UNIVERSITY OF CALIFORNIA
RIVERSIDE

Design Automation of Microfluidics

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Bioengineering

by

Junchao Wang

September 2017

Dissertation Committee:

Dr. William H. Grover, Chairperson
Dr. Philip Brisk
Dr. Victor G. J. Rodgers

The Dissertation of Junchao Wang is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

The writing of this dissertation has been an extraordinary journey that has led to my intellectual and personal growth and it would not have been possible without all the support from my mentors, my colleagues and friends, and my family.

First and foremost, this dissertation would not have been possible without the full support given to me by my advisor, Dr. William H. Grover. One day, I received an email from a Chinese student in CalTech who is interested in joining Grover Lab as a Ph.D. student and asked me if Dr. Grover is a harsh mentor who never care about students because usually that's how a talent and dedicated scientist would do. I replied, "If you joined Grover Lab, you will find Dr. Grover is the nicest professor you will ever meet". His kindness and teaching philosophy will be treasure for my rest of life.

I am very grateful to my committee members: Dr. Victor G. J. Rodgers and Dr. Philip Brisk for all their support, encouragement, and interest in my project.

I wanted to thank my fellow labmates: Dr. Nazila Norouzi, Dr. Shirin Mesbah Oskui, Brittney McKenzie, Astha Arora, Heran Bhakta, Tejas Patel, Joseph Bulone, and Douglas Hill. I learned a lot from every of you. Especially, Nazila taught me about how to fabricate a glass chip. Shirin and Heran gave me a lot of advises to my project.

This dissertation is dedicated to my parents who raised me with endless love and support. As the first people in my family to study abroad, nothing will be possible without spiritual support and financial support from my parents.

I am grateful to my decision to study in UC Riverside. It lead me to find my love, Naiyin Zhang.

Junchao Wang

July 2017

To the best parents in the world, Wang Zhigang and Guan Xueping,

for their endless love and support.

献给世界上最伟大的父母，王志刚和管雪萍，感谢他们对我从小到大的爱和支持！

ABSTRACT OF THE DISSERTATION

Design Automation of Microfluidics

by

Junchao Wang

Doctor of Philosophy, Graduate Program in Bioengineering
University of California, Riverside, September 2017
Dr. William H. Grover, Chairperson

What can microfluidics do? For the past decades, researchers from physics, chemistry, biotechnology, and other fields have used microfluidics to design numerous chips for different applications. As new microfluidic chips keep emerging, we ask ourselves, "what else? Could we develop any microfluidic chips that human beings cannot design or imagine? Can we design better microfluidic chips with the help of computer algorithms?"

This thesis presents several projects that demonstrate the potential that computer algorithms, simulations, and conventional microfluidics together will help researchers find better microfluidic chips, better design methods, and help us explore new phenomena. In Chapter 1, we start with a small review of how microfluidic chips are designed in the past decades and discuss the advantages and disadvantages of the conventional design method. In Chapter 2, we present a "random design" method to design a function microfluidic solute

generator, which is able to generate three arbitrary concentrations at the same time. In Chapter 3, we present a microfluidics-optimized particle simulation algorithm (MOPSA) that simulates the trajectories of cells, droplets, and other particles in microfluidic chips with more lifelike results than particle tracers in existing commercial software. In Chapter 4, we present a microfluidic simulation method that can simulate the behavior of fluids and particles in typical microfluidic chips instantaneously (in around one second), which is able to accelerate the simulation of microfluidic chips. In Chapter 5, we present a microfluidic particle sorter which is designed by the three algorithms in Chapter 2 - 4. In Chapter 6, we automated designed and optimized a microfluidic mixer using Non-dominated Sorting Genetic Algorithm II (NSGA-II). In Chapter 7, we discuss the potential impacts of the projects presented in this thesis and the future directions of computer algorithms that can help microfluidics evolve into the next generation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background: Current microfluidics design method

Since the appearance of microfluidics devices in late 1970s [1], microfluidics have found a wide variety of applications, including cell analysis [2], PCR amplification [3], cell separation [4] and so on. [5]. However, while new applications of microfluidic devices keep emerging, the techniques used for designing and optimizing these devices have remained relatively unchanged since the 1970s. Virtually all researchers design microfluidic chips using computer assisted design (CAD) software, and the design represents the best guess of the researcher to reach their desired functionality. The chip is then fabricated and tested. If the performance of chip is not good enough, the design process will start over again (see Figure 1.1). Thus, this method is mainly depended on the experience of the designer, which makes it labor-intensive and time consuming. At the same time, this process only explores a small fraction of possible designs for functional microfluidic devices. Therefore, a better method is needed to help design different microfluidic chips for the purpose of various

applications in a faster and more accurate way. Figure 1.2 shows how researchers translate a single transistor to a modern CPU. In a modern CPU, there are more than 700 million transistors on it, and obviously human beings cannot imagine how we should organize so many transistors. Thus, software becomes an essential part to help researchers to design a CPU.

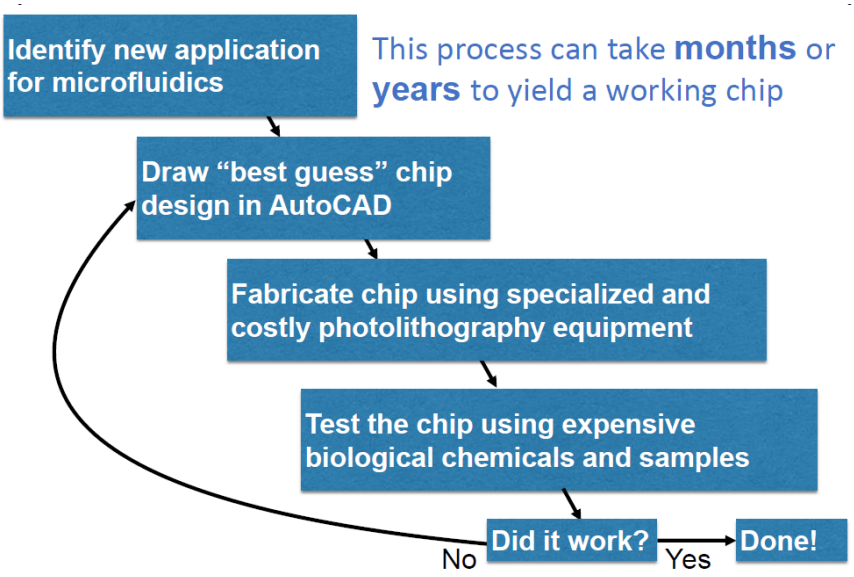

Figure 1.1: How a microfluidic chip is designed?

In contrast, to overcome the obstacle and help microfluidics become a useful tool in different fields, we need design automation tools to simplify the designing process, improve the performance of microfluidic chips, and reduce the time and cost of the designing process.

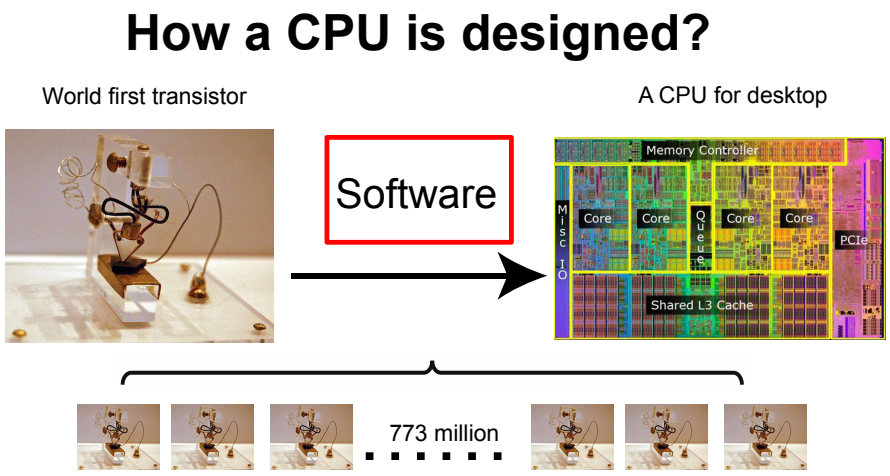

Figure 1.2: How a modern CPU is designed? Software!

1.2 What will the software look like?

In this thesis, we tried to answer this question in several aspects. In Chapter 2, we will discuss a "random design" method. We accomplished this by first generating a library of thousands of different random microfluidic chip designs, then simulating the behavior of each design on a computer using automated finite element analysis. The simulation results were then saved to a database which a user can query via a public website (http://random.groverlab.org) to find chip designs suitable for a specific task. To demonstrate this functionality, we used our library to select chip designs that generate any three desired concentrations of a solute. We also fabricated and tested 16 chips from the library, confirmed that they function as predicted, and used these chips to perform a cell growth rate assay. This is one of many different applications for randomly-designed microfluidics; in principle, *any* microfluidic chip that can be simulated could be designed automatically

using our method. Using this approach, individuals with no training in microfluidics can obtain custom chip designs for their own unique needs in just a few seconds.

Computer simulation plays a growing role in the design of microfluidic chips. However, the particle tracers in some existing commercial computational fluid dynamics software are not well suited for accurately simulating the trajectories of particles such as cells, microbeads, and droplets in microfluidic systems. To address this issue, in Chapter 3, we present a microfluidics-optimized particle simulation algorithm (MOPSA) that simulates the trajectories of cells, droplets, and other particles in microfluidic chips with more lifelike results than particle tracers in existing commercial software. When calculating the velocity of a particle, MOPSA treats the particle as a two-dimensional rigid circular object instead of a single point. MOPSA also checks for unrealistic interactions between particles and channel walls and applies an empirical correcting function to eliminate these errors. To validate the performance of MOPSA, we used it to simulate a variety of important features of microfluidic devices like channel intersections and deterministic lateral displacement (DLD) particle sorter chips. MOPSA successfully predicted that different particle sizes will have different trajectories in six published DLD experiments from three research groups; these DLD chips were used to sort a variety of different cells, particles, and droplets. While some of these particles are not actually rigid or spherical, MOPSA's approximation of these particles as rigid spheres nonetheless resulted in lifelike simulations of the behaviors of these particles (at least for the particle sizes and types shown here). In contrast, existing commercial software failed to replicate these experiments. Finally, to demonstrate that MOPSA

can be extended to simulate other properties of particles, we added support for simulating particle density to MOPSA and then used MOPSA to simulate the operation of a microfluidic chip capable of sorting cells by their density. By enabling researchers to accurately simulate the behavior of some types of particles in microfluidic chips before fabricating the chips, MOPSA should accelerate the development of new microfluidic devices for important applications.

Microfluidics researchers are increasingly using computer simulation in many different aspects of their research. However, these simulations are often computationally intensive: simulating the behavior of a simple microfluidic chip can take hours to complete on typical computing hardware, and even powerful workstations can lack the computational capabilities needed to simulate more complex chips. This slows the development of new microfluidic chips for new applications. In Chapter 4, we present a microfluidic simulation method that can simulate the behavior of fluids and particles in typical microfluidic chips instantaneously (in around one second). Our method decomposes the chip into its primary components: channels and intersections. The behavior of fluid in each channel is determined by leveraging analogies with electronic circuits, and the behavior of fluid and particles in each intersection is determined by querying a database containing nearly 100,000 pre-simulated channel intersections. While constructing this database takes a non-trivial amount of computation time, once built, this database can be queried to determine the behavior of fluids and particles in a given intersection in a fraction of a second. Using this approach, the behavior of a typical microfluidic chip can be simulated in just one second

on a standard laptop computer, without any noticeable degradation in the accuracy of the simulation. As a proof of concept, we show that our simulation method can instantaneously simulate the paths followed by particles in both simple and complex microfluidic chips, with results that are essentially indistinguishable from simulations that took hours or even days to complete using conventional approaches.

Microfluidic chips that can sort mixtures of cells and other particles have important applications in research and healthcare, but designing a sorter chip for a given application is a slow and difficult process. In Chapter 5, we created microfluidic sorter chips without actually designing them. We accomplished this by simulating the paths followed by particles through 10,513 different random microfluidic chip designs. We then defined an application—sorting 1 μm and 10 μm particles—and mined the database of simulations to find 1,061 designs that could perform this application. Finally, we fabricated and tested four of these designs and found that one of the designs can successfully sort the specified particles. Our work shows that by searching libraries of pre-simulated chip designs, researchers with no experience in microfluidics can find chip designs for particle sorters quickly and easily. Additionally, since these sorters were designed at random, we do not know exactly *how* they function. This suggests that entirely new and useful microfluidic phenomena could be discovered in random chip designs using automated techniques like ours, phenomena that might have never been discovered otherwise.

The ability to thoroughly mix two fluids is a fundamental need in microfluidics. While a variety of different microfluidic mixers have been designed by researchers, it remains

unknown which (if any) of these mixers are optimal (that is, which designs provide the most thorough mixing with the smallest possible fluidic resistance across the mixer). In Chapter 6, we automatically designed and rationally optimized a microfluidic mixer. We accomplished this by first generating a library of thousands of different randomly designed mixers, then using the Non-dominated Sorting Genetic Algorithm II (NSGA-II) to optimize the random chips. After 200 generations of evolution, the Pareto-optimal front was found. We examined designs at Pareto-optimal front and found several design criteria that enhance the mixing performance of a mixer while minimizing its fluidic resistance; these observations provide new guidance on how to design microfluidic mixers. We also compared the designs from NSGA-II with some popular microfluidic mixer designs from the literature and found that designs from NSGA-II have a lower fluidic resistance with similar mixing performance.

Chapter 2

Random design of microfluidics

Reprinted with permission from "Random design of microfluidics." by Junchao Wang, Philip Brisk, and William H. Grover. Lab on a Chip 16.21 (2016): 4212-4219.

2.1 Introduction

Since the emergence of the first lab-on-a-chip devices in the late 1970s, [1] microfluidic chips have found applications in a variety of fields. But while the range of possible applications for microfluidics has blossomed, the process of designing microfluidic chips has remained relatively unchanged since the 1970s. Researchers still design new microfluidic chips by hand, drawing on a computer a design that represents a "best guess" of the desired functionality, then fabricating and testing the chip. If the chip does not perform as intended, the researcher alters the chip design and fabricates and tests the new chip. This iterative design process can take months or even years to yield a functional microfluidic chip. The inefficiency of this process slows the development of new microfluidic chips for important applications in research and healthcare. It also creates a significant barrier to

entry for researchers who may wish to create custom microfluidic chips but are not microfluidics experts. Finally, the current design process only explores a tiny fraction of the many possible designs for microfluidic chips. It is reasonable to assume that there are many microfluidic chip designs that are better than our current designs, but these better designs will never be discovered simply because our design process is too slow and inefficient.

Computers can help with the process of microfluidic chip design, but they have not yet completely automated the design process. For example, finite element analysis (FEA) software is sometimes used to simulate the behavior of a microfluidic chip before fabricating it. However, to use FEA software, a researcher still needs to create a chip design first; the software does not design the chip for them. Additionally, the cost of FEA software ($7995 for a single-user license to the popular simulation tool COMSOL Multiphysics) is a practical barrier to widespread use of this software in microfluidics. Recently, software and algorithms from computer science and electrical engineering have been applied to microfluidic chip design with encouraging results. For example, the principles of semiconductor electronic design automation (EDA) can be leveraged to automatically design microfluidic chips for some tasks. [6, 7] However, some physical phenomena in microfluidics do not have clear analogies in electrical circuits. Laminar flow, solute diffusion, sound, [8] light, [9] magnetism, [10] and gravity [11] all affect microfluidic chips in ways that are difficult to model using EDA techniques.

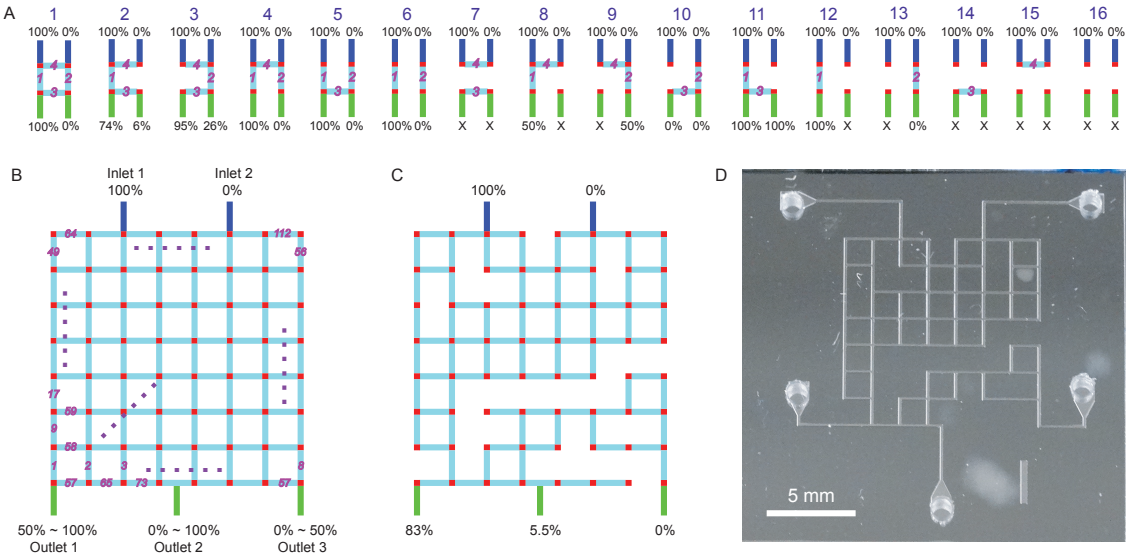

Figure 2.1: Surveying the library of "all possible microfluidic chip designs." (A) For a microfluidic chip with two inlets (dark blue), two outlets (green), four channel intersections arranged in a square (red), and four possible channels connecting those intersections (light blue), there are $2^4 = 16$ different chip designs. If solutions of concentrations 100% and 0% flow into the inlets at a constant volumetric flow rate, these 16 chip designs generate solutions with seven different specific concentrations at the outlets (0%, 6%, 26%, 50%, 74%, 95%, and 100%), and some designs generate no fluid (marked with "X"). (B) An 8×8 grid supports $2^{112} = 5 \times 10^{33}$ different chip designs that can generate essentially any three desired concentrations at the three outlets. One of these chip designs is shown in (C) and fabricated and photographed in (D); this specific design generates concentrations of 83%, 5.5%, and 0% at its outlets.

In this work we were motivated by the question, *is it possible to create functional microfluidic devices without actually designing them?* More precisely, what if we had a library of *all possible designs for microfluidic chips*, and creating a chip for a new application was as simple as searching through the library for the appropriate design? In this thought experiment, every possible application for microfluidics—from diagnosing diseases [12] to searching for life on Mars [13]—would have a suitable design in this library. If the behavior of each chip design was stored with the design in the library, then a researcher could merely search the library for chips with the desired behavior and fabricate and use these chips immediately, without actually designing the chips.

Obviously this hypothetical library of "all possible microfluidic chip designs" would be astronomically large. But it also raises some interesting possibilities. This library would undoubtedly contain chip designs that are far better than the designs created by humans for the same application. Microfluidics researchers have explored such a small fraction of this library of all possible designs—could there be entirely new and useful microfluidic phenomena waiting to be found in this hypothetical library?

While we cannot yet build a full library of "all possible chip designs," we can still explore parts of this library. Specifically, by imposing constraints that limit the number of possible chip designs, we can in some cases actually test all possible chips, or at least explore enough random designs to catch a glimpse of the library of all possible chips. In this work we constrained our microfluidic chips to a rectilinear grid of channels shown in Figure 5.2. We simulated the performance of over ten thousand different random chip designs based

on this grid using FEA software, then created a database of the simulation results. We then queried this database to find chip designs suitable for given tasks. As a demonstration task, we instructed our software to select chip designs that take two fluids as inputs (one fluid, a solution with a known concentration of a solute; and the other fluid, water) and generate three output fluids with user-specified concentrations of the solute (like 92%, 66%, and 23%). We then fabricated and tested several of the chips selected by our software and used them to perform a cell growth rate assay. A graphical overview of our random design process is shown in Figure 2.2. Despite not having been "designed" for any specific purpose, the selected chips successfully performed the desired tasks. This proof-of-concept is just one of many different applications for randomly-designed microfluidics; in principle, *any* microfluidic chip that can be simulated could be designed automatically using our method.

2.2 Results

2.2.1 Grid design

In this work we constrained our microfluidic chip designs to the rectilinear grid patterns shown in Figure 5.2. An $n \times n$ grid has n^2 possible channel intersections and $2n^2 - 2n$ possible channels connecting those intersections. If each of these connecting channels can be either present or absent in a given design, then the total number of different possible chip designs is 2^{2n^2-2n}. If the size of the grid is small, it is feasible to generate all microfluidic chip designs that are possible within that grid. For example, the simplest grid we considered, a 2×2 grid, has only $2^{2\times 2^2 - 2\times 2} = 16$ different designs (Figure 5.2A).

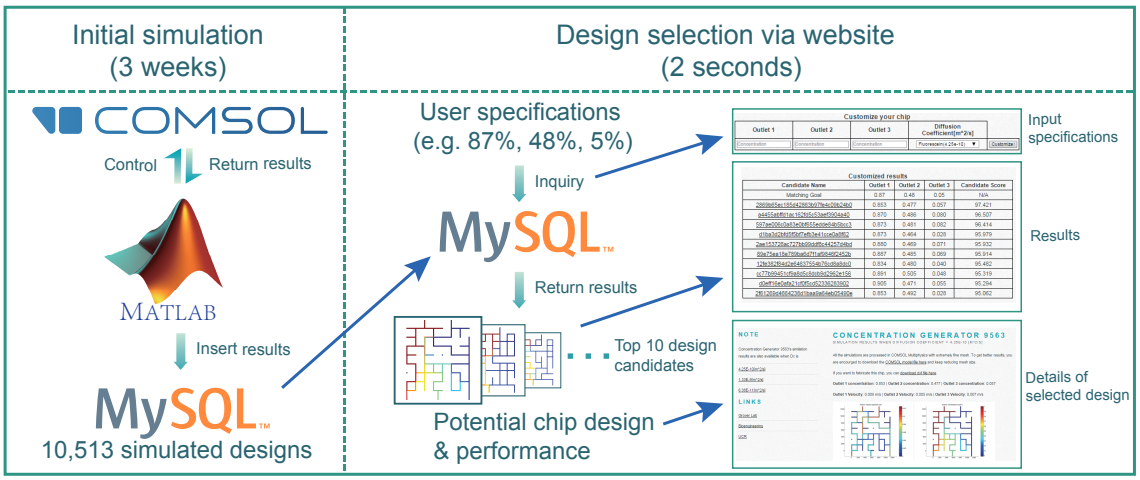

Figure 2.2: Overview of the random microfluidics design process. During the three-week initial simulation phase, a MATLAB program generated 10,513 random chip designs from the grid shown in Figure 5.2B, simulated the performance of each design using the finite element analysis software COMSOL Multiphysics, and saved the simulation results to a MySQL database. This database of random chip designs was then transferred to a web server (http://random.groverlab.org) where a user can specify three desired solution concentrations. Within seconds, the website then returns the top ten chip designs that generate those concentrations, including simulation results and downloadable computer-assisted design (CAD) files for use in fabricating the chips.

As a demonstration of randomly-designed microfluidics, we set out to find chips that are capable of generating any three desired concentrations of a solute. Each chip design contains two inlet channels, one containing a 100% solution of a solute and one containing water (0% solute), flowing into the chip at a constant volumetric flow rate. As these fluid streams split and merge inside the chip, different mixtures of the two fluids are created. We then used finite element analysis (described below) to predict the concentrations of solute in each of the three outlet channels for each chip design.

In the 16 possible designs for the 2 × 2 grid in Figure 5.2A, four designs (7, 14, 15, and 16) have no connections between the inlets and outlets and thus output no fluid; four designs (1, 4, 5, and 6) output solute concentrations that are unchanged from the input

concentrations (still 100% and 0%); two designs (10 and 13) output only 0%; two designs (11 and 12) output only 100%, two designs (8 and 9) output only 50%, one design (2) outputs solute concentrations of 74% and 6%; and one design (3) outputs solute concentrations of 95% and 26%. Thus, seven different specific solute concentrations (0%, 6%, 26%, 50%, 74%, 95%, and 100%) can be generated using chips from the 2×2 library. If these seven concentrations are adequate for a given microfluidic application, then a user may select chip designs from this library and use them. However, if a user requires concentrations that are not generated by chips in the 2×2 library, then a larger and more complex grid is necessary.

As the size of the grid grows larger, there are more opportunities for channel splits and merges that create different mixtures. We hypothesized that an 8×8 grid would support a large enough variety of chip designs to ensure that *any* three desired mixtures will be generated by at least one design in the library. The 8×8 grid in Figure 5.2B supports $2^{2\times 8^2 - 2\times 8} = 5,192,296,858,534,827,628,530,496,329,220,096$ different chip designs, so clearly we will not be able to study every possible design. However, by randomly selecting thousands of designs from the 8×8 grid and simulating their behavior, we can explore the variety of designs supported by this grid. And by fabricating and testing several of these designs, we can confirm that our technique of randomly-designed microfluidics can be used to select chip designs with desired behaviors.

2.2.2 Generating random microfluidic chip designs

A custom MATLAB program was written that generated 21,564 different random chip designs within the constraints of the grid design shown in Figure 5.2B. Each of the 112 variable channels (light blue in Figure 5.2B) has a 90% chance of being present in any given design. This value was chosen as a balance between two practical limits. If the probability of a channel being present is too low, many chip designs many not have a continuous path for fluid to flow between the inlets and outlets, rendering them nonfunctional. If the channel probability is too high, most chips will have nearly all variable channels present, leading to low diversity in the library of random designs. A 90% probability that each variable channel will be present seems to provide a good balance between encouraging functional devices and exploring a large fraction of the chip design space. Of the 21,564 different random chip designs, 10,513 designs have continuous paths for fluid between both inputs and all three outputs; these designs were retained for use in this study. 11,058 designs have paths for fluid between both inputs and two of the three outputs and were not used (although they could be used for applications requiring only two different concentrations of fluids).

2.2.3 Building the library of chip simulations

The behavior of each of the 10,513 random chip designs was simulated using the finite element analysis software COMSOL Multiphysics (COMSOL Inc., Burlington, MA). We used the software's MATLAB API [14] to automate this process and performed all

simulations necessary to construct the library without human assistance. The results of each simulation included plots of fluid velocity, fluid pressure, and solute concentration at each point in the chip, as well as a COMSOL model file containing the simulation results. The two-dimensional simulations used 200 μm channel widths, 1.5 mm channel lengths between vertices, 1×10^{-6} relative repair tolerance, and triangular meshes containing from 5×10^4 to 5×10^5 elements. In the *Laminar Flow* physics module in COMSOL Multiphysics, each inlet was assigned an inlet boundary condition of 10 mm/s normal inflow velocity, and each outlet was assigned an outlet boundary condition of 0 Pa pressure. The remaining boundaries were walls (no-slip boundary condition), and the material filling the channels was water under incompressible flow. In the *Transport of Dilute Species* physics module, Inlet 1 is assigned an inflow of 1 mol/m^3 and Inlet 2 is assigned an inflow of 0 mol/m^3. The three outlets were assigned as outflows. Each chip design was simulated using the diffusion coefficients for sodium ions, fluorescein, and bovine serum albumin, as described below. Two stationary solvers were used in COMSOL Multiphysics: the first solved for laminar flow, and the second solved for transport of diluted species. After each simulation, the linear flow rates and solute concentrations at each of the three outlets were saved to a MySQL database. Additionally, the COMSOL model file, velocity profile, concentration profile, pressure profile, and a computer-assisted design (CAD) file containing the design of the chip in the standard DXF format [15] were saved to local storage. Users can query this database and download chip designs for specific applications at `http://random.groverlab.org`.

2.2.4 Role of diffusion in randomly designed chips

Solutes with different diffusion coefficients may result in different output concentrations in the same chip. To assess the role of diffusion and enable our library to support a wider variety of solutes, each chip design was simulated three times, once for each of three model solutes: sodium ions (diffusion coefficient $D_c = 1.33\times10^{-9}$ m^2/s), fluorescein ($D_c = 4.25\times10^{-10}$ m^2/s), and bovine serum albumin (BSA; $D_c = 6.38\times10^{-11}$ m^2/s). These solutes were chosen to be representative of ions, small molecules, and proteins, respectively. The complete library of solute-specific simulations contains 31,515 simulation results and took three weeks to complete on a desktop computer.

2.2.5 Analyzing the random chip library

Before using our library of simulation results to generate chip designs for specific applications, we first analyzed the entire library to ascertain the range of microfluidic functions it supports. Figure 3 shows the solute concentrations and fluid velocities at each of the three outlets for all 10,513 random chip designs (using the diffusion constant of fluorescein). The distribution of solute concentrations in Figure 3 confirms that the library contains chip designs suitable for generating essentially any desired solute concentration.

Outlet 2 supports the widest variety of solute concentrations; designs yielding concentrations from 0% to 100% at Outlet 2 are present in the library. Concentrations around 50% are most common at Outlet 2, and 80% of the designs generate concentrations between 28% and 72% at Outlet 2. These results make intuitive sense: as the middle outlet

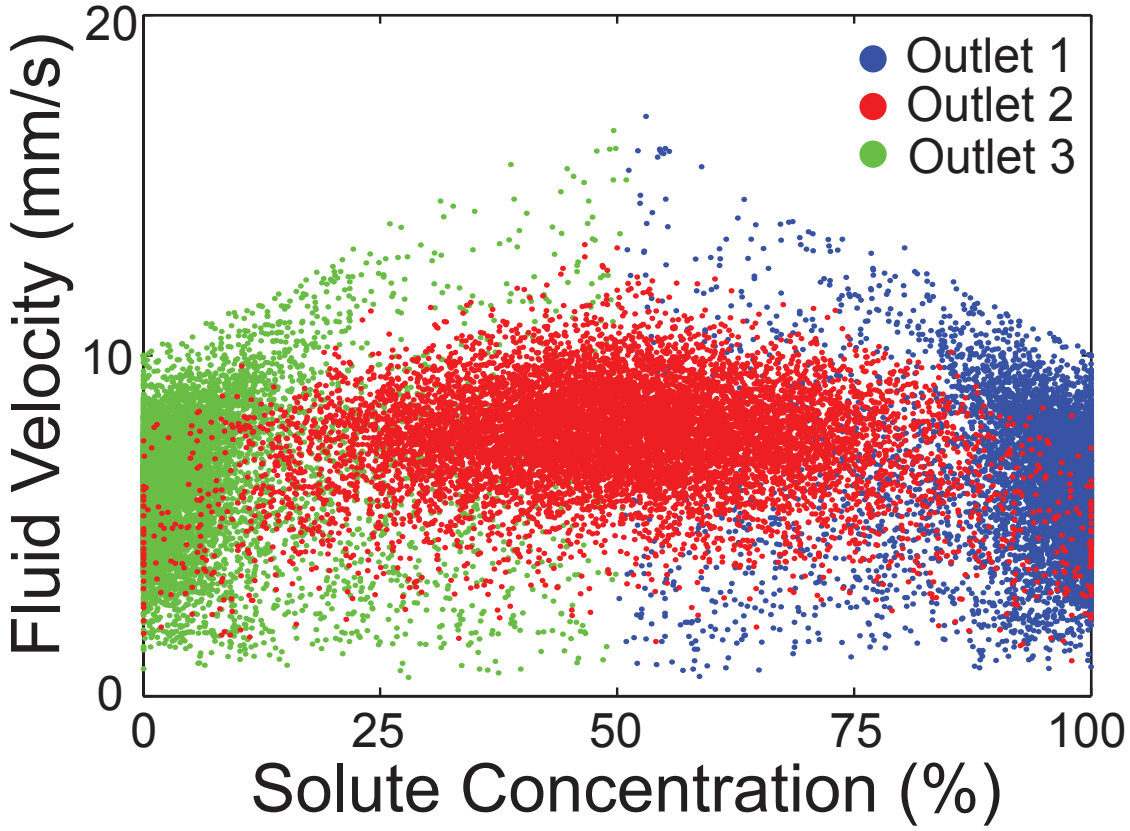

Figure 2.3: Fluid velocity vs. concentration for each outlet in each of the 10,513 chip designs in our simulation library. While this data was obtained using the diffusion coefficient of fluorescein (4.25×10^{-10} m^2/s), the results are essentially identical for solutes with larger or smaller diffusion coefficients (see online *Supplementary Information*). The wide distributions of concentrations and velocities confirm that our proof-of-concept library can provide chip designs that generate any desired fluid concentrations at any desired flow rates up to approximately 10 mm/s.

located halfway between Inlet 1 (100%) and Inlet 2 (0%), Outlet 2 is well suited to yield a range of mixtures centered at 50%.

Outlet 1 yields a narrower range of solute concentrations (from 50% to 100%). The average concentration at Outlet 1 is 95%, the median concentration is 98%, and 90% of the designs yield concentrations between 86% and 100%. These higher concentrations can be explained by the close proximity of Outlet 1 to Inlet 1 (which contains 100% solute); this close proximity favors designs in which most of the fluid from Inlet 1 to flows to Outlet 1 (and therefore increases the solute concentration at Outlet 1).

Outlet 3 also yields a narrower range of solute concentrations (from 0% to 50%), with an average concentration of 5%, a median concentration of 2%, and 90% of the designs yielding concentrations between 0% and 15%. Again, the close proximity of Outlet 3 to Inlet 2 (which contains 0% solute) explains the preference for lower concentrations at Outlet 1.

There is no obvious mathematical relationship between the fluid velocity and solute concentration at each of the three outlets in Figure 3. Consequently, velocity and concentration can be considered as independent variables when selecting designs from the library: a user could specify any desired concentration *and* any desired velocity and a suitable design is likely to exist within the library (at least over the ranges shown in Figure 3).

We then identified which factors are most significant in determining the solute concentrations at the outlets. Specifically, fluids could be mixed by any of three different processes on-chip: the chip design (the various splits and merges fluids undergo in the chip),

diffusion (mixing between adjacent streams of fluid in the chip), and turbulence (chaotic processes that also might mix fluids in the chip). At low flow rates, diffusional mixing would dominate, resulting in identical 50% concentrations at each outlet. At moderate flow rates, the effect of diffusion is reduced, and the mixing ratios would be determined by the channel network and the fluidic resistance of each channel segment. Finally, at high flow rates, the breakdown of laminar flow and emergence of turbulence could affect the mixing ratios in unpredictable ways. The dimensionless Peclet (Pe) and Reynolds (Re) numbers are used to determine in which of these regimes a microfluidic chip is operating:

$$\mathrm{Pe} = \frac{Lu}{D} \tag{2.1}$$

$$\mathrm{Re} = \frac{\rho L u}{\mu} \tag{2.2}$$

where L is a characteristic dimension like channel width, u is the linear fluid flow rate, D is the solute diffusion constant, ρ is the density of the fluid, and μ is the viscosity of the fluid. If $\mathrm{Pe} < 1$, diffusion may have an effect on the mixing behavior of a chip, and if $\mathrm{Re} > 300$, turbulence may have an effect on the mixing behavior. We calculated Re and Pe for each of the 10,513 random chip designs in our library. Values for Re at each location in each chip ranged from 0 to 6 and were all much lower than 300, indicating that no turbulent mixing is occurring in the chip designs. Typical values of Pe were 4500 for channels containing solutions of Na^+, 14,000 for fluorescein, and 94,000 for BSA; these are all much greater than 1 and indicate that diffusional mixing has a negligible effect

on the outlet solute concentrations. These calculations suggest that chip design (and not turbulence or diffusion) is the primary determinant of the output solute concentrations over the range of flow rates we considered.

Finally, to determine if our library of chip designs can be used with a wide variety of solutes (not just the three whose behavior we simulated), we compared simulation results from the two solutes with the greatest difference in diffusion constants: Na^+ and BSA. Even through the diffusion coefficient of Na^+ is about 20 times larger that that of BSA, the average difference in outlet concentrations between Na^+ and BSA for the 10,513 different designs in our library was only 1.02 percentage points, and the largest single difference in outlet concentrations was only 2.76 percentage points. Additionally, versions of Figure 2.6 containing the simulation results for all three solutes (Na^+, fluorescein, and BSA) are provided in online *Supplementary Information*; they are essentially indistinguishable from each other. This further supports our claim that chip design (not diffusion) dominates the mixing behavior of the designs in our library over the range of flow rates we studied (from 0 to 10 mm/s at the outlets), and our library contains designs that are suitable for generating solutions of any desired concentration using a wide variety of different solutes.

2.2.6 Selecting randomly-designed chips for specific tasks

We then transferred our database of 10,513 different random chip designs to a web server and created a website that allows users to search this database to find chips with desired behaviors. The website is available for public use at `http://random.groverlab.org`.

After the user specifies the three desired solution concentrations, the website returns the design and simulation results for each of the top ten chip designs that will generate these concentrations. Note that this server does not run COMSOL or perform any simulations; it simply queries a database populated with simulation results.

To confirm that the chip designs selected by our website actually function as predicted, we chose 16 chip designs at random and fabricated glass microfluidic chips based on those designs. The only modification we made to the computer-generated designs was the deletion of all dead-end channels, which might trap bubbles during use and will have no effect on the mixing behavior of the chips. Conventional photolithography and wet etching were used to etch the randomly-designed chip designs into glass wafers to a depth of 50 μm, inlet and outlet holes were drilled using diamond-tipped drill bits, and glass-glass thermal fusion bonding (668°C for 6 hours) or anodic bonding to silicon (350°C and 400 V for approximately 30 minutes) were used to create finished microfluidic devices.

The simulation results for these 16 designs (labeled A–P) are shown in Figure 2.4, and a photograph of a chip fabricated using Design L is shown in Figure 5.2D. After fabricating chips for each of the 16 random designs in Figure 2.4, we tested the performance of each chip by flowing a solution of 1.0×10^{-5} M fluorescein in 0.1 M Tris buffer (pH = 8.0) in Inlet 1 and an identical solution without fluorescein in Inlet 2. A two-channel syringe pump was used to provide a constant flow rate of 6 μL/min at each inlet (this corresponds to a linear flow rate of 10 mm/s). After 10 minutes, fluid from each of the three outlets was collected for analysis. The fluorescein concentration of fluid from each

outlet was measured using a FlexStation microplate reader (Molecular Devices, Sunnyvale, CA) using an excitation wavelength of 490 nm and an emission wavelength of 514 nm.

Figure 2.5A-D compares our experimental results with the values predicted by our simulation library. The average percent difference between the predicted and experimental values of concentration at each outlet was 2% for Outlet 1, 4% for Outlet 2, and 0.3% for Outlet 3. The greatest single difference between predicted and experimental values was at Outlet 2 on Chip P (9% difference). These differences between predicted and experimental results compare favorably with other microfluidic mixers [16] and are low enough for many microfluidic applications.

Finally, we also used our randomly-designed microfluidic chips to automatically generate five different concentrations of cell media for use in a cell growth assay. We selected three chips (A, G, and H) that generated 5%, 10%, 57%, 83% and 99% concentrations of yeast growth media. On each chip, Inlet 1 received 100% yeast extract peptone dextrose media (YPD; Thermo Fisher Scientific, Waltham, MA USA) and Inlet 2 received water. 10 mL of media from each outlet was collected into test tubes that were then inoculated with identical amounts of *Saccharomyces cerevisiae* yeast and cultured at 25°C for 24 hours. Growth curves for each culture were obtained by periodically measuring the optical absorbance at 600 nm using an UV-Vis-NIR spectrophotometer (V-670, Jasco, Easton, MD).

Yeast growth curves (Figure 2.5E) show that while the initial growth rates were fairly comparable for all five media concentrations generated by our randomly-designed

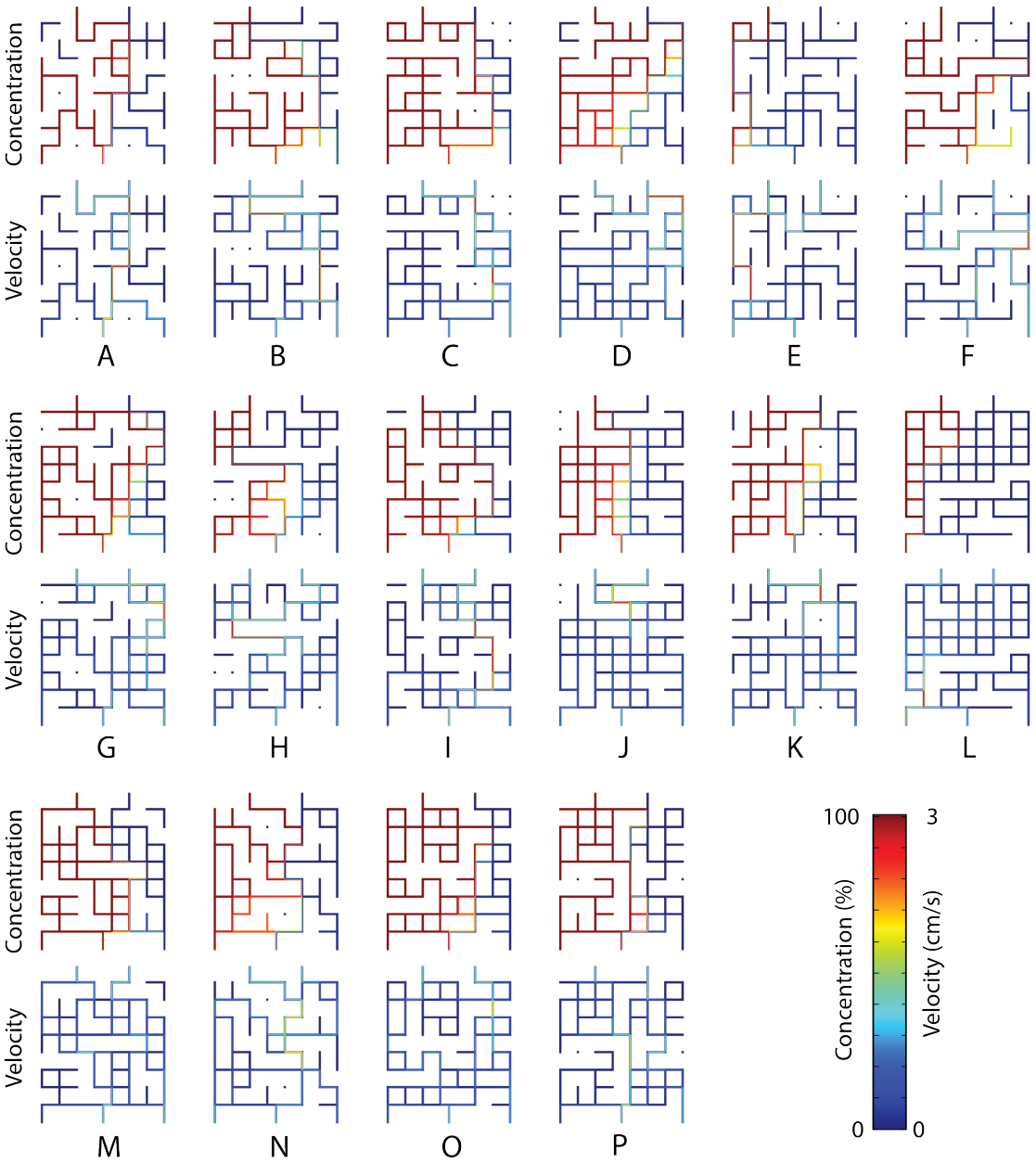

Figure 2.4: Simulation results for 16 chip designs (A–P) selected at random from our library of 10,513 random chip designs, using the diffusion coefficient of fluorescein (4.25×10^{-10} m^2/s). As the channels split and merge in the random designs, the constant solute concentrations (100% and 0%) and constant fluid flow rates (6 μL/min) at the inlets translate into a variety of different concentrations and flow rates at the outlets.

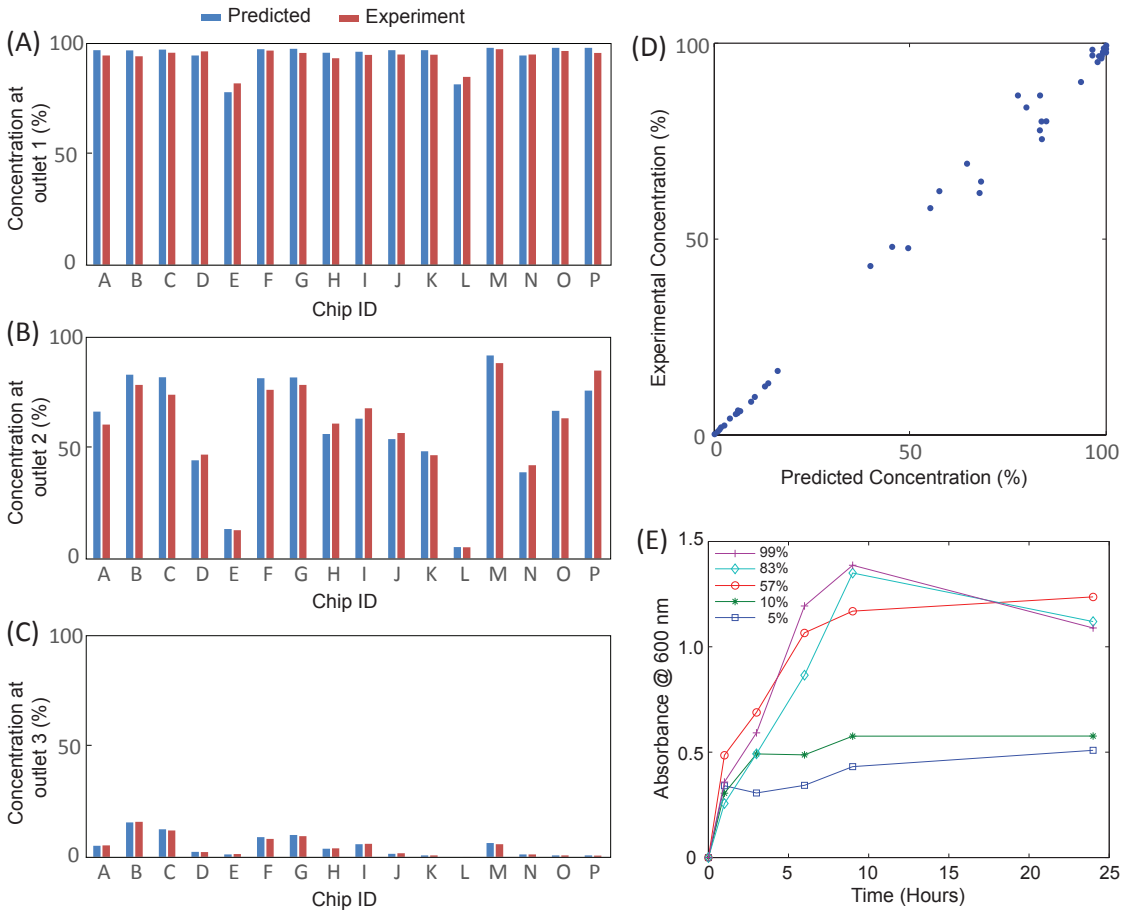

Figure 2.5: (A–D) Comparisons between the predicted and experimentally-determined performance of each of the 16 randomly-designed test chips shown in Figure 2.4. The solute concentrations predicted by our library (blue) agree well with the measured concentrations (red) for each chip at Outlet 1 (A), Outlet 2 (B), and Outlet 3 (C). When combined (D), all 48 solute concentrations generated by these 16 chips are consistent with the library predictions over the full range of concentrations from 0% to 100%. (E) To demonstrate that randomly-generated microfluidic chips can perform real-world biological applications, three chips were used to automatically generate five different concentrations of cell culture media (5%, 10%, 57%, 83% and 99% of yeast extract peptone dextrose). The optical absorbance at 600 nm was measured to obtain growth curves of *Saccharomyces cerevisiae* yeast in each media concentration generated by the randomly-designed chips. As expected, yeast cultures with lower media concentrations reached steady state earlier and with a smaller number of cells than cultures with higher media concentrations.

chips, yeast in the lowest-concentration media (5% and 10%) exhausted their nutrients and entered stationary phase earlier and with fewer cells than the yeast in the higher-concentration media. Growth curves like these play an important role in studies of human conditions like aging and cancer, [17] and our randomly-designed chips could replace manual labor or expensive computer-controlled valves and pumps in an instrument for automated measurement of growth curves. These results suggest that randomly-generated microfluidic chips *can* support real-world research applications.

2.3 Discussion

We demonstrated how to create functional microfluidic chips for specific applications without actually designing the chips. We accomplished this by generating a large library of random chip designs, simulating their behavior using finite element analysis, and saving the results in a database that can be queried by users via a website. Using this website, researchers with no experience in designing microfluidics can easily find chip designs that satisfy their own unique needs.

As a proof-of-concept, we created a library of 10,513 random chip designs that can generate three solutions of any desired concentrations. These random chips have several unique properties compared to existing chips for generating solutions with different concentrations. First, while most existing chips rely on diffusion to create a range of different solute concentrations, [18, 19, 20] our randomly-designed chips use only the series of channel splits and merges in the chip to generate different concentrations. Consequently,

our randomly-designed chips can be operated over a wider range of flow rates than chips that rely on diffusion, and users can specify both the desired concentration *and* the desired flow rate at each outlet. Additionally, while microfluidic valve- and pump-based serial diluters generate waste fluids with undesired concentrations during operation, [21, 22] our randomly-designed chips generate only fluid with the desired concentrations and create no waste fluid. These differences show that randomly-generated microfluidic chips can have unexpected and useful advantages over their human-designed counterparts.

Finally, our technique can be used to find microfluidic chip designs that do more than simply generate solutions with user-specified concentrations. *Any* microfluidic phenomenon that can be simulated could be the basis for a library of chip designs and simulations that could subsequently be queried by users for a wide variety of different applications. For example, a library of random chip designs whose simulations include two-phase flow (oil and water) could be used to automatically design microfluidic droplet generators, and a library whose simulations include particle tracing could be used to automatically design cell sorters. Generating these libraries may require a non-trivial amount of computation time: our proof-of-concept library required three weeks to complete. However, that library was generated using an ordinary desktop computer, and higher-performance hardware could speed up library generation considerably. As libraries of random chip designs proliferate in the future, even complex lab on a chip devices for important research and healthcare applications could be created in seconds without actually designing them.

2.4 Supplementary Materials

2.4.1 Role of diffusion constant in chip behavior

To determine whether solutes with different diffusion coefficients still behave as predicted in our randomly-designed microfluidic chips, we simulated each of the 10,513 random chip designs using three different model solutes: Na^+ (representative of ions), fluorescein (representative of small molecules), and bovine serum albumin (BSA; representative of proteins). The simulation results from all three solutes are summarized in Supplementary Figure 2.6. The similarity of the results from each solute confirm that our library of randomly-designed chips can be used to predict the behavior of a wide range of different solutes from ions to proteins.

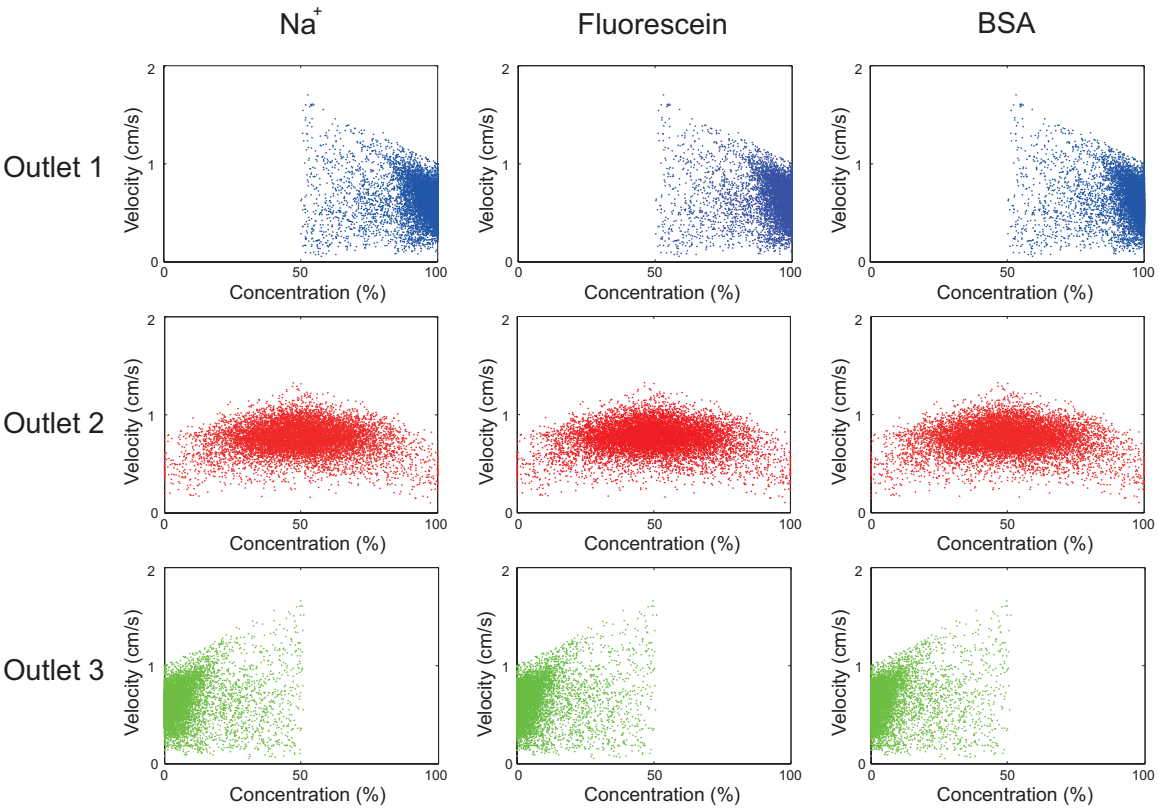

Figure 2.6: Fluid velocity vs. concentration for each outlet in each of the 10,513 chip designs in our simulation library, using the diffusion coefficients of three different solutes: Na^+ (1.33×10^{-9} m^2/s), fluorescein (4.25×10^{-10} m^2/s), and bovine serum albumin (BSA; 6.38×10^{-11} m^2/s). Each solute behaves similarly in each chip design, meaning that our randomly-designed chips will still function as predicted regardless of the specific solute used.

Chapter 3

MOPSA: A microfluidics-optimized particle simulation algorithm

Reprinted with permission from "MOPSA: A microfluidics-optimized particle simulation algorithm" by Junchao Wang, Philip Brisk, and William H. Grover. Biomicrofluidics 11, 034121 (2017).

3.1 Introduction

The field of lab-on-a-chip and micro total analysis devices has been growing rapidly for nearly 40 years, [1] and microfluidic chips have found important applications in biological and chemical analysis. Many of these chips contain particles such as cells, microbeads, or droplets. For example, microfluidic cell sorters are emerging as powerful tools for point-of-care testing and biological research.[23, 24]

As the variety of microfluidic devices increases, computer-based simulations of microfluidics have become more important.[25, 26, 27] Microfluidics researchers are increas-

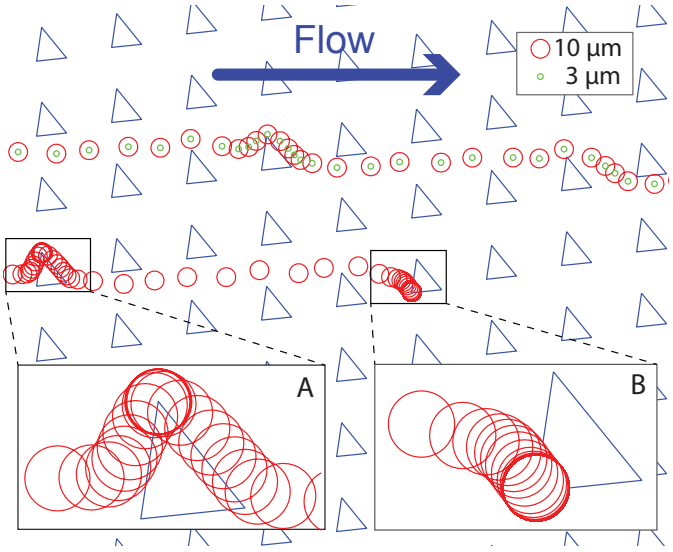

Figure 3.1: Results from using the commercial finite element analysis software COMSOL Multiphysics to simulate the behavior of two sizes of cells flowing through a deterministic lateral displacement (DLD) microfluidic cell sorter chip containing an array of triangular posts. While this DLD chip was designed to separate cells based on size, in this simulation the small cell (3 μm diameter; green) and large cell (10 μm; red) follow exactly the same trajectory and are not separated. Additionally, the simulated cells sometimes impossibly overlap the chip's triangular posts **(A)** or stick permanently **(B)**. Deficiencies like these make it difficult to accurately simulate flowing cells and other particles in microfluidics using existing commercial software tools.

ingly using finite element analysis (FEA) software such as COMSOL Multiphysics (COMSOL Inc., Burlington, MA), Autodesk CFD (Autodesk Inc., Venice, CA), and Fluent (ANSYS Inc., Canonsburg, PA) to simulate and optimize their microfluidic chip designs. These software tools excel at calculating the fluid velocity field in a chip. However, these tools are not well suited for simulating the behavior of particles like cells, beads, and droplets flowing in microfluidic chips. For example, Figure 3.1 shows the results from using COMSOL Multiphysics to simulate the behavior of a deterministic lateral displacement (DLD) cell sorter. DLD sorters use arrays of micron-scale posts to sort cells based on their size. As cells flow through the chip and interact with the posts, large and small cells follow different paths and are therefore separated by the chip.[28, 29] However, in the simulation results from COMSOL Multiphysics in Figure 3.1, the larger 10 μm cells (red) and smaller 3 μm cells (green) follow exactly the same trajectory. If the simulation were accurate, the cells would have followed different trajectories. Additionally, by zooming in to the simulation results in Figure 3.1A, we see that the 10 μm cell actually overlaps with the wall of a triangular post (an impossible situation). These observations confirm that the particle tracing algorithm in COMSOL Multiphysics treats the particle as a mathematical point with no area or volume, an assumption that can be acceptable in macro-scale physics but not in microfluidics. This assumption leads to additional problems: if the simulated single-point particle in COMSOL Multiphysics is located on a mesh node with a fluid velocity magnitude of zero, the particle will remain stuck at that location forever (Figure 3.1B). This sticking behavior remains

regardless of which boundary condition is chosen for the wall in COMSOL Multiphysics (*freeze, stick, bounce, disappear,* or *reflect diffusely*).

The lack of commercial software tools that accurately simulate particles in microfluidics has profound consequences. For example, researchers developing new cell sorters like the DLD cell sorter shown in Figure 3.1 typically have to fabricate and test these chips to determine whether they sort the desired cells. While some empirical models for predicting the sorting behavior of DLD chips do exist for chips with cylindrical posts,[30] these models do not easily extend to posts with arbitrary shapes like the triangles shown in Figure 3.1. And with each chip design iteration requiring fabrication and testing in the lab, it can take months for researchers to develop a functional device. This slows the progress of research and keeps valuable research tools and lifesaving medical diagnostics out of the hands of the people who need them.

To address this problem, we developed a microfluidics-optimized particle simulation algorithm (MOPSA) that is capable of simulating lifelike trajectories for some particles in situations where existing software tools fail. MOPSA treats a particle as a 2D rigid circular object instead of a single point when calculating the particle's velocity. MOPSA also checks whether the particle overlaps a solid object (a post or channel wall) at each simulated time step. If overlap is detected, MOPSA applies an empirical correcting function, *wallEffect*, to shift the particle to a new position where it will not overlap walls. By providing researchers with accurate simulations of some types of particles in microfluidic de-

vices, MOPSA should accelerate the development of new microfluidic devices for important applications in biological research, medical diagnostics, and beyond.

Assuming that a particle is a rigid circle means that MOPSA cannot directly simulate the deformability or non-spherical shapes of some particles. In spite of this limitation, by assuming that the particles are rigid circles, MOPSA nonetheless predicts particle trajectories that are consistent with the experimentally observed behavior of published DLD chips sorting some types of deformable and non-spherical cells and droplets. However, undoubtedly there are particles that cannot be approximated as rigid circles and simulated successfully using MOPSA; for these particles other simulation approaches may be necessary and could even be incorporated into future versions of MOPSA (as discussed in the *Conclusions*).

3.2 Simulation

Simulating a microfluidic chip using MOPSA requires two steps:

1. Calculate the fluid velocity field in the chip using conventional finite element analysis software

2. Calculate the trajectory followed by a particle through the chip using MOPSA

In this proof-of-concept demonstration, we used COMSOL Multiphysics to calculate the fluid velocity field and MATLAB to implement MOPSA. We used this approach to simulate three different microfluidic chip models:

1. The *cross channel model* is a typical microfluidic cross-shaped intersection with 200 μm channel widths. Junctions like this are extremely common in microfluidic chips,[31, 32, 33, 34] so assessing the performance of MOPSA in these intersections is very important.

2. The *triangular DLD model* was chosen because of its complexity. Arrays of hundreds of triangular posts offer many opportunities for interactions between particles and the posts and challenge the robustness of MOPSA.

3. The *cylindrical DLD model* enables comparisons between MOPSA and published experimental results. The vast majority of DLD chips in the literature use arrays of cylindrical posts, and we use MOPSA to simulate the paths followed by particles in six published DLD experiments by three different research groups.[28, 35, 36]

While we chose these three models for this proof-of-concept demonstration, we note that, in principle, MOPSA can simulate virtually any microfluidic chip design containing flowing particles, as long as the chip can be modeled in two dimensions (MOPSA is currently a two-dimensional simulation, although it could easily be extended to three dimensions as discussed in the *Conclusions*) and as long as the particles can be approximated as rigid circles or spheres. Finally, to compare MOPSA to existing commercial software tools for particle simulation in microfluidics, we also simulated these chip models entirely in COMSOL Multiphysics (without using MOPSA).

3.2.1 Calculating fluid velocity field in COMSOL

For each microfluidic chip model simulated here, the fluid velocity fields were calculated using finite element analysis software (COMSOL Multiphysics). For accuracy, we used the *Laminar Flow* physics module and a customized free triangle mesh with a maximum mesh size that was equal to or less than the diameter of the smallest particle that would be simulated using that mesh. In the *cross channel model*, the maximum mesh size was 8 μm (for simulating a particle diameter of 50 μm). Inlets were assigned an inlet boundary condition of 5 mm/s normal inflow velocity. In the *triangular DLD models*, the maximum mesh size was 1 μm (for simulating particle diameters of 1, 3, 10, and 12 μm). In the *cylindrical DLD models*, the maximum mesh size was a quarter of the difference between two simulated particles' diameters (*e.g.*, for simulating a DLD chip separating 8 and 9 μm beads, the maximum mesh size was 0.25 μm). Inlets were assigned an inlet boundary condition of 1 mm/s normal inflow velocity. In all models, outlets were assigned an outlet boundary condition of 0 Pa pressure, the remaining boundaries were walls (no-slip boundary condition), and the material filling the channels was water under incompressible flow. A stationary solver was used for calculating the fluid velocity field.

Algorithm 1 Main MOPSA algorithm

Input: Velocity profile in x and y direction of model ($x.csv$, $y.csv$), diameter of particle (D_p), initial position of particle ($position_i$), model geometry file (geometry.dxf), resolution of generating streamlines ($resolution$), vx_{pre}, vy_{pre}, β.

Output: Simulated particle trajectory ($trajectory$)

$polylines \leftarrow geometry.dxf$ ▷ Load microfluidic chip geometry from DXF file.
$x \leftarrow x.csv$ ▷ Load pre-simulated fluid velocity profiles.
$y \leftarrow y.csv$
$node \leftarrow x.csv$ ▷ Load mesh information from x.csv.
$node_{sorted} \leftarrow \text{SORTROWS}(node)$ ▷ Sort nodes based on x-direction.
$geometry_{boundary}[x_{min}\ x_{max}\ y_{min}\ y_{max}] \leftarrow [\text{MIN}(node(:,1))\ \text{MAX}(node(:,1))\ \text{MIN}(node(:,2))\ \text{MAX}(node(:,2))]$ ▷ Define region of interest.
$position_c \leftarrow position_i$ ▷ Initialize current position of particle.
$trajectory \leftarrow position_i$ ▷ Initialize trajectory of particle.
while $position_c \in geometry_{boundary}$ **do**
    $nodes_{covered} \leftarrow \text{FINDCOVEREDNODES}(D_p, position_c, node_{sorted}, node)$
    $v_x \leftarrow \text{GETMEAN}(nodes_{covered}, x)$ ▷ Set particle velocity to average fluid velocity of covered mesh nodes.
    $v_y \leftarrow \text{GETMEAN}(nodes_{covered}, y)$
    $v_y \leftarrow v_y \cdot \beta$ ▷ Scale lateral (y) component of particle velocity by β if necessary.
    if $v_x = 0$ and $v_y = 0$ **then** ▷ If particle has zero velocity (is stuck)...
        $v_x \leftarrow vx_{pre}$ ▷ ...use the particle's velocity from the previous time step instead.
        $v_y \leftarrow vy_{pre}$
    end if
    $position_n \leftarrow position_c + resolution \cdot [v_x\ v_y]$ ▷ Calculate position of particle at next time step.
    $position_n \leftarrow \text{WALLEFFECT}(D_p/2, position_n, polylines)$ ▷ Apply wallEffect (see Algorithm 2).
    $trajectory \leftarrow [trajectory; position_n]$
    $position_c \leftarrow position_n$ ▷ Record new particle position.
    $vx_{pre} \leftarrow v_x$ ▷ Record new "previous" particle velocity for use in next iteration.
    $vx_{pre} \leftarrow v_y$
end while

3.2.2 Simulating particle trajectory with MOPSA

The MOPSA algorithm is summarized in pseudocode in Algorithm 1. We implemented MOPSA in MATLAB to leverage MATLAB's strengths in matrix calculation. Using the built-in MATLAB function *INPOLYGON* significantly simplifies the MOPSA code. In principle, the pseudocode shown here could be implemented in any programming language.

MOPSA first imports the geometry of the microfluidic chip (the design of the chip in DXF format [15]), the fluid velocity field calculated by COMSOL Multiphysics, and the description of the mesh used by COMSOL (lines 1–4 of Algorithm 1). Instead of treating the particle as a single mathematical point, MOPSA uses the average velocity of all of the mesh nodes covered by the particle to determine the particle's velocity profile (lines 11–12). v_x represents the magnitude of the velocity vector in the x direction, and v_y represents the velocity in the y direction. As shown in Algorithm 1 line 13, MOPSA uses an empirically-obtained parameter β to "weight" the components of the particle velocity vector in direction orthogonal to fluid flow (the y direction). For the *cross channel model*, $\beta = 1$ (meaning that the velocities calculated by COMSOL are used as-is). For the *triangular* and *cylindrical DLD* models, $\beta = 1.45$. Details on how this value was determined are provided in *Results and Discussion*.

MOPSA next considers the possibility of particles with zero velocity. In theory, as long as there is fluid flow in a microfluidic channel, mesh nodes with zero fluid velocity should exist only at channel walls. However, in practice, finite mesh size makes it possible for software like COMSOL Multiphysics to predict that nodes *near* channel walls may also have zero fluid velocity. A particle located at a mesh node with zero fluid velocity can become permanently trapped. To detect this situation, MOPSA utilizes a conditional statement (line 14 in Algorithm 1) to determine if the velocity of the particle in the time step will be zero. If so, MOPSA uses the particle's non-zero velocity from the *previous* time step for the current time step calculation (lines 15 and 16). As long as the amount of time between

steps (*resolution*, defined in Equation 3.1 below) is sufficiently small, this substitution does not seem to affect the accuracy of the simulation. In this manner, MOPSA ensures that the particle will not get permanently stuck. Alternatively, for applications in which particle sticking is desired (for example, using antibody-coated posts to capture cells of interest in a microfluidic chip [37]), a threshold can be added to allow particle sticking below a user-specified particle velocity.

MOPSA then uses Equation 3.1 to calculate the particle's position at the next time step:[38]

$$position_n = position_c + resolution \cdot [v_x \; v_y] \tag{3.1}$$

where $position_c$ (μm) is the current position of the particle, $position_n$ (μm) is the position of the particle in the next time step, *resolution* (s) is the amount of time between each calculation step, and v_x and v_y (μm/s) are the average fluid velocities of the mesh nodes covered by the particle, as defined earlier. The variable *resolution* is a parameter that can be customized by the user. A smaller *resolution* makes the simulation more accurate but also more time consuming. If the flow through a chip is relatively fast, then a small value for *resolution* is needed to accurately predict the path followed by a particle in the fluid. If the flow is relatively slow, then a larger *resolution* can be used without sacrificing simulation accuracy. To find an optimal value for *resolution*, a MOPSA simulation can be repeated with successively smaller values for *resolution*. Eventually the predicted particle

trajectories will converge to a single trajectory that will not change with additional decreases in *resolution*; this is the optimal value for *resolution* for a given application.

Equation 3.1 assumes that:

1. The density of the particle is the same as its surrounding fluid.
2. The net force applied to the particle is always zero.
3. Particles do not interfere with each other or channel walls.
4. Particles have smooth surfaces, so friction between fluid and particle surfaces is negligible.
5. The material of the particle is homogeneous.

If these assumptions are unsuitable for a particular application, MOPSA can be modified by replacing Equation 1 with a more complicated drag law such as Stokes' Law [39] or the work of Haider and Levenspiel [40] (we demonstrate a modification of MOPSA later in this work).

Algorithm 2 function wallEffect

Input: microfluidic chip wall geometry information (*polylines*), radius of particle (R_p), current position of particle ($position_c$), particle mapping parameter (k)
Output: shifted position of particle ($position_s$)

function WALLEFFECT($R_p, position_c, polylines$)
  $x_p \leftarrow R_p \cdot \cos(0 : (2\pi/k) : 2\pi) + position_c(1,1)$
  $y_p \leftarrow R_p \cdot \sin(0 : (2\pi/k) : 2\pi) + position_c(1,2)$
  for $i = 0 \rightarrow total_{polylines}$ **do**
    $polyline_c \leftarrow polylines[i]$
    if $polyline_c(1,:) = polyline_c(end,:)$ **then**
      $x_{polyline} \leftarrow polyline_c(:,1)$
      $y_{polyline} \leftarrow polyline_c(:,2)$
      $in \leftarrow$ INPOLYGON$(x_p, y_p, x_{polyline}, y_{polyline})$
      $count_{in} \leftarrow 0$
      $points_{in} \leftarrow []$
      for $j = 0 \rightarrow total_{in}$ **do**
        if $in(j) = 1$ **then**
          $count_{in}++$
          $points_{in} \leftarrow [points_{in}; x_p\ y_p]$
        end if
      end for
      if $count_{in} > k$ or $count_{in} = 0$ **then**
        $position_s \leftarrow position_c$
      else ▷ However, if the particle *does* overlap with the wall:
        $point_{middle} \leftarrow$ [MEAN$(points_{in}(1,:), points_{in}(end,:))$]
        $direction \leftarrow$ NORM$(position_c - point_{middle})$
        $position_s \leftarrow position_c + direction \cdot (count_{in}/k) \cdot R_p \cdot w$
        BREAK()
      end if
    end if
  end for
  return $position_s$
end function

After the particle's position at the next time step has been calculated, MOPSA's *wallEffect* function (Algorithm 2) determines if the particle will collide or overlap with solid walls. As shown in Figure 2, *wallEffect* uses a large number of points ($k = 200$) to map the boundary of the particle located at position a. If any of these k points are located inside a solid wall (*e.g.*, a triangular post, a channel wall, or any other device feature), then the algorithm recognizes that the particle is overlapping the wall and compensates by shifting the particle's position away from the wall to a new location a'. The direction of

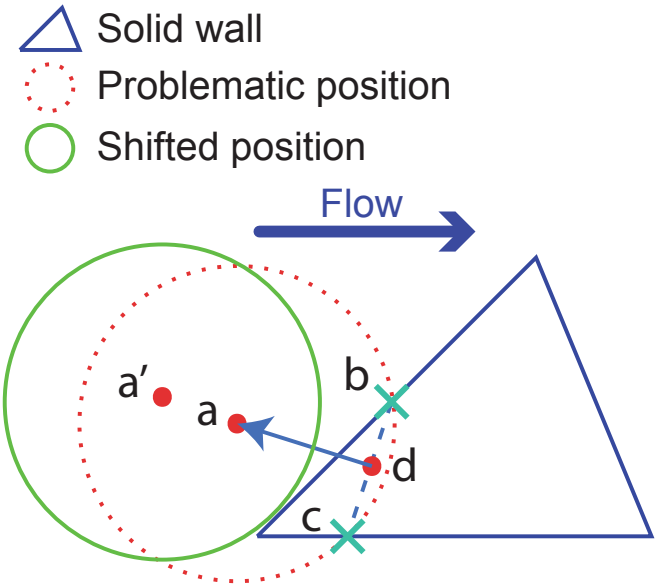

Figure 3.2: Function of the *wallEffect* algorithm. A particle centered at point a ($position_c$ in Algorithm 2) has its boundary represented by 200 points (depicted as small red points). If any of these points overlap with a solid chip feature, such as the triangular post shown (blue), the algorithm shifts the particle to a new location a' (green boundary; $position_s$ in Algorithm 2) in a direction perpendicular to the midpoint d of line bc (defined by the cyan $\times$-shaped points on the intersections between the particle boundary and the edges of the post). The distance of the shift is determined by Equation 3.2. In this illustration, the area of overlap between the particle and post and the distance of the shift are intentionally exaggerated for clarity. In practice, choosing a properly small value of *resolution* (Equation 1) will ensure that the overlap and shift distance are small and the shift direction will be approximately perpendicular to the edge of the post.

this shift is determined by first identifying the two points of contact between the particle perimeter and the wall (b and c), then defining a line segment bc between these two points, and finally computing the midpoint d of the line segment. The direction of the particle shift is perpendicular to line bc (roughly pointing in the opposite direction of the fluid flow) and is calculated using Equation 3.2:

$$position_s = position_c + direction \cdot \frac{count_{in}}{k} \cdot R_p \cdot w \tag{3.2}$$

where $position_s$ is the shifted position of the particle, $position_c$ is the initial (wall-overlapping) position of the particle, $direction$ is a normal vector describing the shifting direction, k is the total number of points used to map the perimeter of the particle, $count_{in}$ is the number of those perimeter mapping points located inside the solid wall, R_p is the radius of the particle, and w is a weight factor that can be adjusted for different models ($w = 1$ was used in this work). Once this algorithm has corrected the overlapping position of the particle, MOPSA moves on to the next simulation time step.

The parameters *resolution* in Equation 1 and w in Equation 2 provide a mechanism for fine-tuning the *wallEffect* algorithm. For example, if the algorithm predicts an unrealistically-large shift distance, reducing *resolution* will decrease the amount of overlap between the particle and wall (and thus decrease the amount of correction applied by *wallEffect*. If the algorithm predicts an unrealistically-small shift distance, increasing w will increase the shift distance. Finally, if the geometry of a wall has concave features, *wallEffect* may yield unexpected results because the particle perimeter could intersect with the

wall in more than two points. In this case, changing w could compensate for unexpected results from concave wall geometries. In any case, $w = 1$ was used for all of the simulations presented here.

3.2.3 Comparison with existing commercial particle tracers

To compare the performance of MOPSA to an existing commercial particle tracer, we repeated all of our particle simulations using the built-in *Particle Tracing for Fluid Flow* physics module in COMSOL Multiphysics. The simulations used a time-dependent solver, and a "drag force" boundary was added into the physics to use Stokes' Law for particle trajectory calculation. For the *triangular DLD model*, the inlet was assigned an inlet boundary condition of 100 particles per release with a uniform distribution. The particles were assigned diameters of 1, 3, 10, or 12 μm (although, as described above, these diameters do not seem to affect the simulation results in COMSOL Multiphysics). The outlet was assigned an outlet boundary condition of *freeze wall*. The six *cylindrical DLD model* chips we simulated used the same conditions as the *triangular DLD model* but with 10 particles per release and the particle diameters shown in Table 3.1. We also tested a range of boundary conditions for the post walls in the DLD models, including *bounce, diffuse scattering,* and *general reflection*. Finally, for the *cross channel model*, the inlet was assigned an inlet boundary condition of 10 particles per release (50 μm particle diameter) with uniform distribution and the outlet was assigned an outlet boundary condition of *freeze wall*.

3.3 Results and discussion

3.3.1 Simulating the cross channel model

The fluid velocity field of the *cross channel model* was solved using COMSOL Multiphysics and is shown in Figure 3.3A. This model simulates a cross-channel with fluid

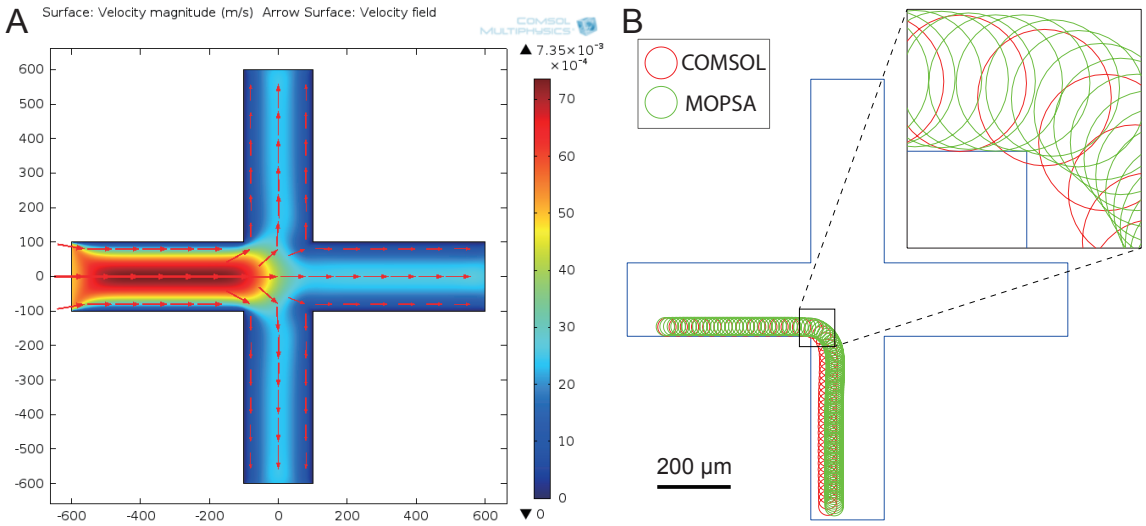

Figure 3.3: (**A**) Fluid velocity field of the *cross channel model* obtained from COMSOL Multiphysics (dimensions in μm and velocity in m/s). (**B**) Simulated trajectories of a 50 μm diameter particle traveling through the *cross channel model*. In the trajectory calculated by the particle tracer in COMSOL Multiphysics (red outlines) the particle overlaps with the channel wall (an impossibility). However, in the trajectory calculated by MOPSA (green outlines) the particle remains separate from the channel wall and exits the intersection offset 18 μm from the COMSOL-predicted trajectory (a significant difference for a particle this size).

entering from the left and exiting out the top, right, and bottom. For rigid spherical particles originating at many locations across the left-hand entrance channel, COMSOL's particle tracer and MOPSA predict very similar particle trajectories. However, for particles that start close to a channel wall, MOPSA's simulation is more realistic than COMSOL's. Figure 3.3B shows trajectories predicted by COMSOL (red) and MOPSA (green) for a

50 μm diameter particle flowing adjacent to the channel wall. In the trajectory predicted by COMSOL, the particle overlaps with the channel wall as it goes around the corner. In contrast, MOPSA detects these channel walls and keeps the particle from overlapping them. The realistic wall interactions simulated by MOPSA result in a particle trajectory that is 18 μm (36% of the particle's size) farther to the right than the trajectory calculated by COMSOL. For simple microfluidic chips with a single intersection, the particle tracing errors introduced by software like COMSOL Multiphysics may be acceptable and may not affect the overall accuracy of the simulation. However, for microfludic chips with hundreds of intersections in parallel (like the hydrodynamic filter of Yamada *et al.* [31]), small errors in each intersection could combine to form a large net error associated with the path followed by particles flowing across the entire chip. In these cases, MOPSA should provide a much more accurate particle trajectory than existing software tools.

3.3.2 Simulating deterministic lateral displacement (DLD)

We also tested MOPSA on the *triangular deterministic lateral displacement (DLD) model* cell sorter to determine how the algorithm performs in a chip with more complex geometry. Figure 3.4 shows the predicted particle trajectories for a 10 μm diameter rigid spherical particle flowing through an array of triangular posts, calculated using COMSOL Multiphysics (red) and MOPSA (green). In both simulations the particle starts at the same location. Since COMSOL treats the particle as a mathematical point, COMSOL's trajectory allows the particle to overlap with a triangular post, which is physically impossible.

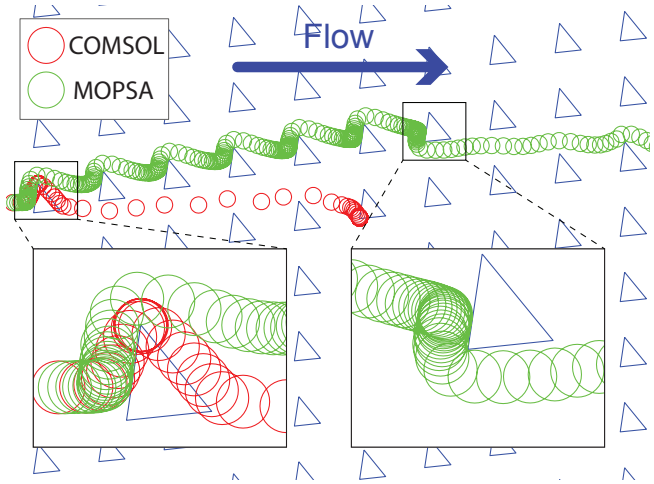

Figure 3.4: Predicted trajectories for a 10 μm diameter particle traveling through an array of triangular posts, obtained using either COMSOL Multiphysics (red) or MOPSA (green). While the COMSOL trajectory allows the particle to overlap with solid posts and stick permanently, the MOPSA trajectory does not. Additionally, the trajectory predicted by MOPSA exhibits the lateral particle displacement expected in DLD chips like this one.

Additionally, later in the COMSOL simulation the particle encountered a zero-velocity mesh node and permanently stopped. In contrast, the trajectory calculated by MOPSA is much more consistent with the known mechanism of DLD. Repeated interactions with triangular posts laterally displace the particle, and no overlapping or sticking with posts is observed (even after repeating the simulation shown in Figure 3.4 hundreds of times with different starting locations for the particle). The design of this chip in DXF format, fluid velocity field, and particle trajectory data are all available for download in online Supplementary Information.

The goal of DLD is to separate different cells and other particles based on their size: smaller particles follow the fluid streamlines straight through the device, while larger particles are "bumped" laterally by the posts and follow a diagonal path through the device.

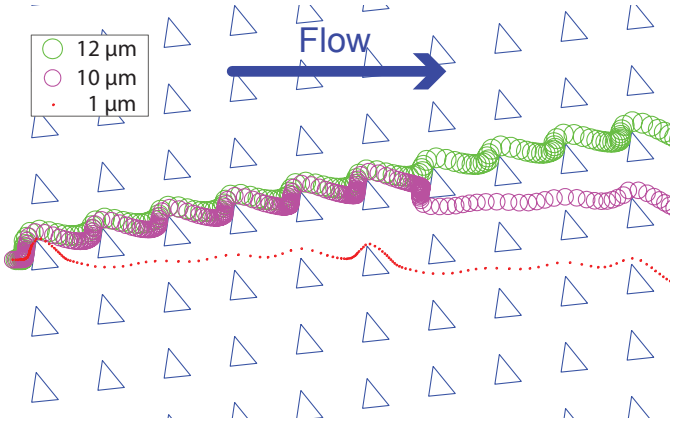

Figure 3.5: Using MOPSA to predict the trajectories of 1 μm (red), 10 μm (purple), and 12 μm (green) diameter particles flowing through a DLD chip containing an array of triangular posts. The simulation predicts that the smallest particle will flow straight through the array and the largest particle will be displaced laterally by interactions with the posts. This is consistent with the known operation of DLD devices. The mid-sized particle (10 μm; purple) is initially displaced laterally but then follows a straight path; this suggests that this particle may be close to the critical diameter for this DLD chip design.

To confirm that MOPSA can simulate the particle sorting capabilities of a DLD chip, in Figure 3.5 we used it to simulate the trajectories of 1, 10, and 12 μm diameter rigid spherical particles flowing through an array of triangular posts. Although particles of all three sizes start in the same location, MOPSA predicts that the particles will exit the chip in different locations: the small 1 μm particle follows the fluid streamlines and moves straight through the chip, and the large 12 μm particle is "bumped" by each post and moves diagonally through the chip. The mid-sized 10 μm particle initially follows a diagonal path but then switches to a straight-through path, leading to a trajectory that lies between the other two particles' trajectories. This suggests that the critical diameter for this DLD chip design (the size of the smallest particle that is laterally displaced in the chip) is close to 10 μm. In contrast, particles of different sizes follow exactly the same trajectory in COMSOL's particle

tracer (Figure 3.1). These results confirm that MOPSA can successfully simulate size-based DLD separations that are difficult or impossible to simulate using existing commercial software, at least for rigid spherical particles. The design of this chip in DXF format, fluid velocity field, and particle trajectory data are all available for download in online Supplementary Information.

3.3.3 Simulating published DLD experiments

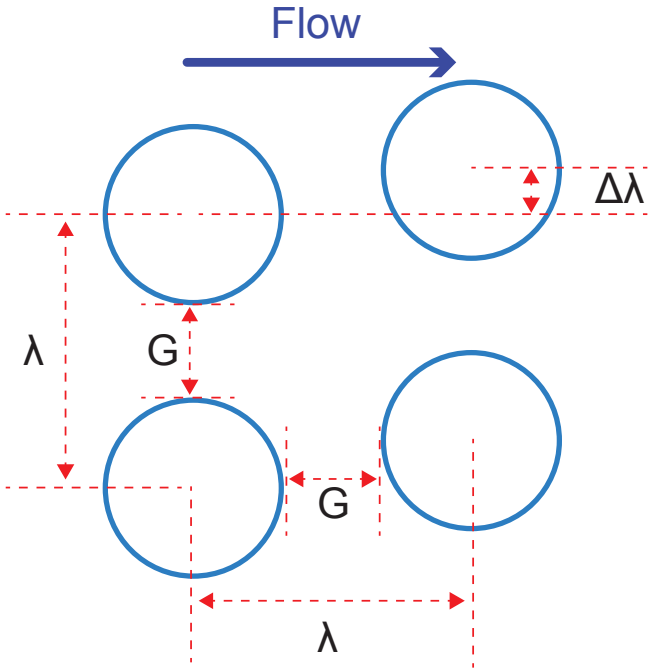

Figure 3.6: Illustration of the key dimensions in a traditional cylindrical-post-based DLD chip: spacing between posts (λ), gap between posts (G), and offset between rows of posts ($\Delta\lambda$).

We next used MOPSA to predict the paths followed by different sized particles in several published DLD chips. We identified six experiments in three published papers

Table 3.1: Experimental details from six published deterministic lateral displacement (DLD) particle separations. These experiments are reproduced by MOPSA in Figure 3.7. The parameters λ, $\Delta\lambda$, and G describe the DLD device design and are defined in Figure 3.6.

Figure	λ (μm)	$\Delta\lambda$ (μm)	G (μm)	Sorted particle types and sizes	Figure in reference
Fig. 3.7A	80	6	25	8 μm and 9 μm beads	Fig. 2A in Li *et al.*[35]
Fig. 3.7B	90	8	25	9 μm and 10 μm beads	Fig. 2B in Li *et al.*[35]
Fig. 3.7C	80	2	20	2 μm platelets and 6 μm red blood cells	Fig. 4A in Li *et al.*[35]
Fig. 3.7D	80	5	20	6 μm red blood cells and 10 μm white blood cells	Fig. 4B in Li *et al.* [35]
Fig. 3.7E	8	0.8	1.6	0.4 μm and 1.0 μm beads	Fig. 2A in Huang *et al.*[28]
Fig. 3.7F	120	6	60	11 μm and 30 μm droplets	Fig. 2C in Joensson *et al.*[36]

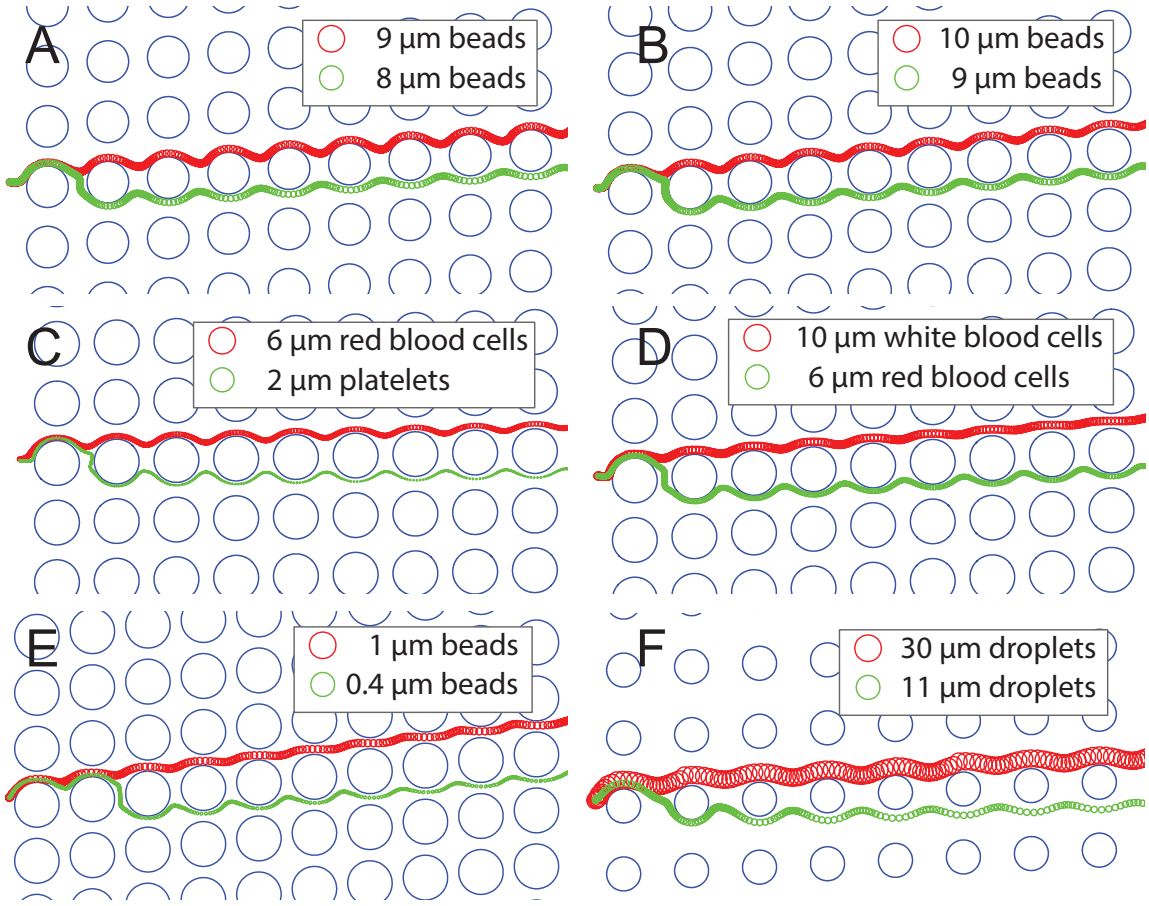

Figure 3.7: Using MOPSA to predict particle trajectories in six published deterministic lateral displacement (DLD) experiments from three research groups.[35, 28, 36] These simulations are based on (**A**) a chip that separates 8 μm and 9 μm beads,[35] (**B**) a chip that separates 9 μm and 10 μm beads,[35] (**C**) a chip that separates 2 μm platelets and 6 μm red blood cells,[35] (**D**) a chip that separates 6 μm red blood cells and 10 μm white blood cells,[35] (**E**) a chip that separates 0.4 μm and 1.0 μm beads,[28] and (**F**) a chip that separates 11 μm and 30 μm droplets.[36] In each case, MOPSA's prediction that the different-sized particles follow different paths is consistent with the experimentally-observed particle sorting behavior of these DLD chips. [35, 28, 36]

[28, 35, 36] in which the chip designs and experimental conditions are described in sufficient detail to allow us to simulate them using MOPSA. These chips use arrays of cylindrical posts with dimensions defined in Figure 3.6 and summarized in Table 3.1.

We first simulated the DLD chip described by Li *et al.*[35] who used this chip to sort 8 μm and 9 μm diameter rigid spherical beads. We found that with a value of $\beta = 1.45$, MOPSA predicts that the 8 μm and 9 μm beads will follow different paths through the DLD chip (Figure 3.7A); this is consistent with the observed bead sorting behavior of this DLD chip.[35]

We then left the value of $\beta = 1.45$ unchanged as we used MOPSA to predict the paths followed by particles through five additional published DLD chips from three research groups (Figure 3.7B–F). In their original publications,[28, 35, 36] these chips were used to sort a wide variety of particle sizes (from 0.4 μm to 30 μm diameters) and types (beads, droplets, red blood cells, white blood cells, and platelets). Some of these particles are not rigid or spherical—droplets are not rigid, and red blood cells, white blood cells, and platelets are neither rigid nor spherical—so one may rightly expect that MOPSA (which assumes rigid spherical particles) may not be able to accurately simulate the trajectories of these particles. However, MOPSA nonetheless successfully predicted that the different-sized particles follow different paths through the chip. This prediction is consistent with the particle-sorting behavior reported by the creators of these chips. In contrast, the built-in particle tracer in COMSOL Multiphysics predicted no differences in the paths followed by the different particles (see Supplementary Figures 2–7). Undoubtedly there are some parti-

cles whose deformability and non-spherical shapes will render MOPSA unable to accurately simulate the behavior of these particles, but at least for the particle types and sizes we studied, MOPSA's assumption of rigid and spherical particles did not seem to adversely affect the quality of the simulation results. The chip design DXF files, fluid velocity profiles, and predicted particle trajectories from both MOPSA and COMSOL for each of the six simulations in Figure 3.7 are available in Supplementary Information.

In some of the DLD chip simulations in Figure 3.7, MOPSA predicted that the smaller particles will follow a somewhat jagged path when the particles flow vertically between two rows of posts. For example, in Figure 3.7C the 2 μm diameter platelets seem to make two sharp turns as they pass from above the third row of posts to below the row. This behavior was unexpected; to determine its source, we examined the fluid velocity fields calculated by COMSOL for these DLD simulations. Close inspection of the gaps between columns of posts (Supplementary Figure 8) reveals that COMSOL predicted asymmetric fluid velocity fields in these gaps in the cases where the MOPSA-predicted particle paths were most jagged. For example, the fluid velocity field calculated by COMOSL for the chip in Figure 3.7C (shown in Supplementary Figure 8C) has a diagonal region of low flow between every two pillars; when MOPSA uses this fluid velocity field to predict the path followed by a particle between the pillars, the asymmetry in the field results in a jagged predicted particle path. These particle trajectories do not seem to affect the overall simulation results (that different-sized particles follow different paths through the DLD chips).

Why was it necessary to increase β to successfully simulate DLD chips in MOPSA? One explanation could be our assumption that the presence of the particle will not affect the fluid velocity profile in the chip. In actuality, the presence of a particle *could* increase the hydrodynamic resistance of the DLD chip.[41] This could reduce the fluid velocity in the direction of flow (the x direction) and make the fluid velocity in the direction perpendicular to flow (the y direction) more significant. Increasing β from 1 to 1.45 adds additional weight to the y component of the fluid velocity and enables MOPSA to accurately simulate particles in DLD devices, at least for the particle sizes and types considered here. While this value of β may not be suitable for all DLD devices or particle sizes and types, it is noteworthy that all six previously-published DLD chips we replicated in this work were successfully simulated using the same value for β (1.45).

3.3.4 Extending MOPSA to simulate particles with different densities

In its current form, MOPSA is capable of simulating rigid spherical particles in a variety of different microfluidic applications. However, in some cases it may be necessary to add additional physics modules to MOPSA to simulate certain particle properties. For example, microfluidic chips have been demonstrated that sort cells and other particles by their densities.[42, 43] Since the current MOPSA assumes that all particles have the same density (and the particles' density equals the density of the fluid around the particles), MOPSA cannot currently be used to simulate the behavior of these density sorter chips.

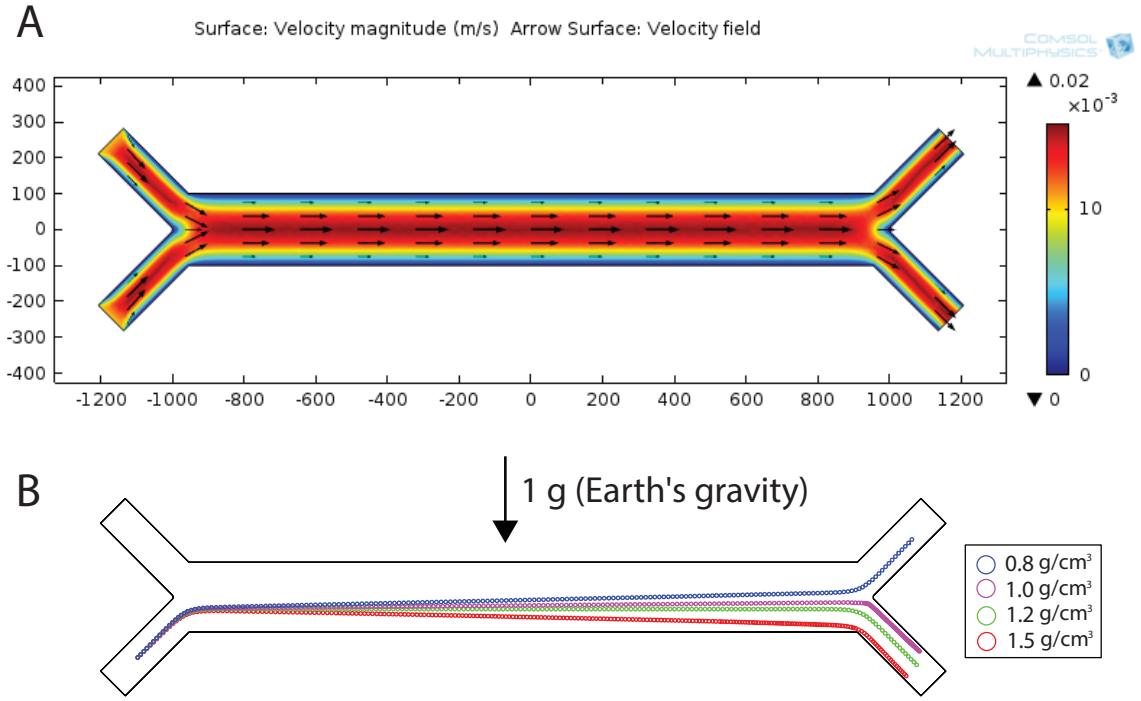

Figure 3.8: Using a modified version of MOPSA to simulate the behavior of a microfluidic chip for sorting cells and other particles by their densities. The chip simulated is based on published results[42, 43] and consists of two inlets that merge into a single horizontal channel before splitting into two outlets. The chip is oriented on its edge relative to Earth's gravity. (**A**) Fluid velocity field for the density sorter chip, obtained using COMSOL Multiphysics. (**B**) After modifying MOPSA using the approach of Haider and Levenspiel,[40] MOPSA predicts the paths followed by four rigid spherical particles of different densities as they enter from the lower input and travel through the chip. As expected, particles with density greater than the fluid density of 1.0 g/cm^3 exit the lower outlet, and particles with density less than the fluid density exit the upper outlet. This demonstrates that additional particle properties (like density) can be added to MOPSA when necessary.

To demonstrate that additional particle properties can be readily added to MOPSA when needed, we added support for particle density to MOPSA. Our approach uses the work of Haider and Levenspiel[40] to predict the effects of buoyancy on particles. We then used our modified MOPSA to predict the paths followed by rigid spherical particles of four different densities (0.8, 1.0, 1.2, and 1.5 g/cm^3) in a microfluidic chip filled with water (1.0 g/cm^3) and oriented on its edge relative to Earth's gravity. In Figure 3.8, all four particles start at the same location in a lower inlet. As the particles flow along the horizontal channel, our modified MOPSA correctly predicts that the particles with density greater than the fluid density will sink downward and exit via the lower outlet, and the particles with density greater than the fluid density will float upward and exit via the upper outlet. Since different cell types often have different densities,[44] our modified MOPSA may be used to help design chips for sorting cells by their type, an important capability in biological research and medical diagnostics.

3.4 Conclusions

In this work we presented MOPSA, an algorithm for optimized particle simulation in microfluidic chips. By treating particles as two-dimensional objects instead of single points and applying corrections when particles interact with channel walls, MOPSA can accurately simulate particle behaviors that particle tracers in existing commercial software tools cannot. Consequently, adopting MOPSA into the design process for microfluidic chips that contain cells, droplets, and other particles should be beneficial.

Limitations of MOPSA and future directions

MOPSA makes several assumptions about the particles it simulates: the particles are circular or spherical, they have the same density as the fluid around them, they have smooth surfaces, they do not interact with other particles, and so on. These assumptions may limit the utility of MOPSA for some applications. For example, in its current form, MOPSA cannot simulate effects of inertia on either the particles or the fluid in a chip; these effects have been used as the basis for particle sorting and focusing in microfluidics.[45] However, for at least some particle sizes and types, MOPSA's assumption of rigid spheres seems to still yield results that are consistent with experimental observations.

For particles that cannot be assumed to be rigid spheres, a number of modeling techniques have been proposed for simulating the behavior of these particles.[46] For example, dissipative particle dynamics [47, 48, 49] has been successfully applied to simulate the deformation of red blood cells in DLD chips.[46, 50] Zhu *et al.* [51] modeled deformable cells as fluid-filled capsules enclosed by neo-Hookean membranes. [**?**] Kruger *et al.* [52] used the immersed-boundary method [53] to model membrane dynamics during cell deformation. These and other models can be used in situations where MOPSA fails to accurately predict the behavior of a non-rigid, non-spherical particle; some of them could even be incorporated into future versions of MOPSA.

We also demonstrated that MOPSA can be easily extended to simulate other particle properties like particle density. This same approach can be used to enable MOPSA

to support other important properties of particles. For example, by altering the *wallEffect* algorithm to allow cells to stick to channel walls, one could simulate the behavior of microfluidic devices that intentionally capture cells in this manner.[37] Additionally, to include the effects of other forces such as electrostatics and acoustics on a particle's trajectory, additional equations can be added to Line 18 in Algorithm 1. To simulate particle-particle interactions that may be important in applications with high particle concentrations, it may be necessary to periodically recalculate the fluid velocity vector field while including particle positions (and particle-induced drag) in the finite element analysis calculation. In our experience, including particle positions in this manner has a negligible effect on simulation accuracy but significantly increases the required computational time, but it may be necessary for some applications.

Finally, MOPSA is currently a two-dimensional simulation technique and cannot be used to simulate three-dimensional microfluidic channel networks. However, nothing about MOPSA precludes extending the algorithm to three dimensions. This would enable MOPSA to simulate particle trajectories in emerging 3D-printed microfluidic chips that can have channels in all three spatial dimensions.[54]

3.5 Supplementary Materials

The supplementary files described in this section are available for free download from the authors' website at `http://groverlab.org/mopsa`.

3.6 Supplementary figures

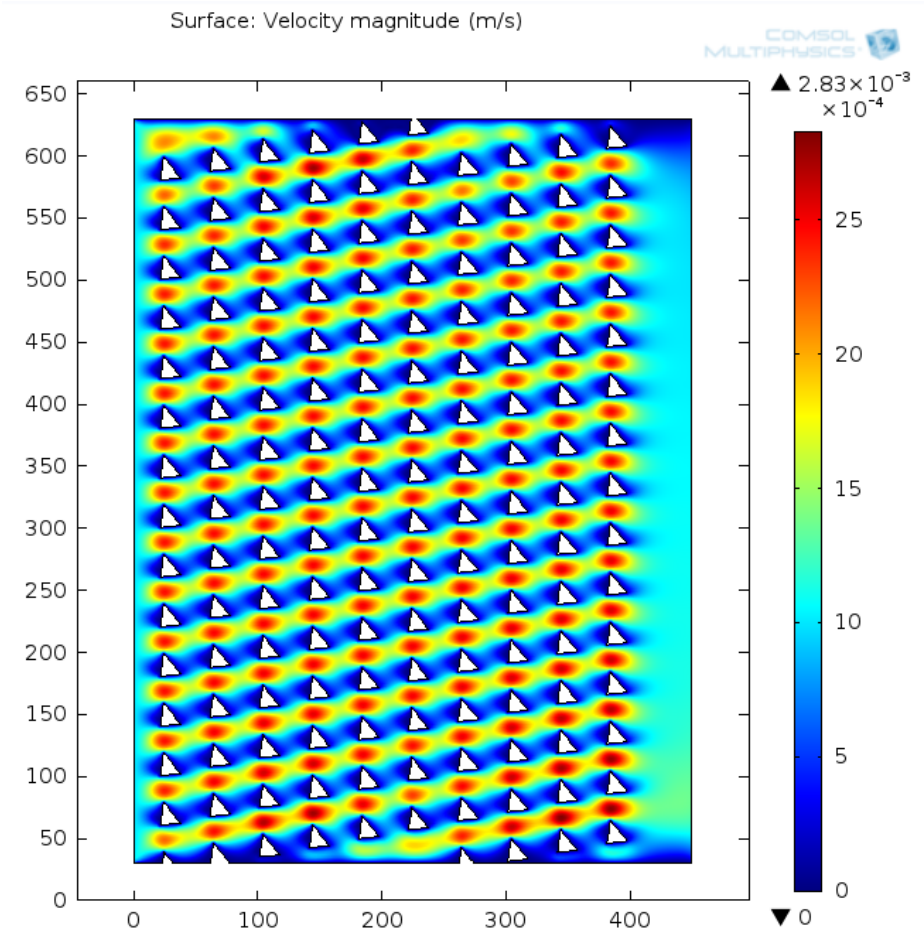

Figure 3.9: Fluid velocity field used in Figures 4 and 5. Flow was from left to right.

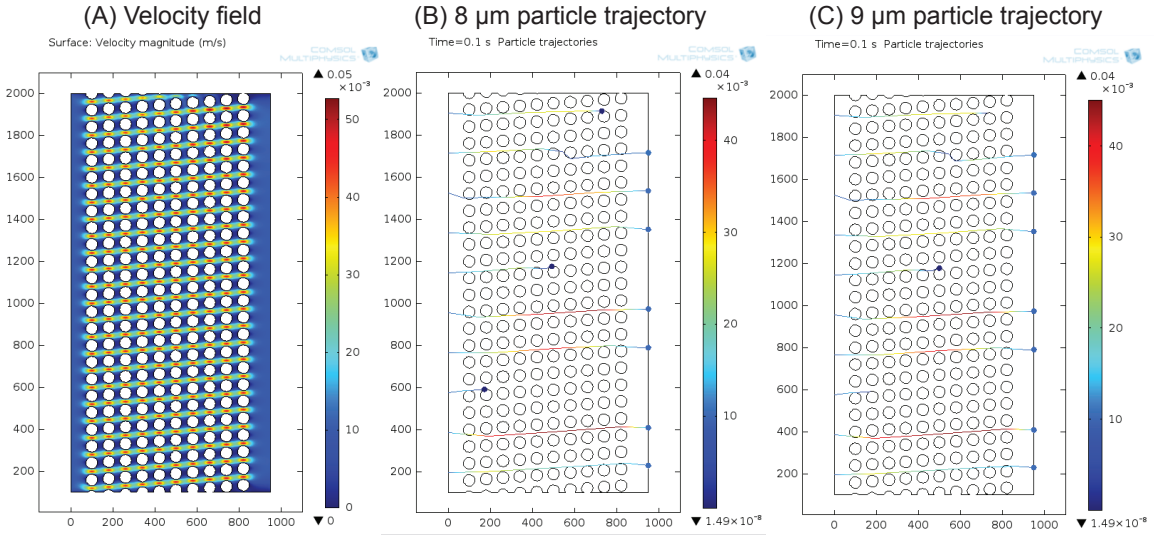

Figure 3.10: Supplementary information for the experiment simulated by MOPSA in Figure 7A. Fluid velocity field (A) and simulated particle trajectories for 8 μm (B) and 9 μm (B) particles obtained using COMSOL Multiphysics. In contrast to the MOPSA simulation in Figure 7A, in the COMSOL simulation the different particle sizes follow identical trajectories (no DLD separation) and occasionally overlap with posts (an impossible occurrence) or become permanently stuck.

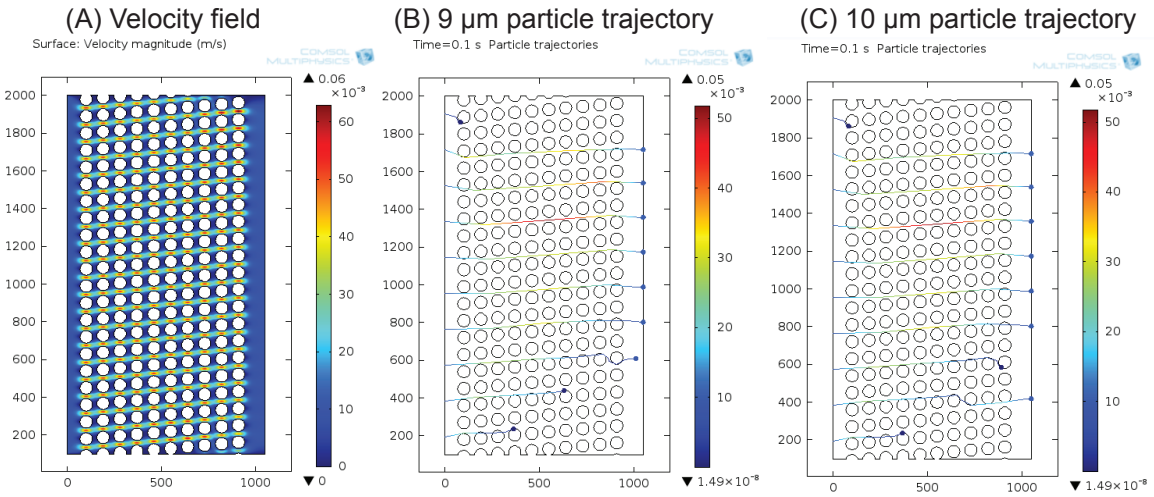

Figure 3.11: Supplementary information for the experiment simulated by MOPSA in Figure 7B. Fluid velocity field (A) and simulated particle trajectories for 9 μm (B) and 10 μm (B) particles obtained using COMSOL Multiphysics. In contrast to the MOPSA simulation in Figure 7B, in the COMSOL simulation the different particle sizes follow identical trajectories (no DLD separation) and occasionally overlap with posts (an impossible occurrence) or become permanently stuck.

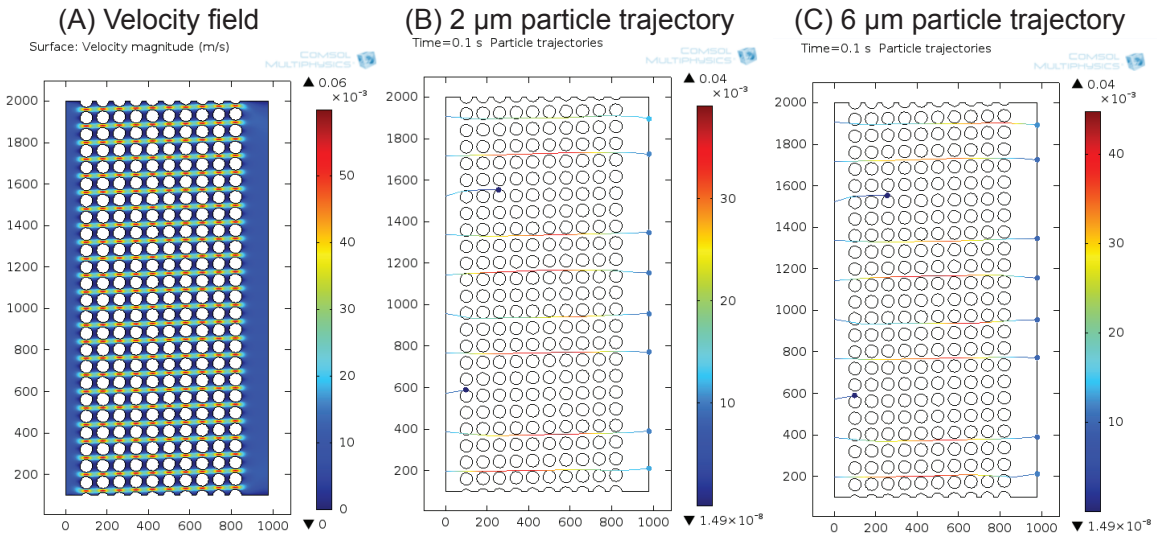

Figure 3.12: Supplementary information for the experiment simulated by MOPSA in Figure 7C. Fluid velocity field (A) and simulated particle trajectories for 2 μm (B) and 6 μm (B) particles obtained using COMSOL Multiphysics. In contrast to the MOPSA simulation in Figure 7C, in the COMSOL simulation the different particle sizes follow identical trajectories (no DLD separation) and occasionally overlap with posts (an impossible occurrence) or become permanently stuck.

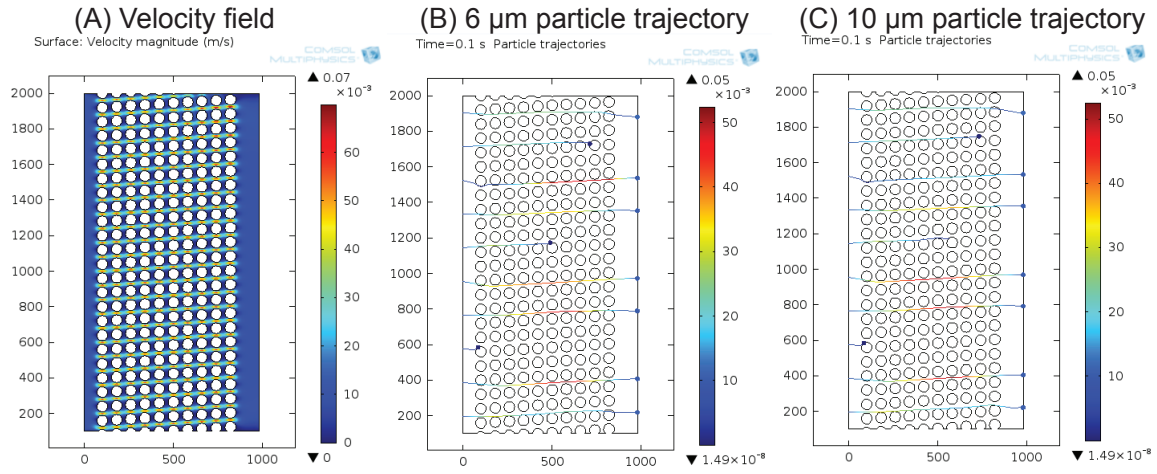

Figure 3.13: Supplementary information for the experiment simulated by MOPSA in Figure 7D. Fluid velocity field (A) and simulated particle trajectories for 6 μm (B) and 10 μm (B) particles obtained using COMSOL Multiphysics. In contrast to the MOPSA simulation in Figure 7D, in the COMSOL simulation the different particle sizes follow identical trajectories (no DLD separation) and occasionally overlap with posts (an impossible occurrence) or become permanently stuck.

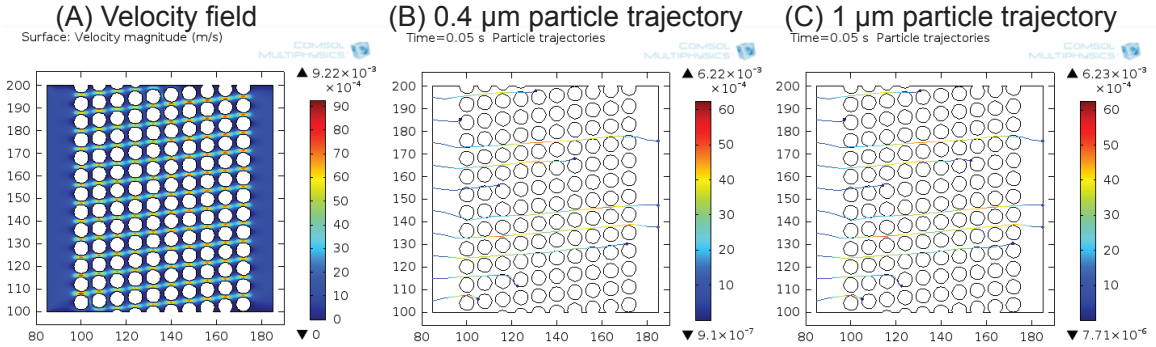

Figure 3.14: Supplementary information for the experiment simulated by MOPSA in Figure 7E. Fluid velocity field (A) and simulated particle trajectories for 0.4 μm (B) and 1 μm (B) particles obtained using COMSOL Multiphysics. In contrast to the MOPSA simulation in Figure 7E, in the COMSOL simulation the different particle sizes follow identical trajectories (no DLD separation) and occasionally overlap with posts (an impossible occurrence) or become permanently stuck.

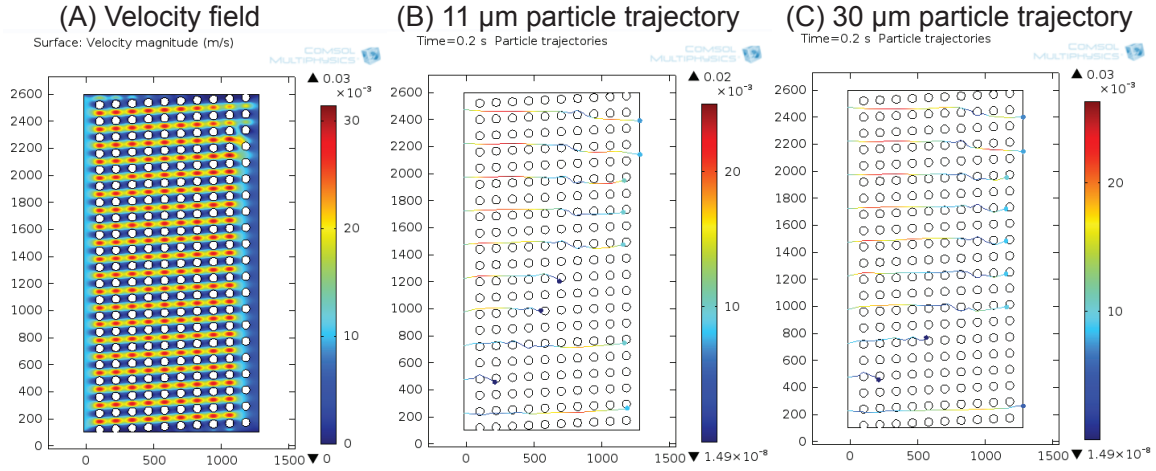

Figure 3.15: Supplementary information for the experiment simulated by MOPSA in Figure 7F. Fluid velocity field (A) and simulated particle trajectories for 11 μm (B) and 30 μm (B) particles obtained using COMSOL Multiphysics. In contrast to the MOPSA simulation in Figure 7F, in the COMSOL simulation the different particle sizes follow the same trajectories (no DLD separation) and occasionally overlap with posts (an impossible occurrence) or become permanently stuck.

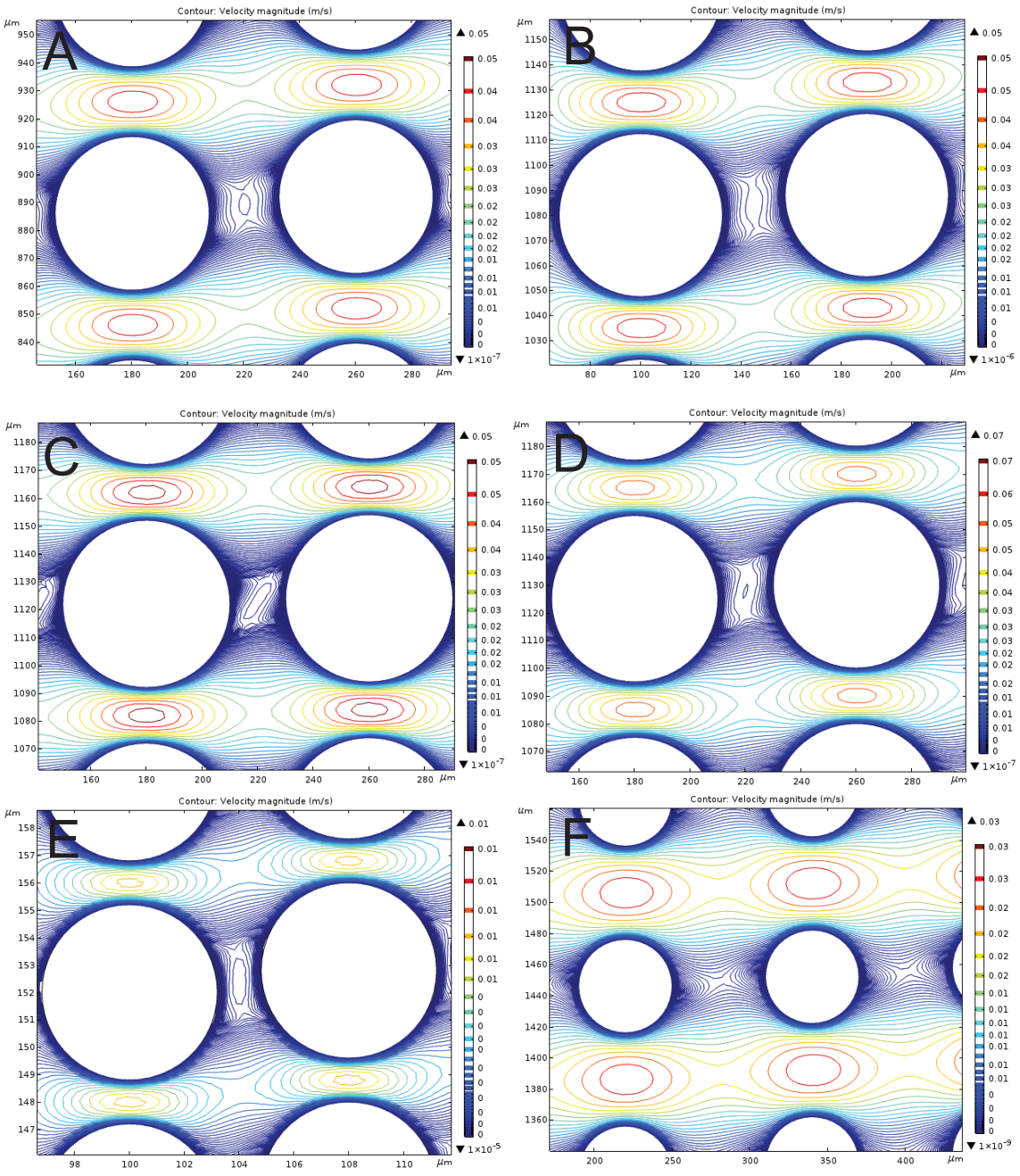

Figure 3.16: Supplementary information for the experiment simulated by MOPSA in Figure 7. Closeups of fluid velocity fields (A–F) corresponding to each MOPSA simulation in Figure 7 (A–F).

Chapter 4

Instantaneous simulation of fluids and particles in complex microfluidic devices

4.1 Introduction

Computer-based simulation is becoming increasingly important in the design of microfluidic devices. Finite element analysis (FEA) software enables researchers to study microfluidic phenomena in their chips [55, 56], optimize existing chip designs [57], and even automate the design of new devices [58]. However, existing simulation tools suffer from several notable drawbacks which have unnecessarily raised the barrier to entry and slowed widespread adoption of these tools in the microfluidics community. First, existing software tools for FEA are complex multi-purpose tools with significant "learning curves" and little customization for microfluidics. Second, the computational requirements of performing the FEA simulation can be prohibitive, leading to exorbitant runtimes. For example, a recent

numerical simulation of the acoustic viscous torque effect on the rotation of a single particle took *several days* to complete using a quad-core Intel Xeon X5570 CPU workstation with 48 GB of RAM [59]. This is tremendously costly, especially during early-stage design space exploration, when chip designers need to rapidly consider a large number of variations in chip designs. Fast simulation times could also be particularly beneficial for emerging computer-based tools that automatically design application-specific custom microfluidic chips [58].

This work presents a new technique to reduce microfluidic simulation times compared to conventional techniques. This technique is inspired by *memoization*, a method for solving complex problems by breaking them into pre-solved sub-problems [60]. We first decompose a microfluidic chip into two types of components: *channels* and *channel intersections*. The behavior of fluid in the channels is determined using fluid flow models like the Hagen-Poiseuille equation and simple analogies from electronic circuits like Kirchhoff's Laws. The behavior of fluid and particles in the intersections is determined by querying a database containing tens of thousands of pre-simulated intersections. By combining intersection simulations retrieved from the database with channel calculations obtained using electrical circuit analogies, we construct a complete simulation of the behavior of fluids and particles in a typical microfluidic chip in around one second on a conventional laptop computer, yielding results that are virtually identical to those obtained in hours or days using conventional software.

4.2 Theory of instantaneous microfluidic simulation

The analogy with electronic circuits has long been used to model fluid flow in microfluidic channels [61, 62]. For example, a voltage difference between two points in an electronic circuit is analogous to a pressure difference ΔP between two points in a microfluidic chip; likewise, the electrical current and resistance of a wire are respectively analogous to the fluid flow rate Q and hydrodynamic resistance R_h of a microfluidic channel. The flow rate in a channel is proportional to the applied pressure drop and inversely proportional to the hydrodynamic resistance of the channel; this is the microfluidic equivalent of Ohm's Law:

$$Q = \frac{\Delta P}{R_h} \tag{4.1}$$

Also like electronic resistors, microfluidic channels with hydrodynamic resistances $R_{h1}, R_{h2}, \ldots R_{hn}$ arranged in series have an equivalent hydrodynamic resistance R_h of

$$R_h = R_{h1} + R_{h2} + \cdots + R_{hn} \tag{4.2}$$

and microfluidic channels with hydrodynamic resistances $R_{h1}, R_{h2}, \ldots R_{hn}$ arranged in parallel have an equivalent hydrodynamic resistance R_h of

$$\frac{1}{R_h} = \frac{1}{R_{h1}} + \frac{1}{R_{h2}} + \cdots + \frac{1}{R_{hn}} \tag{4.3}$$

For microfluidic chips containing a complex network of microfluidic channels, Kirchhoff's laws (which are ordinarily used to calculate currents and voltages in electrical circuits) can

be used to determine the flow rate and pressure at each point in the chip. Specifically, the fluidic analog of Kirchhoff's current law predicts that the sum of the flow rates of fluid flowing into a channel intersection equals the sum of flow rates of fluid flowing out of the intersection, and the fluidic analog of Kirchhoff's voltage law predicts that the sum of all of the pressure drops in a loop of channels equals zero.

The non-negligible viscosity of fluid causes the analogy between electronics and microfluidics to break down. For example, while the current-carrying capacity of a wire is simply proportional to its cross-sectional area, the flow-carrying capacity of a microfluidic channel is a complex function of the cross-sectional size *and shape* of the channel, as well as the viscosity of the fluid in the channel. The hydrodynamic resistance R_h of a channel with a rectangular cross section (a common channel shape in microfluidics) is

$$R_h = \frac{12\eta L}{wh^3 F} \tag{4.4}$$

where η is the viscosity of the fluid, L is the length of the channel, w is the width of the channel, h is the height of the channel, and F is the rectangular geometric form factor of the device [63]. Other expressions for R_h are available for channels with other cross-sectional shapes.

By treating the channels in a microfluidic device as a network of resistors and using Equations 4.1–4.4 and Kirchhoff's laws, it is possible to calculate the average flow rate in each channel in a microfluidic chip. Alternatively, electronic circuit simulation software such as SPICE [64], PSpice (Cadence Design Systems, Inc., Rochester, NY) or Simulink

(MathWorks, Natick, MA) can be used to model the chip as a network of resistors and calculate the current (the average flow rate) in each channel.

This analogy between electronic circuits and microfluidic chips cannot predict the behavior of microfluidic channel *intersections*, where the merging and splitting of different fluid streams in these intersections (and the paths followed by particles in these streams) share no obvious analog in electronic circuits. The behavior of fluids in these intersections is nonetheless very important. For example, merging streams of fluids in intersections can generate new solute concentrations, and changes in streamline width in intersections can be used to sort particles like cells by their size [31, 32, 34]. Computational finite element analysis (FEA) software *can* predict the behavior of fluids and particles in channel intersections, but the computational time required to simulate the behavior of these intersections slows down the design process. In summary, there is an unmet need for simulation approaches that combine the speed of the electrical-fluidic analogy (for simulating channels) with the accuracy of finite element analysis (for simulating intersections).

Our proposed solution is to create a database of pre-characterized microfluidic channel intersections. We first introduced a generic Unit Intersection model, shown in Fig 4.1, which has up to four channels (labeled North, East, South, and West), any of which can be an inlet or an outlet with varying inflow or outflow rates. For example, intersection b in Fig 4.1 has a 1 cm/s inlet at North, a 2 cm/s inlet at South, a 3 cm/s outlet at East, and no connection (*i.e.,* a 0 cm/s inlet) at West. We then populated our database with nearly

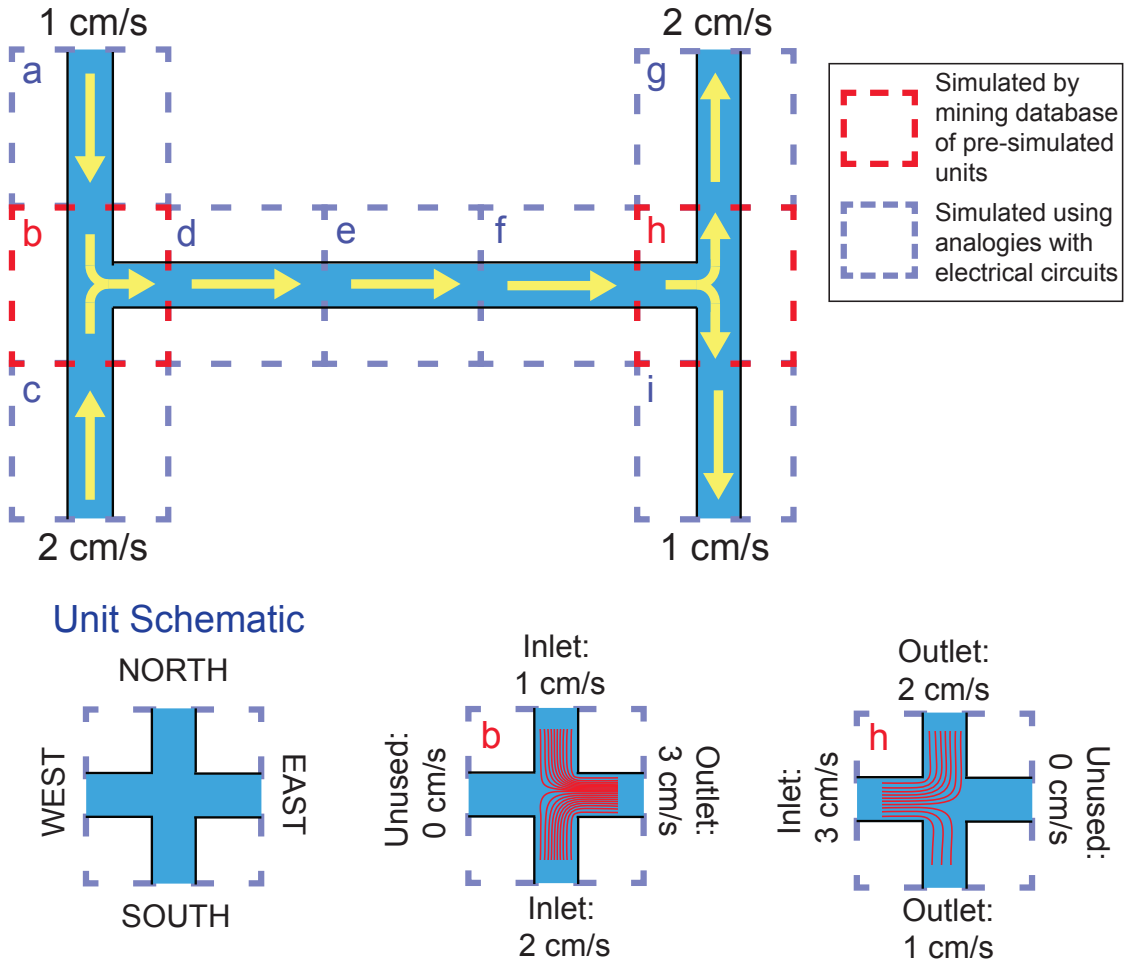

Figure 4.1: Using instantaneous simulation of a microfluidic chip to predict the paths followed by fluids and particles flowing through the chip. A basic "H" channel chip (top) is first separated into nine units ($a - i$). Units a, c, d, e, f, g, and i are simple channels; the flow of fluid in these channels is akin to the flow of electricity in a network of resistors and therefore can be modeled using principles of electrical circuit analysis (Equations 1–4 and Kirchhoff's laws). Units b and h are channel intersections where multiple fluid inlets and outlets come together; the paths followed by particles through these intersections cannot be predicted using analogies with electric circuits. Instead, each unit is described in terms of a prototype intersection (the "Unit Intersection," bottom left) with different boundary conditions. For example, unit b has two inlets (1 cm/s at North, and 2 cm/s at South), one outlet (3 cm/s at East), and one unused connection (West). By querying a database containing nearly 100,000 pre-simulated unit intersections, suitable simulation results for units b and h are retrieved. Particle trajectories from these unit intersections (red lines in b and h) are then expanded through the rest of the chip using streamline theory [65, 66]. In this manner, the paths followed by fluids and particles through the entire chip are predicted in around one second.

100,000 different randomly-generated instances of our Unit Intersection, each of which has a different random assignment of Inlets and Outlets and different flow rates at each of the four channels. We then used commercial finite element software to simulate the behavior of fluids and particles in each intersection and stored the results in the database. To determine the behavior of a specific intersection in a given microfluidic chip, one can simply query the database to find the pre-simulated intersection that is the closest match in terms of inlets, outlets, and flow rates, then retrieve that intersection's simulation results and use them in the overall chip simulation.

Fig 4.1 shows how our instantaneous microfluidic simulation method is applied to predict the paths followed by particles in a simple "H" channel microfluidic device. Our method first separates the whole chip into nine units (marked $a - i$) and then determines whether each unit is a channel or an intersection. Units a, c, d, e, f, g, and i are channels, so we can use the electronic-fluidic analogy (Equations 1–4 and Kirchhoff's laws) to predict the rate of fluid flow through these units. The remaining two units, b and h, are channel intersections, so we query our database nearly 100,000 pre-simulated channel intersections to find similar intersections and retrieve their simulation results. Since intersection units b and h have only three channels, the unused fourth channel in the Unit Intersection is specified to have a flow rate of 0 cm/s. After obtaining fluid streamlines for units b and h from our database, we then expand these streamlines throughout the rest of the chip based on streamline theory, which predicts that at low Reynolds number, a massless particle will

follow fluid streamlines through these channels [66, 65]. In this manner, we can predict the paths followed by fluids and particles in this chip in under one second.

4.3 Materials and Methods

The overall process for instantaneous simulation of microfluidic chips is illustrated Fig 4.2. We implemented this proof-of-concept demonstration in MATLAB (MathWorks, Natick, MA) using the LiveLink API to automate the simulation of microfluidic intersections in COMSOL Multiphysics and saving device designs and simulation results into a MySQL database [67]. All experimental data was collected on a typical laboratory workstation with a 6-core 3.5 GHz Intel Xeon CPU and 32 GB RAM.

4.3.1 Step 1: Constructing the database of Unit Intersection instances

Each of the four channels in the generic Unit Intersection shown in Fig 4.1 can be configured as an inlet or outlet, leading to 16 different possible configurations. The law of conservation of mass (and the fluidic analog of Kirchhoff's current law) forbids configuring all four channels as inlets or all four channels as outlets, so only 14 of the intersection configurations are actually feasible. When generating random instances of the Unit Intersection, we randomly vary the fluid inflow rates from 0 to 2 cm/s (a typical range of flow rates in microfluidic devices). For an intersection with $N = 1$, 2, or 3 outlets, we randomly vary the outflow rate of $N - 1$ of the outlets from 0 to 2 cm/s. The outflow rate of the remaining outlet is set to be the total inflow rate minus the total outflow rate to

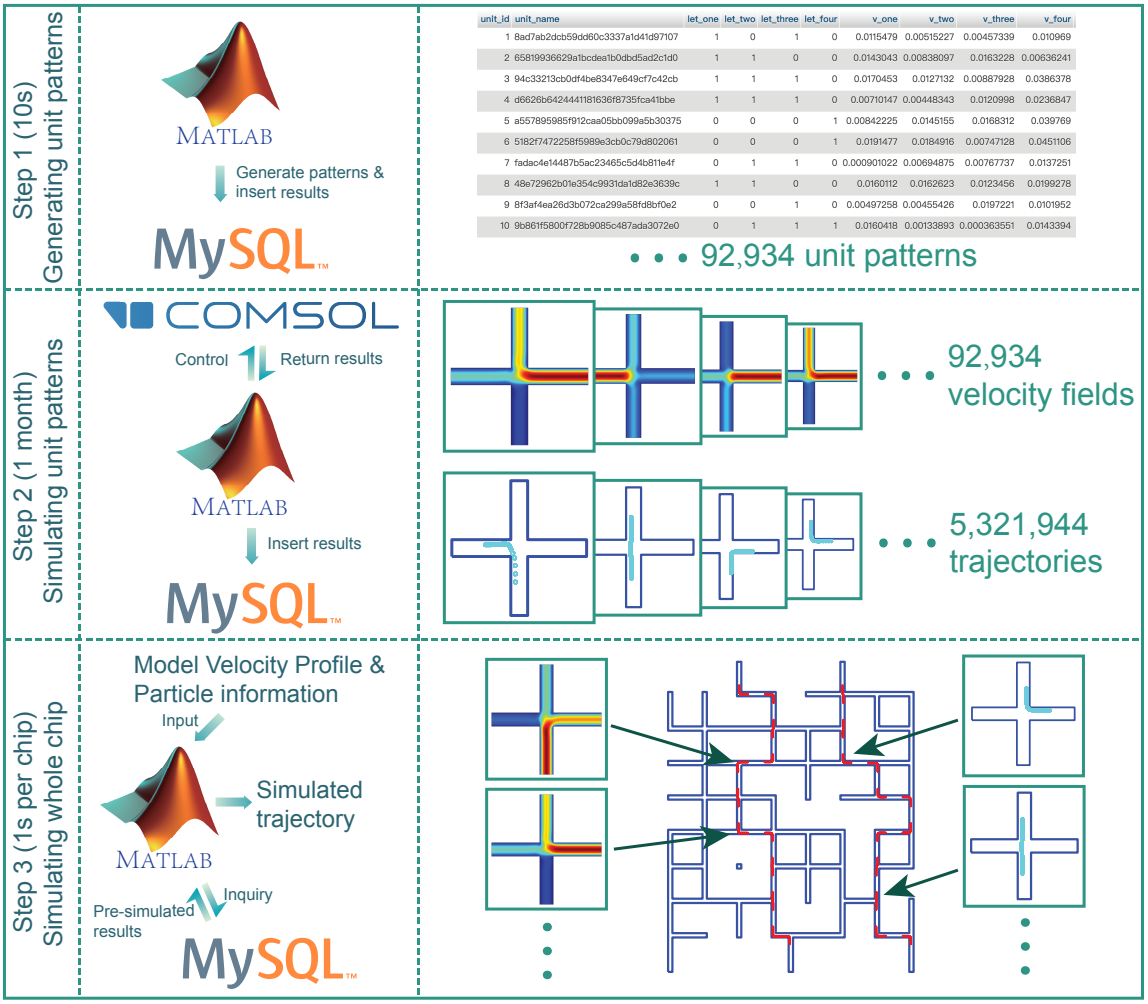

Figure 4.2: Graphical overview of the instantaneous microfluidic simulation method. In Step 1, MATLAB was used to generate 92,934 unit intersections with different random assignments of inlets and outlets and different random flow rates at each of the four connections North, East, South, and West (see also Fig 1). The resulting unit intersections were generated in 10 seconds and saved into a MySQL database. In Step 2, MATLAB was used to control COMSOL Multiphysics to calculate the fluid velocity fields and particle trajectories of each unit intersection and save the simulation results to the MySQL database. A total of 92,934 fluid velocity fields and 5,321,944 particle trajectories were calculated; the entire simulation process took one month to complete, but this step only needs to be performed once. In Step 3, our method is used to predict the path followed by a particle in a given microfluidic chip. First, the fluid velocity profile and particle information of the chip design are imported into MATLAB. Then MATLAB matches each intersection in the chip design with the closest pre-simulated intersection in the MySQL database and returns the corresponding fluid velocity profile and particle trajectory. Finally, the entire path of the particle through the chip is expanded and generated. Simulating a given chip using this approach takes around one second.

ensure conservation of mass. Using this approach, we used MATLAB (MathWorks, Natick, MA) to generate 92,934 different random instances of the Unit intersection, each of which has a channel width of 200 μm and an overall intersection size of 1.6 $\times$ 1.6 mm.

4.3.2 Step 2: Simulating the behavior of each Unit Intersection instance

The LiveLink API for MATLAB was used to automate the simulation of each of the 92,934 microfluidic intersections in COMSOL Multiphysics. The fluid velocity field for each intersection was solved using the *Laminar Flow* physics module with a customized mesh (1 μm maximum mesh size); we confirmed that finer meshes do not alter the simulation results (thus demonstrating mesh independence). The boundary conditions of the inlets and outlets were defined as described in Step 1 above; the remaining boundaries were specified as walls (no-slip boundary condition) and the material filling the channels was water under incompressible flow. A stationary solver was used for calculation.

We then used the *Particle Tracing for Fluid Flow* physics module in COMSOL Multiphysics to calculate the paths followed by particles through each intersection in our database. A "Drag Force" boundary condition was added to each channel, and a particle "inlet" boundary condition with "Uniform distribution" of initial positions was added to all inlets (10 particles per release). The remaining channels were assigned "Outlet" boundary conditions, and the "Freeze" boundary condition was applied to the walls (meaning that particles in contact with the channel walls will stick there, a realistic assumption in many microfluidic chips). This process was repeated for each intersection using a range of particle

diameters from 1 μm to 20 μm. The resulting 5,321,944 simulated particle trajectories were stored in the simulation database. This approach assumes that the entrance lengths of the channels in each intersection (the distance required for a fully-developed flow profile to emerge) is smaller than the lengths of the channels in the simulated intersections.

4.3.3 Step 3: Simulating the behavior of a given chip

Our code for using our instantaneous technique to simulate a given chip is written entirely in MATLAB; it does not use COMSOL Multiphysics or any other FEA simulation because all necessary FEA results for simulating the chip are available in the MySQL database. The software first loads the fluid velocity profile of the chip, which is used to provide the boundary conditions for each intersection. This fluid velocity profile can be obtained manually (using Equations 1–4 and Kirchhoff's laws), using circuit simulation software like SPICE or PSpice, or using computational fluid dynamics simulations obtained from software such as COMSOL Multiphysics. The user then provides the diameter and starting location for each particle to simulate. As simulation proceeds, the software uses streamline theory to project the current position of the particle to the entrance of the next junction. When the software encounters an intersection, it queries the MySQL database to find the pre-characterized unit intersection in the library that most closely matches the boundary conditions of the given intersection. The software retrieves the pre-simulated particle trajectory of the best-matching unit intersection and uses it to predict the path taken by the particle through the intersection. This process repeats until each particle

reaches an outlet of the chip, and the software then provides the user with the complete path followed by the particle through the chip. The overall process for simulating the behavior of a given chip takes around one second to complete.

4.3.4 Comparisons with existing simulation software

To compare the computation time and accuracy of the results of our instantaneous simulation method with existing software tools, we also simulated microfluidic chips using only COMSOL Multiphysics. We compared our technique with COMSOL Multiphysics using two chip designs: a simple chip with three inlets and two outlets shown in Fig 3, and a more complex randomly-designed microfluidic chip [58] shown in Fig 4.

For the simple chip in Fig 3, the fluid velocity field of the chip was solved using the *Laminar Flow* physics module in COMSOL Multiphysics and a stationary solver. Each inlet was assigned a boundary condition of 10 Pa absolute pressure and each outlet was assigned a boundary condition of 0 Pa absolute pressure. The simulation used the default mesh size setting, "Extremely Fine." The *Particle Tracing for Fluid Flow* physics module was then used to predict particle trajectories across the entire chip. A "Drag Force" boundary condition was added to the entire chip, and a particle "Inlet" boundary condition with initial position "Uniform Distribution" and 1.0 μm particle diameter was added to all inlets in the *Laminar Flow* module. The outlets in the *Laminar Flow* module were assigned "Outlet" boundary conditions, and the remaining boundaries were walls ("freeze" boundary

condition). The number of particles per release was set to three, meaning that one particle was released in each of the three inlets.

For the randomly designed microfluidic chip in Fig 4, we attempted to solve the fluid velocity field of the chip was again modeled using the *Laminar Flow* physics module in COMSOL Multiphysics as described above, but the simulation failed after five days (details below in *Results and Discussion*). Each inlet was assigned an inlet boundary condition of 0.01 m/s normal inflow rate and each outlet was assigned an outlet boundary condition of 0 Pa absolute pressure. The maximum mesh size was 20 μm.

4.4 Results

Generating the library of 92,934 unit intersection simulations (92,934 fluid velocity fields and 5,321,944 particle trajectories; Step 2 in Fig 2) took approximately one month of continuous computation on a desktop workstation. Each intersection took an average of 27 s to simulate: 5 s to generate mesh coordinates, 2 s to calculate the fluid velocity field, and 20 s to trace the trajectories of particles through the intersection. The resulting database is about 2.1 TB in size (600 MB for the fluid velocity field data, and 1.5 TB for the particle trajectory data). To the best of our knowledge, mesh coordinate generation and particle trajectory calculation in COMSOL Multiphysics is not amenable to parallelization on multiple processors, which limits the performance benefits that can be accrued by using multi-core CPUs in this step. Consequently, the time required to generate the MySQL

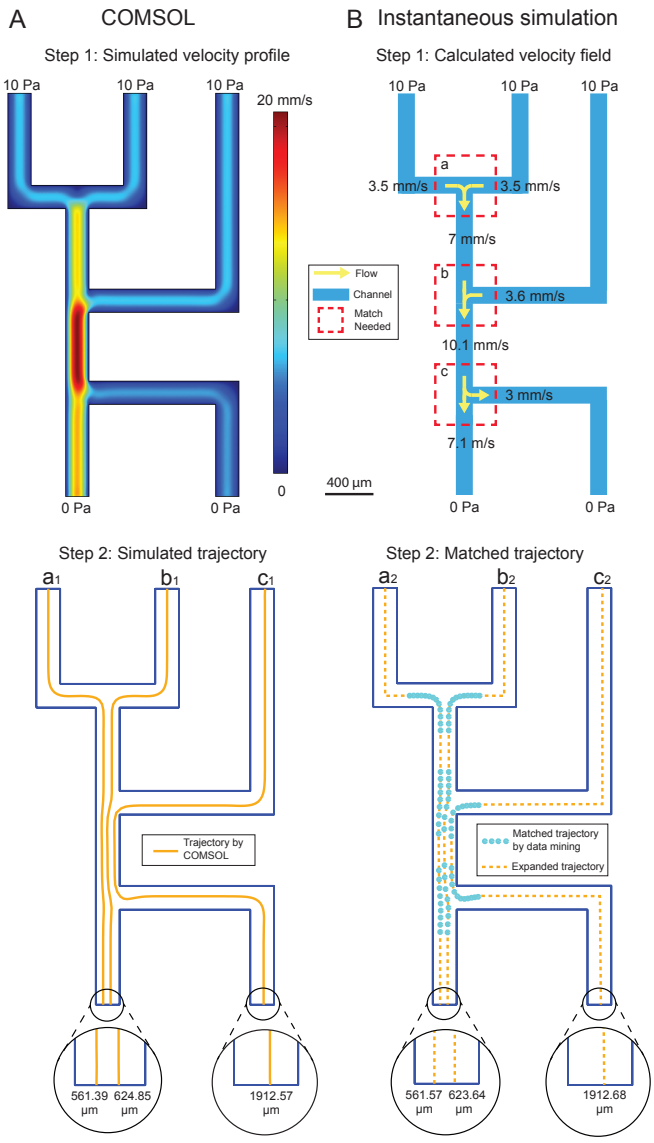

Figure 4.3: Comparison of results from simulating a simple microfluidic chip design using existing commercial software (COMSOL Multiphysics) and our instantaneous simulation method. In the COMSOL Multiphysics simulation (A), the fluid velocity field is calculated using finite element analysis (Step 1) and the *Particle Tracing for Fluid Flow* physics module is used to calculate the paths followed by particles (Step 2). In our instantaneous simulation (B), the flow rates in the channels were calculated using Equations 1-4 and fluidic analogs of Kirchhoff's circuit laws, then simulation results for the channel intersections (dashed red boxes) were found by searching for similar intersections in our database of nearly 100,000 pre-simulated intersections (Step 1). The software retrieved the corresponding particle trajectories from these intersection simulations (blue points in Step 2) and expanded them into whole-chip particle trajectories (yellow points in Step 2). Our instantaneous simulation method was 45 times faster than COMSOL Multiphysics, and the predicted locations of each particle at the exit channels agree to within about 1 μm in a 200 μm wide channel. Raw data of these simulations are available for download (S1 File).

database depends primarily on CPU clock speed. Memory consumption during library construction did not exceed 4 GB.

Fig 4.3 presents results using both COMSOL Multiphysics (Fig 4.3A) and our instantaneous method (Fig 4.3B) to simulate a simple microfluidic chip with 3 inlets, 2 outlets, and 3 intersections. We simulated the paths followed by 1 μm diameter particles originating at the center of each inlet. The COMSOL-based simulation took 135 s to complete: 15 s to solve the fluid velocity field, and 120 seconds to calculate the particle trajectories. For the instantaneous simulation, Equations 1–4 and fluidic analogs of Kirchhoff's laws were used to determine the flow rate boundary conditions of intersections a, b, and c; these boundary conditions are are 3.5 mm/s for the East and West inlets and 7 mm/s for the South outlet of intersection a; 7 mm/s for the North inlet, 3.5 mm/s for the East inlet, and 10.1 mm/s for the South outlet of intersection b; and 10.1 mm/s for the North inlet, 3 mm/s for the East outlet, and 7.1 mm/s for the South outlet of intersection c. We then queried the MySQL database to identify the best matches for each of these three intersections, then expanded the pre-simulated particle trajectories from the database (blue dots) into trajectories that extend through the entire chip (yellow dashes). The entire process of simulating the chip using the instantaneous method took 3 s to complete: 2 s for determining the boundary conditions of the three intersections and obtaining simulation results from the MySQL database, and 1 s to generate the whole-chip particle trajectories. In addition to being 45 times faster than COMSOL Multiphysics, our instantaneous method's results were

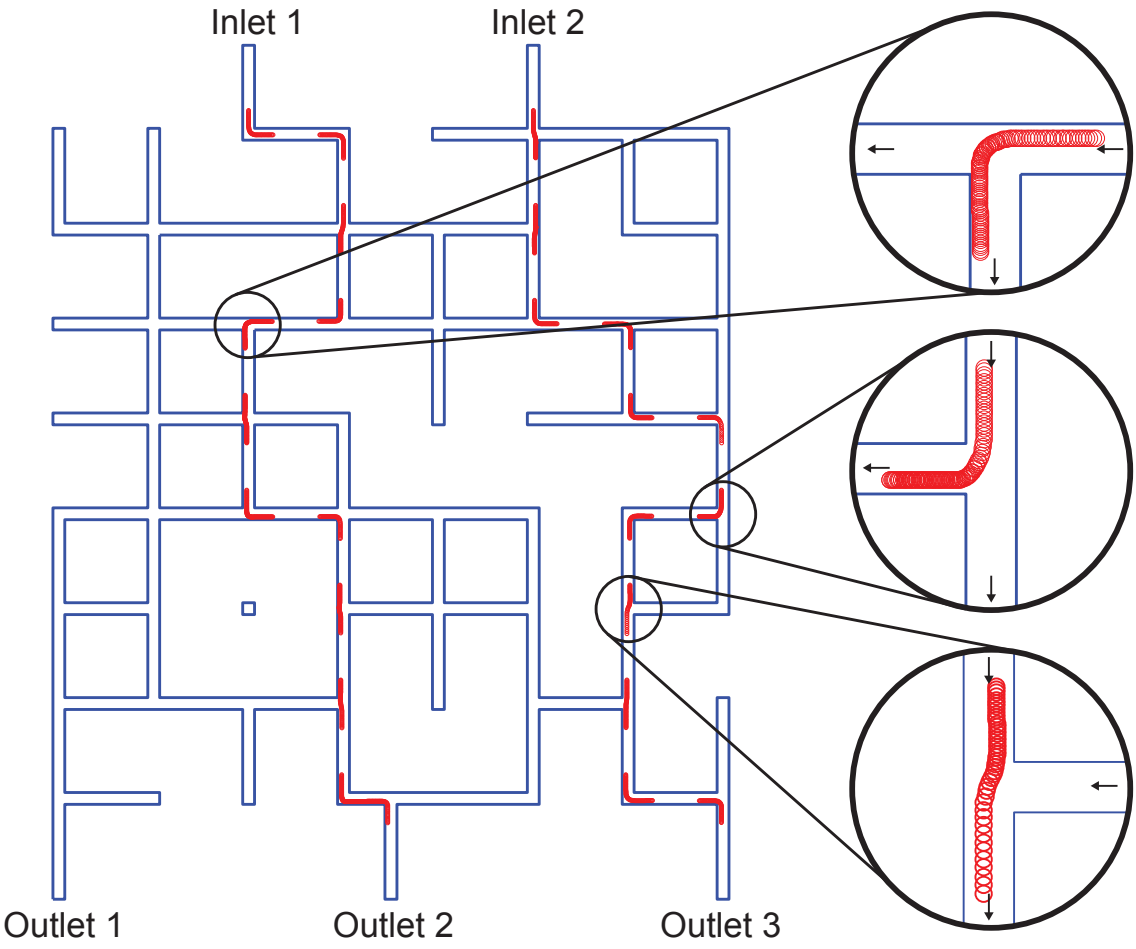

Figure 4.4: Results from using instantaneous simulation to predict the paths of two 1 μm particles traveling through a randomly-designed microfluidic chip [58], with close-ups of some channel intersections (black circles). The red lines/circles indicate the particle trajectories obtained from the database of pre-simulated intersections, and the gaps between the red lines are regions where no calculation of particle trajectory is necessary because the channels have no intersections in these regions. This simulation was completed in around one second by our instantaneous simulation method. In contrast, the same simulation failed after five days of computation when using an existing simulation tool (COMSOL Multiphysics). Raw data of these simulations are available for download (S1 File).

virtually identical to COMSOL's results: the exit locations of the three particles predicted by the two simulations differ by only 0.18, 1.21, and 0.11 μm in a 200 μm wide channel.

To demonstrate instantaneous microfluidic simulation on a more complex chip design, we used the method to predict the paths followed by particles on the large randomly designed microfluidic chip [58] shown in Fig 4.4. In this simulation, two 1 μm diameter particles start in the center of each inlet channel. The instantaneous simulation method was able to predict the paths followed by the particles in around 1 s of computational time. We attempted to replicate this simulation using COMSOL Multiphysics, but the simulation took *five days* just to generate the coordinates of each mesh node in this design; the subsequent calculation of the fluid velocity field failed, so we were unable to calculate particle trajectories using COMSOL alone.

4.5 Conclusions

We have presented a method to instantaneously simulate the behavior of microfluidic chips using common computing hardware. The efficiency of our method arises from two key innovations: leveraging existing electrical-fluidic analogies to efficiently compute fluid velocity wherever possible, and querying a database of pre-simulated intersections when necessary. By enabling researchers to simulate microfluidic chip designs in seconds instead of hours or days, our simulation method should accelerate the development of new microfluidic chips.

In its current form, our instantaneous simulation method has some limitations. For example, our current database of intersection simulations only supports intersections with up to four channels and assumes that all channels in the intersection have the same width. These limitations could be alleviated by creating a larger database that includes intersections with more than four channels and various channel widths. While generating this larger database would require significant computational resources, it would only need to be generated once, and using specialized computing hardware could accelerate the process. As the database of pre-simulated chip features grows, more and more microfluidic chips could be simulated in seconds using this technique.

Chapter 5

Automated design of microfluidic particle sorters

5.1 Introduction

Tools for sorting mixtures of particles play important roles in many different fields. In biological research and healthcare, cell sorters are routinely used to sort cells by their type. Fluorescence-activated cell sorters (FACS) are particularly powerful, but these sorters typically require a fluorescent tag or label that is specific to the cells of interest [68, 69]. Techniques that sort particles by their intrinsic properties—like size and density—are attractive because all particles have these properties, and a number of microfluidic sorters have been developed that sort cells and other particles by their physical properties [70, 71, 72, 24, 73]. However, designing a microfluidic sorter for a given task is currently a slow and labor-intensive process. Typically, researchers select an existing particle sorting principle (such as pinched flow fractionation [74] or deterministic lateral displacement [28]), tailor the design to suit their given application, and then fabricate and test the chip. If the chip does

not function as intended, the researchers must identify the problem and repeat the design, fabricate, and test process. This may continue through several iterations (and months of work) before a suitable sorter chip design is found (if one is found at all).

At the beginning of this work, we were motivated by the question, "is it possible to create a custom sorter chip for a given application without actually designing the chip?" Specifically, what if we had a library of *all possible microfluidic chip designs*? This hypothetical library would contain chips that perform previously-demonstrated sorting techniques, like pinched flow fractionation and deterministic lateral displacement. Perhaps more interestingly, the library might also contain *new* sorting techniques that have not been previously demonstrated. By searching through this library to find a chip that performs the desired sorting operations, a researcher with no experience in microfluidics could quickly and easily find a custom chip for their particular application.

Obviously, this hypothetical library of all possible microfluidic chip designs would be astronomically large. But by applying some constraints on the variety of possible chip designs, one can reduce the universe of "all possible chips" to a more manageable size. Recently, we generated a library of 10,513 different random chip designs, all of which are constrained within an 8×8 grid of fluid channels [75]. We then simulated the flow of fluids through these random chips and saved the results in database with a publicly-accessible website front-end (random.groverlab.org). Using this database, a user can specify three desired concentrations of a solute, and the website will select the random chips from the library that can generate solutions with those three concentrations. This work confirmed

that functional microfluidic chip designs for a given application can be found by mining a database of random chip designs, at least for one type of application (generating solutions with different concentrations). But can random chips do anything else?

In this work, we searched our library of 10,513 random microfluidic chip designs to find *chips that can sort particles by their size.* We initially doubted that we would find *any* functional sorters in the random library—only a handful of different techniques exist for sorting particles by their size, and sorter chips are always carefully designed by hand—so it seemed unlikely that a functional sorter could appear at random. But after simulating the paths followed by particles through all 10,513 random designs, we found 1,061 candidate designs that in principle could sort particles. We then fabricated and tested four of these designs and found that one of the four random chips can successfully sort particles based on their size. Thus, we showed that a library of random microfluidic chips that was created for one purpose can actually provide functional chip designs for a completely different purpose. This suggests that libraries of pre-simulated microfluidic chips can be used as general-purpose sources for designs for a wide variety of applications.

5.2 Materials and methods

This work was performed on an HP workstation with a E5-1650v2 CPU and 32GB RAM, using MATLAB 2014b [76] to control COMSOL Multiphysics 4.4 [77, 14] and MySQL 5.4 [67]. COMSOL Multiphysics used the Laminar Flow module to model the physics of fluid

flow in chips and a stationary solver to calculate the results. MOPSA [78] was implemented in MATLAB and used to predict particle trajectories.

After candidate particle sorter designs were found in our library of random chips, two modifications were made to the designs before fabricating the chips. First, all dead-end channels were deleted (these channels can trap bubbles or particles during use and have no effect on the velocity profile or particle trajectories of the chips, so it is safe to delete them; compare Figs. 5.2B and C for an example). Second, we added additional inlets and channels to flow focus beads to the center of the inlet channels (visible in Fig. 5.2C, Part 1) because in our particle trajectories simulations, all particles started at the centers of the inlet channels. The fluorescence micrograph of Node 1 in Fig. 5.4B shows the flow focusing channels in operation.

Conventional photolithography and wet etching were used to etch the randomly-designed chip designs into glass wafers to a depth of 80 μm and a width of 200 μm, and glass-glass thermal fusion bonding (668°C for 6 hours) was used to create finished microfluidic devices. 1 μm red fluorescent polystyrene beads (Magsphere Inc.) and 10 μm green fluorescent polystyrene beads (Magsphere Inc.) were used to test the performance of selected chips. We diluted 500 μL 1 μm and 10 μm beads in the 20 ml water and mixed with one droplet of TWEEN-20. The volumetric flow rate of four inlets was 0.05 ml/min and provided by two 2-channel syringe pumps (Harvard Apparatus Inc.). A fluorescence microscope (Eclipse Ti-S, Nikon, Japan) was used to image the trajectories of fluorescent beads. Fluorescent beads were also collected from three outlets and counted with hemocytometer.

5.3 Results

A graphical overview of our process for finding functional microfluidic chips in a library of random chips is shown in Fig. 5.1.

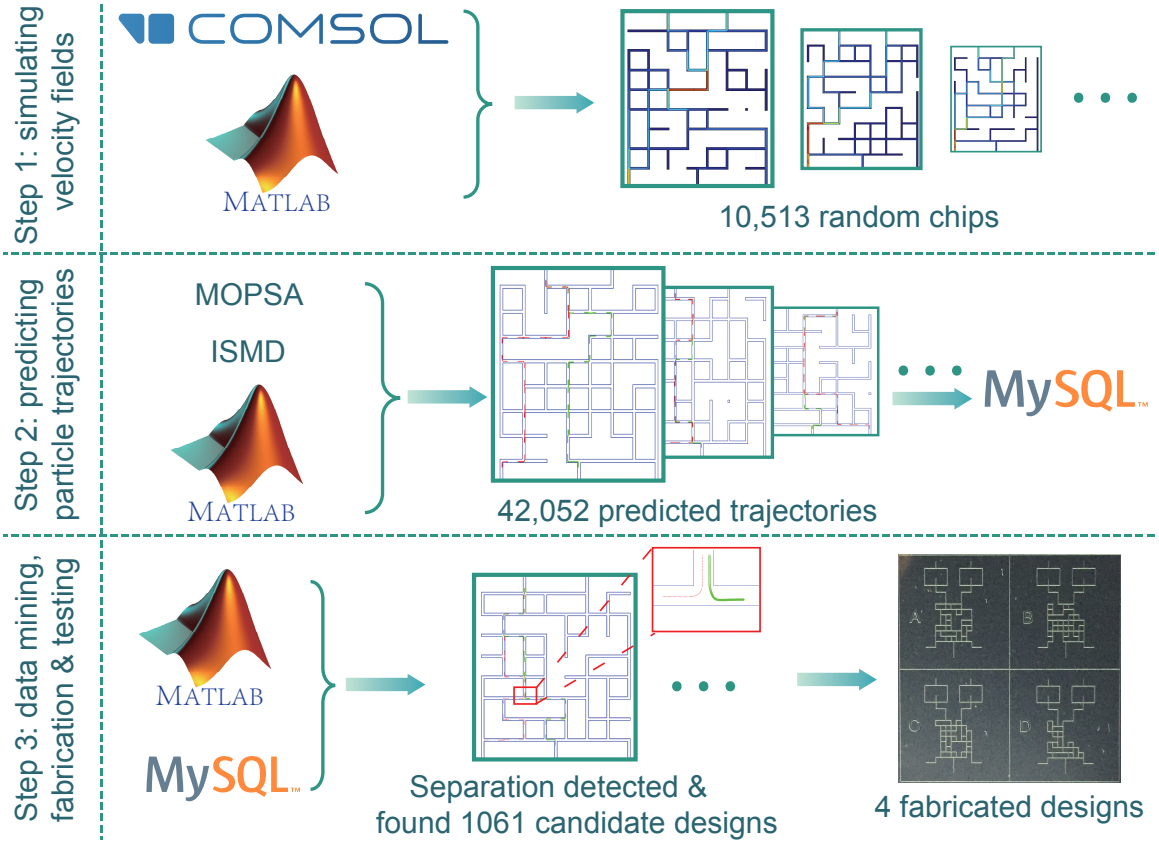

Figure 5.1: Overview of process for automated design of microfluidic particle sorters. In Step 1, MATLAB was used to generate 10,513 random microfluidic chip designs, and COMSOL Multiphysics was used to calculate the velocity fields of fluids inside each random chip. In Step 2, the microfluidics-optimized particle sorting algorithm (MOPSA; [78]) instantaneous simulation of microfluidic devices (ISMD; [79] were implemented in MATLAB to simulate the trajectories followed by 1 μm and 10 μm diameter particles through all 10,513 random chips. In total, 42,052 particle trajectories were predicted and saved into a MySQL database. In Step 3, MATLAB was used to mine the database for designs in which the different particle sizes exited through different outlets (that is, the designs that can sort particles). Of the resulting 1061 candidate particle sorters, 4 designs were fabricated in glass and tested experimentally.

5.3.1 Generating random microfluidic chip designs

In Step 1 of Fig. 5.1, we used a custom MATLAB program to generate a large number of random microfluidic chip designs. The resulting library of 10,513 chip designs is the same library we used to find chips for creating solutions with three user-specified solute concentrations [75].

We constrained our possible microfluidic chip designs to the rectilinear grid patterns shown in Fig. 5.2A. An 8×8 grid has 8^2 possible channel intersections and $2 \times 8^2 - 2 \times 8$ possible channels connecting those intersections. Each of these connecting channels can be either present or absent in a given design, which gives us 5,192,296,858,534,827,628,530,496, 329,220,096 possible different chip designs. Obviously, it is impossible for us to simulate all of the designs possible in the 8×8 grid, so we randomly selected 10,513 random chip designs as our candidate library.

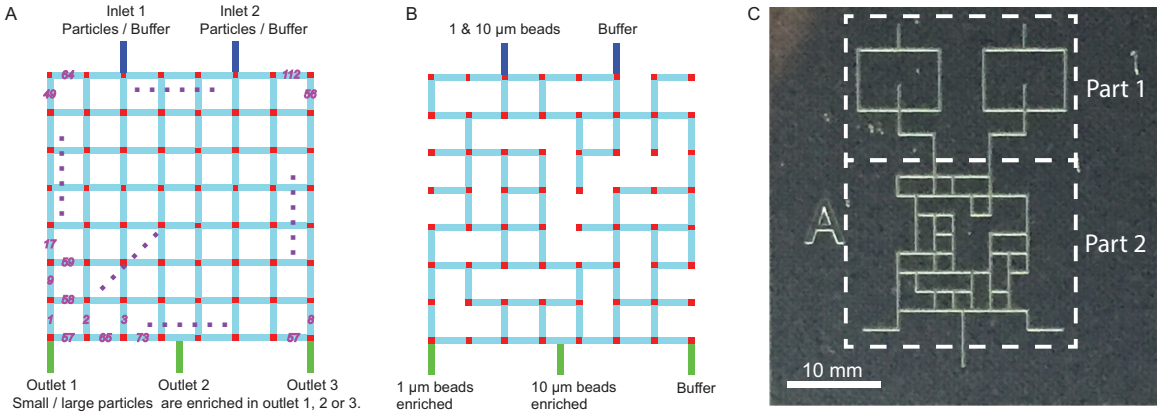

Figure 5.2: (A) The 8×8 grid constraint for our random microfluidic chips. Each light blue channel has a 90% chance of being present in any one chip design. (B) Design of a specific random chip ("Chip A") that our technique identified as a likely particle sorter. (C) Photograph of Chip A fabricated in glass using conventional photolithography and wet etching. The structures added in Part 1 focus the flow of beads to a streamline in the center of the inlet channel, and Part 2 is the random section where sorting occurs.

Each random chip design has two inlets and three outlets. In our previous work, the inlets received two solutions with different concentrations, and the outlets produced solutions with three user-specified concentrations [75]. In this work, we planned to inject a mixture of particles into Inlet 1 and buffer into Inlet 2 (or vice versa). If particles with different sizes exit the chip through different outlets, then that chip is deemed a functional sorter. For example, Fig.5.2B shows a random chip that is capable of sorting 1 μm and 10 μm particles based on simulation results. The particles enter the chip through Inlet 1, and 1 μm particles will be enriched in Outlet 1 and 10 μm particles will be enriched in Outlet 2. Fig.5.2C shows a photograph of this random chip fabricated in glass using conventional photolithography and wet etching (details in Methods below).

5.3.2 Simulating the behavior of the random chips

In Step 2 of Fig. 5.1, we simulated the behavior of all 10,513 random chips in our library. In our previous work with this library, we simulated only the flow of fluids and the transport of solutes in these chips [75]. In this work, since we are searching for effective particle sorters in the random chips, we had to (A) add lifelike particle sorting simulation results to our library, and (B) do so in a reasonable amount of computing time.

To add particle sorting simulation results to our library, one could use the built-in particle tracing module in COMSOL Multiphysics. However, using this module would have two major disadvantages. First, the predicted particle trajectories are unrealistic in most cases because the particle in COMSOL is a mathematical point without mass or volume,

which makes the particle fail to interact with channel walls. Since the particle separation principle of random chips might be pinched flow fractionation[74, 80, 81] and hydrodynamic filtration [31], interacting with channel walls is crucial for realistic simulation results, while COMSOL cannot provide. Second, COMSOL is extremely computationally expensive when solving particle trajectories. In our case, we need to find out the difference of trajectories between 1 μm and 10 μm particles, which means that our maximum mesh size should be smaller than 1 μm, which is the diameter of the smaller particle. If we configured 1 μm as the maximum mesh size in a random chip, it will take us more than 24 hours to solve the velocity filed only for a single random chip by using laminar flow module and about 2 hours to simulate the trajectories of particles by using particle tracing for fluid flow module.

We accomplished this using the *microfluidics-optimized particle simulation algorithm* (MOPSA) we developed previously [78]. Briefly, unlike the particle tracers built in to simulation tools like COMSOL Multiphysics, MOPSA does not assume that the particle is an infinitely-small point. Instead, MOPSA models the particle as a circle or sphere. MOPSA also applies an empirical correction when the particle touches or overlaps with a channel wall. We have previously used MOPSA to accurately reproduce the observed particle trajectories in several studies [78], so we are confident that it will accurately predict the behavior of particles flowing through our random chips.

To further accelerate the simulations, we leveraged the *instantaneous simulation of microfluidic devices* (ISMD) technique we developed previously [79]. ISMD simulates the behavior of microfluidic chips with the same accuracy as conventional techniques but in a

fraction of the time. To simulate a given random chip, ISMD first breaks the chip down into its component parts: channels and intersections. The flow in each channel is determined using analogies with electronic circuits or finite element analysis (FEA), and the flow in each intersection is determined by querying a database of nearly 100,000 pre-simulated channel intersections. By combining conventional FEA simulation with data mining of pre-simulated intersections, the trajectories of particles in a complete microfluidic device can be generated in about one second on an ordinary laptop computer and yields results that are indistinguishable from simulations that take hours or days to complete using conventional techniques.

Overall, by applying MOPSA and ISMD, the computational time required to simulate 42,052 different particle trajectories in 10,513 different random microfluidic chips was reduced from more than a year (about 70 min per chip when simulated by COMSOL Multiphysics alone) to only 2 weeks.

5.3.3 Analyzing the random chip library

In Step 3 of Fig. 5.1, we searched our library of simulation results to find chips that have the ability to separate 1 μm and 10 μm particles (that is, the chips where these particles exit through different Outlets). When a candidate sorter chip was found, our code examined the predicted particle trajectories to identify the channel intersection where the two particles' paths diverged.

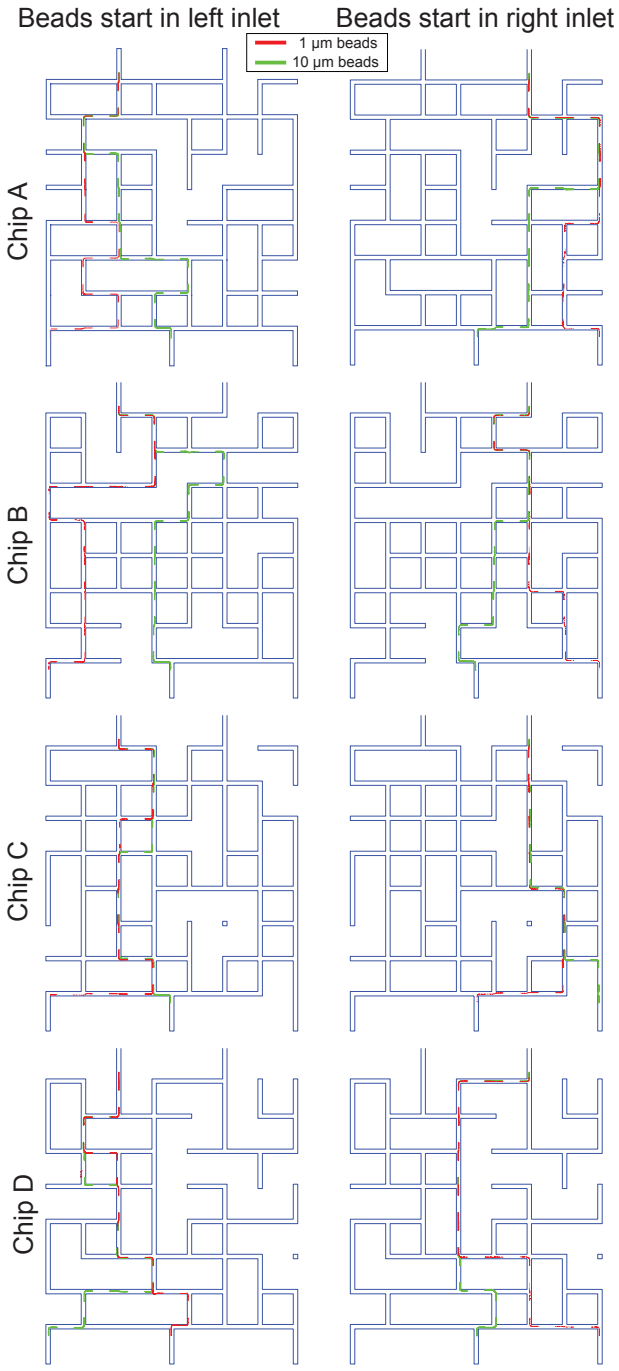

Figure 5.3: Predicted trajectories of 1 μm particles (red lines) and 10 μm particles (green lines) through random chips A, B, C, and D. While both particle sizes always start at the same inlet (either the left Inlet 1 or the right Inlet 2), they always exit through different outlets, indicating that these four random chip designs have the potential to be particle sorters.

We found that of the 10,513 random chips in our library, 485 designs (4.6%) could separate particle mixtures originating from Inlet 1, and 576 designs (5.5%) could separate particle mixtures originating from Inlet 2. Of the 485 + 576 = 1,061 random designs that can sort particles from *at least one* input, 12 of these designs can sort particles from *both* Inlet 1 *and* Inlet 2. From these 12 designs, we randomly selected four designs for fabrication and experimental testing.

The simulated trajectories of the four chosen candidates (labeled A–D) are shown in Fig. 5.3. Photographs of the microfabricated chips are shown in Fig. 5.1 (Chips A–D) and Fig. 5.2C (Chip A).

5.3.4 Experimentally confirming particle sorting

To determine whether the four chosen random chip designs can actually sort particles as predicted, we fabricated and tested these chips. A mixture of 1 μm diameter red fluorescent polystyrene beads and 10 μm green fluorescent polystyrene beads was pumped into one Inlet, and buffer without beads was pumped into the other Inlet. Fluorescence microscopy was used to visualize the paths followed by the beads in the chips, and a hemocytometer was used to count the beads of each size emerging from each Outlet of the chips. Additional details in *Methods*.

Of the four tested random chips, one design (Chip A) successfully sorted 1 μm and 10 μm particles to different Outlets. The other three designs (Chips B–D) yielded

identical mixtures of beads at each outlet (possible reasons why Chips B–D failed to sort are considered in the *Discussion* below).

Fig. 5.4 compares the predicted trajectories and experiment results of particles in Chip A. At Node 1, the simulation predicts that the 1 μm and 10 μm beads will be flow focused to the center of channel. The fluorescence micrograph of Node 1 confirms that this is the case. At Node 2, the simulation predicted that the 10 μm beads (green) would be offset to the right of the 1 μm beads (red). The fluorescence micrograph of Node 2 confirms that the 10 μm beads have been offset to the right of the red 1 μm beads (though the 1 μm beads seem to fill most of the channel and are not as well aligned as the 10 μm beads).

To quantify the separation performance of random Chip A, the sorting experiment shown in Fig. 5.4 was repeated three times and beads were collected from all three outlets and counted. Fig. 5.5A shows that, of the smaller 1 μm beads that exited the chip, an average of 68% exited through Outlet 1 and 32% exited through Outlet 2. Conversely, of the larger 10 μm beads that exited the chip, an average of 31% exited through Outlet 1 and 68% exited through Outlet 2. One-way ANOVA confirms that there is a statistically significant difference between the contents of Outlet 1 and Outlet 2 for both the 1 μm and 10 μm beads ($p < 0.001$).

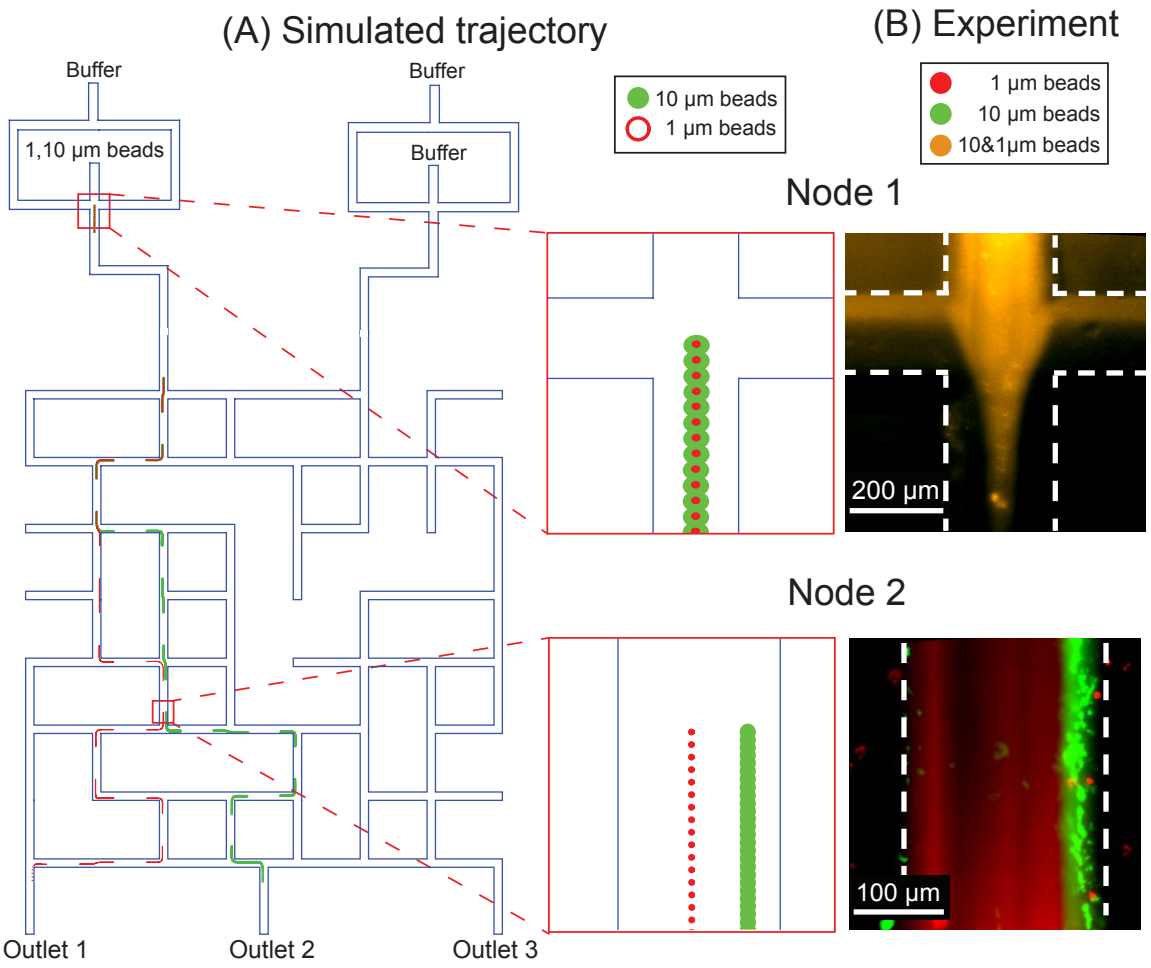

Figure 5.4: (A) Predicted trajectories of 1 μm particles (red) and 10 μm particles (green) through random Chip A, and (B) fluorescence micrographs showing the paths actually followed by 1 μm and 10 μm fluorescent particles at two critical nodes in the chip. At Node 1, the simulation predicts that both sizes of beads will be flow-focused to the center of the channel, and the experimental results confirm this (the yellow color indicates that both bead sizes are present). By the time the beads arrive at Node 2, the simulation predicts that the 10 μm green particles will be offset to the right of the 1 μm red particles, and the experimental results confirm that this separation by size has occurred. When the channel subsequently splits downstream of Node 2, most of the small red particles exit through Outlet 1, and most of the large green particles exit through Outlet 2.

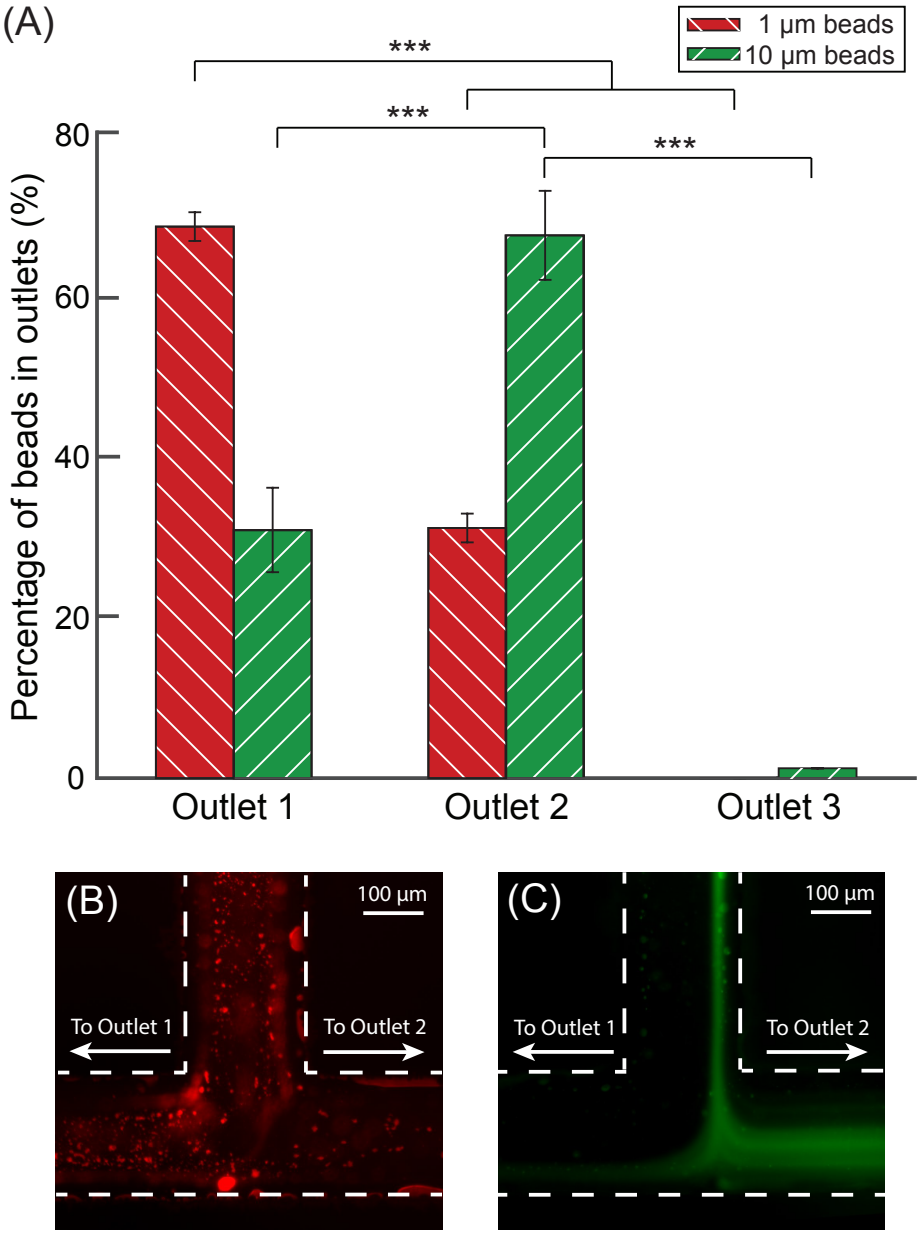

Figure 5.5: (A) Summary of results from three replicates of using Chip A to sort 1 μm (red) and 10 μm (green) beads. Error bars indicate the standard deviation and *** indicates $p < 0.001$. Fluorescence micrographs showing the trajectories of 1 μm (B; red) and 10 μm (C; green) beads, in the key sorting intersection (Node 2 in Fig. 5.4).

5.4 Discussion

Chip A in Fig. 5.4 is the first microfluidic particle sorter that was not designed for that purpose (or any other purpose, for that matter). Admittedly, Chip A's performance is inferior to most microfluidic sorters that were designed by hand to sort. Also, Chip A was the only one of the four random designs we tested that actually sorted as predicted. Why?

Perhaps the simplest answer is that *our technique is only as good as our simulations,* and if our simulations do not always agree with experimental observations, then we are more likely to find candidate chips that will not actually work as expected. For example, while our simulations accurately predicted the locations of the green 10 μm particles in Node 2 of Fig. 5.4, they did not predict the observed range of locations for the red 1 μm particles. This could be attributed to Brownian motion, which will widen the range of paths followed by the red 1 μm particles much more than the larger green 10 μm particles [82]. Since our simulations currently do not account for Brownian motion, our technique could not have foreseen this source of error (though it would be straightforward to incorporate Brownian motion in MOPSA and avoid these errors in the future).

Additionally, small differences between predicted and experimentally observed particle trajectories early in a chip could be magnified to large errors by the end of the chip. For example, the key intersection after which the different particle sizes follow different paths in Chip 1, Node 2 in Fig. 5.4, occurs after the particles have already passed through *seven* other intersections. Any errors in these seven intersections could make Node 2 not

function as predicted; this could be the reason why most (but *not quite all*) of the green 10 μm beads flow to the right as predicted downstream of Node 2 in Fig. 5.5C. To reduce the impact of errors like this, we might algorithmically prefer designs in which the key sorting intersection appears early in the chip and the paths of the sorted particles do not recombine afterwards.

We also observed that some intersections in the random chips are more prone to errors than other intersections. For example, the predicted path for the red 1 μm particle in Node 1 in Fig. 5.6 comes very close to the southeast corner of that intersection before following the channel to the south. A particle that enters Node 1 slightly to the east of the predicted path (perhaps because of Brownian motion) could easily exit to the east, and indeed we observed this behavior experimentally. Even though the two possible paths for the red 1 μm particle are expected to recombine in Node 2, having multiple unintended paths for particles only adds uncertainty to our technique. Preferring designs that avoid particle paths near intersection corners could reduce the impact of these errors.

Finally, we found that a few simulated intersections have nonsensical behavior due to the limited size of the intersection simulation database we use in our ISMD technique. For example, the predicted path for the red 1 μm particle in Node 3 in Fig. 5.6 contains a swerve to the south in a straight channel; in reality the particles follow a straight path through this section of channel (though happily, in this case the error did not affect which Outlet the particle used when leaving the chip) This error arose when our algorithm could

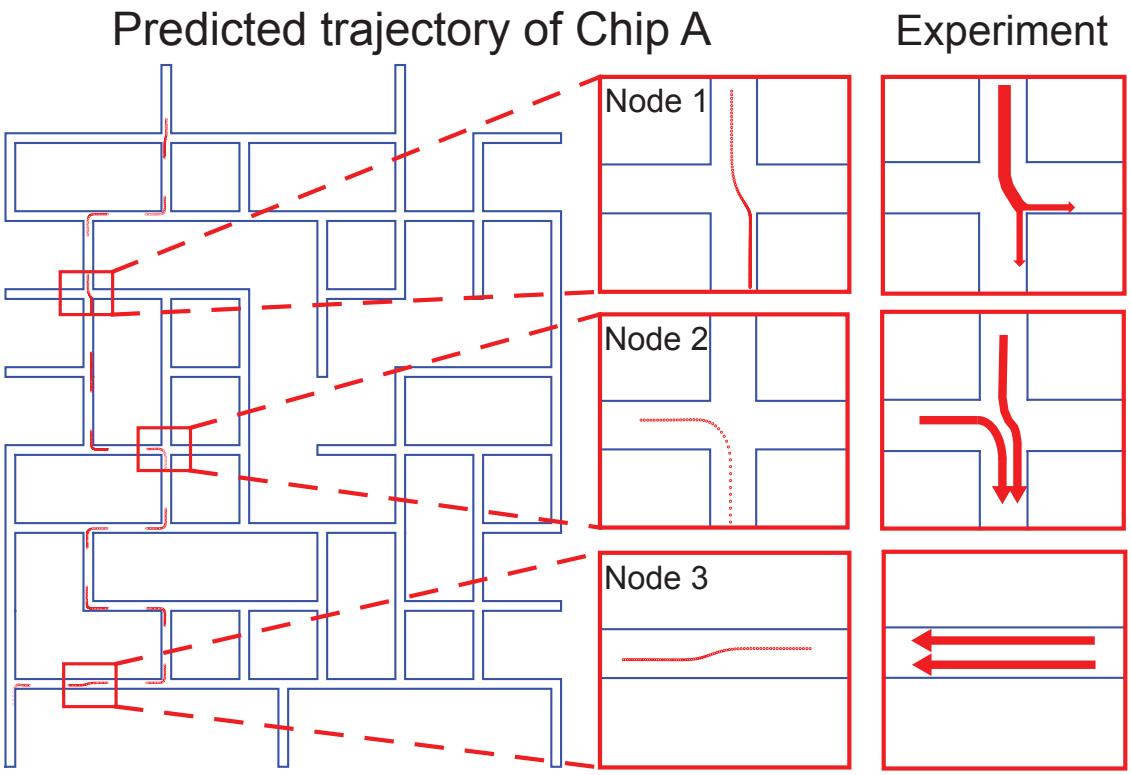

Figure 5.6: Comparisons between predicted and experimentally-observed trajectories of particles in three nodes of Chip A.

not find a pre-simulated intersection in our ISMD database that was similar enough to this section of channel. Having a larger number of different intersection simulations in our ISMD database will reduce the likelihood of these errors.

Looking at Chip A, we find ourselves in the unusual situation of having a working microfluidic chip and not knowing *how* it works. The chip may sort particles using aspects of pinched flow fractionation [74, 80, 81] or hydrodynamic filtration [31] because MOPSA can accurately simulate those physical phenomena. Additionally, Chip A shares some structural similarities with hydrodynamic filtration chips [31]. However, in hydrodynamic filtration, the size of the largest particle to be separated is usually on the same scale as the channel width. In contrast, in Chip A the channel width is *20 times larger* than the diameter of the largest separated particle. Inertial effects [83] could also be responsible for particle separation in Chip A, since the Reynolds number in some of the channels of random chips ($\sim$20) is large enough for inertial effects to affect the trajectories of particles. However, since MOPSA does not currently simulate inertial effects, it seems very unlikely that our technique would have selected a chip that depends on inertial effects to sort particles. Finally, it is possible that entirely new and undiscovered phenomena are responsible for the sorting abilities of Chip A. New and useful microfluidic phenomena are discovered regularly (*e.g.*, [84]), and as long as our simulation tools can accurately simulate a phenomenon, it seems reasonable that our approach can discover that phenomenon. For example, assume that Chip A functions by pinched flow fractionation. If pinched flow fractionation had never been discovered by Yamada *et al.* in 2004 [74], our software would have discovered it in this

study. It appears to be only a matter of time before automated approaches like ours find useful microfluidic phenomena that would have never been discovered by humans on their own.

Chapter 6

Automated and rational design of a microfluidic mixer

6.1 Introduction

Mixing is one of fundamental functions in microfluidic chips. For the past decade, different microfluidic mixers have been designed [85]. Microfluidic mixers are usually categorized as either "active" (an external energy force or an external physical field is present to accelerate mixing phenomenon) or "passive" (mixing is accomplished only by diffusion and is dependent only on the area of contact between the two fluids and the amount of time the fluids are in contact). Active mixers generally outperform passive mixers, but integrating an external force or field in the chip adds unwanted complexity and cost to the chip. Passive mixers are simpler and more economical, but increasing the area and time of contact between the two fluids has undesirable consequences: increasing contact area by lengthening the channel containing the two fluids adds unwanted additional fluidic resistance to the channel, and increasing contact time by slowing the flow rate decreases the

overall throughput of the microfluidic chip. Thus, there is a need for mixer designs that combine high mixing performance with low fluidic resistance and high flow rates.

Several studies have been conducted on the optimization of microfluidic mixers. Li *et al.* optimized a chaotic microfluidic mixer using lattice Boltzmann method [86]. Three continuous studies from Wang *et al.* were focused on the optimization of layout of obstacles for enhanced mixing in microchannels using a fluid dynamics software for different applications [87, 88, 89]. Hossain *et al.* conducted a research of optimizing a modified Tesla structure based on topology optimization [90]. Finally, Cortes-Quiroz *et al.* optimized a grooved microfluidic mixer using a multi-objective optimization approach [91]. However, their approaches either only optimized one parameters or the optimized designs were too complicated and impossible to fabricated.

Obviously, different microfluidic mixers will have different performance. In this work, we set out to answer the question, *is it possible to find the most optimized mixer in certain conditions?* Specifically, are we able to explore the limitation of how good a microfluidic mixer can possibly be within a certain limit on fluidic resistance?

In this work we developed an approach to automatically design and optimize ideal microfluidic mixers for specific conditions. We accomplished this in two steps. First, we generated a library of 6068 different random mixer designs and simulated the performance of each of them. We have previously used this technique to generate designs of functional microfluidic chips that can deliver solutes of any desired concentrations [92], so adapting it to generating random mixer designs was relatively straightforward. Second, we used the

Non-dominated Sorting Genetic Algorithm II (NSGA-II) [93] to optimize multiple design parameters of our microfluidic mixer at the same time. After 200 generations of evolution, the mixer designs converged near the true Pareto-optimal front; this allows us to explore the fundamental performance limits of a microfluidic mixer. A user can select a design from these optimized mixers and have confidence that the design is optimal for a given fluidic resistance. Additionally, we identified certain design trends in the optimized mixers, hand-designed several mixers that incorporate these trends, and compared the performance of our hand-designed mixers to that of our automatically-designed optimal designs. Finally, we also hand-designed versions of popular microfluidic mixers from the literature and compared their performance to that of our optimal designs. In each case, our automatically-designed and optimized mixers equaled or exceeded the mixing performance of hand-designed mixers.

6.2 Materials and Methods

6.2.1 Generating initial random mixer designs

We created our first generation of microfluidic mixers by generating mixer designs at random [92]. Of course, there is an essentially limitless variety of possible mixer designs, so we applied certain constraints to our random designs. Figure 6.1A shows the basic design template of our random mixers. Each mixer has two inlets, two outlets, and a 500 μm x 500 μm design domain where the random mixing structures are located. In the design domain are ten cylindrical posts with random sizes and locations. Two or more cylinder posts can

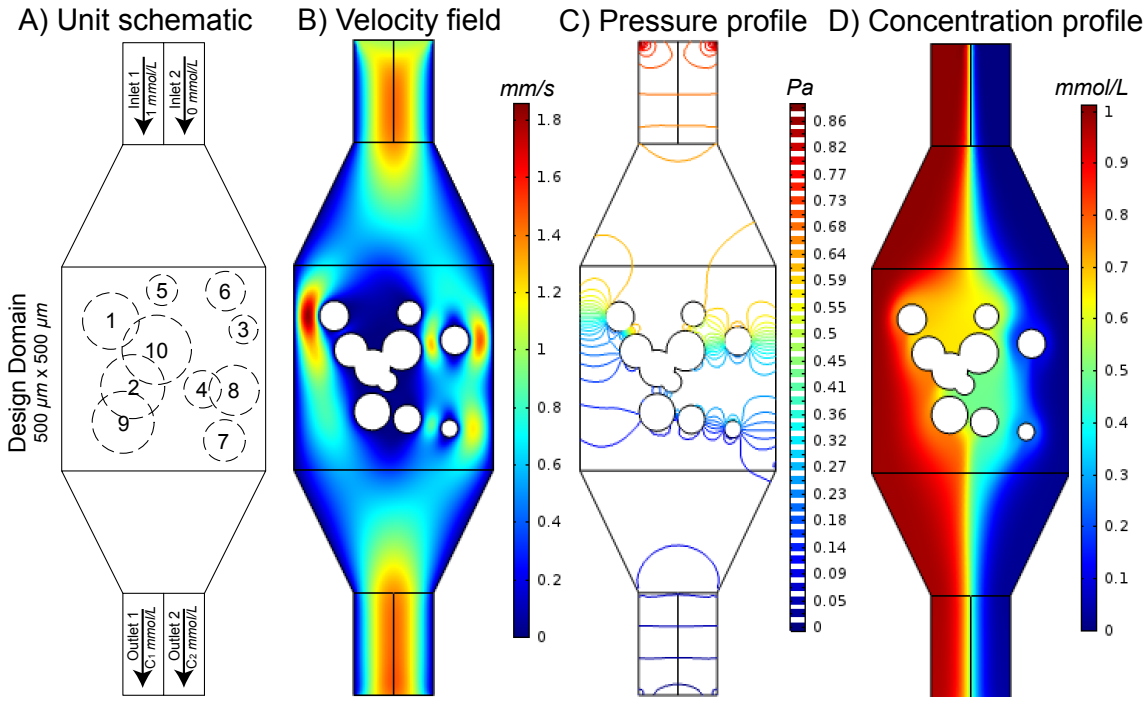

Figure 6.1: (A) Schematic of a simulated microfluidic mixer unit. A simulated unit has two inlets and two outlets. Between inlets and outlets is a 500 μm x 500 μm design domain. In the design domain, each mixer has ten cylindrical posts with random sizes and locations. Different cylinder posts were allowed to overlap to create additional structures like walls. (B) The predicted velocity field of a typical mixer unit. This velocity field is used for simulating the solute concentration profile in the mixer. (C) The predicted pressure profile of the mixer unit. This pressure profile is used to calculated the fluidic resistance of the mixer. (D) The predicted solute concentration profile of the mixer unit. This concentration profile is used to determine the mixing performance of this mixer unit.

overlap, which enables the mixer designs to include non-circular features as well (like walls). In total, 6096 different mixer designs were generated and stored to a database.

In addition to randomly-generated designs, we also manually designed five mixer units based on our experience so as to compare them with the randomly generated designs as well as NSGA-II designs.

6.2.2 Simulating mixer performance

All simulations were performed using the finite element analysis software COMSOL Multiphysics (COMSOL Inc., Burlington, MA). We used the software's MATLAB API to automate this process and performed all simulations. The Laminar Flow physics module and Transport of Dilute Species physics module as well as two stationary solvers were used in COMSOL Multiphysics. In the Laminar Flow physics module in COMSOL Multiphysics, each inlet was assigned an inlet boundary condition of 1 mm/s normal inflow velocity, and each outlet was assigned an outlet boundary condition of 0 Pa pressure. The remaining boundaries were walls (no-slip boundary condition), and the material filling the channels was water under incompressible flow. In the Transport of Dilute Species physics module, inlet 1 is assigned an inflow concentration of 1 mmol/L and inlet 2 is assigned an inflow concentration of 0 mmol/L. The two outlets were assigned as outflows. The solute diffusion coefficient used in simulation is 4.25×10^{-10} m^2/s. Figures 6.1B and C show the calculated velocity field and pressure field of one mixer unit design, and Figure 6.1D shows the concentration mixing field of the same design.

6.2.3 Evolving mixer designs and finding the Pareto-optimal front

The genetic algorithm NGSGA-II [93] was used to evolve optimized versions of our random mixers. A flow chart representation of our custom NSGA-II implementation is shown in Figure 6.2. The fitness function for fluidic resistance ($S_{\text{resistance}}$) is

$$S_{\text{resistance}} = P_2 - P_1 \tag{6.1}$$

where P_2 is the pressure at the outlets and P_1 is the pressure at the inlets. This means that the lower the pressure drop across the mixer, the better the performance of the mixture. The fitness function for mixing performance ($S_{\text{concentration}}$) is

$$S_{\text{concentration}} = (1 - C_1) + (C_2 - 0) \tag{6.2}$$

where C_1 is the average concentration of Outlet 1; $1 - C_1$ calculates the mixing performance between Inlet 1 and Outlet 1; C_2 is the average concentration of Outlet 2; $C_2 - 0$ calculates the mixing performance between Inlet 2 and Outlet 2. This means that the closer the concentrations of the fluids in Outlet 1 and Outlet 2, the better performance of the mixer.

The total population for each generation was 200 due to the limitation of computational expense. The first generation starts with a randomly selected design. After the non-dominant sorting operator, the typical selection, crossover, and mutation operators were conducted so as to generate the new population. The parameters in the numerical

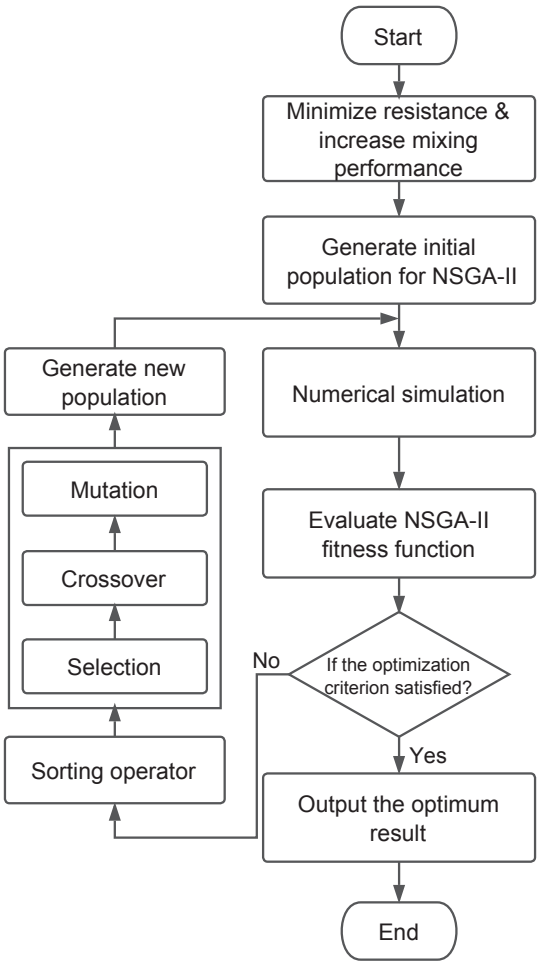

Figure 6.2: A flow chart depicting our custom NSGA-II process for optimizing mixers and finding the Pareto-optimal front. The overall goal was to minimize the pressure drop or fluidic resistance of mixer while increasing the mixing performance. Numerical simulation was conducted by COMSOL Multiphysics and MATLAB. Typical genetic operators were conducted after a non-dominant sorting operator.

simulation section were the same as the randomly generated designs in above. In total, 200 generations were calculated to find the Pareto-optimal front.

6.3 Results and discussions

Figure 6.3A plots pressure drop versus concentration score for each of the randomly-generated designs (small blue circles), NSGA-II-evolved optimal designs (red stars), and human-created designs (yellow stars). The NSGA-II designs distribute at the boundary of the randomly generated designs. This means that NSGA-II successfully found the Pareto-optimal front. To achieve a similar concentration score, NSGA-II designs always need less pressure drop or generate less resistance of a mixer unit. In another words, within a certain pressure drop condition, the NSGA-II designs will always have better mixing performance than the random-design mixers.

Figure 6.3B-D are three common microfluidic mixer designs with being constrained in the design domain and using cylinder posts to map the geometry. We also plot their mixing performance in Figure 6.3A and we can clearly see that the yellow stars of design B, C and D are all above the red stars. This tells us that the common mixer designs for microfluidics still have some room to be optimized.

Figure 6.3 profiles G0–G200 are the concentration profiles of NSGA-II designs in generations 0, 25, 50, 75, 100, 125, 150, 175 and 200. As the generation number increases, the mixing performance improves and the geometry converges into a S-shape. The S-shape suggests that NSGA-II selects S-shape designs as elite designs and retains the S-shape

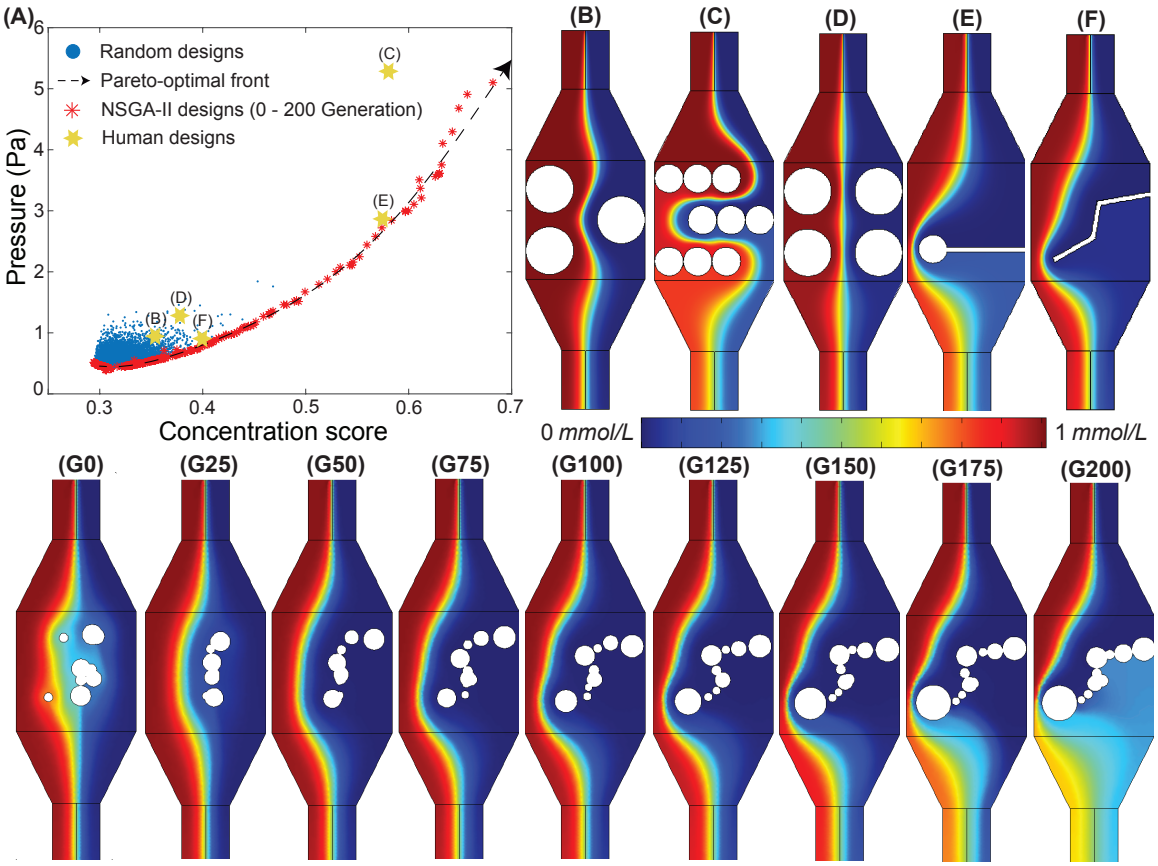

Figure 6.3: (A) Pressure drop vs. concentration for random microfluidic mixer designs (blue dots) and NSGA-II-evolved mixer designs (red stars). The random designs are distributed in the bottom-left corner while the NSGA-II designs are at the boundary of all random designs. By connecting all the NSGA-II designs, we can draw a Pareto-optimal front. Human-generated designs B–F (yellow stars) are above the Pareto-optimal front, which means that their mixing performance is not as good as NSGA-II designs with a certain pressure drop. (B) Human design B has a pressure drop 0.98 Pa and a mixing score of 0.35. (C) Human design C has a pressure drop of 5.36 Pa and a mixing score of 0.59. (D) Human design D has a pressure drop of 1.29 Pa and a mixing score of 0.38. (E) Human design E has a pressure drop of 2.87 Pa and a mixing score is 0.59. (F) Human design F has a pressure drop of 0.81 Pa and a score of 0.40. (G0 — G200) The concentration profiles of NSGA-II designs of generation 0, 25, 50, 75, 100, 125, 150, 175 and 200. Additional concentration profiles, pressure profiles, and velocity fields of 0–200 generations are available in Supporting Information.

feature into next generations. The S-shape could increase the mixing contact area as well as minimizing the fluid resistance. The small gaps between each post also appear to be crucial to the performance of the mixer. From the concentration and pressure profiles of each generation (see Supporting Information), we know that each small gap allows fluid with no chance to mix (around 0 mmol/L) to go through the S-shape and reduce the fluid resistance. We are unaware of any similar designs that have been created by human designers. Figures 6.3E and F are inspired by NSGA-II designs. Their performance is close to the Pareto-optimal front but they do not have small gaps in dark blue area (around 0 mmol/L) to reduce the fluid resistance.

Since we only have 200 populations in each generation while the size and position of cylinder posts in design domain are infinite, the Pareto-optimal front we found is only close to the real Pareto-optimal front. Figure 6.4 shows the comparison of NGSA-II designs between two separate evolution runs. Figure 6.4A is the first run and the results in Figure 6.3 are from this run. Figure 6.4B is the second run, and in this run we found that the geometry converged into a Y-shape instead of a S-shape. While the concentration scores of these two separate evolution runs are similar (around 0.68), the design from first run has a lower pressure drop. This indicates that the Pareto-optimal front found from first run is closer to the real Pareto-optimal front. Although the geometries resulting from the two evolution runs are different, they do share two important similarities. First, they both created a narrow gap near the left edge with a large cylinder post. Secondly, they used

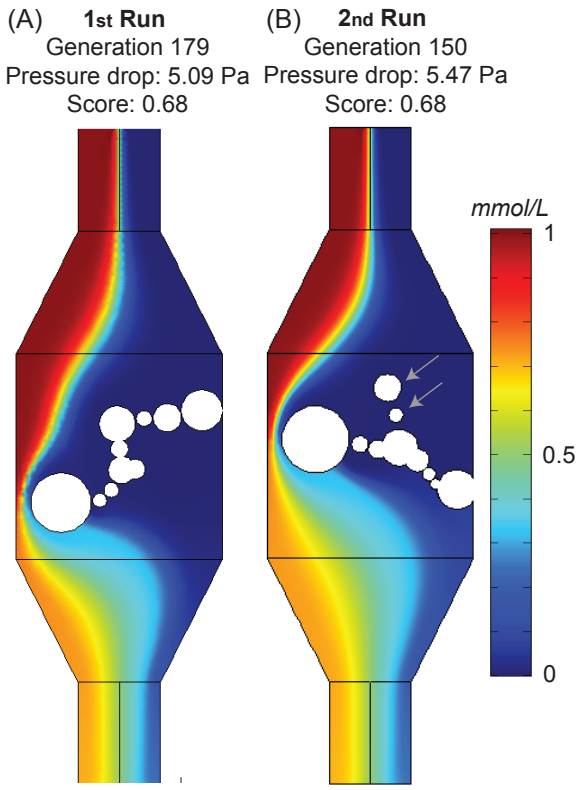

Figure 6.4: (A) The NSGA-II design selected at the end of the first run of evolution. (B) The NSGA-II design selected at the end of a second run of evolution. In the second run, the geometry converged into a Y-shape. To achieve a similar concentration score as the S-shape design from run 1, the Y-shape design from run 2 will have a higher fluidic resistance.

the rest of the cylinder posts to generate a wall containing small gaps in dark blue area (around 0 mmol/L) so as to minimize the pressure drop. So, why does the design from the first evolution run have a lower pressure drop? From the concentration profiles, we can see that in the first run design, fluid had a longer contact time and contact area before entering the critical gap (generated by the largest post). Additionally, it seems that the second-evolution designs only used eight posts to create a wall instead of ten. Two upper cylinder posts (pointed by gray arrows) seem to have no function and increase the fluidic resistance of this design.

6.4 Conclusions

We demonstrated how to optimize a functional microfluidic mixer for two parameters, pressure drop and mixing performance, using NSGA-II. We accomplished this by using MATLAB and COMSOL Multiphysics as our simulation platform and implementing NSGA-II in MATLAB. We also first found the pressure drop versus concentration score Pareto-optimal front. After that, we compared the designs at the Pareto-optimal front with human-created designs and random designs. Our simulations indicate that designs from NSGA-II have lower pressure drops while achieving a similar mixing performance (concentration score) than designs by humans or random designs. Based on the NSGA-II designs, we have better understanding about how to design a microfluidic mixer rationally: a mixer should have a constriction to increase contact area and contact time between the fluids, as

well as some features that are not for mixing but rather for reducing the overall resistance of the mixer.

Our approach is not limited to optimizing only mixer performance—it should be able to optimize additional parameters as well. For instance, the size is also crucial to a microfluidic chip. Instead of constraining a mixer into a fixed design domain, we could try to minimize the size of design domain as well. Additionally, microfluidic mixers are just one one of many components in microfluidic chips [5, 94]. We are confident that our approach can be applied to other applications of microfluidics. For example, cell sorting is a major application in microfluidics [24]. In different sorting technologies, inertial microfluidics shows the potential to efficiently separate different cells based on the sizes of cells [83, 95, 96, 97]. However, inertial microfluidics usually are operated at a high Reynolds number, whose shear stress could be harmful for the target cells [98]. In this case, coupled with our previous work (MOPSA, microfluidics-optimized particle simulation algorithm [99]), NSGA-II could be used to optimize a inertial microfluidic chip so as to increase the separation performance while minimizing the damage to cells from shear stress.

Chapter 7

Conclusions

In this work, we introduced several methods on designing microfluidic chips with the help of computational tools. These methods are:

- *The random design algorithm*: with the help of random design algorithm and simulation platform (COMSOL and MATLAB), we are able to create thousands of microfluidic solute generators without actually designing them.
- *MOPSA, a microfluidics-optimized particle simulation algorithm*: allowing simulating the trajectories of different types of particles in greater accuracy than commercial software.
- *ISMD, instantaneous simulation of microfluidics*: allowing instantaneous simulation of fluids and particles in complex microfluidic devices.
- *A randomly designed particle sorter*: by taking advantages of random design algorithm, MOPSA and ISMD, we automated designed a microfluidic particle sorter which is able to separate 1 μm and 10 μm particles.

- *An optimized microfluidic mixer by NSGA-II*: NSGA-II is a multi-object genetic algorithm which allows us to optimize multiple parameters of a microfluidic mixer. With the help of NSGA-II, the Pareto-optimal front was found and the designs from NSGA-II have better performance than designs from human being or rational designs.

In future, there will be more and more commercial microfluidic products in both clinics and point of care testing market instead of being constrained in laboratories. To achieve that, computer-assisted design will play an important role in this journey. The algorithms and tools used in this thesis are fundamental in computer science. We are confident that more microfluidics optimized algorithms will emerge in future. Microfluidics and computer science will become a new promising field that help human beings live a better life.

The obstacles are obvious as well. Until now, simulation is still too complicated even for researchers. In future, we should make encourage researchers in not only related field but also unrelated fields to try microfluidics in simulation first and then know the beauty of microfluidics. After that, researchers in other fields can take advantages of microfluidics in their own fields. For example, we could build an Online version of MOPSA so that researchers who are interested in MOPSA could deploy their simulation in cloud and get simulated results in seconds. **Random.groverlab.org** is a good demonstration of how an easy simulation platform could help researchers. Even a year after *Random design of microfluidics* was published, we still have visitors coming to visit the website and play

with random chips every day on average. The accelerating algorithm described in Chapter 4 could also possibly be deployed in the cloud and facilitate the cloud simulation platform.

Cell sorting is one of the important applications in biological research. **Randomly designed particle sorter** presented in Chapter 5 demonstrated that after we utilized multiple algorithms, we were able to design complex microfluidic chip for sorting purpose. Before that, every cell sorter was designed by human being with limited computational aid. In chapter 6, we presented an optimizing process using the genetic algorithm NSGA-II. Both resistance and mixing performance of a microfluidic mixer are investigated and optimized simultaneously. We are confident that **Random particle sorter** and **NSGA optimized mixer** could be a good start for using computer algorithms to design better microfluidic chips.

Bibliography

[1] Stephen C Terry, John H Jerman, and James B Angell. A gas chromatographic air analyzer fabricated on a silicon wafer. *IEEE Transactions on Electron Devices*, 26(12):1880–1886, 1979.

[2] William H Grover, Andrea K Bryan, Monica Diez-Silva, Subra Suresh, John M Higgins, and Scott R Manalis. Measuring single-cell density. *Proceedings of the National Academy of Sciences*, 108(27):10992–10996, 2011.

[3] Chunsun Zhang, Jinliang Xu, Wenli Ma, and Wenling Zheng. PCR microfluidic devices for DNA amplification. *Biotechnology advances*, 24(3):243–284, 2006.

[4] Ali Asgar S Bhagat, Hansen Bow, Han Wei Hou, Swee Jin Tan, Jongyoon Han, and Chwee Teck Lim. Microfluidics for cell separation. *Medical & biological engineering & computing*, 48(10):999–1014, 2010.

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

[6] Jeffrey McDaniel, Brian Crites, Philip Brisk, and William H Grover. Flow-layer physical design for microchips based on monolithic membrane valves. *IEEE Design and Test*, 32(6):51–59, 2015.

[7] Hailong Yao, Qin Wang, Yizhong Ru, Yici Cai, and Tsung-Yi Ho. Integrated flow-control codesign methodology for flow-based microfluidic biochips. *IEEE Design and Test*, 32(6):60–68, 2015.

[8] James Friend and Leslie Y Yeo. Microscale acoustofluidics: Microfluidics driven via acoustics and ultrasonics. *Reviews of Modern Physics*, 83(2):647, 2011.

[9] C Monat, P Domachuk, and BJ Eggleton. Integrated optofluidics: A new river of light. *Nature Photonics*, 1(2):106–114, 2007.

[10] Nicole Pamme. Magnetism and microfluidics. *Lab on a Chip*, 6(1):24–38, 2006.

[11] Nazila Norouzi, Heran C Bhakta, and William H Grover. Orientation-based control of microfluidics. *PLoS One*, 11(3):e0149259, 2016.

[12] Paul Yager, Thayne Edwards, Elain Fu, Kristen Helton, Kjell Nelson, Milton R Tam, and Bernhard H Weigl. Microfluidic diagnostic technologies for global public health. *Nature*, 442(7101):412–418, 2006.

[13] Alison M Skelley, James R Scherer, Andrew D Aubrey, William H Grover, Robin HC Ivester, Pascale Ehrenfreund, Frank J Grunthaner, Jeffrey L Bada, and Richard A Mathies. Development and evaluation of a microdevice for amino acid biomarker detection and analysis on Mars. *Proceedings of the National Academy of Sciences of the United States of America*, 102(4):1041–1046, 2005.

[14] LiveLink for Matlab. `https://www.comsol.com/ livelink-for-matlab`. Accessed: 2016-03-23.

[15] ASCII DXF Files. `http://www.autodesk.com/ techpubs/autocad/acad2000/dxf/ ascii_dxf_files_dxf_aa.htm`. Accessed: 2016-09-26.

[16] Matthew A Holden, Saurabh Kumar, Edward T Castellana, Ali Beskok, and Paul S Cremer. Generating fixed concentration arrays in a microfluidic device. *Sensors and Actuators B: Chemical*, 92(1):199–207, 2003.

[17] Luciano Galdieri, Swati Mehrotra, Sean Yu, and Ales Vancura. Transcriptional regulation in yeast during diauxic shift and stationary phase. *OMICS*, 14(6):629–38, Dec 2010.

[18] Kangsun Lee, Choong Kim, Byungwook Ahn, Rajagopal Panchapakesan, Anthony R Full, Ledum Nordee, Ji Yoon Kang, and Kwang W Oh. Generalized serial dilution module for monotonic and arbitrary microfluidic gradient generators. *Lab on a Chip*, 9(5):709–717, 2009.

[19] Daniel Irimia, Dan A Geba, and Mehmet Toner. Universal microfluidic gradient generator. *Analytical Chemistry*, 78(10):3472–3477, 2006.

[20] Noo Li Jeon, Stephan KW Dertinger, Daniel T Chiu, Insung S Choi, Abraham D Stroock, and George M Whitesides. Generation of solution and surface gradients using microfluidic systems. *Langmuir*, 16(22):8311–8316, 2000.

[21] Frederick K Balagaddé, Lingchong You, Carl L Hansen, Frances H Arnold, and Stephen R Quake. Long-term monitoring of bacteria undergoing programmed population control in a microchemostat. *Science*, 309(5731):137–140, 2005.

[22] Brian M Paegel, William H Grover, Alison M Skelley, Richard A Mathies, and Gerald F Joyce. Microfluidic serial dilution circuit. *Analytical Chemistry*, 78(21):7522–7527, 2006.

[23] Gerald J Kost. *Principles & Practice of Point-of-care Testing*. Lippincott Williams & Wilkins, 2002.

[24] C Wyatt Shields IV, Catherine D Reyes, and Gabriel P López. Microfluidic cell sorting: a review of the advances in the separation of cells from debulking to rare cell isolation. *Lab on a Chip*, 15(5):1230–1249, 2015.

[25] Martin Wörner. Numerical modeling of multiphase flows in microfluidics and micro process engineering: a review of methods and applications. *Microfluidics and Nanofluidics*, 12(6):841–886, 2012.

[26] Matthew K Runyon, Christian J Kastrup, Bethany L Johnson-Kerner, Thuong G Van Ha, and Rustem F Ismagilov. Effects of shear rate on propagation of blood clotting determined using microfluidics and numerical simulations. *Journal of the American Chemical Society*, 130(11):3458–3464, 2008.

[27] Jonathan Siegrist, Mary Amasia, Navdeep Singh, Debjyoti Banerjee, and Marc Madou. Numerical modeling and experimental validation of uniform microchamber filling in centrifugal microfluidics. *Lab on a Chip*, 10(7):876–886, 2010.

[28] Lotien Richard Huang, Edward C Cox, Robert H Austin, and James C Sturm. Continuous particle separation through deterministic lateral displacement. *Science*, 304(5673):987–990, 2004.

[29] Kevin Loutherback, Kevin S Chou, Jonathan Newman, Jason Puchalla, Robert H Austin, and James C Sturm. Improved performance of deterministic lateral displacement arrays with triangular posts. *Microfluidics and Nanofluidics*, 9(6):1143–1149, 2010.

[30] David W Inglis, John A Davis, Robert H Austin, and James C Sturm. Critical particle size for fractionation by deterministic lateral displacement. *Lab on a Chip*, 6(5):655–658, 2006.

[31] Masumi Yamada, Kyoko Kano, Yukiko Tsuda, Jun Kobayashi, Masayuki Yamato, Minoru Seki, and Teruo Okano. Microfluidic devices for size-dependent separation of liver cells. *Biomedical Microdevices*, 9(5):637–645, 2007.

[32] Anne Y Fu, Charles Spence, Axel Scherer, Frances H Arnold, and Stephen R Quake. A microfabricated fluorescence-activated cell sorter. *Nature Biotechnology*, 17(11):1109–1111, 1999.

[33] Xiaolin Wang, Shuxun Chen, Marco Kong, Zuankai Wang, Kevin D Costa, Ronald A Li, and Dong Sun. Enhanced cell sorting and manipulation with combined optical tweezer and microfluidic chip technologies. *Lab on a Chip*, 11(21):3656–3662, 2011.

[34] Hong Miao Ji, Victor Samper, Yu Chen, Chew Kiat Heng, Tit Meng Lim, and Levent Yobas. Silicon-based microfilters for whole blood cell separation. *Biomedical Microdevices*, 10(2):251–257, 2008.

[35] Nan Li, Daniel T Kamei, and Chih-Ming Ho. On-chip continuous blood cell subtype separation by deterministic lateral displacement. In *2007 2nd IEEE International Conference on Nano/Micro Engineered and Molecular Systems*, pages 932–936. IEEE, 2007.

[36] Haakan N Joensson, Mathias Uhlén, and Helene Andersson Svahn. Droplet size based separation by deterministic lateral displacementseparating droplets by cell-induced shrinking. *Lab on a Chip*, 11(7):1305–1310, 2011.

[37] Sunitha Nagrath, Lecia V. Sequist, Shyamala Maheswaran, Daphne W. Bell, Daniel Irimia, Lindsey Ulkus, Matthew R. Smith, Eunice L. Kwak, Subba Digumarthy, Alona Muzikansky, Paula Ryan, Ulysses J. Balis, Ronald G. Tompkins, Daniel A. Haber, and Mehmet Toner. Isolation of rare circulating tumour cells in cancer patients by microchip technology. *Nature*, 450(7173):1235–1239, 12 2007.

[38] Isaac Newton, Andrew Motte, and NW Chittenden. *Newton's Principia: The Mathematical Principles of Natural Philosophy*. Geo. P. Putnam, 1850.

[39] George Keith Batchelor. *An Introduction to Fluid Dynamics*. Cambridge university press, 2000.

[40] A Haider and O Levenspiel. Drag coefficient and terminal velocity of spherical and nonspherical particles. *Powder Technology*, 58(1):63–70, 1989.

[41] Marco A Cartas-Ayala, Mohamed Raafat, and Rohit Karnik. Self-sorting of deformable particles in an asynchronous logic microfluidic circuit. *Small*, 9(3):375–381, 2013.

[42] Dongeun Huh, Joong Hwan Bahng, Yibo Ling, Hsien-Hung Wei, Oliver D Kripfgans, J Brian Fowlkes, James B Grotberg, and Shuichi Takayama. Gravity-driven microfluidic particle sorting device with hydrodynamic separation amplification. *Anal Chem*, 79(4):1369–76, Feb 2007.

[43] Jihwan Song, Minsun Song, Taewook Kang, Dongchoul Kim, and Luke P Lee. Label-free density difference amplification-based cell sorting. *Biomicrofluidics*, 8(6):064108, Nov 2014.

[44] William H Grover, Andrea K Bryan, Monica Diez-Silva, Subra Suresh, John M Higgins, and Scott R Manalis. Measuring single-cell density. *Proc Natl Acad Sci U S A*, 108(27):10992–6, Jul 2011.

[45] Dino Di Carlo. Inertial microfluidics. *Lab Chip*, 9(21):3038–46, Nov 2009.

[46] Ewan Henry, Stefan H Holm, Zunmin Zhang, Jason P Beech, Jonas O Tegenfeldt, Dmitry A Fedosov, and Gerhard Gompper. Sorting cells by their dynamical properties. *Scientific Reports*, 6, 2016.

[47] PJ Hoogerbrugge and JMVA Koelman. Simulating microscopic hydrodynamic phenomena with dissipative particle dynamics. *EPL (Europhysics Letters)*, 19(3):155, 1992.

[48] Pep Espanol and Patrick Warren. Statistical mechanics of dissipative particle dynamics. *EPL (Europhysics Letters)*, 30(4):191, 1995.

[49] Pep Espanol and Mariano Revenga. Smoothed dissipative particle dynamics. *Physical Review E*, 67(2):026705, 2003.

[50] Zunmin Zhang, Ewan Henry, Gerhard Gompper, and Dmitry A Fedosov. Behavior of rigid and deformable particles in deterministic lateral displacement devices with different post shapes. *The Journal of chemical physics*, 143(24):243145, 2015.

[51] Lailai Zhu, Cecilia Rorai, Dhrubaditya Mitra, and Luca Brandt. A microfluidic device to sort capsules by deformability: a numerical study. *Soft Matter*, 10(39):7705–7711, 2014.

[52] Timm Krüger, David Holmes, and Peter V Coveney. Deformability-based red blood cell separation in deterministic lateral displacement devices—a simulation study. *Biomicrofluidics*, 8(5):054114, 2014.

[53] Charles S Peskin. The immersed boundary method. *Acta numerica*, 11:479–517, 2002.

[54] Nirveek Bhattacharjee, Arturo Urrios, Shawn Kang, and Albert Folch. The upcoming 3d-printing revolution in microfluidics. *Lab Chip*, 16(10):1720–42, May 2016.

[55] Nazila Norouzi, Heran C Bhakta, and William H Grover. Orientation-based control of microfluidics. *PLOS ONE*, 11(3):e0149259, 2016.

[56] Johannes Srajer, Burhanuddin J Majlis, and Ille C Gebeshuber. Microfluidic simulation of a colonial diatom chain reveals oscillatory movement. *Acta Botanica Croatica*, 68(2):431–441, 2009.

[57] A Rasmussen, C Mavriplis, ME Zaghloul, O Mikulchenko, and K Mayaram. Simulation and optimization of a microfluidic flow sensor. *Sensors and Actuators A: Physical*, 88(2):121–132, 2001.

[58] Junchao Wang, Philip Brisk, and William H. Grover. Random design of microfluidics. *Lab on a Chip*, 16:4212–4219, 2016.

[59] Philipp Hahn, Andreas Lamprecht, and Jurg Dual. Numerical simulation of microparticle rotation by the acoustic viscous torque. *Lab Chip*, 16:4581–4594, 2016.

[60] Donald Michie. Memo functions and machine learning. *Nature*, 218(5136):19–22, 1968.

[61] Dongshin Kim, Naomi C Chesler, and David J Beebe. A method for dynamic system characterization using hydraulic series resistance. *Lab on a Chip*, 6(5):639–644, 2006.

[62] David J Beebe, Glennys A Mensing, and Glenn M Walker. Physics and applications of microfluidics in biology. *Annual Review of Biomedical Engineering*, 4(1):261–286, 2002.

[63] Leah A Godwin, Kennon S Deal, Lauren D Hoepfner, Louis A Jackson, and Christopher J Easley. Measurement of microchannel fluidic resistance with a standard voltage meter. *Analytica Chimica Acta*, 758:101–107, 2013.

[64] Laurence William Nagel and Donald O Pederson. *SPICE: Simulation program with integrated circuit emphasis*. Electronics Research Laboratory, College of Engineering, University of California, 1973.

[65] LS Han. Hydrodynamic entrance lengths for incompressible laminar flow in rectangular ducts. *Journal of Applied Mechanics*, 27(3):403–409, 1960.

[66] Robert A. Granger. *Fluid Mechanics*. Dover Publications, 1995.

[67] Mysql. `http://www.mysql.com`. Accessed: 2016-03-23.

[68] WA Bonner, HR Hulett, RG Sweet, and LA Herzenberg. Fluorescence activated cell sorting. *Review of Scientific Instruments*, 43(3):404–409, 1972.

[69] Howard M Shapiro and Howard M Shapiro. Practical flow cytometry. 2003.

[70] Sunitha Nagrath, Lecia V Sequist, Shyamala Maheswaran, Daphne W Bell, Daniel Irimia, Lindsey Ulkus, Matthew R Smith, Eunice L Kwak, Subba Digumarthy, Alona Muzikansky, et al. Isolation of rare circulating tumour cells in cancer patients by microchip technology. *Nature*, 450(7173):1235, 2007.

[71] Nezihi Murat Karabacak, Philipp S Spuhler, Fabio Fachin, Eugene J Lim, Vincent Pai, Emre Ozkumur, Joseph M Martel, Nikola Kojic, Kyle Smith, Pin-i Chen, et al. Microfluidic, marker-free isolation of circulating tumor cells from blood samples. *Nature protocols*, 9(3):694, 2014.

[72] Wooseok Jung, Jungyoup Han, Jin-Woo Choi, and Chong H Ahn. Point-of-care testing (poct) diagnostic systems using microfluidic lab-on-a-chip technologies. *Microelectronic Engineering*, 132:46–57, 2015.

[73] Nazila Norouzi, Heran C Bhakta, and William H Grover. Sorting cells by their density. *PLoS One*, 12(7):e0180520, 2017.

[74] Masumi Yamada, Megumi Nakashima, and Minoru Seki. Pinched flow fractionation: continuous size separation of particles utilizing a laminar flow profile in a pinched microchannel. *Analytical chemistry*, 76(18):5465–5471, 2004.

[75] Junchao Wang, Philip Brisk, and William H Grover. Random design of microfluidics. *Lab on a Chip*, 16(21):4212–4219, 2016.

[76] MATLAB. `http://www.mathworks.com/products/matlab` . Accessed: 2016-03-23.

[77] COMSOL Multiphysics. `http://comsol.com`. Accessed: 2016-09-26.

[78] Junchao Wang, Victor G. J. Rodgers, Philip Brisk, and William H. Grover. Mopsa: A microfluidics-optimized particle simulation algorithm. *Biomicrofluidics*, 11(3):034121, 2017.

[79] Junchao Wang, Victor G. J. Rodgers, Philip Brisk, and William H. Grover. Instantaneous simulation of fluids and particles in complex microfluidic devices. *PloS ONE In revision.*

[80] Junya Takagi, Masumi Yamada, Masahiro Yasuda, and Minoru Seki. Continuous particle separation in a microchannel having asymmetrically arranged multiple branches. *Lab on a Chip*, 5(7):778–784, 2005.

[81] Dongeun Huh, Joong Hwan Bahng, Yibo Ling, Hsien-Hung Wei, Oliver D Kripfgans, J Brian Fowlkes, James B Grotberg, and Shuichi Takayama. A gravity-driven microfluidic particle sorting device with hydrodynamic separation amplification. *Analytical chemistry*, 79(4):1369, 2007.

[82] Hendrik Anthony Kramers. Brownian motion in a field of force and the diffusion model of chemical reactions. *Physica*, 7(4):284–304, 1940.

[83] Dino Di Carlo. Inertial microfluidics. *Lab on a Chip*, 9(21):3038–3046, 2009.

[84] Manuchehr Aminian, Francesca Bernardi, Roberto Camassa, Daniel M Harris, and Richard M McLaughlin. How boundaries shape chemical delivery in microfluidics. *Science*, 354(6317):1252–1256, 2016.

[85] Chia-Yen Lee, Chin-Lung Chang, Yao-Nan Wang, and Lung-Ming Fu. Microfluidic mixing: a review. *International journal of molecular sciences*, 12(5):3263–3287, 2011.

[86] Chuang Li and Tianning Chen. Simulation and optimization of chaotic micromixer using lattice boltzmann method. *Sensors and Actuators B: Chemical*, 106(2):871–877, 2005.

[87] David E Hertzog, Benjamin Ivorra, Bijan Mohammadi, Olgica Bakajin, and Juan G Santiago. Optimization of a microfluidic mixer for studying protein folding kinetics. *Analytical chemistry*, 78(13):4299–4306, 2006.

[88] Hengzi Wang, Pio Iovenitti, Erol Harvey, and Syed Masood. Numerical investigation of mixing in microchannels with patterned grooves. *Journal of Micromechanics and Microengineering*, 13(6):801, 2003.

[89] Hengzi Wang, Pio Iovenitti, Erol Harvey, and Syed Masood. Optimizing layout of obstacles for enhanced mixing in microchannels. *Smart materials and structures*, 11(5):662, 2002.

[90] Shakhawat Hossain, Mubashshir A Ansari, Afzal Husain, and Kwang-Yong Kim. Analysis and optimization of a micromixer with a modified tesla structure. *Chemical Engineering Journal*, 158(2):305–314, 2010.

[91] Cesar A Cortes-Quiroz, Alireza Azarbadegan, Mehrdad Zangeneh, and Akira Goto. Analysis and multi-criteria design optimization of geometric characteristics of grooved micromixer. *Chemical Engineering Journal*, 160(3):852–864, 2010.

[92] Junchao Wang, Philip Brisk, and William H Grover. Random design of microfluidics. *Lab on a Chip*, 16(21):4212–4219, 2016.

[93] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[94] Todd M Squires and Stephen R Quake. Microfluidics: Fluid physics at the nanoliter scale. *Reviews of modern physics*, 77(3):977, 2005.

[95] Sathyakumar S Kuntaegowdanahalli, Ali Asgar S Bhagat, Girish Kumar, and Ian Papautsky. Inertial microfluidics for continuous particle separation in spiral microchannels. *Lab on a Chip*, 9(20):2973–2980, 2009.

[96] Soojung Claire Hur, Nicole K Henderson-MacLennan, Edward RB McCabe, and Dino Di Carlo. Deformability-based cell classification and enrichment using inertial microfluidics. *Lab on a Chip*, 11(5):912–920, 2011.

[97] Ali Asgar S Bhagat, Han Wei Hou, Leon D Li, Chwee Teck Lim, and Jongyoon Han. Pinched flow coupled shear-modulated inertial microfluidics for high-throughput rare blood cell separation. *Lab on a Chip*, 11(11):1870–1878, 2011.

[98] Myung Gwon Lee, Joong Ho Shin, Chae Yun Bae, Sungyoung Choi, and Je-Kyun Park. Label-free cancer cell separation from human whole blood using inertial microfluidics at low shear stress. *Analytical chemistry*, 85(13):6213–6218, 2013.

[99] Junchao Wang, Victor GJ Rodgers, Philip Brisk, and William H Grover. Mopsa: A microfluidics-optimized particle simulation algorithm. *Biomicrofluidics*, 11(3):034121, 2017.

[100] Christopher T Culbertson, Stephen C Jacobson, and J Michael Ramsey. Diffusion coefficient measurements in microfluidic devices. *Talanta*, 56(2):365–373, 2002.

[101] Li Yuan-Hui and Sandra Gregory. Diffusion of ions in sea water and in deep-sea sediments. *Geochimica et cosmochimica acta*, 38(5):703–714, 1974.

[102] Jeffrey McDaniel, Daniel Grissom, and Philip Brisk. Multi-terminal PCB escape routing for digital microfluidic biochips using negotiated congestion. In *Very Large Scale Integration (VLSI-SoC), 2014 22nd International Conference on*, pages 1–6. IEEE, 2014.

[103] Daniel Grissom, Christopher Curtis, and Philip Brisk. Interpreting assays with control flow on digital microfluidic biochips. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 10(3):24, 2014.

[104] Barna Aladar Szabo and Ivo Babuška. *Finite element analysis*. John Wiley & Sons, 1991.

[105] William BJ Zimmerman. *Multiphysics modeling with finite element methods (series on Stability, vibration and control of systems, serie)*. World Scientific Publishing Co., Inc., 2006.

[106] Glenn M Walker, Jiqing Sai, Ann Richmond, Mark Stremler, Chang Y Chung, and John P Wikswo. Effects of flow and diffusion on chemotaxis studies in a microfabricated gradient generator. *Lab on a Chip*, 5(6):611–618, 2005.

[107] C Barthou, J Benoit, P Benalloul, and A Morell. Mn2+ concentration effect on the optical properties of Zn2SiO4: Mn phosphors. *Journal of The Electrochemical Society*, 141(2):524–528, 1994.

[108] Nadia Zaari, Padmavathy Rajagopalan, Sooyoung K Kim, Adam J Engler, and Joyce Y Wong. Photopolymerization in microfluidic gradient generators: microscale control of substrate compliance to manipulate cell response. *Advanced Materials*, 16(23-24):2133–2137, 2004.

[109] Luis M Fidalgo and Sebastian J Maerkl. A software-programmable microfluidic device for automated biology. *Lab on a Chip*, 11(9):1612–1619, 2011.

[110] Jessica Melin and Stephen R Quake. Microfluidic large-scale integration: the evolution of design rules for biological automation. *Annu. Rev. Biophys. Biomol. Struct.*, 36:213–231, 2007.

[111] William Thies, John Paul Urbanski, Todd Thorsen, and Saman Amarasinghe. Abstraction layers for scalable microfluidic biocomputers. In DNA Computing , pages 308–323. Springer, 2006.

[112] Chia-Hung Liu, Ting-Wei Chiang, and Juinn-Dar Huang. Reactant minimization in sample preparation on digital microfluidic biochips. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems , 34(9):1429–1440, 2015.

[113] Chia-Hung Liu, Kuo-Cheng Shen, and Juinn-Dar Huang. Reactant minimization for sample preparation on microfluidic biochips with various mixing models. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems , 34(12):1918–1927, 2015.

[114] Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. Dynamic programming and optimal control , volume 1. Athena Scientific Belmont, MA, 1995.