A Discrete Model for Networked Labs-on-Chips: Linking the Physical World to Design Automation

Andreas Grimmer[1] Werner Haselmayr[2] Andreas Springer[2] Robert Wille[1]
[1]Institute for Integrated Circuits
[2]Institute for Communications Engineering and RF-Systems
Johannes Kepler University Linz, Austria
{andreas.grimmer,werner.haselmayr,andreas.springer,robert.wille}@jku.at

ABSTRACT

Labs-on-Chip integrate and minimize the functionality of complete conventional laboratories on a single chip. An upcoming and especially biocompatible realization are *Networked Labs-on-Chips* (NLoCs). In NLoCs, small volumes of reagents, so-called droplets, flow in an immiscible fluid in closed channels. An external pump applies a force to this immiscible fluid driving the droplets through the channels of the NLoC. However, the exact flow behavior of droplets in NLoCs physically depends on many factors and interdependencies. This makes it cumbersome to manually determine the taken path of a droplet and the time it needs to pass the NLoC. For the same reason, also almost no automated design solutions exist for NLoCs yet. In this work, we present a discrete model enabling designers and design automation tools to efficiently determine the droplets' path and positions. The precision of the proposed model is evaluated by a systematic examination for basic building blocks of NLoCs as well as for a complete architecture. The resulting model can be used for manual inspections of the droplets' behavior in an NLoC and, additionally, provides the basis for automated design solutions.

1. INTRODUCTION

The domain of *microfluidics* is a distinct new field, which deals with the manipulation of small amounts of fluids [25]. Corresponding devices are called *Labs-on-Chips* (LoCs) and have the potential to revolutionize chemical and biomedical procedures [9]. LoCs offer significant advantages compared to conventional processes, e.g. they only require small sample quantities, work without human interaction, and, at the same time, provide high precision and throughput. Therefore, their usage strongly increased in the last decade, e.g. for in-vitro diagnostics, DNA sequencing, cell analysis, drug screening, or protein crystallization [15, 20].

An upcoming technology for LoCs are so called *Networked Labs-on-Chips* (NLoCs, [7, 8, 24]), where small volumes (in the order of few micro- to pico-liters) – so-called *droplets* – flow within an immiscible fluid in closed channels. An external pump applies a force to the fluid driving the droplets through the channels of the NLoC. By this, the droplets are transported to different modules, which execute chemical/biological operations on the droplets. The closed channels allow an especially biocompatible realization as they prevent evaporation and unwanted contamination. Hence, they allow for a long-term incubation and storage of droplets [11, 20]. As discussed in [10, 11], this addresses severe shortcomings of alternative LoC technologies such as electrowetting- or flow-valve-based LoCs.

However, the design and realization of a desired chemical or biomedical procedure onto a given NLoC architecture is a non-trivial task. In fact, a dedicated *routing* has to be determined which passes a so-called *payload* droplet (containing the biological sample) through a sequence of modules (conducting the desired operations). To this end, so-called *header* droplets are employed which change the flow in a way so that the desired path is taken [7, 10]. Despite the combinatorial complexity of this problem, the exact flow behavior of the droplets in an NLoC additionally depends on many physical factors and interdependencies.

Because of this, it is a cumbersome task to manually create a proper droplet sequence consisting of a payload and header droplets realizing the desired experiment or to even simply predict the taken path of a single droplet within a given NLoC architecture. Moreover, for the same reason, only few automated design solutions exist for NLoCs yet. As a consequence, experiments are thus far directly designed by "trial-and-error" approaches, i.e. manually testing various droplet sequences as long as, eventually, one sequence realizing the desired experiment is obtained. Obviously, this is a time-consuming and costly endeavor which does not even always guarantee success.

In this work, we present a main prerequisite to overcome these problems: A discrete model which allows for a consideration of these design tasks on a more abstract level, i.e. without an explicit consideration of the physical behavior or even by means of methods for design automation. To this end, we investigate the physical behavior of arbitrary NLoC architectures which leads to a description that requires a constant (re-)evaluation of complex equation systems. As the resulting complexity (particularly combined with actual design problems) makes this description infeasible for both, a manual consideration but also possible design automation methods, we afterwards propose an abstraction which yields a discrete model for this purpose.

The resulting model enables designers and design automation tools to intuitively and efficiently determine the droplets' paths and positions during the execution of an experiment and, by this, to tackle design tasks such as simulation, droplet sequence generation, and verification. As an example, the proposed model has already successfully been applied to verify whether an NLoC architecture indeed allows for the realization of the desired experiments [13]. Our evaluations show the precision of the discrete model in general as well as with respect to different resolutions. Overall, with this work, we are linking the physical world of NLoCs to the domain of design and design automation.

2. MAIN CONCEPT OF NLOCS

In *Networked Labs-on-Chips* (NLoC, [7, 8, 24]), a small volume of a biological sample, the so-called *payload droplet*, flows in *closed channels* of sub-millimeter diameters. In order to realize an experiment, this payload droplet has to flow through a desired sequence of *modules* [8], which are connected by channels. These modules execute elementary operations like mixing, splitting, fusing, detecting, or heating on the payload droplet. In order to route the payload droplet to the desired modules, so-called *header* droplets are utilized to temporary block channels that must not be taken by the payload. The header droplets used for this routing must not *coalesce* (mix) with the payload droplet because this would destroy the sample in the payload droplet. In this section, we review the respective concepts, i.e. how channels and modules are employed to build an NLoC architecture and how payload and header droplets are routed through this architecture. Finally, we discuss the resulting design problems.

2.1 Architecture

An NLoC architecture consists of an external *pump*, a set of *channels*, and a set of *modules*, which have the following functionalities:

- The *pump* injects a *continuous fluid* into the input channel of the NLoC by applying a force on this fluid. Additionally, the pump is combined with a logic for the droplet generation (e.g. T-junctions or flow focusing geometries [14]). The generated payload and header droplets flow inside this continuous fluid through the NLoC.
- The set of *channels* C allows for a directed flow of the continuous fluid and, therefore, of the droplets.
- The set of *modules* M defines the available operations, which can be executed on the payload droplet. Since header droplets must not be executed by a module, they are forwarded. To this end, an integrated *sorter* [23] is applied which distinguishes the payload droplet from the header droplets by their different volumes.

The *architecture* defines how the pump, the channels, and the modules are connected in a closed network. Using so-called *bifurcations* (i.e. splitting a channel into two or more successor channels), multiple paths through which the droplets can flow are realized. Each of these paths represents a different experiment.

Example 1. *Figure 1 shows a sketch of an NLoC architecture, which consists of a pump and channels* $C = \{c_{in}, c_1, c_2, \ldots, c_9\}$ *connecting two modules* $M = \{m_1, m_2\}$. *Additionally, the architecture contains a bifurcation which splits the input channel* $c_{in} \in C$ *in two successor channels* $c_1, c_2 \in C$. *This bifurcation and a corresponding droplet routing enables the designer to decide whether the payload droplet should first be heated or not before it gets analyzed by the detector module (i.e. this architecture allows the designer to choose between two different experiments). The sorters (denoted by* Ⓢ*) ensure that only the payload droplet is executed by the module, while header droplets are forwarded.*

The principle of how to route the payload droplet through the architecture using header droplets is described next.

2.2 Droplet Routing

The droplet routing in NLoCs is based on the different fluidic resistances of channels. The fluidic resistance of a channel, is mainly defined by its *geometry* [7, 17], e.g. the smaller the section and the longer the channel, the higher the resistance. For supporting the droplet routing, the successor channels of bifurcations have different fluidic resistances. When a droplet arrives at a bifurcation, it flows along the successor channel with the lowest fluidic resistance (denoted *default successor* in the following).

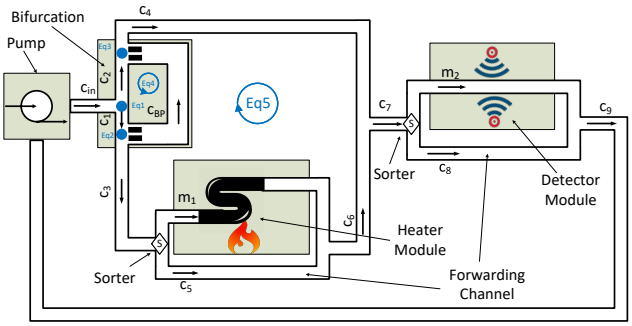


Figure 1: Sketch of an NLoC architecture

Example 2. *The architecture in Figure 1 contains a bifurcation, which splits the input channel* c_{in} *into two successor channels* c_1 *and* c_2. *Assume that the fluidic resistance of successor channel* c_1 *is lower than those of* c_2 *(i.e.* c_1 *is the default successor). When the pump with the droplet generation logic now injects a payload droplet, it will flow along the successor channel* c_1 *(since its fluidic resistance is lower than those of* c_2*), gets heated by module* m_1*, analyzed by module* m_2*, and eventually flows back to the pump.*

However, droplets themselves increase the fluidic resistance of a channel [7, 10]. This principle is used at bifurcations, i.e. they are designed so that, when the default successor already contains a droplet, a closely following droplet will eventually take the other channel. This way, a payload can be routed to other modules.

Example 3. *Consider the bifurcation from the architecture of Figure 1. Additionally, assume a header droplet in channel* c_1 *and a closely following payload droplet. This payload droplet will flow into channel* c_2 *because the header droplet increases the resistance of* c_1 *so that, now, the resistance of* c_2 *is lower*[1]. *Hence, the payload droplet does not get heated but directly flows into the detector module.*

Note that the successor channels of a bifurcation are connected with a wide channel, a so-called *bypass*, which cannot be entered by any droplet. This bypass decouples the bifurcation from the rest of the NLoC architecture, which makes the droplet routing only dependent on the resistances of the immediate successor channels as well as the fact whether they are occupied by a droplet [6].

2.3 Resulting Design Problems

Based on the concepts from above, several design problems emerge. A selection of design tasks includes e.g.:

- *Droplet Sequence Generation*, i.e. determine a sequence of a payload and header droplets which realizes the desired experiment. Potentially multiple header droplets are used to route the payload droplet through the desired modules and, for all those, respective injection times have to be determined.
- *Simulation*, i.e. determine the evaluation of a sequence of droplets. This is required e.g. to validate a droplet sequence or to estimate the durations of experiments.
- *Verification* [13], i.e. check whether an NLoC architecture allows to execute a given set of experiments.

Thus far, these design tasks have often been conducted in a "trial-and-error" fashion, i.e. by manually testing various hand-crafted droplet sequences. This resulted in a time-consuming and costly process as several droplet sequences explicitly need to be tested. In this work, we aim to overcome this problem by providing a discrete model that allows for a consideration of these design tasks on a more abstract level. To this end, we first investigate the physical behavior of NLoCs as a basis for the contribution of this work.

[1] A video at http://www.jku.at/iic/eda/nloc shows a physical realization where two consecutive droplets take different successor channels.

3. PHYSICAL BEHAVIOR OF NLOCS

While the concepts reviewed in the previous section are rather straightforward, the physical behavior of NLoCs depends on multiple properties and interdependencies. The exact flow behavior of droplets in NLoCs physically depends not only on the geometry of the channels and modules but also interdependencies throughout the architecture as well as all involved droplets. In this section, we investigate the "real world" behavior of NLoCs, i.e. their physical behavior, and, by this, provide the basis out of which a compatible model for design automation is derived.

3.1 Flow Distribution

The pump injects the continuous fluid so that a flow through the channels and modules of the NLoC results. Inside this continuous fluid, the droplets flow through the NLoC. Each channel and module of the NLoC poses a resistance for the flow and, therefore, the overall flow distributes over all channels and modules depending on the respective resistances. The following physical properties describe this flow distribution over the architecture:

- The *volumetric flow rate* Q provides the volume of the fluid which passes a channel $c \in C$ or module $m \in M$ per time unit (in $[m^3/s]$).
- The *fluidic resistance* R provides the difficulty for passing a volumetric flow through a channel $c \in C$ or module $m \in M$ (in $[Pa\ s/m^3]$).
- The *pressure gradient* ΔP provides the change of pressure between the ends of a channel $c \in C$ or module $m \in M$ (in $[Pa]$).

The *Hagen-Poiseuille* equation [1] describes the proportional relation between these physical properties with $\Delta P = R\ Q$. This equation is analogous to the *Ohm's law* of electronic circuits (i.e. $U = R\ I$), which describes the relation between the current I (corresponds to the volumetric flow rate Q), the resistor R of a conductor (corresponds to the fluidic resistance R of a channel or module), and the voltage U measured across the conductor (corresponds to the pressure gradient ΔP). Hence, the physical behavior of NLoCs can be described using the laws from electronic engineering [2, 21]. In the following, this is used to describe (1) the behavior of the pump producing the force driving the droplets through the NLoC, (2) the resistances of channels and modules, and (3) how these resistances determine the flow rates and pressure gradients.

(1) Behavior of the Pump: A *pump* injects a *continuous fluid* with a given viscosity μ_{cont} (in $[Pa\ s]$) into the input channel $c_{in} \in C$ of the NLoC. Two different realizations of pumps can be used for this purpose: a *syringe pump* applies a volumetric flow rate Q_{in} to c_{in} (cf. a current source in electronic circuits), while a *peristaltic pump* imposes a pressure gradient ΔP_{in} to c_{in} (cf. a voltage source in electronic circuits). How either the incoming volumetric flow rate Q_{in} or the applied pressure gradient ΔP_{in} distributes over the channels and modules of the NLoC depends on their specification and their arrangement within the architecture.

(2) Resistances of Channels and Modules: The specification of a channel $c \in C$ defines its fluidic resistance R_c. Assuming the channel is *not occupied* by any droplets, then the resistance R_c of a channel is exclusively specified by its rectangular section with width w_c and height h_c as well as its length l_c (all in $[m]$). More precisely [12], the resistance R_c of a channel c is

$$R_c = \frac{\alpha\ \mu_{cont}\ l_c}{w_c\ h_c^3}, \tag{1}$$

where α denotes a dimensionless parameter defined as

$$\alpha = 12 \left[1 - \frac{192\ h_c}{\pi^5\ w_c} \tanh\left(\frac{\pi\ w_c}{2\ h_c} \right) \right]^{-1}. \tag{2}$$

Accordingly, a module $m \in M$ also defines a fluidic resistances R_m, which also depends on its component specification.

Besides that, an NLoC employs these channels and modules in an architecture, for which the same rules as in electronic circuits are applicable, i.e.

- the resistance of *serial* channels or modules adds together, i.e. the resistance of two serial channels is $R_{c_1+c_2} = R_{c_1} + R_{c_2}$, and
- the resistance of *parallel* channels or modules is defined by adding their reciprocal resistances and building the inverse, i.e. the resistance of two parallel channels is $R_{c_1||c_2} = (1/R_{c_1} + 1/R_{c_2})^{-1}$.

Overall, this allows us to determine the resistances of an NLoC architecture.

(3) Resulting Flow Rates: Now, these basics allow for a determination of the respective flow rates for each channel $c \in C$ and each module $m \in M$. In fact, the flow rate Q of each channel and module depends on (1) the applied input flow rate or pressure gradient of the pump (depending on what pump is applied), (2) the resistance R of the channel or module itself and on all other resistances and their composition in the NLoC[2]. All this is incorporated by the *Kirchhoff's Laws* [2, 21] which state the following:

- The sum of flow rates into a *node* is equal to the sum of flows rates out of that node. A node is a point in the architecture where a channel splits into multiple channels or where multiple channels merge to one channel.
- The directed sum of pressure gradients (cf. Hagen-Poiseuille with $\Delta P = R\ Q$) around any closed *cycle* in the architecture is zero. The sign of the pressure gradients thereby depend on the direction of the flow rates.

EXAMPLE 4. *Consider again the architecture shown in Figure 1. In order to determine the flow rates, an equation system is defined using the Kirchhoff's Laws. For example, the equations for three nodes and two cycles (namely, the ones highlighted by blue dots and blue cycles in Figure 1, respectively) are as follows:*

Eq1: $Q_{c_{in}} = Q_{c_1} + Q_{c_2}$
Eq2: $Q_{c_1} = Q_{BP} + Q_{c_3}$
Eq3: $Q_{c_2} + Q_{BP} = Q_{c_4}$
...
Eq4: $Q_{c_1}\ R_{c_1} + Q_{BP}\ R_{BP} - Q_{c_2}\ R_{c_2} = 0$
Eq5: $Q_{BP}\ R_{BP} + Q_{c_4}\ R_{c_4} - Q_{c_6}\ R_{c_6} - Q_{m_1}\ R_{m_1} - Q_{c_3}\ R_{c_3} = 0$
...

By solving this equation system, the flow rates Q of each channel $c \in C$ and each module $m \in M$ of the architecture are obtained.

3.2 Effect of Droplets

Determining the flow rates for each channel and module is a first step in order to completely describe the physical behavior of an NLoC. But as mentioned above, also the fact whether a channel or module is occupied by a droplet increases its fluidic resistance and, hence, has an effect on the flow rates [3,5,18,19]. This increase of the resistance is given by

$$\rho_c = (\mu_d - \mu_{cont}) \frac{l_d \alpha}{w_c\ h_c^3}, \tag{3}$$

where l_d is the length of the droplet and μ_d is the given viscosity of the droplet. Therefore, the overall fluidic resistance of a channel or module containing a droplet is given by $R_c + \rho_c$ or $R_m + \rho_m$, respectively.

EXAMPLE 5. *Consider again the architecture shown in Figure 1 and additionally assume a droplet in channel c_{in} and another one in channel c_1. The flow of these two droplets causes additional resistances in these channels, which have to be considered in the equation system from Example 4. For example, the flow of the droplet in channel c_1 changes Eq4 to*

Eq4: $Q_{c_1}\ (R_{c_1} + \rho_{c_1}) + Q_{BP}\ R_{BP} - Q_{c_2}\ R_{c_2} = 0.$

[2]Note that, additionally, the fact whether a channel or module is occupied by a droplet affects the flow rate. However, this is omitted here and addressed separately in Section 3.2.

3.3 Droplet-Routing with Resistance Changes

In order to exactly determine the flow distribution, also the chosen successor channel of droplets at bifurcations has to be considered. As discussed in Section 2.2 and illustrated by Example 3, this depends on the resistances of the successor channels which can be determined following the formalism from above.

EXAMPLE 6. *Again, consider the architecture shown in Figure 1 and especially its bifurcation with the following channel specification:*

	c_{in}	c_1	c_2	*given in*
height h	50	50	50	$10^{-6}m$
width w	50	50	50	$10^{-6}m$
length l	300	175	200	$10^{-6}m$

Because of that, channel c_{in}, c_1 and c_2 have the following resistances (assuming a droplet-free NLoC and a fluid viscosity of $\mu_{cont} = 10^{-3}$ Pa s):

	c_{in}	c_1	c_2	*given in*
R	1.3567	0.7914	0.9044	10^{12} $Pa\ s/m^3$

Since the resistance R_{c_1} is smaller than the resistance R_{c_2}, a single droplet occupying channel c_{in} will flow into successor channel c_1. Afterwards, the flow of this droplet (with the viscosity of $\mu_d = 1.5931 \cdot 10^{-3} Pa\ s$ and a length of $l_d = 60 \cdot 10^{-6} m$) through c_1 would increase its resistance to

$$R_{c_1} + \rho_{c_1} = 0.7914 + 0.17434 = 0.96574.$$

Since this resistance is now greater than the resistance R_{c_2}, a closely following second droplet will flow into the successor channel c_2.

Note that the bypass channel allows to decide the routing by only considering the resistances of the successor channels[3].

3.4 Overall Behavior

All the considerations from above allow for a comprehensive description of the physical behavior of droplets in an NLoC. In fact, using that, we can compute the *velocity* u (in $[m/s]$) in a channel and module by dividing its flow rate Q by its section $w\ h$, i.e.

$$u = \frac{Q}{w\ h}. \tag{4}$$

Therefore, a droplet flows with velocity u through the channel or module. Using these velocities in combination with the injection time of a droplet allows to determine its position. By additionally considering the behavior of droplets at bifurcations, the flow of all droplets in an NLoC architecture can be determined.

EXAMPLE 7. *Consider again the architecture shown in Figure 1 and its channel specification from Example 6. Additionally, assume that just now a droplet is injected into channel c_{in} and another flows in channel c_1. As long as both occupy these two channels, they flow with the following velocities:*

	c_{in}	c_1	c_2	*given in*
Q	10.00	4.84	5.16	10^{-12} m^3/s
u	4.00	1.93	2.07	10^{-3} m/s

These velocities now allow to determine how long the droplets require to pass a channel, e.g. the droplet in c_{in} requires $\frac{l_{in}}{u_{in}} = \frac{300 \cdot 10^{-6} m}{4 \cdot 10^{-3} m/s} = 75 \cdot 10^{-3}$ s. By this, the positions of each droplet in the architecture can be obtained.

However, since the droplets affect the resistances and, therefore, the flow rates of all channels, the corresponding velocities have to be re-calculated whenever a droplet gets injected into the architecture or moves from one channel to a succeeding channel [4].

[3]Technically, a bypass channel levels out the pressure gradients between the ends of the successor channels. Details can be found in [6].

EXAMPLE 8. *Consider again the situation from Example 7 (i.e. a droplet in channel c_{in} and another droplet in channel c_1). As soon as the droplet in c_{in} passes the bifurcation (whose exact time can be determined by the velocities), it will flow into the successor channel c_2 (while the other droplet still flows in channel c_1 at that time). This changes the resistances and, hence, also the flow rates of* all *channels in the NLoC. Accordingly, the velocities change as well:*

	c_{in}	c_1	c_2	*given in*
Q	10.00	4.87	5.12	10^{-12} m^3/s
u	4.00	1.95	2.05	10^{-3} m/s

As can be seen, the velocity in channel c_1 increases while the velocity in c_2 decreases. These values have to be re-evaluated for all other channels as well as whenever a droplet gets injected into the architecture or moves a channel.

Overall, the consideration from above indeed allows to exactly determine the velocities and, by this, the position of each droplet at each time. But as illustrated in the example, the flow rates and resistances of the channels are subject to constant changes. As a consequence, the equations systems and all dependencies discussed in the previous sections have to be constantly re-evaluated in order to guarantee a correct determination of the physical behavior. Obviously, the resulting complexity makes it infeasible to use this physical description for purposes of design automation. Hence, it is necessary to abstract from the physical behavior. In the remainder of this work, we address this issue by introducing a discrete model which is suitable for the (automatic) design of NLoCs, while still aims to rely as much as possible on the "real world" given by these physical descriptions.

4. DISCRETE MODEL FOR NLOCS

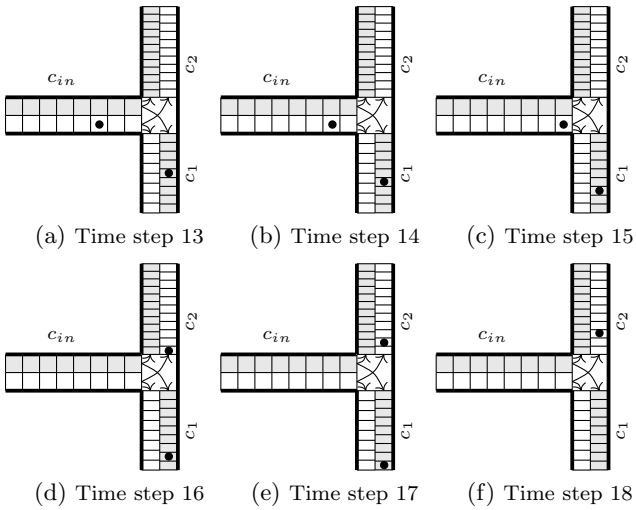
The discrete model introduced in this work shall be applied for the design of NLoCs involving typical tasks as discussed in Section 2.3. To this end, a discrete abstraction is proposed which enables designers and design automation tools to intuitively and efficiently determine the droplets' paths and positions during the execution of an experiment. At the same time, it avoids the complex determination of the physical behavior with its constant re-evaluations. In this section, we first describe the proposed model and show how an instance of the model for a given NLoC architecture can be derived. An evaluation of the precision of the model is afterwards provided in Section 5.

4.1 Definition of the Model

The proposed model is based on the following main concepts: (1) A discrete representation of time, (2) a distinction between payload and header droplets, (3) a discrete consideration of droplet behavior at bifurcations and sorters, and (4) constraints restricting the distance of droplets to avoid coalescences of droplets. These concepts are briefly discussed next before the resulting model is illustrated by means of an example.

Discrete Representation of Time: The model discretizes the continuous time during the droplet flow into atomic *time steps*. This allows to describe the duration a droplet requires to flow through a channel or to execute a module's operation in terms of a number of time steps.

Distinction between Payload/Header Droplets: Since the payload droplets and the header droplets are of different volumes, they cause different resistances in the channels and modules. Accordingly, the discrete model differentiates between these droplet types. More precisely, the number of time steps a payload droplet requires to flow through a channel $c_i \in C$ or to execute a module $m_i \in M$ is defined by the function $pSteps : C \cup M \rightarrow \mathbb{N}$. Accordingly, the function $hSteps : C \cup M \rightarrow \mathbb{N}$ defines the respective number of time steps for header droplets. Hence, depending on the type, a droplet takes a certain amount of time steps before it enters the succeeding channel or module in the architecture.



(a) Time step 13 (b) Time step 14 (c) Time step 15

(d) Time step 16 (e) Time step 17 (f) Time step 18

Figure 2: Bifurcation with time steps

Behavior at Bifurcations/Sorters: The physical behavior at bifurcations is abstracted by taking the number of time steps into account which are required for the respective droplet to flow through either of the successor channels. More precisely, a droplet flows into the successor channel requiring the *least* amount of time steps. If this channel already contains a droplet, it flows into the other successor channel. Sorters are handled as already described in Section 2.1, i.e. the payload droplet flows into the module, while header droplets are forwarded to the outlet of the module.

Constraints to Avoid Coalescences of Droplets: Unintended coalescences of droplets are avoided by restricting the distance of two droplets to a minimum value (in $[m]$). In the discrete model, this is abstracted to the corresponding number T_Δ of time steps a droplet would need to pass this distance. All droplets represented in the model must have a distance of at least T_Δ time steps.

Example 9. *Consider a visualization of the proposed discrete model for a bifurcation with its three channels c_{in}, c_1, and c_2 as shown in Figure 2a. The segments in the channels represent the number of time steps a payload droplet (white background) or a header droplet (gray background) needs to pass this channel, i.e. they respectively visualize the functions pSteps and hSteps defined as follows:*

	c_{in}	c_1	c_2
pSteps	8	8	11
hSteps	8	9	12

Using this model, e.g. the droplets' path and positions can be simulated in a much more intuitive and efficient fashion as shown in the Figures 2a-2f. Each of these figures represent the current state of this NLoC (including the positions of a payload droplet and a header droplet) at various time steps. By this, the "real world" behavior at the bifurcation in the architecture from Figure 1 (as already considered before in Section 3) is simulated. Note that all involved droplets satisfy the distance constraints, i.e. have a distance of at least $T_\Delta = 5$ time steps (applies for both, hSteps and pSteps). All other components of the architecture from Figure 1 can be represented and simulated in a similar fashion.

4.2 Determination of a Model Instance

The model proposed above allows to determine a discrete representation of arbitrary NLoC architectures. To this end, for a given NLoC architecture, a designer needs to (1) define the "real world" time of an atomic time step, (2) determine the functions *pSteps* and *hSteps* for each channel $c \in C$ and module $m \in M$, and (3) determine the minimal distance T_Δ of time steps between droplets. By this, a model instance can be derived which represents the given NLoC architecture.

The first step is the responsibility of the designer who, by choosing the "real world" time of an atomic time step T_a (in $[s]$) implicitly defines the resolution (and, hence, also the precision) of the model instance. With this information (together with the specs from the given NLoC architecture), the *pSteps*-function can be determined. This is done by performing the following steps for all channels $c \in C$ and all modules $m \in M$, i.e. for all entities $e \in C \cup M$:

- Place a payload droplet in the currently considered entity e.
- Determine the velocity u of the currently considered entity by solving the equation system defined by the Kirchhoff's laws (cf. Section 3).
- Use the resulting velocity together with the respective length of the entity to determine the duration d a payload droplet takes to flow through the channel or to execute the module, i.e. determine $d = \frac{l}{u}$.
- Abstract the resulting duration d to the corresponding discrete amount of time steps, i.e. set $pSteps(e) := \lceil \frac{d}{T_a} \rceil$.

The same is similarly conducted for a header droplet and its respective volume in order to determine the *hSteps*-function.

Example 10. *Consider again the bifurcation from the NLoC architecture shown in Figure 1 with the channel specification from Example 6. Following the steps from above (using the flow rate Q obtained by the NLoC spec as well as the respectively considered droplet sizes, cf. Section 3), yields the following velocities u and, hence, durations d for each payload/header droplet and channel:*

	c_{in}		c_1		c_2		*given in*
	p	h	p	h	p	h	
Q	10	10	5.29	4.84	4.63	4.23	$10^{-12}\ m^3/s$
u	4	4	2.12	1.93	1.85	1.69	$10^{-3}\ m/s$
d	75	75	82.68	90.45	107.98	118.13	$10^{-3}\ s$

By setting the "real world" time of an atomic time step to 10 ms, values for the functions pSteps and hSteps are defined as already used before in Example 9.

Finally, the model requires a minimum time difference T_Δ between droplets to prevent an unintended coalescence of droplets. This time difference is determined by dividing the required minimum distance *dist* (in $[m]$) by the minimum length a droplet flows in a single time step in any channel, i.e. $T_\Delta = \lceil \frac{dist}{MinLength} \rceil$.

Example 11. *To ensure a minimum distance of $dist = 80\mu m$, we first have to determine the minimum length a droplet flows in a single time step. Considering the values from the previous example, a droplet flows at least $\frac{l_2}{hSteps(c_2)} = \frac{200\mu m}{12} = 16.67\mu m$ in one time step. Hence, the minimum time difference between droplets is defined as $T_\Delta = \lceil \frac{80\mu m}{14.58} \rceil = 5$.*

5. PRECISION OF THE MODEL

The proposed model is an abstraction of the "real world" behavior, which allows designers to intuitively simulate the droplet flow and to efficiently conduct automated design tasks as e.g. determining a droplet sequence or verifying whether an architecture allows to execute a set of experiments. In this section, we evaluate the precision of the proposed discrete model in general as well as with respect to different resolutions. To this end, we implemented a *simulator* (in Java) which is capable of simulating the flow of droplets based on the discrete model proposed in Section 4. Afterwards, the results obtained by this simulator have been compared to the physical behavior as described in Section 3. To this end, the respective equation systems have been solved using the tool of [4] implemented in *Matlab*.

As use cases we considered various single building blocks such as *bifurcations*, *modules* [16], as well as cascades of them (in order to evaluate the precision of the model for these basic NLoC building blocks) and a *complete NLoC*

Table 1: Precision Evaluation

$T_a = 1ms$			$T_a = 5ms$			$T_a = 10ms$			$T_a = 15ms$		
Max	Bh.?	Prc.	Max	Bh.?	Prc.	Max	Bh.?	Prc.	Max	Bh.?	Prc.
Bifurcation:											
372	✓	99.7%	75	✓	98.9%	37	✓	96.3%	26	✓	95.1%
388	✓	99.9%	78	✓	99.4%	39	✓	99.4%	26	✓	99.4%
397	✓	99.9%	80	✓	96.3%	40	✓	96.3%	27	✓	95.1%
Cascade of three bifurcations:											
890	✓	99.8%	179	✓	99.2%	89	✓	99.8%	62	✓	95.5%
938	✓	100%	188	✓	99.8%	95	✓	98.7%	62	✓	99.2%
965	✓	96.4%	194	✓	95.9%	98	✓	94.9%	65	✓	95.4%
Module [16]:											
1713	✓	100%	342	✓	99.8%	171	✓	99.8%	114	✓	99.8%
2440	✓	100%	487	✓	99.8%	245	✓	99.6%	162	✓	99.6%
2047	✓	99.6%	408	✓	99.7%	206	✓	99.3%	136	✓	99.7%
2222	✓	99.8%	443	✓	99.7%	223	✓	99.2%	147	✓	99.3%
Cascade of three modules [16]:											
5826	✓	99.9%	1164	✓	100%	580	✓	99.6%	388	✓	100%
8007	✓	100%	1599	✓	99.9%	802	✓	99.8%	532	✓	99.7%
7820	✓	99.9%	1562	✓	99.9%	783	✓	99.8%	520	✓	99.8%
7164	✓	99.8%	1430	✓	99.9%	718	✓	99.7%	476	✓	99.7%
6414	✓	99.8%	1280	✓	99.9%	643	✓	99.7%	426	✓	99.7%
Complete NLoC Architecture:											
14267	✓	99.9%	2851	✓	99.9%	1429	✓	99.7%	948	✓	99.8%
12678	✓	99.9%	2533	✓	99.9%	1268	✓	99.8%	843	✓	99.6%
13240	✓	99.9%	2645	✓	99.9%	1324	✓	99.7%	880	✓	99.7%

Max: Total number of discrete time steps Bh.?: Does the discrete behavior match the "real world" (i.e. physical) behavior? Prc.: Precision of the discrete time steps

architecture generated from a benchmark of [22]. For each use case, a number of representative droplet sequences have been considered and simulated with respect to both, their actual physical behavior and their discrete behavior. The differences between both show the precision of the model. Since the designer defines the resolution (and, hence, the precision) of the model by choosing the "real world" time of an atomic time step, we additionally considered different atomic time step configurations, i.e. different values for T_a.

Table 1 summarizes the respectively obtained results. Each line provides the results obtained for a droplet sequence applied to the respective use case. For each considered resolution, the columns give the total number of time steps in the resulting model (*Max*), state whether the behavior of the discrete model matches with the actual physical behavior (*Bh.?*), and provide the timing precision the model accomplishes for the respectively considered droplet sequence (*Prc.*). For the latter, we compare the duration obtained by the physical simulation with those derived from the discrete model (a precision of 100% states that both values match exactly).

All our evaluations confirm that the discrete model indeed correctly abstracts from the physical behavior: All droplets always take the expected paths and are never involved in an unintended coalescence (indicated by "✓" in column *Bh.?*). Moreover, also the execution durations are abstracted in a precise fashion, i.e. in all cases a precision of more than 95% can be reported. This precision can additionally be refined by the designer by varying the time T_a of an atomic time step. This gives the NLoC designer the possibility to trade-off between the total amount of time steps to be considered in the model (and, hence, its complexity) and the obtained precision[4]. In our simulations, the computation time of all droplet sequences are only a few CPU-seconds. Overall, the results show that the proposed model is a suitable representation for the design of NLoCs.

6. CONCLUSION

In this work, we linked the physical world of *Networked Labs-on-Chip* (NLoC) to the domain of design and design automation. To this end, we proposed a discrete model which allows for the development of design tools for droplet sequence generation, simulation, and verification. A detailed consideration of the physical behavior of NLoCs as well as an intensive evaluation for representative droplet sequences confirmed the precision of the model. The resulting model allows for a deep consideration of design issues for this emerging LoC technology and, by this, provides the basis for several further works in this direction.

[4]Note that, in some cases, the precision increases by a lower resolution. This is the case when the imprecision caused by the discretization cancels out the discrete model simplifications.

7. REFERENCES

[1] D. J. Acheson. *Elementary fluid dynamics*. Oxford University Press, 1990.

[2] A. Ajdari. Steady flows in networks of microfluidic channels: building on the analogy with electrical circuits. *Journal on Comptes Rendus Physique*, 5(5):539–546, 2004.

[3] A. Biral and A. Zanella. Introducing purely hydrodynamic networking functionalities into microfluidic systems. *Journal of Nano Communication Networks*, 4(4):205–215, 2013.

[4] A. Biral, D. Zordan, and A. Zanella. Modeling, simulation and experimentation of droplet-based microfluidic networks. *IEEE Trans. on Molecular, Biological, and Multi-scale Communications*, 1(2):122–134, 2015.

[5] A. Carlson, M. Do-Quang, and G. Amberg. Droplet dynamics in a bifurcating channel. *Journal of Multiphase Flow*, 36(5):397–405, 2010.

[6] G. Cristobal, J.-P. Benoit, M. Joanicot, and A. Ajdari. Microfluidic bypass for efficient passive regulation of droplet traffic at a junction. *Applied Physics Letters*, 89(3):34104–34104, 2006.

[7] E. De Leo, L. Donvito, L. Galluccio, A. Lombardo, G. Morabito, and L. M. Zanoli. Communications and switching in microfluidic systems: Pure hydrodynamic control for networking Labs-on-a-Chip. *IEEE Trans. on Communications*, 61(11):4663–4677, 2013.

[8] E. De Leo, L. Galluccio, A. Lombardo, and G. Morabito. Networked labs-on-a-chip (NLoC): Introducing networking technologies in microfluidic systems. *Journal of Nano Communication Networks*, 3(4):217–228, 2012.

[9] A. J. Demello. Control and detection of chemical reactions in microfluidic systems. *Nature*, 442(7101):394–402, 2006.

[10] L. Donvito, L. Galluccio, A. Lombardo, and G. Morabito. On the assessment of microfluidic switching capabilities in NLoC networks. In *Int'l Conf. on Nanoscale Computing and Communication*, page 19, 2014.

[11] L. Donvito, L. Galluccio, A. Lombardo, and G. Morabito. μ-NET: a network for molecular biology applications in microfluidic chips. *IEEE/ACM Trans. on Networking*, 2015.

[12] M. J. Fuerstman, A. Lai, M. E. Thurlow, S. S. Shevkoplyas, H. A. Stone, and G. M. Whitesides. The pressure drop along rectangular microchannels containing bubbles. *Journal on Lab on a Chip*, 7(11):1479–1489, 2007.

[13] A. Grimmer, W. Haselmayr, A. Springer, and R. Wille. Verification of networked labs-on-chip architectures. In *Design, Automation and Test in Europe*, 2017.

[14] H. Gu, M. H. Duits, and F. Mugele. Droplets formation and merging in two-phase flow microfluidics. *Journal of Molecular Sciences*, 12(4):2572–2597, 2011.

[15] S. Haeberle and R. Zengerle. Microfluidic platforms for Lab-on-a-Chip applications. *Journal on Lab on a Chip*, 7:1094–1110, 2007.

[16] W. Haselmayr, A. Biral, A. Grimmer, A. Zanella, A. Springer, and R. Wille. Addressing multiple nodes in Networked Labs-on-Chips without payload re-injection. 2017.

[17] E. D. Leo, L. Donvito, L. Galluccio, A. Lombardo, and G. Morabito. Microfluidic networks: design and test of a pure hydrodynamic switching function. In *Int'l Conf. on Communications*, pages 787–791, 2013.

[18] A. Leshansky and L. Pismen. Breakup of drops in a microfluidic T junction. *Physics of Fluids*, 21(2):023303, 2009.

[19] D. Link, S. L. Anna, D. Weitz, and H. Stone. Geometrically mediated breakup of drops in microfluidic devices. *Physical Review Letters*, 92(5):054503, 2004.

[20] D. Mark, S. Haeberle, G. Roth, F. von Stetten, and R. Zengerle. Microfluidic Lab-on-a-Chip platforms: requirements, characteristics and applications. *Journal of Chemical Society Reviews*, 39(3):1153–1182, 2010.

[21] K. W. Oh, K. Lee, B. Ahn, and E. P. Furlani. Design of pressure-driven microfluidic networks using electric circuit analogy. *Journal on Lab on a Chip*, 12(3):515–545, 2012.

[22] M. F. Schmidt. Microfluidic flow-based biochips. https://sites.google.com/site/mlsibiochips/, 2012.

[23] Y.-C. Tan, Y. L. Ho, and A. Lee. Microfluidic sorting of droplets by size. *Journal of Microfluidics and Nanofluidics*, 4(4):343–348, 2008.

[24] S.-Y. Teh, R. Lin, L.-H. Hung, and A. P. Lee. Droplet microfluidics. *Journal on Lab on a Chip*, 8:198–220, 2008.

[25] G. M. Whitesides. The origins and the future of microfluidics. *Nature*, 442(7101):368–373, 2006.