A General and Exact Routing Methodology for Digital Microfluidic Biochips

Oliver Keszocze[1,2] Robert Wille[1,2] Krishnendu Chakrabarty[3] Rolf Drechsler[1,2]
[1]Institute of Computer Science, University of Bremen, Bremen, Germany
[2]Cyber-Physical Systems, DFKI GmbH, Bremen, Germany
[3]Department of Electrical and Computer Engineering, Duke University, Durham, NC
{keszocze,rwille,drechsle}@informatik.uni-bremen.de krish@ee.duke.edu

***Abstract*—Advances in microfluidic technologies have led to the emergence of *Digital Microfluidic Biochips* (DMFBs), which are capable of automating laboratory procedures in biochemistry and molecular biology. During the design and use of these devices, droplet routing represents a particularly critical challenge. Here, various design tasks have to be addressed for which, depending on the corresponding scenario, different solutions are available. However, all these developments eventually result in an "inflation" of different design approaches for routing of DMFBs – many of them addressing a very dedicated routing task only. In this work, we propose a comprehensive routing methodology which (1) provides one (generic) solution capable of addressing a variety of different design tasks, (2) employs a "push-button"-scheme that requires no (manual) composition of partial results, and (3) guarantees minimality e.g., with respect to the number of timesteps or the number of required control pins. Experimental evaluations demonstrate the benefits of the solution, i.e., the applicability for a wide range of design tasks as well as improvements compared to specialized solutions presented in the past.**

I. Introduction

Recent advances in microfluidic technologies have enabled the emergence of *Digital Microfluidic Biochips* (DMFBs) as a new paradigm in automating laboratory procedures in biochemistry and molecular biology [1]. These devices provide a platform in which small volumes of fluids (in terms of *droplets*) can be moved around and typical laboratory operations such as mixing, heating, analyzing, etc. can be conducted. To this end, several positions are available at each DMFB (eventually forming a *layout*) onto which droplets can be placed. The movements of droplets between different positions is realized through electrodes at all positions. These can be turned on/off thereby realizing electrical actuations that move droplets using the principle of *electrowetting-on-dielectric* [2]). Compared to conventional experiments conducted by laboratory technicians, this leads to significant advantages such as high throughput, low reagent consumption, and minimal to no manual intervention.

Accordingly, DMFBs received significant consideration in the recent past – especially in the domain of design automation. Several approaches have been introduced which aid engineers in the task of realizing particular laboratory experiments onto given DMBFs. They basically follow the well-known steps known from the design of conventional systems (e.g., allocation, binding, scheduling, placement, and routing), but additionally consider the intrinsic properties and constraints of DMFBs. Examples of corresponding design solutions are available e.g., in [3], [4], [5], [6], [7], [8].

Within this design flow, the problem of *routing* is of particular importance. It principally deals with the movement of droplets in a given layout and, hence, is one of the major problems that must be tackled in order to efficiently and correctly realize an experiment on the DMFB. The main objective is to move all considered droplets from their respective source positions to their desired target positions. Moreover, the electrical actuations, i.e. the proper control of all electrodes which eventually perform the movement of droplets, becomes a crucial design issue. Overall, this leads to a variety of routing tasks which have to be considered when designing DMFBs.

The investigation of all these routing tasks led to several powerful EDA methods whereby each of them usually addresses rather specific scenarios. More precisely, dedicated solutions have been proposed for

- Droplet routing, i.e., the determination of routes onto a grid along which the considered droplets are moved to the desired position (see e.g., [5], [9], [10]),
- Pin mapping, i.e., the determination of an appropriate pin-assignment which realizes a given routing (see e.g., [11], [12]),
- Pin-aware routing, i.e., the simultaneous determination of an actual routing and a pin assignment (see e.g., [3], [4], [13], [14]), as well as
- Routing for fixed pin mappings, i.e., the determination of the desired routing with respect to a given pin-assignment (see e.g., [15], [16]).

The recent surge of interest in this research area, as highlighted by the above references, has resulted in an "inflation" of routing methods and, by this, a state-of-the-art in which very specialized solutions have to be applied for each dedicated routing problem. Such focused and point-solution methods do not take into account the global optimization problem that must be solved to obtain efficient solutions. Moreover, for follow-up design tasks such as verifying the applicability

978-1-4673-8388-2/15/$31.00

of a given pin mapping, no solution exists today. In other words, designers are have to choose from a huge selection of point solutions for a subset of routing problems, but have no EDA support for the remaining routing-related design tasks. Moreover, although existing routing solutions consider dedicated sub-problems only, the majority of them are of heuristic nature or rely on a composition of exact results. As a consequence, most of them do not guarantee minimality e.g., with respect to the number of time steps and/or pins.

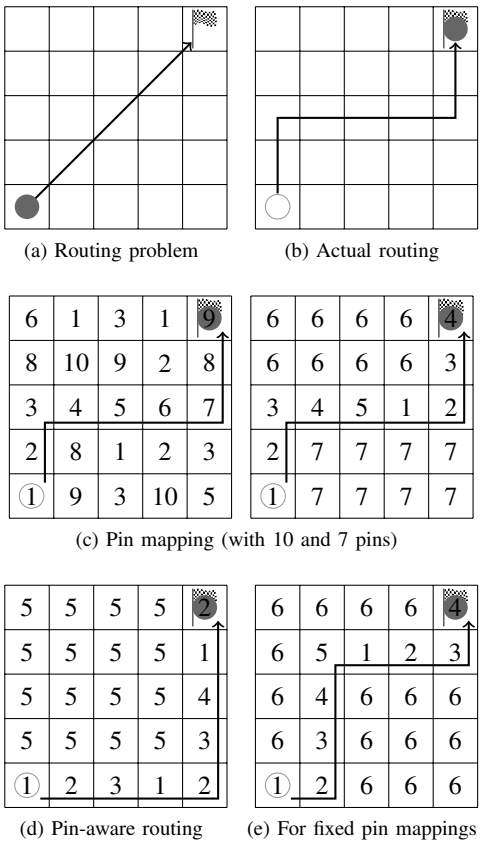
In this work, we show that point solutions for specific routing problems are not always necessary, and better results can be obtained using a more general problem formulation and solution technique. We propose a generic routing methodology which aims for comprehensively and exactly covering routing problems for DMFBs. More precisely, a solution is proposed which (1) provides *one* (generic) solution capable of addressing a wide variety of routing tasks, (2) employs a "push-button" scheme, i.e., requires no (manual) composition of partial results but directly delivers the entire solution, and (3) is exact, i.e. guarantees minimality e.g., with respect to the number of timesteps or the number of required actuation pins. Those goals are accomplished by formulating the respective routing tasks in a generic fashion that symbolically represents all desired scenarios. Afterwards, this formulation can be configured in a fashion which, together with a proper solving engine (e.g., based on Integer Linear Programming, Boolean satisfiability, SAT Modulo Theories, etc.), solves the considered routing task.

Several case studies using commercial as well as academic designs confirm the applicability of the proposed methodology. Overall, they show the following benefits:

- Various routing tasks, for which a selection of dedicated solutions have to be applied thus far, can comprehensively be tackled using a unified and generic methodology.
- Improvements, e.g., in the number of required control pins, are achieved, since the proposed methodology guarantees minimality of the results.
- Routing-related problems that have not been considered yet but will gain relevance in the future (such as the verification of pre-defined pin mappings) are, for the first time, covered as part of an optimization framework.

The proposed generic optimization framework and our simulation results therefore show that prior heuristic methods on specific aspects of droplet routing are no longer necessary.

The remainder of this paper is organized as follows. The next section gives an overview on the variety of routing tasks as well as corresponding solutions that have been suggested for this in previous work. Next, the proposed symbolic formulation is introduced and described in Section III followed by Section IV with a summary of the various case studies that we have considered. Finally, the paper is concluded in Section V.



(a) Routing problem (b) Actual routing

(c) Pin mapping (with 10 and 7 pins)

(d) Pin-aware routing (e) For fixed pin mappings

Fig. 1. Different routing tasks

II. Background & Motivation

In this work, we consider the problem of routing in DMFBs. Here, the overall problem is to realize the movement of all considered droplets from the respective initial positions (denoted as *source positions*) to the respective desired positions (denoted as *target positions*). Although routing in DMFBs is similar to conventional routing for VLSI systems, it inherits some intrinsic differences. For example, the respective routes may cross each other as long as unintended mixing of droplets is avoided. At the same time, routing is realized through electrodes under each cell which are activated by dedicated control pins. Overall, this leads to two major issues to be considered when solving routing problems for DMFBs, namely

- determining an actual routing, i.e. (preferably short) paths from the source positions to the target positions, for all droplets involved in an experiment and
- determining a precise pin mapping for a routing, i.e. a mapping of control pins to electrodes which allow for the realization of the desired routing.

In practice, this eventually results in various design tasks which, depending on the given scenario, need to be addressed. Fig. 1 provides an (incomplete) illustration of some of these tasks which are reviewed in more detail next.

A. Actual Routing

The most obvious and heavily investigated problem is the determination of a routing for all droplets as illustrated in the following example. The main objective is to keep the number of needed timesteps as small as possible (or at least below a given threshold).

Example 1. *Consider the routing problem as sketched in Fig. 1(a): A droplet has to be routed from position (0,0) to position (5,5). A possible solution (requiring 8 timesteps) is sketched in Fig. 1(b).*

Several approaches for routing in DMFBs have been presented in the recent past. This includes greedy approaches [5], [10] as well methods based on network-flows [9] or the concept of entropy [17] together with clever heuristics or solving schemes such as dynamic programming. While these solutions lead to heuristic results, also exact methods aiming for minimal routings have been considered e.g., in [18] (based on *Integer Linear Programming*, ILP) or [19], [20] (based on *Boolean satisfiability*, i.e. SAT). For a specific bioassay, the respective droplet routing is decomposed into a series of subproblems.

B. Pin Mapping

The movements of droplets onto a given DMFB is realized through *electrodes* at all positions which can be turned on and off at different timesteps. This can be coordinated so that electrical actuations result which allow for moving the droplets around the entire DMFB. More precisely, if an electrode is turned on and all neighboring electrodes are turned off, a droplet is held at the respective cell. If, afterwards, the same electrode is turned off, but one neighboring electrode is turned on, the droplet moves to this cell. In order to control the electrodes, each of them is connected to a *control pin*.

To realize this concept, a naive solution could assume a direct-addressing scheme, where each electrode is connected to a dedicated pin (allowing for activating each cell independently). As this would lead to rather complex systems, the number of available pins is usually assumed restricted [11]. Consequently, a single control pin is connected to several electrodes. In order to still realize the desired routing, this requires to determine a proper *pin mapping* which keeps the number of required control pins as small as possible (or at least, below a given threshold). In practice, this particularly is of importance when a single assay is supposed to be realized in a large amount of chips. Then, saving pins directly saves production costs.

Example 2. *Consider again the routing derived before as shown in Fig. 1(b). The left-hand side of Fig. 1(c) shows a possible pin mapping which would realize this routing (the respective pins are denoted as numbers in the respective cells, i.e. a cell denoted by* 1 *is supposed to be controlled by pin number 1). While this would require a total of* 10 *control pins, a better (in fact, minimal) pin mapping realizing the same routing but requiring only* 7 *control pins is shown on the right-hand side of Fig. 1(c).*

Pin mapping has explicitly been addressed e.g., in [11]. Here, the movement of droplets is analyzed in order to create partitions of the DMFB. As soon as the partitions become small enough, a direct addressing pin assignment scheme is applied. In [12], the droplet movements are used to generate so-called actuation vectors. This allows for representing the pin mapping problem as a clique-partitioning problem for which an existing (heuristic) solution can be applied.

C. Pin-aware Routing

Besides considering the actual routing as well as the pin mapping separately, both tasks may also be tackled at the same time (usually called *pin-aware routing*). In general, this significantly increases the complexity of the tasks and make them harder to solve, but allows for determining better routings with respect to the number of control pins.

Example 3. *Consider again the routing problem as sketched in Fig. 1(a). Conducting pin-aware routing might lead to the result shown in Fig. 1(d). Although this routing would require the same number of timesteps as the result obtained by the actual routing (see Fig. 1(b)), it can be realized with 5 pins only.*

Dedicated solutions for this task have been presented e.g., in [11] or [3], [4], [13], [14] – again, based on partitioning and ILP, respectively. However, these methods generally suffer from high computational costs. Computations may require multiple days or even entirely fail to generate results within the considered time limit. As a consequence, often rather preliminary results are derived from those.

D. Routing for Fixed Pin Mappings

Commercially available DMBFs already have successfully been used to perform assays such as *polymerase chain reaction* (PCR) [15] or *bead-based immunoassays* [16]. Usually they include a fixed pin mapping. Hence, design methods are required which realize the desired routing dependent on a given pin mapping.

Example 4. *Consider the (fixed) pin mapping as shown in Fig. 1(e). Assuming this configuration would neither allow for the realization of the routing from Fig. 1(b) nor the one from Fig. 1(d), since the movement from cell (0,1) to cell (0,2) is impossible with both cells controlled by pin 6. Hence, an alternative routing (as the one shown in Fig. 1(e)) has to be determined.*

E. Discussion

The tasks reviewed above represent only a selection of issues which may have to be considered during the design of DMBFs. Besides that, many further derivations (e.g., additionally for online-routing [21], error correction [22], and more) exist. The respective developments aiming at solving these

tasks eventually resulted in an "inflation" of different routing approaches for DMFBs – many of them often only addressing a very dedicated routing task. Moreover, the majority of them are of heuristic nature or rely on a composition of exact results – either way harming minimality and not guaranteeing optimal results e.g., with respect to the number of time steps and/or pins. Overall, no solution exists yet which comprehensively covers routing tasks as discussed above in an efficient and holistic fashion.

III. Symbolic Formulation

The general idea of the proposed methodology is to formulate the different routing possibilities in a symbolic fashion which includes all possible paths of the droplets, pin assignments, pin-actuations, etc. Afterwards, precise derivations of routing tasks can be addressed by pre-assigning certain parameters of the formulation (i.e. *configuring* the problem) and passing the resulting instance to a proper solving engine which derives an explicit solution for the considered task (i.e. *solving* the problem). In this section, the notation applied in order to describe routing problems is provided first. Afterwards, the symbolic formulation itself is presented.

A. Notation

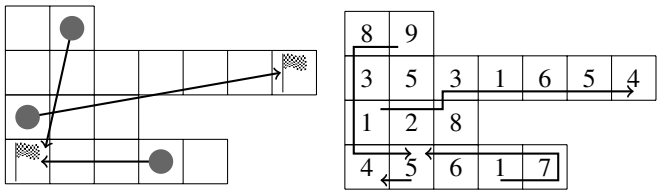
The proposed formulation shall be applicable to a variety of possible DMFB-layouts. In order to properly represent those, the following notation is applied.

Definition 1 (DMFB Layout)**.** *By P we denote the* set of all positions *(or* cells*) of the DMFB. For a position $p \in P$, the sets $N_5(p)$ as well as $N_9(p)$ denote the horizontal/vertical-neighborhood as well as the complete surrounding neighborhood of p, respectively*[1]*. An additional asterisk (i.e. N^*) denotes that the center of the neighborhood (i.e. the actual position p) is not included in the set.*

For a given DMFB-layout, the routing problem is defined as determining a route from given source positions of droplets to the desired target positions. The respective routing tasks are defined in terms of so-called *nets*.

Definition 2 (Droplets and Nets)**.** *Let D denote the* set of all droplets *considered in the given routing problem. For each droplet $d \in D$, the variables p_d^* and $p_d^\dagger$ denote the droplet's* source position *and* target position*, respectively. Similarly, the variables t_d^* and $t_d^\dagger$ denote the* spawn-time *and the targeted* arrival time *of the droplet d, respectively. Then, a k-*droplet net *N is defined by a subset of droplets $N \subseteq D$ that inherit different source positions but eventually are supposed to reach the same target position $p_N^\dagger$ at timestep $t_N^\dagger$ (with $p_N^\dagger = p_d^\dagger$ and $t_N^\dagger = p_d^\dagger$ for an arbitrary droplet $d \in N$). The set of all nets is defined as $\mathcal{N}$. In order to consider the entire routing problem including all nets, $t^\dagger = \max_{d \in D}\{t_d^\dagger\}$ timesteps have to be considered in the worst case.*

[1]Note that this definition does not imply any constraints on the layout of the chip. For example, a set N_9 does not necessarily contain 9 positions.



(a) Routing problem (b) Solution

Fig. 2. Illustrating the symbolic formulation

The notation of the layout, the droplets, and the nets to be realized allows for a formal description of an explicit routing task as illustrated by the following example.

Example 5. *Consider the routing problem sketched in Fig. 2(a). In total, the layout leads to* 17 *positions $P = \{p_1, \ldots, p_{17}\}$ (enumerated line-wise from top to bottom) to be considered which can be occupied by* 3 *droplets $D = \{d_1, d_2, d_3\}$. These droplets form two nets $\mathcal{N} = \{N_1, N_2\}$,*

- *a 1-droplet net N_1 composed of the droplet d_1 which is assumed to spawn at position $p_1^* = p_{10}$ at timestep $t_1^* = 1$ and supposed to reach position $p_1^\dagger = p_3$ at timestep $t_1^\dagger = 8$ and*
- *a 2-droplet net N_2 composed of the droplets d_2 and d_3 which are assumed to spawn at position $p_2^* = p_2$ at timestep $t_2^* = 1$ as well as position $p_3^* = p_{16}$ at timestep $t_3^* = 1$, respectively, and supposed to reach position $p_2^\dagger = p_3^\dagger = p_{13}$ at timestep $t_2^\dagger = t_3^\dagger = 5$.*

In order to consider this routing problem, at most $t^\dagger = \max\{t_1^\dagger, t_2^\dagger, t_3^\dagger\} = 8$ timesteps have to be considered.

Note that in many cases, the maximal number $t^\dagger$ of timesteps is not considered. Instead, the objective is to derive a routing in as few timesteps as possible.

B. Formulation

As a next step, all possible routing solutions – including the actual routing, respective pin assignments, as well as actuations – are represented symbolically.

Definition 3 (Symbolic Representation)**.** *Consider a DMFB with positions P, droplets D, a maximal number $t^\dagger$ of timesteps to be considered, and a maximal number n_{pins} of pins to be considered. Then, a* symbolic representation *of all possible solutions can be defined over*

- *Boolean variables $c_{p,d}^t$ which represent whether the cell at position $p \in P$ is occupied by the droplet $d \in D$ at timestep $t \leq t^\dagger$,*
- *natural numbers pin_p (including 0) that represent the pin number that controls the electrode of the cell at position $p \in P$, and*
- *Boolean variables act_p^t representing whether in timestep $t < t^\dagger$ the pin controlling the cell at position $p \in P$ is activated or not.*

In fact, these variables are sufficient to symbolically represent *all* possible routing paths of droplets, pin assignments, pin-actuations, etc. independently from a specific routing task.

Example 6. *Consider again the routing problem from Fig. 2(a) which has been discussed before in Example 5. Additionally assume that the number of timesteps is restricted to $t^\dagger = 8$ and the number of pins is restricted to $n_{pins} = 9$. Then, the symbolic formulation representing all possible routing paths, pin assignments, pin-actuations, etc. for this DMFB is composed of 408 $c^t_{p,d}$-variables, 9 pin_p-variables, and 136 act^t_p-variables. Fig. 2(b) shows a possible solution for the problem from Example 5. The paths of the droplets are depicted by the arrows while the corresponding pin assignment for each cell is depicted by the number printed in the center of the cell. Note that at position p_{14} the droplets d_2 and d_3 merge and continue merged to their destination. Furthermore d_3 has to move right first in order to avoid unintentional merging with d_1.*

This solution can be derived from the symbolic formulation by the following assignments:

- *Realization of net N_1 (N_2 is done analogously):* $c^1_{p_{10},d_1} = c^2_{p_{11},d_1} = c^3_{p_{12},d_1} = c^4_{p_5,d_1} = c^5_{p_6,d_1} = \ldots = 1$
- *Pin assignment:* $pin_{p_1} = 8$, $pin_{p_2} = 9$, $pin_{p_3} = 3$, $pin_{p4} = 5$, $\ldots$
- *Activation of cells (exemplarily at timestep 1):* $act^1_{p_2} = 1$, $act^1_{p_{10}} = 1$, $act^1_{p_{16}} = 1$

All remaining variables are set to 0.

However, in its current form this symbolic formulation also enables the derivation of invalid solutions, e.g., by arbitrarily assigning the variables despite their semantic meaning. In order to avoid that, the following constraints are invoked on the variables introduced in Definition 3:

Source and Target Configuration

Source and target positions must be occupied by the droplets at the respective spawn time and targeted arrival time, i.e.

$$\bigwedge_{d \in D} c^{t^*_d}_{p^*_d,d} \ \wedge \ c^{t^\dagger_d}_{p^\dagger_d,d}.$$

Droplet Movement

Droplets must not arbitrarily occupy a cell position. In fact, a droplet may be present at a position p, iff it was present in the neighborhood of this position in the previous timestep, i.e.

$$\bigwedge_{d \in D} \bigwedge_{p \in P} \bigwedge_{t > t^*_d}^{t^\dagger_d} \left(c^t_{p,d} \Rightarrow \bigvee_{p' \in N_5(p)} c^{t-1}_{p',d} \right).$$

Note that this constraint does not apply at the spawn-time of a droplet (hence the condition $t > t^*_d$ rather than $t \geq t^*_d$). In this timestep, the droplet is directly placed onto the chip by the constraint for source and target configuration introduced above.

Consistency Constraints

The constraints above do not prohibit that a droplet $d \in D$ may simultaneously occur at different positions within the same timestep (particularly the constraints for droplet movement allow several instances of the same droplet to occur onto the DMFB). Obviously, this should be avoided. Hence, the following consistency constraints are added which ensure (1) that droplets are not present before being spawned and after they reached their target position as well as (2) that they only appear once while they are present:

$$\bigwedge_{d \in D} \bigwedge_{t=1}^{t^\dagger} \begin{cases} \overline{\bigvee_{p \in P} c^t_{p,d}} & (t < t^*_d) \vee (t^\dagger_d < t) \\ \sum_{p \in P} c^t_{p,d} = 1 & t^*_d \leq t \leq t^\dagger_d \end{cases}$$

Fluidic Constraints

In order to avoid unintended mixing of droplets, so called *fluidic constraints* must be satisfied (see e.g., [5]). The basic idea is to ensure that two droplets from different nets are supposed to never occupy cells in their respective neighborhood. This can be accomplished by enforcing

$$\bigwedge_{n \in N} \bigwedge_{d \in n} \bigwedge_{\substack{d' \in D \\ d' \notin n}} \bigwedge_{\substack{1 \leq t \leq t^\dagger \\ p \in P}} \left(c^t_{p,d} \Rightarrow \bigwedge_{p' \in N_9(p)} \overline{c^t_{p',d'}} \wedge \overline{c^{t-1}_{p',d'}} \right).$$

Blockages

Droplets must not occupy cells which are blocked. Blockages are caused e.g., by operations which are in progress on the DMBF at given timesteps. Assume that all blockages are given as tuples (b,t) with $b \in P$ being the position assumed to be blocked at timestep $1 \leq t \leq t^\dagger$. The set of all blockages is denoted by B. Then, blockages are taken into account by

$$\bigwedge_{(b,t) \in B} \left(\overline{\bigvee_{d \in D} c^t_{b,d}} \right).$$

Pin Assignment

Finally, constraints are added which ensure a valid consideration of pins, pin assignments, as well as the respective actuation. First, it is constrained that all electrodes are controlled by not more than n_{pins} pins, i.e.

$$\bigwedge_{p \in P} pin_p < n_{pins}.$$

Next, it has to be ensured that, whenever a droplet is occupying a cell, the corresponding pin controlling the electrode of that cell (i.e. position) must be activated, i.e.

$$\bigwedge_{p \in P} \bigwedge_{d \in D} \bigwedge_{t=1}^{t^\dagger} c^t_{p,d} \Rightarrow act^t_p.$$

The next constraint ensures that if two cells are not actuated the same way (in a given timestep) their corresponding pins are also different, i.e.

$$\bigwedge_{t=1}^{t^\dagger} \bigwedge_{p \in P} \bigwedge_{p' \in P} act^t_p \neq act^t_{p'} \Rightarrow pin_p \neq pin_{p'}.$$

Finally, pins must never control electrodes of cells in a fashion that would lead to undesired splittings. This may happen when a droplet occupies a cell at position $p \in P$ (requiring the corresponding pin being activated, i.e. act_p^t is true) and, at the same time, a pin controlling one of the neighboring cells with position $p' \in N_9(P)$ is activated as well. One further has to ensure that only one neighboring cell is being actuated in the following timestep. This is finally prohibited by the constraint

$$\bigwedge_{d \in D} \bigwedge_{p \in P} \bigwedge_{t=1}^{t^\dagger} \left(c_{p,d}^t \Rightarrow \bigwedge_{p' \in N_9^*(p)} \overline{act_{p'}^t} \wedge \sum_{p' \in N_9(p)} act_{p'}^{t+1} = 1 \right).$$

IV. Applications & Evaluation

The symbolic formulation presented in the previous section can be applied for solving a variety of routing tasks. To this end, the precise task just needs to be

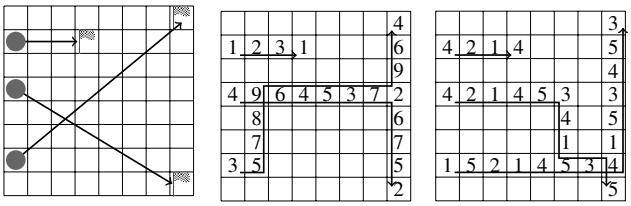
1) *configured*, i.e. the variables of the symbolic formulation (in particular, the corresponding position/time and target position/time for all droplets) have to be set appropriately, and
2) *solved*, i.e. a suitable solving engine (e.g., based on ILP, SAT, SMT, etc.) has to be applied in order to derive an explicit assignment for all remaining variables and, finally, an explicit solution.

For instance, if a precise pin assignment for a given routing is desired, the values of the $c_{p,d}^t$-variables (representing the cell positions which are occupied by the droplets according to the given routing) are set first. Afterwards, the values of the remaining variables (particularly for the pin_p-variables representing the desired pin assignment) are to be determined by the solving engine. If this was successful, a valid pin assignment for the given routing is derived. If, in contrast the solving engine concludes that no assignment satisfying all constraints exists, it has been proven that – under the given parameters – no valid solution exists. This may be the case if e.g., the number n_{pins} of pins was set too low. Then, the same task can be repeated with a larger value of n_{pins}.

In this fashion, the full variety of routing tasks reviewed in Section II (and much more) can eventually be addressed by a single (generic) solution. In the remainder of this section, we exemplary discuss the application of this methodology and compare the obtained results with solutions determined by related work. To this end, *Z3* [23] is used as the corresponding solving engine. All evaluations have been conducted on a Fedora 20 machine with 3.5 GHz and 32 GB of memory.

A. Performing Pin-aware Routing

The most general case that can be addressed by the proposed methodology is pin-aware routing, as reviewed in Section II-C. Recall that the main objective is to determine the proper movements of all droplets *and* a corresponding pin mapping which realizes the routing. In order to configure this task, the source positions p_d^* and target positions $p_d^\dagger$ as well as the corresponding spawn times t_d^* and, if needed, desired target times $t_d^\dagger$ have to be properly initialized for each droplet $d \in D$. Afterwards, the solving engine determines an assignment for, besides others, the remaining $c_{p,d}^t$- and pin_p-variables. From those, the cell positions which are occupied by the droplets and, hence, define the resulting routing as well as the precise pin mapping can be derived.



(a) Routing Task (b) Sol. from [14] (c) Minimal sol.

Fig. 3. Performing pin-aware routing

In order to illustrate the applicability and efficiency of the solution, consider the routing task shown in Fig. 3(a), which has been taken from [14]. The goal is to determine a routing and pin mapping with as few pins as possible (i.e. with n_{pins} being as small as possible). In [14] a heuristic approach was utilized, which leads to the result as shown in Fig. 3(b), requiring a total of $n_{pins} = 9$ pins.

In contrast, the generic solution proposed in this work can be applied to determine an *optimal* result. To this end, a pin mapping with $n_{pins} = 1$ is considered first. If the solver proves that no such solution exists, n_{pins} is iteratively increased by 1 until a solution and, hence, a routing as well as pin mapping is found. This leads to the *optimal* solution with $n_{pins} = 5$ pins as shown in Fig 3(c). Obtaining this solution required negligible run-time. Hence we have demonstrated that, using a general optimization framework that targets all the routing-related problems in a unified manner, we have obtained a significant advance over a previous heuristic technique that targets only this sub-problem.

Moreover, the proposed solution can even be applied for the evaluation of commercially produced DMFBs. To demonstrate that, we considered the investigations from [13] in which the authors aimed to evaluate whether a fixed pin mapping (chosen by the manufacturer) is optimal for a given set of assays. As an example, we consider the result for the reaction region of the commercial DMFBs which performs an "n-Plex Immunoassay" (the layout of this chip is shown in Fig. 4). This chip was produced by Advanced Liquid Logic in 2013 and, subsequently, the company and all its intellectual property were acquired by the DNA sequencing company Illumina. The original pin mapping (chosen by the manufacturer) required a total of 19 pins. In [13], the authors were able to show that the assay can be realized with a total of 13 pins, but their approach required more than a day of compute time in order to determine this result. In contrast, we were able to determine a *minimal* pin mapping requiring a total of only 4 pins in less than 5 minutes. This is again a clear and striking improvement compared to the previously proposed solutions.

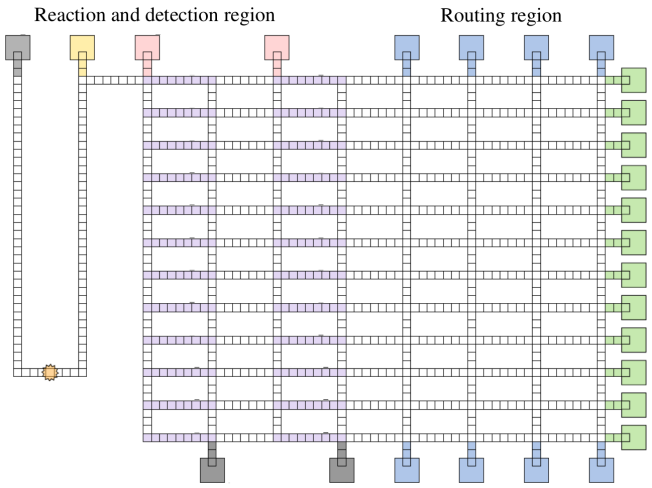


Fig. 4. Layout of a commercially available chip manufactured by [24].

TABLE I
EVALUATION OF THE EFFICIENCY OF THE PROPOSED METHOD

Benchmark	$\|P\|$	$\|D\|$	$\|\mathcal{N}\|$	CPU time (s)
protein2.02	169	3	3	48.47
protein2.07	169	4	4	141.65
protein2.23	169	2	2	29.88
protein2.49	169	4	4	31.74
protein2.40	169	4	4	77.60
protein2.64	169	2	2	9.90
protein2.70	169	4	3	92.88
in-vitro1.09	256	2	2	140.38
in-vitro1.10	256	1	1	91.6
in-vitro2.05	196	4	3	75.18
in-vitro2.12	196	2	2	14.40
in-vitro2.15	196	1	1	15.54

The proposed generic formulation can easily be applied to further pin-aware routing tasks. However, due to a lack of descriptions of the actual experiments conducted in the related work, previously reported results are not reproducible. Since precise comparison to previously obtained solutions is not possible, we summarize some evaluations on the efficiency of the proposed solution in Table I. Here, the obtained results for a variety of different pin-aware routing tasks (inspired by related work) are listed. The first columns provide the applied benchmarks (taken from [18] and denoted by *Benchmark*), the number of considered positions (denoted by $|P|$), the number of considered droplets (denoted by $|D|$), and the number of applied nets (denoted by $|\mathcal{N}|$). Finally, the required run-time (in CPU seconds) needed to obtain both the routing as well as the pin mapping is listed (denoted by *CPU time*). As can clearly be seen, for a variety of pin-aware routing tasks which similarly have also been considered in related work (such as [14], [20]), (minimal) solutions can be obtained in a rather efficient fashion.

Pin-aware routing represents the most complex case of routing tasks which can be handled by the proposed approach – all remaining routing tasks can be solved in much less run-time. Since Table I already confirmed the efficiency in this regard and due to page limitations, further run-time evaluations are omitted. Instead, we focus on demonstrating the generic applicability of the proposed solution.

B. Performing Actual Routing

As the next example of a routing task, we consider actual routing as reviewed in Section II-A. Recall that the main objective is to determine routes for all the droplets without an explicit consideration of the respective pin assignments. In order to configure this task, a simple direct-addressing scheme can be applied on the pins, i.e. in a DMFB layout composed of $|P|$ positions, the cell at position p_1 is controlled by the first pin (i.e. $pin_{p_1} = 1$), the cell at position p_2 is controlled by the second pin (i.e. $pin_{p_2} = 2$), and so on. By this, the precise pin mapping becomes irrelevant and the solver only has to determine the actual movements of the droplets in order to realize the routing. In doing so, all problems as considered in [9], [10], [17], [20] can easily be addressed without the need for the specific suboptimal solutions proposed in previous work.

C. Optimizing the Pin Mapping

Another task is to optimize a pin mapping for a given plan for droplet movement – again with the objective of keeping the total number n_{pins} of pins as small as possible (as reviewed in Section II-B). In order to configure this task, simply all $c^t_{p,d}$-variables which represent the given movement of droplets have to be set to 1. Following the iterative scheme as applied for pin-aware routing (first looking for a pin mapping with $n_{pins} = 1$; if this fails, increase n_{pins} by 1), this leads to a minimal result and, hence, allows for addressing all the problems as e.g., considered in [12]. Moreover, since our approach guarantees minimality, some very substantial improvements can be obtained. For example, the authors of [12] intensely discussed the pin mapping problem for PCR for which they obtained a solution with $n_{pins} = 14$ pins (see [12, Fig. 13]) – our approach yields a solution that requires only $n_{pins} = 5$ pins (see Fig. 5). Again, our results are striking, they imply that exact optimization is computationally feasible.

D. Verifying a Given Pin Mapping

Finally, we discuss a task that has not been considered yet in the literature, but is crucial in practice: As discussed in Section II-D, many commercially available DMFBs already include a fixed pin mapping. However, since the pin mapping significantly affects how and even whether droplets may be moved on the chip, it is not obvious whether all positions of a DMBF indeed can be reached under a fixed pin mapping. As an example, consider the 4×4-layout with a fixed pin assignment as shown in Fig. 6. The given routing task is to

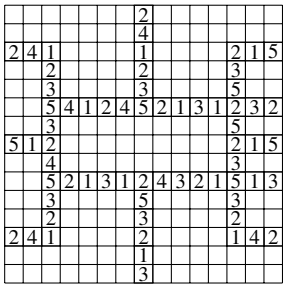


Fig. 5: Optimizing the pin mapping

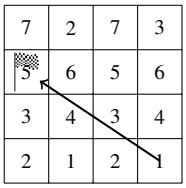


Fig. 6: Pin mapping with impossible movement

move a droplet from position p_4 to position p_9. However, under the applied pin assignment, no route can be realized which can accomplish that.

While this might not be obvious at a first glance (and even heuristics would be incapable of *proving* the non-existence of such a route), our generic formulation symbolically considers *all* possible solutions. Hence, the given pin assignments (i.e. assignments to the pin_p-variables) as well as the source position p_d^* and the target position $p_d^\dagger$ simply have to be set to corresponding values. Then, a solving engine would conclude that, under this assignment, no valid solution exists anymore – thereby proving that this movement is not possible under the given pin mapping. Depending on the considered application scenario, serious design flaws might be detected.

V. Conclusions

The development of EDA methods for DMFBs resulted in an "inflation" of different design approaches for the routing problem of DMFBs. While most of them are applicable for very dedicated tasks only, we proposed a generic and exact solution which comprehensively covers various routing issues. To this end, a generic formulation has been proposed that symbolically represents all possible solutions and can accordingly be configured in order to derive the actually desired one. Several case studies have shown the applicability of the proposed methodology for various scenarios. Besides that, also many further derivations of routing tasks (e.g., for online-routing [21], error correction [22], and more) can in principle be handled with this solution.

Acknowledgements

This research was supported in part by a Humboldt Research Award to Krishnendu Chakrabarty, conferred by the Alexander von Humboldt Foundation, Germany.

References

[1] Tsung-Yi Ho, Jun Zeng, and Krishnendu Chakrabarty. Digital microfluidic biochips: A vision for functional diversity and more than Moore. In *Int'l Conf. on CAD*, pages 578–585. IEEE Press, 2010.

[2] MG Pollack, AD Shenderov, and RB Fair. Electrowetting-based actuation of droplets for integrated microfluidics. *Lab on a Chip*, 2(2):96–101, 2002.

[3] Cliff Chiung-Yu Lin and Yao-Wen Chang. ILP-Based Pin-Count Aware Design Methodology for Microfluidic Biochips. *IEEE Trans. on CAD*, 29(9):1315–1327, 2010.

[4] Tsung-Wei Huang, Jia-Wen Chang, and Tsung-Yi Ho. Integrated Fluidic-Chip Co-Design Methodology for Digital Microfluidic Biochips. In *International Symposium on Physical Design*, pages 49–56. ACM, 2012.

[5] Fei Su, William Hwang, and Krishnendu Chakrabarty. Droplet routing in the synthesis of digital microfluidic biochips. In *Design, Automation and Test in Europe*, volume 1, pages 1–6. IEEE, 2006.

[6] Ying-Han Chen, Chung-Lun Hsu, Li-Chen Tsai, Tsung-Wei Huang, and Tsung-Yi Ho. A reliability-oriented placement algorithm for reconfigurable digital microfluidic biochips using 3D deferred decision making technique. *IEEE TCAD*, 32(8):1151–1162, 2013.

[7] Andrew J Ricketts, Kevin Irick, Narayanan Vijaykrishnan, and Mary Jane Irwin. Priority scheduling in digital microfluidics-based biochips. In *DATE*, pages 329–334, 2006.

[8] Oliver Keszocze, Robert Wille, Tsung-Yi Ho, and Rolf Drechsler. Exact One-pass Synthesis of Digital Microfluidic Biochips. In *Design Automation Conf.*, 2014.

[9] Ping-Hung Yuh, Chia-Lin Yang, and Yao-Wen Chang. BioRoute: A network-flow based routing algorithm for digital microfluidic biochips. In *Int'l Conf. on CAD*, pages 752–757. IEEE Press, 2007.

[10] Minsik Cho and David Z Pan. A high-performance droplet routing algorithm for digital microfluidic biochips. *IEEE Trans. on CAD*, 27(10):1714–1724, 2008.

[11] Tao Xu and Krishnendu Chakrabarty. Droplet-Trace-Based Array Partitioning and a Pin Assignment Algorithm for the Automated Design of Digital Microfluidic Biochips. In *Conference on Hardware/Software Codesign and System Synthesis*, pages 112–117. ACM, 2006.

[12] Tao Xu and Krishnendu Chakrabarty. Broadcast Electrode-Addressing for Pin-Constrained Multi-Functional Digital Microfluidic Biochips. In *Design Automation Conf.*, pages 173–178. ACM, 2008.

[13] Yang Zhao and Krishnendu Chakrabarty. Simultaneous Optimization of Droplet Routing and Control-Pin Mapping to Electrodes in Digital Microfluidic Biochips. *IEEE Trans. on CAD*, 31(2):242–254, 2012.

[14] Tsung-Wei Huang and Tsung-Yi Ho. A Two-Stage ILP-Based Droplet Routing Algorithm for Pin-Constrained Digital Microfluidic Biochips. In *International Symposium on Physical Design*, pages 201–208. ACM, 2010.

[15] Zhishan Hua, Jeremy L. Rouse, Allen E. Eckhardt, Vijay Srinivasan, Vamsee K. Pamula, Wiley A. Schell, Jonathan L. Benton, Thomas G. Mitchell, and Michael G. Pollack. Multiplexed Real-Time Polymerase Chain Reaction on a Digital Microfluidic Platform. *Analytical Chemistry*, 82(6):2310–2316, 2010. PMID: 20151681.

[16] Ramakrishna Sista, Zhishan Hua, Prasanna Thwar, Arjun Sudarsan, Vijay Srinivasan, Allen Eckhardt, Michael Pollack, and Vamsee Pamula. Development of a digital microfluidic platform for point of care testing. *Lab on a Chip*, 8(12):2091–2104, 2008.

[17] Tsung-Wei Huang and Tsung-Yi Ho. A fast routability- and performance-driven droplet routing algorithm for digital microfluidic biochips. In *Int'l Conf. on Comp. Design*, pages 445–450. IEEE, 2009.

[18] Ping-Hung Yuh, Sachin Sapatnekar, Chia-Lin Yang, and Yao-Wen Chang. A progressive-ilp based routing algorithm for cross-referencing biochips. In *Design Automation Conf.*, pages 284–289. ACM, 2008.

[19] Ping-Hung Yuh, Cliff Chiung-Yu Lin, Tsung-Wei Huang, Tsung-Yi Ho, Chia-Lin Yang, and Yao-Wen Chang. A SAT-based routing algorithm for cross-referencing biochips. In *System Level Interconnect Prediction Workshop*, pages 6:1–6:7. IEEE Press, 2011.

[20] Oliver Keszocze, Robert Wille, and Rolf Drechsler. Exact routing for digital microfluidic biochips with temporary blockages. In *Int'l Conf. on CAD*, pages 405–410. IEEE Press, 2014.

[21] Daniel Grissom and Philip Brisk. Fast online synthesis of generally programmable digital microfluidic biochips. In *International Conference on Hardware/Software Codesign and System Synthesis*, pages 413–422. ACM, 2012.

[22] Kai Hu, Bang-Ning Hsu, Andrew Madison, Krishnendu Chakrabarty, and Richard Fair. Fault detection, real-time error recovery, and experimental demonstration for digital microfluidic biochips. In *Design Automation Conf.*, pages 559–564. EDA Consortium, 2013.

[23] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008. Z3 is available at http://z3.codeplex.com/.

[24] Advanced liquid logic, inc. http://www.liquid-logic.com.