA-II PARAMETERS.

Parameter	Value
Population size	50
Maximum generation	50000
Crossover1 rate	0.3
Crossover2 rate	0.3
Crossover3 rate	0.4
Mutation1 rate	0.1
Mutation2 rate	0.21

worth noting that these fronts are the two most populated fronts of the analyzed instances. It is important to mention that even if two or more individuals share equal values of T_{latest} (3) and C_{used} (4), to be distinct it is sufficient, that they differ from one another, in a single cell in the same time stamp of the same route. Finding multiple solutions with the same objective function values can be considered as a good characteristic of the algorithm. These similar solutions may be used to design fault tolerant routes. To show the differences between three individuals equally valued in objective functions (3) and (4) that were found by the proposed method, their corresponding routes are listed below:

$T_{latest} = 14$, $C_{used} = 45$
$rd_1 = \{c_{11}, c_{12}, c_{18}, c_{24}, c_{18}, c_{12}, c_{18}, c_{24}, c_{30}, c_{36}, c_{49}, c_{48}\}$
$rd_2 = \{c_1, c_{14}, c_{20}, c_{26}, c_{32}, c_{38}\}$
$rd_3 = \{c_6, c_7, c_8, c_9, c_{10}, c_{16}, c_{22}, c_{28}, c_{34}, c_{47}, c_{46}, c_{45}, c_{44}, c_{43}\}$
$rd_4 = \{c_{105}, c_{106}, c_{107}, \mathbf{c_{97}, c_{98}, c_{91}}, c_{92}, c_{93}, c_{94}, c_{101}, c_{111}\}$
$rd_5 = \{c_{75}, c_{85}, c_{92}, c_{93}, c_{94}, c_{101}, c_{111}, c_{112}, c_{113}, c_{114}, c_{115}\}$
(See Fig. 8(a))

$T_{latest} = 14$, $C_{used} = 45$
$rd_1 = \{c_{11}, c_{12}, c_{18}, c_{24}, c_{18}, c_{12}, c_{18}, c_{24}, c_{30}, c_{36}, c_{49}, c_{48}\}$
$rd_2 = \{c_1, c_{14}, c_{20}, c_{26}, c_{32}, c_{38}\}$
$rd_3 = \{c_6, c_7, c_8, c_9, c_{10}, c_{16}, c_{22}, c_{28}, c_{34}, c_{47}, c_{46}, c_{45}, c_{44}, c_{43}\}$
$rd_4 = \{c_{105}, c_{106}, c_{107}, \mathbf{c_{108}, c_{109}, c_{99}}, c_{92}, c_{93}, c_{94}, c_{101}, c_{111}\}$
$rd_5 = \{c_{75}, c_{85}, c_{92}, c_{93}, c_{94}, c_{101}, c_{111}, c_{112}, c_{113}, c_{114}, c_{115}\}$
(See Fig. 8(b))

$T_{latest} = 14$, $C_{used} = 45$
$rd_1 = \{c_{11}, \mathbf{c_{17}, c_{23}, c_{29}, c_{35}}, c_{48}\}$
$rd_2 = \{c_1, \mathbf{c_0, c_{13}, c_{13}, c_0, c_0, c_{13}, c_{19}, c_{25}, c_{31}, c_{37}}, c_{38}\}$
$rd_3 = \{c_6, \mathbf{c_5, c_4, c_3, c_2, c_{15}, c_{21}, c_{27}, c_{33}, c_{39}, c_{40}, c_{41}, c_{42}}, c_{43}\}$
$rd_4 = \{c_{105}, c_{106}, c_{107}, \mathbf{c_{97}, c_{90}, c_{91}}, c_{92}, c_{93}, c_{94}, c_{101}, c_{111}\}$
$rd_5 = \{c_{75}, c_{85}, c_{92}, c_{93}, c_{94}, c_{101}, c_{111}, c_{112}, c_{113}, c_{114}, c_{115}\}$
(See Fig. 8(c))

The differences on these solutions are shown by using boldface characters. Fig. 8 shows the three solutions previously described.

In Table IV, is shown the overall running time taken to initialize the population and the remaining part of the NSGA-II. The column labeled as *Size (%B)* indicates the size of the instance and the average percentage of $B \in C^*$, respectively. The table shows how the computing time of initializing the population increases with instances when the size of B is above 29% of the array size.

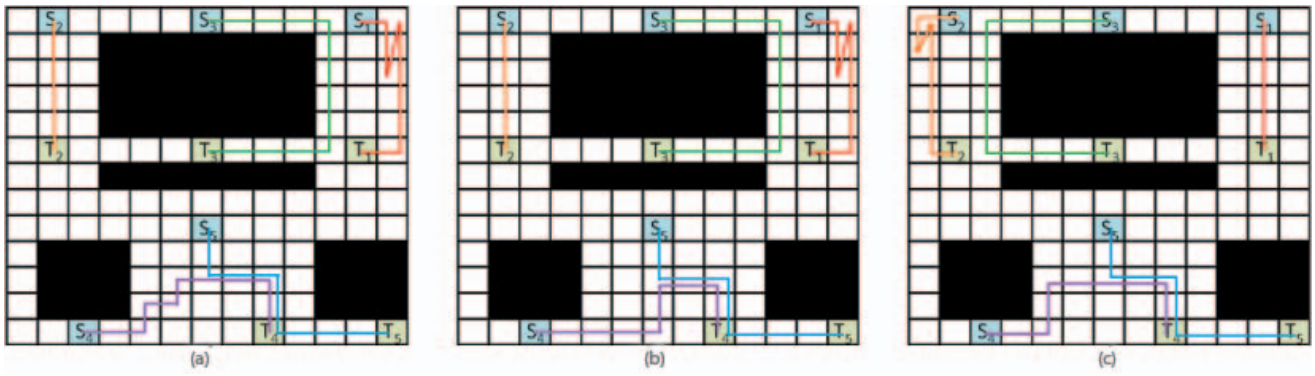

Fig. 8. Three individuals equally valued on objective functions (3) and (4), $T_{latest} = 14$, $C_{used} = 45$.

TABLE II

COMPARISON RESULTS OF THE PROPOSED APPROACH WITH THE STATE-OF-THE-ART METHODS.

BENCHMARK SUITE I													
				Prioritized A* [1]		N.-Flow [25]		High Perf. [5]		Fast Rout. [10]		Ours	
Name	Size	#Drops	#Blk	T_{latest}	C_{used}	T_{latest}	C_{used}	T_{latest}	C_{used}	T_{latest}	C_{used}	T_{latest}	C_{used}
Test 1	12x12	12	23	**37**	**66**	n/a	n/a	100	67	39	73	**36**	**92**
Test 2	12x12	12	25	n/a	n/a	n/a	n/a	n/a	n/a	**47**	**65**	**32**	**86**
Test 3	12x12	12	28	n/a	n/a	n/a	n/a	n/a	n/a	**41**	**58**	**39**	**90**
Test 4	12x12	12	31	n/a	n/a	n/a	n/a	**70**	**64**	**38**	**71**	**20**	**79**
Test 5	16x16	16	39	**28**	**108**	n/a	n/a	78	118	**40**	**100**	29	141
Test 6	16x16	16	30	**43**	**116**	44	132	55	119	**47**	**98**	**38**	**154**
Test 7	16x16	16	52	**33**	**104**	n/a	n/a	89	113	**44**	**93**	42	147
Test 8	16x16	16	54	n/a	n/a	47	129	**41**	**94**	49	96	**35**	**148**
Test 9	16x16	16	72	n/a	n/a	n/a	n/a	n/a	n/a	**49**	**91**	**28**	**135**
Test 10	16x16	16	67	n/a	n/a	n/a	n/a	77	110	**51**	**94**	**41**	**154**
Test 11	24x24	24	106	62	252	100	264	**47**	**249**	**56**	**228**	60	304
Test 12	24x24	24	104	n/a	n/a	80	242	**52**	**219**	**62**	**231**	**50**	**280**
Test 13	24x24	24	137	**60**	**241**	n/a	n/a	**52**	**247**	**62**	**221**	**49**	**286**
Test 14	24x24	24	143	n/a	n/a	n/a	n/a	**57**	**234**	**64**	**219**	60	321
Test 15	24x24	24	173	**63**	**243**	74	233	83	230	**64**	**227**	65	313
Test 16	24x24	24	185	n/a	n/a	n/a	n/a	**63**	**223**	**58**	**220**	66	305

TABLE III

COMPARISON RESULTS OF THE PROPOSED APPROACH WITH THE STATE-OF-THE-ART METHODS.

BENCHMARK SUITE II									
				High Performance [5]		Fast Routability [10]		Ours	
Name	Size	#Drops	#Blk	T_{latest}	C_{used}	T_{latest}	C_{used}	T_{latest}	C_{used}
Test 1	13x13	5	53	n/a	n/a	**13**	**33**	14	44
Test 2	13x13	6	69	n/a	n/a	17	51	**16**	**50**
Test 3	16x16	9	133	n/a	n/a	**27**	**87**	**29**	**72**

TABLE IV

OVERALL BENCHMARK SUITE I INSTANCES RUNNING TIME.

Size ($\%B$)	Init. pop. (sec)	NSGA-II (sec)
12 x 12 (18)	4	56
16 x 16 (20)	7	104
24 x 24 (21)	53	252
24 x 24 (31)	6126	414

V. CONCLUSIONS

A new representation and genetic operators, under the NSGA-II framework, for the droplet routing problem have been proposed. The maximum completion time and the total number of used cells are simultaneously minimized. The representation and genetic operators are designed to maximize the chance of generating valid individuals, which is one of the major challenges to achieve in the droplet routing problem.

Results from computational experiments show that the approach produces competitive results when comparing it with state-of-the-art approaches. Few solutions in the resulting nondominated fronts show that a strategy that does not expend much effort on diversity may be used to address the problem.

Future work is aimed at improving the generation of the initial population since much of the computation cost is expended here. Also, the inclusion of a repairing procedure for invalid offspring, resulting from a genetic operator, will be explored as an alternative to discard invalid individuals. Finally, addressing the problem with washing operations will be considered.

VI. ACKNOWLEDGMENT

We would like to thank Dr. Minsik Cho and Prof. Dr. David Z. Pan for providing Benchmark Suite I instances.

REFERENCES

[1] K.-F. Böhringer. Towards optimal strategies for moving droplets in digital microfluidic systems. In *ICRA*, pages 1468–1474. IEEE, 2004.

[2] K.-F. Böhringer. Modeling and controlling parallel tasks in droplet-based microfluidic systems. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 25(2):334–344, 2006.

[3] J. Brassil and R. L. Cruz. Nonuniform traffic in the manhattan street network. *Perform. Eval.*, 25(3):233–242, 1996.

[4] S. Chatterjee, H. Rahaman, and T. Samanta. Multi-objective optimization algorithm for efficient pin-constrained droplet routing technique in digital microfluidic biochip. In *Quality Electronic Design (ISQED), 2013 14th International Symposium on*, pages 252–256, 2013.

[5] M. Cho and D. Z. Pan. A high-performance droplet routing algorithm for digital microfluidic biochips. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 27(10):1714–1724, 2008.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[7] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, 2003.

[8] R. Fair, A. Khlystov, T. Tailor, V. Ivanov, R. Evans, P. Griffin, V. Srinivasan, V. Pamula, M. Pollack, and J. Zhou. Chemical and biological applications of digital-microfluidic devices. *Design Test of Computers, IEEE*, 24(1):10–24, 2007.

[9] E. Griffith and S. Akella. Coordinating multiple droplets in planar array digital microfluidics systems. In *Workshop on the Algorithmic Foundations of Robotics*, pages 219–234. Springer, 2005.

[10] T.-W. Huang and T.-Y. Ho. A fast routability- and performance-driven droplet routing algorithm for digital microfluidic biochips. In *ICCD*, pages 445–450. IEEE, 2009.

[11] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA, USA, 1991.

[12] W. C. Nelson and C.-J. C. Kim. Droplet actuation by electrowetting-on-dielectric (ewod): A review. *Journal of Adhesion Science and Technology*, 26(12-17):1747–1771, 2012.

[13] B. M. Paegel, R. G. Blazej, and R. A. Mathies. Microfluidic devices for {DNA} sequencing: sample preparation and electrophoretic analysis. *Current Opinion in Biotechnology*, 14(1):42 – 50, 2003.

[14] V. Pamula, V. Srinivasan, H. Chakrapani, R. Fair, and E. J. Toone. A droplet-based lab-on-a-chip for colorimetric detection of nitroaromatic explosives. In *Micro Electro Mechanical Systems, 2005. MEMS 2005. 18th IEEE International Conference on*, pages 722–725, 2005.

[15] I. Pan, P. Dasgupta, H. Rahaman, and T. Samantam. Ant colony optimization based droplet routing technique in digital microfluidic biochip. In *Electronic System Design (ISED), 2011 International Symposium on*, pages 223–229, 2011.

[16] I. Pan and T. Samanta. Efficient droplet router for digital microfluidic biochip using particle swarm optimizer, 2013.

[17] J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. *The International Journal of Robotics Research*, 24(4):295–310, 2005.

[18] P. Roy, P. Howladar, R. Bhattacharjee, H. Rahaman, and P. Dasgupta. A new cross contamination aware routing method with intelligent path exploration in digital microfluidic biochips. In *Design Technology of Integrated Systems in Nanoscale Era (DTIS), 2013 8th International Conference on*, pages 50–55, 2013.

[19] V. Srinivasan. Droplet-based microfluidic lab-on-a-chip for glucose detection. *Analytica Chimica Acta*, 507(1):145–150, Apr. 2004.

[20] V. Srinivasan, V. K. Pamula, M. G. Pollack, and R. B. Fair. Clinical diagnostics on human whole blood, plasma, serum, urine, saliva, sweat, and tears on a digital microfluidic platform. 2003.

[21] F. Su and K. Chakrabarty. Design of fault-tolerant and dynamically-reconfigurable microfluidic biochips. *CoRR*, abs/0710.4673, 2007.

[22] F. Su, W. L. Hwang, and K. Chakrabarty. Droplet routing in the synthesis of digital microfluidic biochips. In G. G. E. Gielen, editor, *DATE*, pages 323–328. European Design and Automation Association, Leuven, Belgium, 2006.

[23] A. S. Tanenbaum. *Computer networks*. Prentice-Hall, 3 edition, 1996.

[24] T. Xu and K. Chakrabarty. Integrated droplet routing and defect tolerance in the synthesis of digital microfluidic biochips. *J. Emerg. Technol. Comput. Syst.*, 4(3):11:1–11:24, Aug. 2008.

[25] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang. BioRoute: a network-flow based routing algorithm for digital microfluidic biochips. pages 752–757, Nov. 2007.

[26] Y. Zhao and K. Chakrabarty. Synchronization of washing operations with droplet routing for cross-contamination avoidance in digital microfluidic biochips. In *Proceedings of the 47th Design Automation Conference*, DAC '10, pages 635–640, New York, NY, USA, 2010. ACM.

[27] Y. Zhao and S. K. Cho. A micro particle sampler using electrowetting-actuated droplet sweeping. In *Solid-State Sensors, Actuators and Microsystems, 2005. Digest of Technical Papers. TRANSDUCERS '05. The 13th International Conference on*, volume 1, pages 129–134 Vol. 1, 2005.