Performance Characterization of a Reconfigurable Planar-Array Digital Microfluidic System

Eric J. Griffith, Srinivas Akella, *Member, IEEE*, and Mark K. Goldberg

***Abstract*—This paper describes a computational approach to designing a digital microfluidic system (DMFS) that can be rapidly reconfigured for new biochemical analyses. Such a "lab-on-a-chip" system for biochemical analysis, based on electrowetting or dielectrophoresis, must coordinate the motions of discrete droplets or biological cells using a planar array of electrodes. The authors have earlier introduced a layout-based system and demonstrated its flexibility through simulation, including the system's ability to perform multiple assays simultaneously. Since array-layout design and droplet-routing strategies are closely related in such a DMFS, their goal is to provide designers with algorithms that enable rapid simulation and control of these DMFS devices. In this paper, the effects of variations in the basic array-layout design, droplet-routing control algorithms, and droplet spacing on system performance are characterized. DMFS arrays with hardware limited row-column addressing are considered, and a polynomial-time algorithm for coordinating droplet movement under such hardware limitations is developed. To demonstrate the capabilities of our system, we describe example scenarios, including dilution control and minimalist layouts, in which our system can be successfully applied.**

***Index Terms*—Array layout, biochips, digital microfluidics, droplet routing, lab-on-a-chip, performance analysis, row–column addressing.**

I. Introduction

MINIATURE biochemical analysis systems that use microfluidics technology have the potential to function as complete "lab-on-a-chip" systems. These systems offer a number of advantages, including reduced reagent requirements, size reduction, power reduction, increased throughput, and increased reliability. An important goal is to create reconfigurable and reprogrammable systems capable of handling a variety of biochemical analysis tasks.

A promising new class of lab-on-a-chip systems are digital microfluidic systems (DMFSs) that use phenomena such as electrowetting [8], [29], [31] and dielectrophoresis [22], [26]. Electrowetting-based microfluidic systems manipulate discrete droplets by modulating the interfacial tension of the droplets with a voltage [29]. Droplets have been moved at 12–25 cm/s on planar arrays of 0.15-cm-wide electrodes [8], [14]. Dielectrophoresis-based systems apply a spatially nonuniform electric field to actuate neutral charge particles [22], [26]. Arrays with 20-μm-wide electrodes that manipulate biological cells have been demonstrated [16]. The ability to control individual droplets or biological cells on a planar array enables complex analysis operations to be performed in biochemical lab-on-a-chip systems (Fig. 1). For example, they can be used to perform deoxyribonucleic acid (DNA) polymerase chain reactions (PCRs) for DNA-sequence analysis, to perform glucose assays, or to fuse biological cells with drug molecules. These systems have the potential to rapidly process hundreds or even thousands of samples on a single biochip. A key challenge in using DMFSs is developing computationally tractable algorithms to automate the simultaneous coordination of operations on a potentially large number of droplets or biological cells.

Our focus is the development of algorithms to automatically coordinate the transport and reaction operations on droplets or biological cells in a DMFS. We describe our approach in the context of droplet-based systems that use electrowetting; the same approach and algorithms may also be applied to dielectrophoresis-based systems that manipulate biological cells. The broad problem we are interested in is: Given a chemical analysis graph describing the sequence in which chemicals should mix, coordinate the droplet operations on the DMFS array for a set of droplets so as to permit mixing with prescribed mix times while avoiding undesired contact between droplets. Our approach to countering the complexity of this problem is to impose a virtual layout on the DMFS array and coordinate droplet operations by dynamically routing droplets to components in the layout. The layout permits us to abstract away from the underlying array hardware and provides an additional structure that simplifies droplet coordination. We previously described this approach to creating a general-purpose DMFS [18], [19], which combines a semiautomated approach to array-layout design using modular virtual components with algorithms for components to dynamically route the droplets. The resulting system has been simulated in software to perform analyses such as DNA PCR. The algorithms have been able to coordinate hundreds of droplets simultaneously and perform one or more chemical analyses in parallel.

In this paper, we explore variations on the basic DMFS layout design and routing control for increased versatility and performance, and describe example scenarios in which our system can be applied. Since array-layout design and droplet-routing strategies are closely related in a reconfigurable DMFS, our goal is to provide designers with simulation tools for both rapid evaluation and real-time control of these DMFS devices. After summarizing our previous work in Section III to provide the background, we describe the effects on system performance

Manuscript received March 25, 2005; revised July 21, 2005. This work was supported in part by National Science Foundation (NSF) under Award IIS-0093233 and Award IIS-0541224. This paper was recommended by Associate Editor K. Chakrabarty.

E. J. Griffith is with the Delft University of Technology, 2600 AA Delft, The Netherlands.

S. Akella and M. K. Goldberg are with Rensselaer Polytechnic Institute, Troy, NY 12180-3590 USA.

Digital Object Identifier 10.1109/TCAD.2005.859515

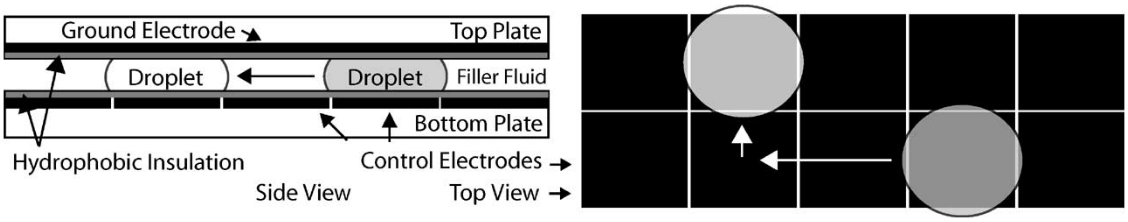


Fig. 1. Droplets on an electrowetting array (side and top views). A droplet moves to a neighboring control electrode when the electrode is turned on. The electrode is turned off when the droplet completes its motion. Based on [29].

of variations in design and control including different layout schemes, routing algorithms, and increased spacing between droplets in Section IV. We then develop a new approach to droplet coordination with limited row-column addressing in Section V. We use a polynomial-time graph coloring algorithm to coordinate droplet movements under such hardware limitations. Finally, in Section VI, we outline two application scenarios involving droplet-dilution control and minimal layouts to demonstrate the capabilities of our system.

II. Related Work

DMFSs: DMFSs are a novel and emerging class of lab-on-a-chip systems. Most work in this area has focused on developing hardware to demonstrate the feasibility of this new technology. Pollack *et al.* [31] demonstrated rapid manipulation of discrete microdroplets by electrowetting-based actuation. Fair *et al.* [14] describe experiments on injection, dispensing, dilution, and mixing of samples in an electrowetting DMFS. Cho *et al.* [8] developed an orthogonal cross-reference grid of single-layer electrodes to manipulate droplets with limited row–column addressing. Fan *et al.* [15] demonstrated creating, merging, splitting, and move operations using electrodes covered with dielectrics, and identified conditions under which these operations can be performed in an air environment. Gong *et al.* [17] developed a portable digital microfluidics lab-on-chip platform using electrowetting. They use a time-multiplexed control scheme to control droplets with limited row–column addressing, where the number of steps is proportional to the number of array rows. Paik *et al.* [29] studied the effects of droplet aspect ratios and mixing strategies on the rate of droplet mixing. Dielectrophoresis is another mechanism to actuate neutral charge particles and cells by applying a spatially nonuniform electric field [22], [26]. Jones *et al.* [22] demonstrated dielectrophoresis-based liquid actuation and nanodroplet formation. Arrays with 20-μm-wide electrodes that manipulate biological cells have been demonstrated [16].

More recently, work on DMFS has focused on applications. Srinivasan *et al.* [39] demonstrate the use of a DMFS as a biosensor for glucose, lactate, glutamate, and pyruvate assays, and use it for clinical diagnostics on blood, plasma, serum, urine, saliva, sweat, and tears [40]. Pollack *et al.* [32] have demonstrated the use of electrowetting-based microfluidics for real-time PCR applications. Wheeler *et al.* [46] demonstrate an electrowetting-based DMFS for the analysis of proteins by matrix-assisted laser desorption/ionization mass spectrometry, for high-throughput proteomics applications.

Coordination of droplet operations and architectural design for DMFS, the topics most closely related to the current paper, have been far less studied. In early work, Ding *et al.* [11] described an architectural design and optimization methodology for scheduling biochemical reactions using electrowetting arrays. They identified a basic set of droplet operations and used an integer-programming formulation to minimize completion time. Droplet paths and areas on the array for storage, mixing, and splitting operations are predefined by the user. Zhang *et al.* [47] describe hierarchical techniques for the modeling, design, performance evaluation, and optimization of microfluidic systems. They compared the performance of a continuous-flow system and a droplet-based system and showed that the droplet-based system has a less complex design that provides higher throughput and processing capacity. Su and Chakrabarty [41] recently proposed architectural-level synthesis techniques for digital microfluidics-based biochips, and describe an integer-programming formulation and heuristic techniques to schedule assay operations under resource constraints, prior to geometry-level synthesis. Our work is motivated by the above body of work, as well as the work of Böhringer [4], [5], who viewed each droplet in a DMFS as a simple robot that translates on an array and outlined an approach for moving droplets from start to goal locations, subject to droplet-separation constraints, obstacles, and control-circuitry limitations. He uses an A^* search algorithm to generate optimal plans for droplets. To overcome the exponential complexity of this approach, he plans the droplet motions in prioritized order. However a DMFS must have additional capabilities, such as the ability to combine and split droplets as needed, sometimes with different mixing durations.

Multiple-Robot Coordination: The coordination of droplets in a DMFS is closely related to multiple-robot-motion coordination, as pointed out above. Hopcroft *et al.* [21] showed that even a simplified two-dimensional case of motion planning for multiple translating robots is PSPACE-hard. Erdmann and Lozano-Perez [13] developed a heuristic approach for planning the motions of multiple robots that orders robots by assigned priority and sequentially searches for collision-free paths; this approach was used by Böhringer [5]. Owing to the computational complexity of the multiple-robot-motion-planning problem, recent efforts have focused on probabilistic approaches [35], [44].

When the paths of the robots are specified, as in the DMFS model of Ding *et al.* [11], a path-coordination problem arises. Path coordination was first studied by O'Donnell and Lozano-Perez [28] for two robots. LaValle and Hutchinson addressed a similar problem in [24], where each robot was constrained to a C-space roadmap during its motion. Simeon *et al.* [37] coordinated over 100 carlike robots, where robots with intersecting paths are partitioned into smaller sets. Akella and Hutchinson [1] developed a mixed-integer linear programming (MILP) formulation for the trajectory coordination of 20 robots by changing robot start times. Peng and Akella [30] developed an MILP formulation to coordinate many robots with simple double-integrator dynamics along specified paths. Conflict resolution among multiple aircraft in a shared airspace [3], [36], [43] is also closely related to multiple-robot coordination.

Flexible Manufacturing Systems: Our approach to droplet coordination in a DMFS shares similarities with flexible manufacturing systems, where product assembly is like droplet mixing. One example is a reconfigurable automated precision-assembly system that uses cooperating modular robots [34]. Such systems have been modeled and analyzed using several techniques including Petri nets [10]. Of particular interest to flexible manufacturing systems is the issue of deadlock avoidance, which has been analyzed for certain classes of systems [25], [33].

Networking: We can view our DMFS as a network. This system differs from typical networking systems in nontrivial ways, including the fact that droplets cannot be dropped and that the system has multiple classes of nodes and operations. However, techniques for network flow and rate control [2], [42] may be modified for a DMFS. Related research in networking includes work on hot-potato or deflection routing [7], [9] for different classes of networks, and work on rate control to ensure stability [23].

III. System Overview

In this section, we provide an overview of our system, previously described in [18], [19]. We create a general-purpose reconfigurable DMFS by first generating a virtual layout that logically partitions the array into virtual components that perform different functions, and then applying specialized algorithms for routing droplets to appropriate components. The layout is created by combining one or more modular tiles that each contain the same pattern of virtual components. Each virtual component is a logical grouping of cells that can perform one or more functions. A cell corresponds to an electrode of the array, and may have additional capabilities, such as the ability to optically sense droplets. We initially assume that individual cells of the array are addressable by direct activation of individual electrodes. A droplet moves to a neighboring cell (electrode) when that electrode is activated; the electrode is turned off when the droplet completes its motion. We assume each droplet has a unit volume, except during mixing. Each mix operation is followed by a split operation, which is performed by simultaneously activating the two electrodes on either side of the droplet. Droplets are dynamically allocated to virtual components based on the operation (such as mixing or transport) to be performed on them. We adapt network-routing algorithms to route the droplets to destination components in the layout. When the routing algorithms, provided with knowledge of the electrode-addressing mechanism, are used as the software controller for a DMFS, the droplet motions can be downloaded to a microcontroller at each clock cycle. The microcontroller will activate the requested set of electrodes to enable droplet motion.

Our approach of imposing a layout on a digital microfluidic array to suit given chemical reactions is similar to programming a reconfigurable field-programmable gate array (FPGA) [27]. However, unlike an FPGA, whose elements have distinct functions such as logic or routing, the interchangeable functionality of the DMFS cells permits instantaneous reconfigurations of the layout through software changes only. For example, a cell with a droplet-transport function in one layout may be used for droplet mixing or sensing in another layout.

This DMFS is reconfigurable in several ways. In the simplest sense, it can be reconfigured to run a variety of analyses that require moving, mixing, and splitting of different types of droplets just by changing the types of the input droplets and their associated mixing operations. One or more of these reactions can also be run in parallel. This reconfigurability potentially requires no actual change of the layout, but only changes to inputs to the software. Second, the actual layout design itself can be modified by altering the number of tiles and their arrangement, the number of components in a tile and their arrangement, and the locations of the sources and the sinks. We can even partition a large array into multiple DMFS layouts. This type of reconfigurability offers control over the system performance, and supports a wider variety of biochemical analyses. Third, the system offers reconfigurability by the ability to introduce new component types such as droplet-storage components or, if supported by the array hardware, optical sensor components. This offers flexibility for tailoring to specific analysis needs and for future expansion. Finally, the system can easily incorporate changes to the droplet-routing-and-scheduling algorithms to optimize performance.

A. Array-Layout Design Using Components

We partition the array into a set of "virtual" components, where each type of component performs a specific set of operations. This partitioning is enabled by the versatility of the array electrodes, which can perform droplet movement, merging, mixing, and splitting operations practically anywhere on the array. Each component controls droplets within its cells, and, by linking a sufficient set of components together, a DMFS can be created to perform one or more biochemical analyses. Fig. 2 illustrates an example system comprised of six component types. These six virtual components (Fig. 3) perform droplet transportation (street, connector, and intersection components) or droplet mixing, input, and output operations (work area, source, and sink components).

Street Component: The street component is the general-purpose droplet-transportation component. Streets are one-way to prevent two droplets from moving in opposite directions through the component.

Connector Component: The connector component is a specialized version of a street component where a droplet only moves through a single cell. A droplet in a connector is adjacent to two components simultaneously.

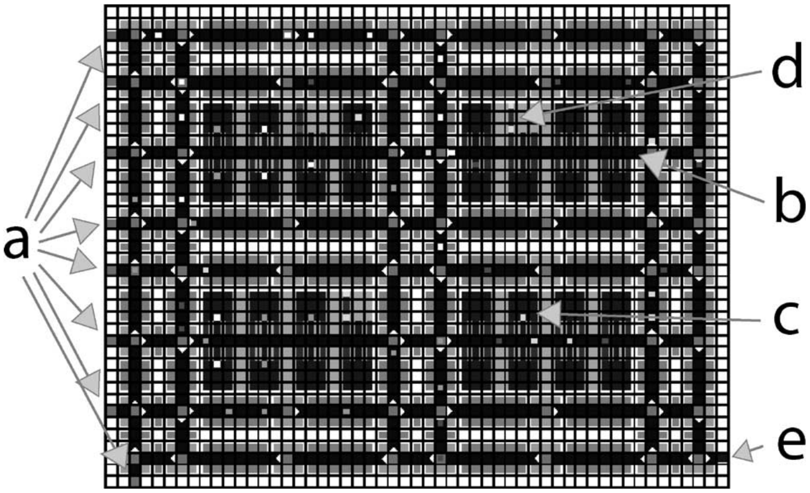


Fig. 2. Array layout for the PCR analysis described in Section III-C. Each cell of the array is represented by a square; arrowheads indicate valid droplet-motion directions. On the left side of the array are (a) eight sources, which supply the input sample droplets to the system. There are (b) four work areas on the array, in which droplets are (c) mixed together and (d) split apart. In the lower right corner of the array is a (e) sink, which moves the droplets of the final products off the array.

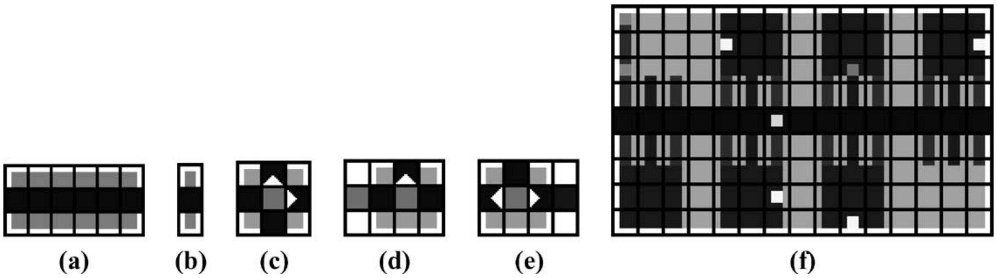


Fig. 3. Components. (a) A street. (b) A connector component. (c) An intersection. (d) A source connected to an intersection. (e) A sink connected to an intersection. (f) An active work area, showing several mixing units with droplets (depicted as small squares).

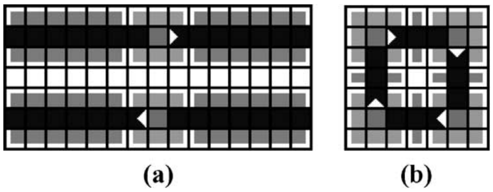


Fig. 4. Simulating two-way transportation: (a) Two-way street, (b) rotary.

Intersection Component: The intersection components route droplets through the system, using the algorithms described in Section III-B.

Work Area Component: The work area component is where mixing and splitting take place. Each work area has a transit area and multiple mixing units. Each mixing unit may function as a mixer and/or as a splitter. A work area can mix and split multiple droplets at the same time.

Source Component: The source component represents an input point for droplets into the array.

Sink Component: The sink component represents an output point for droplets from the array.

The layout is designed to have sufficient capacity to both transport droplets between components and to process droplets. We do this by first grouping one-way streets and intersections into two-way streets and rotaries (Fig. 4). Then we couple this with a work area to form a pattern, shown in Fig. 5, which can be tiled periodically to create the layout. The layout is completed with an alternating sequence of rotaries and streets along its upper and right edges. To generate the layout, the user must know the physical size of the array and specify the locations of sources and sinks. Our design can be expanded to accommodate new types of components for specific or general operations.

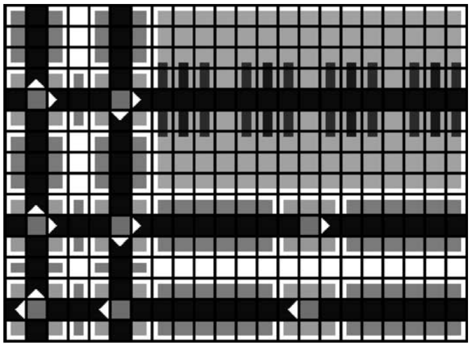


Fig. 5. Pattern tile that is a modular building block for the layout.

B. Droplet-Destination Selection and Routing Algorithms

The core algorithms in our approach deal with deciding where to send droplets, and how to get them there. With these droplet-destination selection and routing algorithms, we transform a set of interconnected components into a functional DMFS. The intersection components execute these algorithms to route droplets through the system.

Assigning a destination to a droplet depends on the droplet type and the available components. The droplet type determines whether it is to mix with another type of droplet in a work area or leave the array from a sink. An available work area is either one that has already had one of the two droplets for a mixing operation assigned and is requesting the other type, or one with free mixing units that can accept any type of droplet. Each available work area and sink adds itself to a (global) ordered list of components accepting droplets for operations. There is also a (global) ordered list of higher priority containing requests from work areas for specific droplet types required to complete a mix-and-split operation. Intersections assign work areas and sinks on a rotating basis, except when the second droplet in a mixing operation is being requested.

When a new droplet enters the system, or is created through a mixing operation, the droplet type determines the operation it is assigned. When the droplet enters an intersection, the intersection tries to find a destination component to send the droplet to by first checking the high-priority list and then, if necessary, the low-priority list. If any component is actively requesting that droplet type for its operation, the droplet is assigned to that component. Failing that, the droplet is assigned to the first component that can accept droplets of its type. If no components are available to assign the droplet to, then the next intersection the droplet enters attempts to assign it a destination.

The droplet-routing method we use can be viewed as a deflection-routing variant [6] of the Open Shortest Path First (OSPF) network protocol [42]. When the system is initialized, each intersection uses Dijkstra's algorithm to compute a routing table, which maps the shortest legal path between the intersection and each component to a corresponding exit from which to leave the intersection.

At each clock cycle, the intersections are processed in a fixed order to select their droplet-routing moves, as described in Section III-C. Subsequently, a synchronous motion of droplets is executed. If a droplet entering the intersection has no destination, then the intersection attempts to assign it one. If that fails, then the droplet is sent to a random valid exit. For droplets with destinations, the intersection finds the destination component in its routing table and selects the exit that corresponds to the shortest path to the destination. If the droplet is able to move toward that exit, it does so. Otherwise, the intersection randomly chooses a valid exit for the droplet. If no viable exit is available, then the droplet waits.

C. General-Purpose DMFS

We create a general-purpose DMFS by combining the component-based-layout-design approach and droplet-destination-selection-and-routing algorithms. The basic layout is designed to handle a variety of analyses. Furthermore, the DMFS can be reconfigured by altering the number of mixing units in the work areas, the overall size of the layout, the locations of the sources and sinks, and the types of analyses it is to perform. The layout approach presented here can be extended to produce new layouts, and to incorporate new types of components into the system. To fully define the system, the user must specify additional parameters based on the chemical analyses to be performed, including the type of droplets introduced at each source, when and how often they are produced, the types of droplets to send to the sinks, and information about the various intermediate operations to perform on the droplets. A complete example 2 × 2 layout with eight sources and one sink can be seen in Fig. 2.

1) DMFS Control: The above approach to DMFS organization yields a collection of communicating components organized into a network. Components may move droplets at will within themselves, but before moving droplets into cells bordering a neighboring component or into a neighboring component, they must consult the neighbor to ensure this would not result in two droplets being adjacent. Therefore, the system first processes the components serially at each clock cycle and then executes motion in parallel. The system does this by maintaining an ordered master list of components. At each clock cycle, each component in the list is instructed to attempt to move its droplets. When a particular component wishes to move a droplet into an array cell adjacent to or into a neighbor component, it first asks that component if the move will result in two droplets being adjacent. If it will, then it requests the neighbor component to attempt to move its droplets, and then it asks again if the move will result in two droplets being adjacent. If the move would still result in adjacent droplets, then it waits to move those droplets that would result in violations. A separate master list is kept containing the current location of all droplets and their desired location in the next clock cycle. As each component is processed, it updates the list of droplets to reflect the current and desired locations of each droplet within it. The set of consistent droplet movements can then be collected so motion can be performed in parallel.

2) System Stability: The behavior of a general-purpose system changes with the chemical analysis it performs. We define a DMFS to be stable if it does not get deadlocked after ten million clock cycles of operation. We define a DMFS to be in deadlock if no droplet in the system is able to move. A system operating continuously may or may not be stable depending on its parameters, especially the input flow rate of droplets. In an unstable system, droplets enter the system faster than the system is able to process them, and a steady-state flow cannot be guaranteed [20]. In time, such a system will become heavily congested and finally become deadlocked. We identify stable systems by simulating them and checking at each clock cycle whether they are in a state where no droplet may move.

3) System Simulation: We have simulated several analyses, including one based on the DNA PCR operations outlined in [11]. The analysis involves eight input-droplet types and seven mixing operations. See Fig. 6 for an analysis graph of the system (note that the PCR analysis requires heating steps and that we assume that droplets may be routed off-chip for heating). Immediately following each mixing operation, the resulting droplet is split into two droplets. The layout is set up with four work areas, eight sources, each introducing an input droplet type, and one sink to collect the final product (Fig. 2). This layout with a 2 × 2 tile arrangement has 53 × 41 cells.

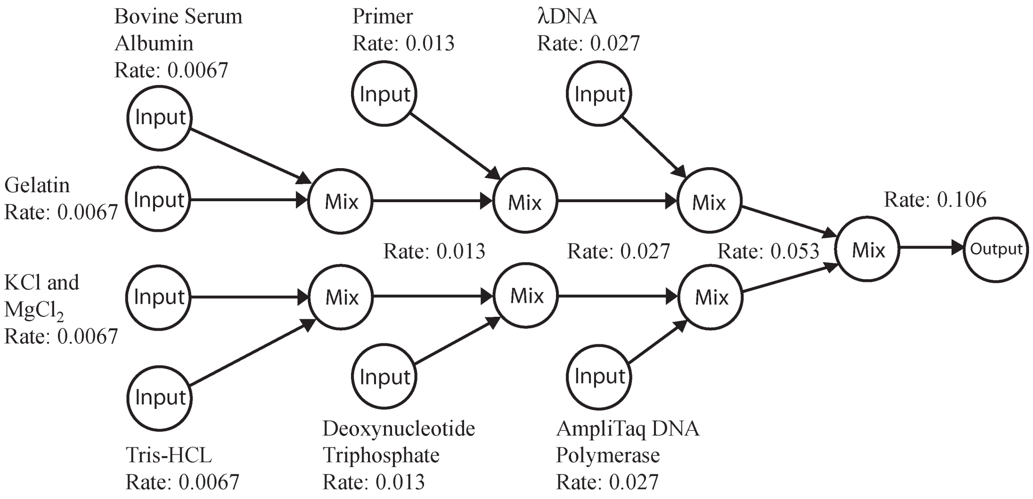
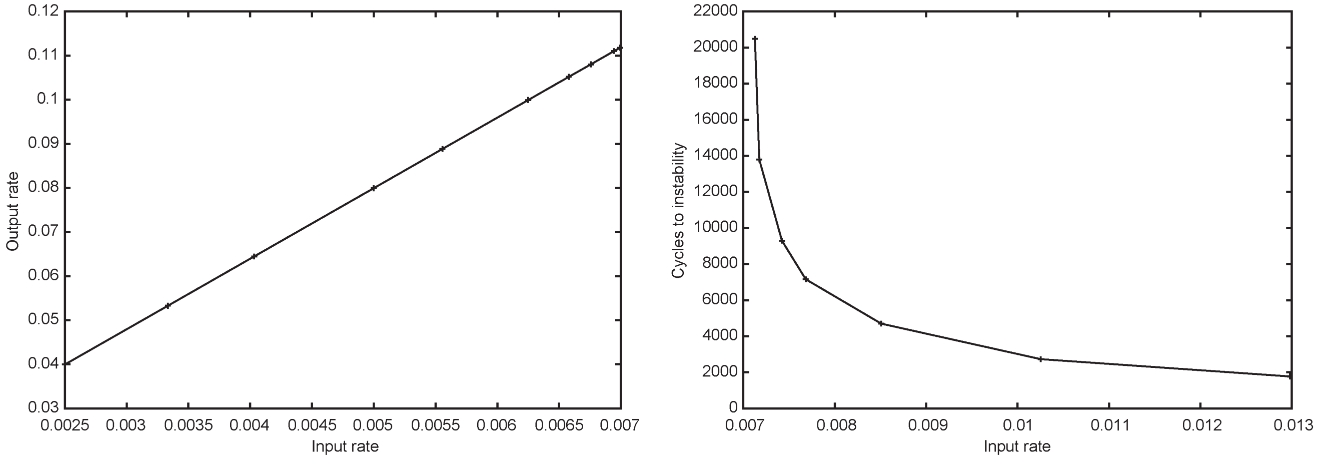


Fig. 6. PCR analysis graph. Input nodes are labeled with the samples they introduce and the rate at which they introduce them, in droplets per cycle. Edges out of mix nodes are labeled with the droplet rate resulting from the operation.



(a) (b)

Fig. 7. Simulation data for the PCR analysis illustrating (a) variation of droplet output rate with input rate in the stable range, and (b) number of cycles at which the system goes into deadlock, as input rate is increased in the unstable range. For this example, mixing time is 128 cycles, the number of mixing units per work area is 8, and the tiles are in a 2 × 2 pattern.

The system has an average of 66 droplets on the array. Our simulation environment is the stand-alone C++ software that we have created for this application; this software may also be used in a controller for a DMFS chip. The routing computations for this array are performed at a rate of about 60 000–70 000 Hz on a 1.7-GHz Pentium-M laptop with 512 MB of RAM. This enables rapid simulation of the system to verify stability. For example, at this speed, we can simulate 1 000 000 cycles in approximately 15–20 s. Animations of the PCR analysis, as well as multiple analyses running in parallel, are available at www.cs.rpi.edu/~sakella/microfluidics/.

The simulation approach has provided insight into the behavior of the system. When the system is in its stable operating range, there is a linear relation between the input droplet rate and output droplet rate, since no droplets are accumulating on the array [Fig. 7(a)]. Once a critical input rate is exceeded, there is a rapid dropoff in the number of clock cycles at which deadlock occurs [Fig. 7(b)]. Here, the "input rate" is the rate at which each of the four chemicals on the left of Fig. 6 is introduced. The subsequent input chemicals are introduced at correspondingly higher multiples of the input rate. We have observed sharp variations in behavior when simulating systems that are on the borderline between stability and instability. Small changes in the input rate at which droplets enter the system can mean the difference between becoming deadlocked in 5000 cycles, becoming deadlocked in 2 000 000 cycles, or running continuously for 10 000 000 cycles without deadlock.

IV. Variations on the Existing System

We now briefly describe our efforts to optimize the system performance. We experimented with a variety of modifications to the original system to gauge their effects on the stability of the system, and to determine which modifications allowed the system to be stable at the highest input rates.

A. *Variations on the Layout Tile*

We first experimented with altering the modular tile pattern used to create the layout (Fig. 5). Our goal was to increase

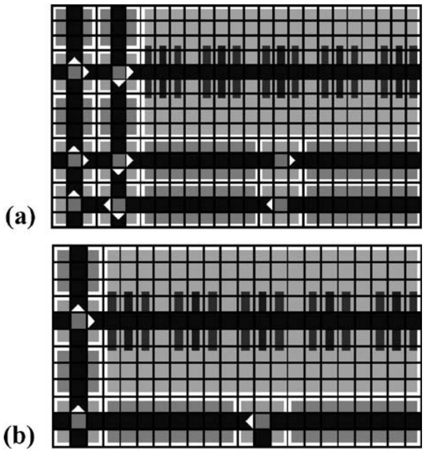


Fig. 8. Tile variations: (a) With no connectors between streets. (b) With only one-way streets.

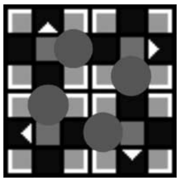


Fig. 9. When droplets are in this particular configuration, they cannot move again. Attempting to advance any droplet would require activating the adjacent electrode, which is also diagonally adjacent to another droplet. This activation could result in unexpected droplet movement or mixing, and therefore, is disallowed.

TABLE I
COMPARISON OF THE STABILITY OF THREE TILE-LAYOUT PATTERNS WITH A 2 × 2 TILE ARRANGEMENT, FOR THE PCR ANALYSIS. INPUT RATE IS MEASURED IN DROPLETS PER CLOCK CYCLE

Tile Layout	Mixing Units per Work Area	Highest Stable Rate (Approx.)
Default	8	0.0065
No Connectors	8	0.0040
One Way	8	0.0050
Default	10	0.0080
No Connectors	10	0.0030
One Way	10	0.0055
Default	12	0.0090
No Connectors	12	0.0040
One Way	12	0.0060

the percentage of space on the tile devoted to droplet-mix-and-split operations. We created two alternative layouts, shown in Fig. 8. The first tile removes the connector components between streets, and the second tile has only one horizontal and vertical street, rather than oppositely directed pairs of each.

These alternative tiles were not effective, however. In the tile without the connectors between the streets, rotaries become deadlocked whenever the situation in Fig. 9 arises. Once one set of intersections has become deadlocked, the system usually ceases being able to operate soon after due to the resulting droplet-traffic backup. The layout with only one way streets suffers from a diminished capacity for droplet traffic, which is exacerbated by droplets often needing to travel a greater distance to reach their destinations. The three layout designs are compared in Table I.

B. Variations in Routing Control

We also experimented with three changes to component behavior. The first change was to modify the droplet-destination-selection-and-routing algorithm to assign droplets to the closest available component instead of the original method of assigning them to components on a rotating basis. The second change was to have half of the work areas on the array be right to left (i.e., droplets enter from the right and exit from the left side of the work area) instead of all work areas being left to right. The third change was varying the order in which components attempt to move their droplets. In the original implementation, the components were assigned an initial order, and they attempted to move their droplets in that order at each cycle. The order is, generally, sources and work areas first, and then, the remaining components; the order could vary a little at each cycle based on droplet-movement dependencies. We instead compute a random permutation of the components at each clock cycle, and then the components try to move their droplets in that order, subject to droplet-movement-dependence variations.

The effects of these variations are depicted in Fig. 10. The best performance is obtained by using the new routing algorithm with the original work areas and fixed component order. In general, all combinations with the new routing algorithm performed better than their counterparts with the old routing algorithm. The opposite is true with the mixture of left-to-right work areas with right-to-left work areas versus just left-to-right work areas. Similarly, the new component order offers slightly inferior performance to the original component ordering. The other interesting characteristic is that the effects of the various changes are negligible with small arrays that can only operate at lower input rates, but, as the size of the array and thus its capacity for processing droplets increases, the effects of the changes become more pronounced (Table II).

C. Increased Droplet Spacing

We earlier assumed that multiple droplets moving in a line could be moved in synchrony in the same direction with only a single empty array cell between droplets. However, this assumption requires a high degree of synchronization of electrode activation, and may make this type of movement hard to implement or even infeasible. We now assume that in addition to the requirement that droplets must have at least one empty array cell on all sides except when mixing is about to occur, that any droplets moving in the same direction simultaneously must have at least two empty cells between them to avoid undesired mixing or splitting (Fig. 11). There should be at least three empty cells between droplets when there is a 90° bend in the path. This change has not significantly affected the performance of the system because it is rare, under stable conditions, for droplets to be moving in the same direction with only one empty array cell between them.

D. Additional Enhancements

Although we have implicitly described all mixing operations as taking the same amount of time, the system accommodates

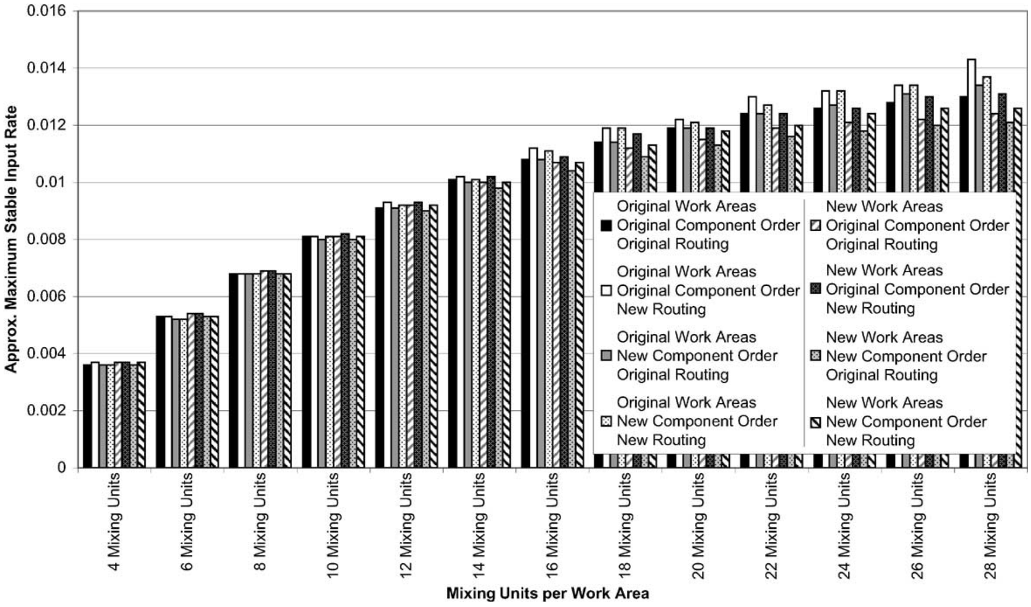


Fig. 10. Chart depicting the effects of each of the three routing-control variations on a 2 × 2-tile PCR simulation. Input rate is measured in droplets per clock cycle. Each bar in the graph corresponds to operating the system under a certain set of parameters. Parameters labeled as "new" correspond to the new methods in Section IV-B. Parameters labeled as "original" correspond to the original methods described in Section III.

TABLE II
COMPARISON OF THE MAXIMUM INCREASE IN STABLE RATE DUE TO DIFFERENT VARIATIONS IN ROUTING CONTROL, FOR DIFFERENT VALUES OF MIXING UNITS PER WORK AREA. DATA IS FOR A 2 × 2 TILE-LAYOUT SIMULATION OF THE PCR ANALYSIS. FOR A LOWER NUMBER OF MIXING UNITS PER WORK AREA, THE MAXIMUM INCREASE IS ACHIEVED WITH NEW WORK AREAS AND NEW ROUTING, WHILE FOR A HIGHER NUMBER OF MIXING UNITS PER WORK AREA, IT IS ACHIEVED WITH NEW ROUTING AND THE ORIGINAL WORK AREAS AND COMPONENT ORDER. RATE IS MEASURED IN DROPLETS PER CLOCK CYCLE

Mixing Units per Work Area	Maximum % Increase in Stable Rate
4	2.778
6	1.887
8	1.471
10	1.235
12	2.198
14	0.990
16	3.704
18	4.386
20	2.521
22	4.839
24	4.762
26	4.688
28	10.0

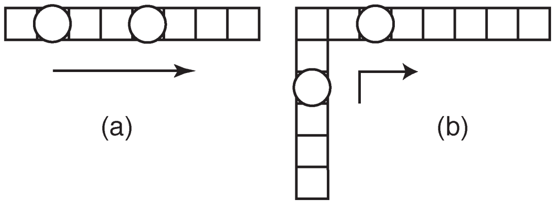


Fig. 11. Minimum number of empty cells between two occupied cells to ensure that the droplets cannot combine or split inadvertently depends on the path shape. (a) When the two cells are on a straight line. (b) When the two cells are around a bend in the path.

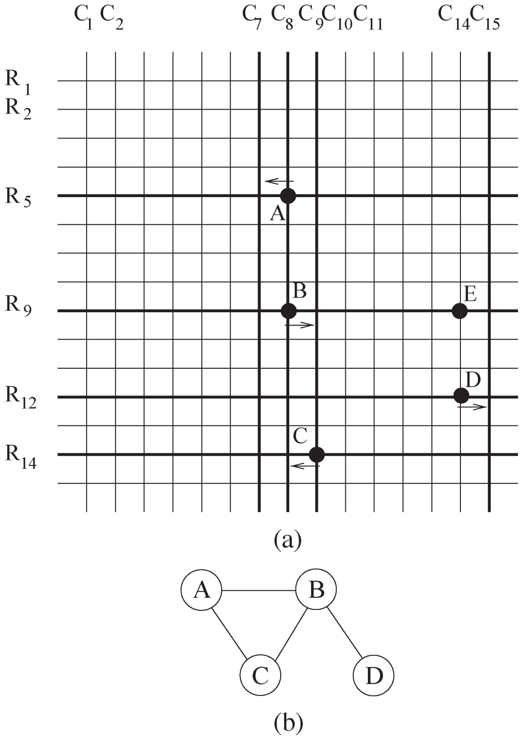
mixing operations with differing durations based on the droplet types. There are other enhancements to the system that can be easily incorporated. We can add virtual-storage components to the layout by treating one or more of the mixing units in a work area as storage units. Similarly, if some or all of the array cells have optical-sensing capabilities, we can create sensing components for the layout, located in the work areas, for example, or even in the streets or intersections. These sensors can permit monitoring of reaction results based on droplet color.

V. LIMITED ROW–COLUMN ADDRESSING

We have so far assumed that every electrode on the 2-D array can be individually addressed, so an arbitrary set of cells can be activated at each cycle. In a limited row–column addressing scheme, individual cells are not directly addressable. Only entire rows and columns can be activated and only electrodes at intersections of activated rows and columns will be turned on [5], [15], [17]. For example, Fan *et al.* [15] developed a cross-referencing scheme by arranging two vertically separated electrode layers orthogonal to each other. While this simplifies the hardware and reduces fabrication and packaging costs, it provides less flexibility in moving several droplets in synchrony and complicates droplet control. The interference graph (Fig. 12) represents potential conflicts between droplet movements. Here, two vertices connected by an edge represent droplets that cannot be moved in the same clock cycle.

A. Modified Schemes for Limited Row–Column Addressing

The central issue with limited row–column addressing is how to serialize the previously synchronous motion of the droplets at each clock cycle. In direct addressing mode, the movements



(a)

(b)

Fig. 12. Schematic illustration of droplet motion in an array with limited row–column addressing. (a) Each line represents a control wire connected to all electrodes in the corresponding row or column. Bold lines represent columns or rows to be activated. Droplet A is to be moved from the cell at (C_8, R_5) to (C_7, R_5), droplet B from (C_8, R_9) to (C_9, R_9), droplet C from (C_9, R_{14}) to (C_8, R_{14}), droplet D from (C_{14}, R_{12}) to (C_{15}, R_{12}), and droplet E is to remain stationary. (b) The interference graph indicates the conflicts for simultaneous droplet motion. Each vertex represents a droplet, and two vertices connected by an edge represent droplets that cannot be moved at the same time. Simultaneously activating the rows R_5, R_9, R_{14} and columns C_7, C_8, C_9 would not guarantee the desired motion for droplets A, B, and C. Moving droplets B and D simultaneously would also move droplet E. So instead, in one clock cycle, droplet A can be moved by activating R_5 and C_7 and droplet D by activating R_{12} and C_{15}, in the next clock cycle, droplet C can be moved by activating R_{14} and C_8, and in the next clock cycle droplet B can be moved by activating R_9 and C_9.

for all droplets are calculated at each clock cycle, and they are then executed in parallel. For clarity, we will refer to one clock cycle in direct addressing mode as a virtual clock cycle. For row–column addressing, the droplet movements are computed at the beginning of each virtual clock cycle and then the droplet movements are executed over one or more real clock cycles.

We have developed two schemes to perform limited row–column addressing for the DMFS. The first is a simple row–column addressing scheme where only one cell is addressed each cycle, by simultaneously activating both its row and column. Hence, only one droplet is moved each real clock cycle. Moving any droplet by a planned move will not result in it being inadvertently adjacent to any other droplet either before or after the droplet's movement. This is because the planning of the droplet movements (Section III-C) ensures that no motions are allowed for droplets that would move adjacent to either the starting or ending location of a droplet in a particular virtual cycle.

We next describe a more complex row–column addressing scheme where multiple cells may be addressed by simultaneously activating their rows and columns. In this scheme, multiple droplets may be moved at each clock cycle such that their activation does not cause other droplets to move inadvertently, and they do not inadvertently move next to another droplet. See Fig. 12 for an example scenario.

B. Graph-Coloring Approach

We have developed a graph-coloring approach to limited row–column addressing, to reduce the number of real clock cycles per virtual clock cycle by performing multiple droplet motions simultaneously. The results below are quite general and in fact apply to any array layout with a planar grid of electrodes. Scheduling an interference-free movement of the droplets may be modeled as a vertex-coloring problem. It is known that the general vertex-coloring problem is NP-complete (see [38]); furthermore, it is NP-complete even on the class of three-colorable graphs. The fastest algorithms for three-colorable graphs are exponential [12]. We introduce a heuristic, polynomial-time algorithm for coloring the interference graph (or equivalently, the transition graph introduced below). Note that this algorithm is not guaranteed to produce an optimal coloring.

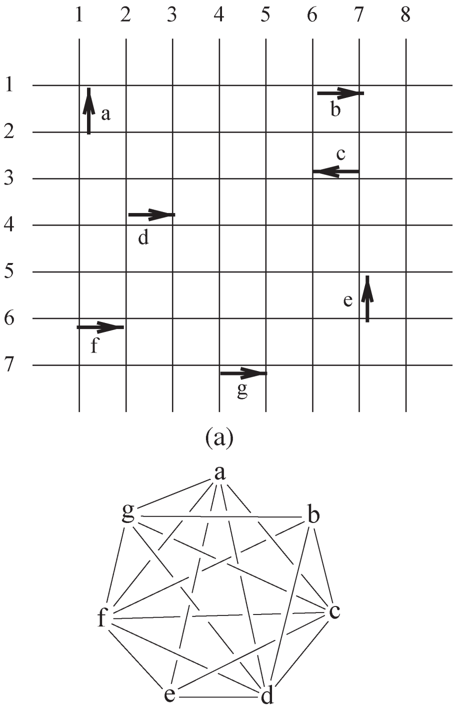
To address the problem of scheduling the movements of the droplets, we define a transition graph $T(V, E)$. The input to such a graph consists of a set L of the current locations of the droplets and the set M of the droplets' movements that are to be performed in the current virtual clock cycle. Every movement is an ordered pair of coordinates $[(x_\mathrm{s}, y_\mathrm{s}); (x_\mathrm{d}, y_\mathrm{d})]$, where the first term, $(x_\mathrm{s}, y_\mathrm{s})$ is the current (start) location of the droplet, and the second one, $(x_\mathrm{d}, y_\mathrm{d})$, is the next destination. Since all movements are either horizontal or vertical movements in the grid, the pair describing a movement satisfies the following condition

$$|x_\mathrm{s} - x_\mathrm{d}| = 1 \text{ and } y_\mathrm{s} = y_\mathrm{d}, \quad \text{for a horizontal movement}$$
$$|y_\mathrm{s} - y_\mathrm{d}| = 1 \text{ and } x_\mathrm{s} = x_\mathrm{d}, \quad \text{for a vertical movement.}$$

In Fig. 13 below, we present an example set of movements, including [(2, 4); (3, 4)], a horizontal movement, and [(7, 6), (7, 5)], a vertical movement.

The vertex set $V(T)$ of the transition graph T is the set of all movements that must be performed during a virtual clock cycle. The set $E(T)$ of edges of T consists of all pairs (u, v), u, $v \in V(T)$, such that the corresponding movements cannot be performed in the same real clock cycle of a given virtual clock cycle.

For an arbitrary graph G, a (legal) vertex coloring of the vertex set $V(G)$ is an assignment $F : V(G) \rightarrow C$, where C is a finite set called a color set, such that no two adjacent vertices are colored the same color. Usually, C is a set of nonnegative integers $\{0, 1, 2, \ldots\}$. The chromatic number $\chi(G)$ is the smallest number of colors needed to legally color the vertices of G. In the context of the transition graph T, the set of vertices with the same color correspond to a set of movements that can be performed simultaneously. Thus, the chromatic number $\chi(T)$ is the smallest number of real clock cycles in which all movements of the current virtual clock cycle can be performed.



(a)

(b)

Fig. 13. (a) Grid of control wires indicating droplets with horizontal and vertical movements. (b) Corresponding transition graph for droplet movements.

Let $m_1 = [(x_s^1, y_s^1); (x_d^1, y_d^1)]$ and $m_2 = [(x_s^2, y_s^2); (x_d^2, y_d^2)]$ be two vertices of T. Then, m_1 and m_2 are adjacent, $(m_1, m_2) \in E(T)$, iff there exists some vertex $v = [(x_s^v, y_s^v); (x_d^v, y_d^v)]$, where (x_s^v, y_s^v) may be the same as (x_d^v, y_d^v) and v may be m_1 or m_2 such that one of the following holds:

1) $|x_d^1 - x_s^v| \leq 1$ and $|y_d^2 - y_s^v| \leq 1$, and (x_d^1, y_d^2) is not (x_d^v, y_d^v);
2) $|x_d^2 - x_s^v| \leq 1$ and $|y_d^1 - y_s^v| \leq 1$, and (x_d^2, y_d^1) is not (x_d^v, y_d^v);
3) $|x_d^1 - x_d^v| \leq 1$ and $|y_d^2 - y_d^v| \leq 1$, and (x_d^1, y_d^2) is not (x_d^v, y_d^v);
4) $|x_d^2 - x_d^v| \leq 1$ and $|y_d^1 - y_d^v| \leq 1$, and (x_d^2, y_d^1) is not (x_d^v, y_d^v).

Briefly, when two droplets move simultaneously, four electrodes are activated (unless both droplets have the same row or column as their destination). Two of these electrodes perform the desired droplet movements, but the other two can cause unwanted droplet movement. These conditions check if that is the case (see Fig. 14).

C. Coloring Algorithm

We now describe Algorithm 1 that can be used for coloring the transition graph T. We use a heuristic approach for this.

```
Algorithm 1 Color
    Input: T // The input graph
    Output: F // The output coloring assignment
    $c = 0$ // Color index
    while $V(T) \neq \emptyset$ do
        $M \leftarrow V(T)$
        while $M \neq \emptyset$ do
            pick random vertex $v \in M$
            for all $u =$ neighbor($v$) do
                $M = M \backslash u$
            end for
            $M = M \backslash v$
            $T = T \backslash v$
            $F(v) = c$
        end while
        $c = c + 1$
    end while
    return
```

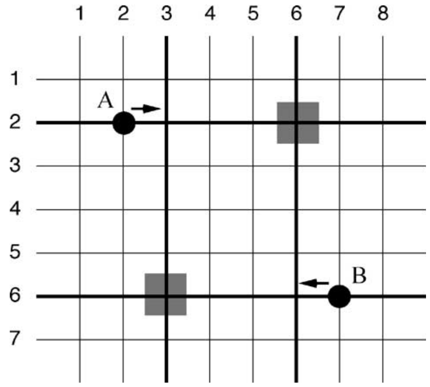


Fig. 14. Small grid of control wires with two droplets to be moved. Droplet A must move to the right and droplet B must move to the left. Actuating them simultaneously will also activate the electrodes marked with gray squares. If these electrodes cause undesired droplet movement, then droplets A and B interfere with each other.

The above procedure takes $O(|V|^3)$ time in the worst case, where $|V|$ is the number of vertices in T.

See Table III for a summary of the number of cycles taken by each addressing scheme. The number of real cycles for the simple scheme depends on the number of droplets on the array, while the number of real cycles for the graph-coloring scheme depends on the connectivity of the transition graph. The stability behavior of the system remains the same under these addressing schemes.

VI. System-Application Scenarios

In this section, we discuss two scenarios that our system is capable of handling. The first scenario deals with adjusting the concentration levels of the droplets being used on the array. The second scenario describes an approach to use a minimal layout for glucose assays.

A. Dilution Control

Having the ability to dilute chemicals on chip is useful for improving the sensitivity and accuracy of bioanalyte detection [39]. Fair *et al.* [14] describe an interpolating serial-dilution scheme. Each exponential dilution step mixes a unit-volume chemical droplet with a unit-volume buffer droplet to obtain two-unit-volume droplets of half the concentration. Each interpolation step combines unit-volume droplets of concentrations

TABLE III
COMPARISON OF THE EFFICIENCY OF THREE ADDRESSING SCHEMES FOR A 2 × 2 TILE-LAYOUT SIMULATION OF THE PCR ANALYSIS

Addressing Scheme	Virtual Cycles Completed	Real Clock Cycles Taken
Direct Addressing	1,000,000	1,000,000
Simple Row-Column	1,000,000	39,579,750
Coloring-based Row-Column	1,000,000	10,035,243

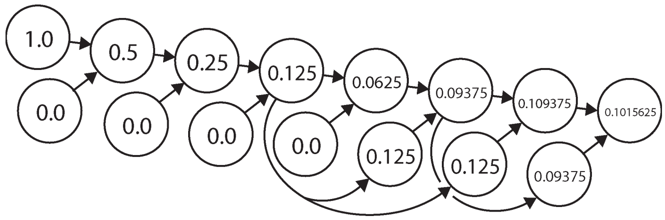


Fig. 15. An example mixing graph for dilution control. The scheme assumes that droplets of a specified concentration level are given and that buffer droplets of 0% concentration are available. Any desired reduced concentration can be achieved; our approach is to identify the intermediate droplet concentrations through a binary search strategy. Here, the concentration is reduced to approximately 10% of its original level.

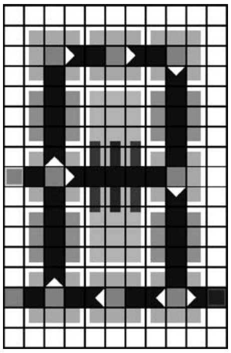


Fig. 16. 11 × 17-array layout for sample preparation for glucose assay.

C_1 and C_2 to obtain two droplets of concentration $(C_1 + C_2)/2$. In principle, a droplet with an arbitrary dilution level can be created through a sequence of interpolating and exponential-dilution steps.

We have implemented an algorithm for automated-droplet-dilution control. We associate a concentration level with each droplet type the system is to process. If a droplet of a particular type and concentration is specified as an input to the system, and a mixing operation is specified that takes that droplet type but with a lower concentration as input, then the system will recognize that the input droplet needs to be diluted. A set of mixing operations to create the desired concentration is computed by applying Algorithm 2, which is based on a binary search strategy. To facilitate the dilution, two special droplet types are introduced. The first, a buffer droplet, has a concentration level of 0 and can be used to reduce the concentration of any droplet it mixes with by half. The second is a waste droplet; any unwanted, extra droplets produced by the dilution process that are to be discarded are designated as waste droplets. Once the set of mixing operations M has been computed, droplets of matching concentrations can be linked together in a mixing graph, by comparing the input and output concentrations of pairs of operations. See the example graph in Fig. 15.

Algorithm 2 Droplet Dilution

```
Input: $d_i$, $d_b$ // Input droplet type with known concentration
                // and the buffer droplet type
       $c$ // Desired concentration level.
       $tol$ // The tolerance within which concentrations
             // are considered equal
Output: $M$ // Set of mixing operations $\{((d_j, d_k) \rightarrow (d_{jk}^{mix_1}, d_{jk}^{mix_2})\}$ that yield concentration $c$.
$D \leftarrow \{d_i, d_b\}$ // Initializing $D$, set of droplets of varying
                // concentrations available for mixing
$M \leftarrow \emptyset$
$range \leftarrow$ Concentration$(d_i)$ − Concentration$(d_b)$
$d_H \leftarrow d_i$ // $d_H$ is upper bound for concentration
$d_L \leftarrow d_b$ // $d_L$ is lower bound for concentration
while $range > tol$ do
    for all $d_l$, $d_\mathrm{h} \in D$ do
        if Concentration$(d_l) < c$ and Concentration$(d_\mathrm{h}) > c$
        then
            if Concentration$(d_\mathrm{h})$ − Concentration$(d_l) < range$
            then
                $range \leftarrow$ Concentration$(d_\mathrm{h})$ − Concentration$(d_l)$
                $d_H \leftarrow d_\mathrm{h}$
                $d_L \leftarrow d_l$
            end if
        end if
    end for
    $m \leftarrow ((d_H, d_L) \rightarrow (d_{HL}, d_w))$ // $d_w$ is identical to $d_{HL}$
            // but designated a waste droplet
    $M \leftarrow M \bigcup m$
    $D \leftarrow D \bigcup d_{HL}$
end while
return $M$
```

B. Minimalistic Layout for Glucose Assays

Experimentally demonstrated DMFSs range in size from small electrowetting arrays (for example, 5 × 5 cells [15]) to large dielectrophoresis arrays (for example, 320 × 320 cells [26]). The layouts we described above for our system are intermediate in size. We can also create a small layout of 11 × 17 cells (Fig. 16), comparable in size to existing electrowetting-based arrays [17]. These small layouts are most appropriate for simple reactions that require only a small number of droplet types.

Srinivasan *et al.* [40] describe the use of a prototype DMFS for glucose assays in a variety of biological fluids. They mix sample droplets and reagent droplets in the system to dilute the sample. After splitting, one resulting droplet is discarded as waste and the other is sent to an on-chip concentration-detection cell. We have successfully simulated the sample-preparation phase of this glucose assay using the minimal 11 × 17 layout in Fig. 16. Currently, we assume that the diluted samples are sent off-chip for glucose-concentration sensing; an optical-sensor component can be easily incorporated into the layout, in the work area or at the sink intersection. This glucose-assay example, along with the PCR example, demonstrates that our system is highly scalable; it is able to operate successfully on a range of sizes consistent with current experimental systems.

VII. Conclusion

Our approach to creating a general-purpose DMFS, previously described in [18], [19], consists of imposing a virtual layout of components on the planar array and coordinating the motions of droplets by developing decentralized-routing algorithms. The system can perform real-time droplet manipulation, and can be easily used to act as a controller for a physical array. The same array can perform a variety of chemical analyses including the DNA PCR and glucose assays, and can even perform multiple analyses in parallel.

In this paper, we enhanced the original system in a number of ways for greater versatility and performance. These included support for new layout schemes, routing algorithms, and increased spacing between droplets, and characterization of their effects on system performance. We found the system relatively stable to these variations, which implies the overall design is relatively robust. We then considered DMFS arrays with hardware limited row-column addressing and developed a polynomial-time graph coloring algorithm for the problem of droplet coordination under such hardware limitations. We demonstrated the capabilities of our system on example scenarios, including dilution control and minimalist layouts.

There are several directions for future work. Identifying the minimum number of steps to execute a set of droplet movements under limited row–column addressing is an open problem that we are working on using the graph-coloring approach. The overall design of the components and the system allows for the introduction of new component types, such as droplet-heater components, for example. Automatically generating the optimal layout for a given analysis requires methods for optimizing the number of tiles and their arrangement, as well as the locations of the sinks and sources on the array. Modeling the system as a network can potentially provide insights into changes to the array design and improve system performance. The design and control of dynamically reconfigurable layouts, where any part of the array may be reallocated for any desired operation, pose particularly interesting challenges. Developing layouts that can adapt to electrode failures is another direction that will lead to robust systems.

Acknowledgment

The authors extend many thanks to K. Böhringer for introducing them to this problem and providing encouragement and advice.

References

[1] S. Akella and S. Hutchinson, "Coordinating the motions of multiple robots with specified trajectories," in *IEEE Int. Conf. Robotics and Automation*, Washington, DC, May 2002, pp. 624–631.

[2] D. P. Bertsekas and R. G. Gallagher, *Data Networks*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[3] A. Bicchi and L. Pallottino, "On optimal cooperative conflict resolution for air traffic management systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 4, pp. 221–231, Dec. 2000.

[4] K.-F. Böhringer, "Optimal strategies for moving droplets in digital microfluidic systems," in *Proc. 7th Int. Conf. Micro Total Analysis Systems (MicroTAS)*, Squaw Valley, CA, Oct. 2003, pp. 591–594.

[5] ——, "Towards optimal strategies for moving droplets in digital microfluidic systems," in *Proc. IEEE Int. Conf. Robotics and Automation*, New Orleans, LA, Apr. 2004, pp. 1468–1474.

[6] J. Brassil and R. Cruz, "Nonuniform traffic in the Manhattan street network," in *Proc. IEEE Int. Conf. Communications (ICC)*, Denver, CO, Jun. 1991, pp. 1647–1651.

[7] C. Busch, M. Herlihy, and R. Wattenhofer, "Hard-potato routing," in *Proc. 32nd Annu. ACM Symp. Theory Computing (STOC)*, Portland, OR, May 2000, pp. 278–285.

[8] S. K. Cho, H. Moon, and C.-J. Kim, "Creating, transporting, cutting, and merging liquid droplets by electrowetting-based actuation for digital microfluidic circuits," *J. Microelectromech. Syst.*, vol. 12, no. 1, pp. 70–80, Feb. 2003.

[9] A. K. Choudhury and V. O. K. Li, "An approximate analysis of the performance of deflection routing in regular networks," *IEEE J. Sel. Areas Commun.*, vol. 11, no. 8, pp. 1302–1316, Oct. 1993.

[10] A. A. Desrochers, *Modeling and Control of Automated Manufacturing Systems*. Washington, DC: IEEE Comput. Soc., 1990.

[11] J. Ding, K. Chakrabarty, and R. B. Fair, "Scheduling of microfluidic operations for reconfigurable two-dimensional electrowetting arrays," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 20, no. 12, pp. 1463–1468, Dec. 2001.

[12] D. Eppstein, "Improved algorithms for 3-coloring, 3-edge-coloring, and constraint satisfaction," in *Proc. 12th Symp. Discrete Algorithms*, Washington, DC, New York: ACM and SIAM, Jan. 2001, pp. 329–337.

[13] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," *Algorithmica*, vol. 2, no. 4, pp. 477–521, 1987.

[14] R. B. Fair, V. Srinivasan, H. Ren, P. Paik, V. Pamula, and M. G. Pollack, "Electrowetting-based on-chip sample processing for integrated microfluidics," in *Proc. IEEE Int. Electron Devices Meeting (IEDM)*, Washington, DC, 2003, pp. 32.5.1–32.5.4.

[15] S.-K. Fan, C. Hashi, and C.-J. Kim, "Manipulation of multiple droplets on N × M grid by cross-reference EWOD driving scheme and pressure-contact packaging," in *Proc. IEEE Conf. Microelectro Mechanical Systems (MEMS)*, Kyoto, Japan, Jan. 2003, pp. 694–697.

[16] A. Fuchs, N. Manaresi, D. Freida, L. Altomare, C. L. Villiers, G. Medoro, A. Romani, I. Chartier, C. Bory, M. Tartagni, P. N. Marche, F. Chatelain, and R. Guerrieri, "A microelectronic chip opens new fields in rare cell population analysis and individual cell biology," in *Proc. 7th Int. Conf. Micro Total Analysis Systems (MicroTAS)*, Squaw Valley, CA, Oct. 2003, pp. 911–914.

[17] J. Gong, S.-K. Fan, and C.-J. Kim, "Portable digital microfluidics platform with active but disposable lab-on-chip," in *Tech. Dig. 17th IEEE Int. Conf. Microelectro Mechanical Systems (MEMS)*, Maastricht, The Netherlands, Jan. 2004, pp. 355–358.

[18] E. Griffith and S. Akella, "Coordinating multiple droplets in planar array digital microfluidics systems," in *Algorithmic Foundations of Robotics VI*, M. Erdmann, D. Hsu, M. Overmars, and A. F. van der Stappen, Eds. Berlin, Germany: Springer-Verlag, 2005, pp. 219–234.

[19] ——, "Coordinating multiple droplets in planar array digital microfluidic systems," *Int. J. Rob. Res.*, vol. 24, no. 11, pp. 933–949, Nov. 2005.

[20] D. Gross and C. M. Harris, *Fundamentals of Queueing Theory*, 3rd ed. New York: Wiley, 1998.

[21] J. E. Hopcroft, J. T. Schwartz, and M. Sharir, "On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the 'warehouseman's problem'," *Int. J. Rob. Res.*, vol. 3, no. 4, pp. 76–88, 1984.

[22] T. B. Jones, M. Gunji, M. Washizu, and M. J. Feldman, "Dielectrophoretic liquid actuation and nanodroplet formation," *J. Appl. Phys.*, vol. 89, no. 2, pp. 1441–1448, Jan. 2001.

[23] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, no. 3, pp. 237–252, Mar. 1998.

[24] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Trans. Robot. Autom.*, vol. 14, no. 6, pp. 912–925, Dec. 1998.

[25] M. A. Lawley, "Deadlock avoidance for production systems with flexible routing," *IEEE Trans. Robot. Autom.*, vol. 15, no. 3, pp. 497–509, Jun. 1999.

[26] N. Manaresi, A. Romani, G. Medoro, L. Altomare, A. Leonardi, M. Tartagni, and R. Guerrieri, "A CMOS chip for individual cell manipulation and detection," *IEEE J. Solid-State Circuits*, vol. 38, no. 12, pp. 2297–2305, Dec. 2003.

[27] C. Maxfield, *The Design Warrior's Guide to FPGAs: Devices, Tools, and Flows*. Burlington, MA: Elsevier, 2004.

[28] P. A. O'Donnell and T. Lozano-Perez, "Deadlock-free and collision-free coordination of two robot manipulators," in *Proc. IEEE Int. Conf. Robotics and Automation*, Scottsdale, AZ, May 1989, pp. 484–489.

[29] P. Paik, V. K. Pamula, and R. B. Fair, "Rapid droplet mixers for digital microfluidic systems," *Lab Chip*, vol. 3, no. 4, pp. 253–259, 2003.

[30] J. Peng and S. Akella, "Coordinating multiple robots with kinodynamic constraints along specified paths," in *Algorithmic Foundations of Robotics V*, J.-D. Boissonnat, J. Burdick, K. Goldberg, and S. Hutchinson, Eds. Heidelberg, Germany: Springer-Verlag, 2003, pp. 221–237.

[31] M. G. Pollack, R. B. Fair, and A. D. Shenderov, "Electrowetting-based actuation of liquid droplets for microfluidic applications," *Appl. Phys. Lett.*, vol. 77, no. 11, pp. 1725–1726, Sep. 2000.

[32] M. G. Pollack, P. Y. Paik, A. D. Shenderov, V. K. Pamula, F. S. Dietrich, and R. B. Fair, "Investigation of electrowetting-based microfluidics for real-time PCR applications," in *Proc. 7th Int. Conf. Miniaturized Chemical and Biochemical Analysis Systems (MicroTAS)*, Squaw Valley, CA, Oct. 2003, pp. 619–622.

[33] S. A. Reveliotis, M. A. Lawley, and P. M. Ferreira, "Polynomial-complexity deadlock avoidance policies for sequential resource allocation systems," *IEEE Trans. Autom. Control*, vol. 42, no. 10, pp. 1344–1357, Oct. 1997.

[34] A. A. Rizzi, J. Gowdy, and R. L. Hollis, "Distributed coordination in modular precision assembly systems," *Int. J. Rob. Res.*, vol. 20, no. 10, pp. 819–838, Oct. 2001.

[35] G. Sanchez and J. Latombe, "On delaying collision checking in PRM planning—Application to multi-robot coordination," *Int. J. Rob. Res.*, vol. 21, no. 1, pp. 5–26, Jan. 2002.

[36] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *Proc. Eur. Control Conf.*, Porto, Portugal, 2001, pp. 2603–2608.

[37] T. Simeon, S. Leroy, and J.-P. Laumond, "Path coordination for multiple mobile robots: A resolution-complete algorithm," *IEEE Trans. Robot. Autom.*, vol. 18, no. 1, pp. 42–49, Feb. 2002.

[38] S. S. Skiena, *The Algorithm Design Manual*. New York: Springer-Verlag, 1998.

[39] V. Srinivasan, V. Pamula, M. Pollack, and R. Fair, "A digital microfluidic biosensor for multianalyte detection," in *Proc. IEEE 16th Annu. Int. Conf. Microelectro Mechanical Systems (MEMS)*, 2003, pp. 327–330.

[40] V. Srinivasan, V. K. Pamula, and R. B. Fair, "An integrated digital microfluidic lab-on-a-chip for clinical diagnostics on human physiological fluids," *Lab Chip*, vol. 4, no. 4, pp. 310–315, Aug. 2004.

[41] F. Su and K. Chakrabarty, "Architectural-level synthesis of digital microfluidics-based biochips," in *Proc. IEEE Int. Conf. Computer Aided Design (ICCAD)*, San Jose, CA, 2004, pp. 223–228.

[42] A. S. Tanenbaum, *Computer Networks*, 3rd ed. Upper Saddle River, NJ: Prentice-Hall, 1996.

[43] C. Tomlin, G. J. Pappas, and S. Sastry, "Conflict resolution for air traffic management: A study in multi-agent hybrid systems," *IEEE Trans. Autom. Control*, vol. 43, no. 4, pp. 509–521, Apr. 1998.

[44] P. Švestka and M. Overmars, "Coordinated path planning for multiple robots," *Robot. Auton. Syst.*, vol. 23, no. 3, pp. 125–152, Apr. 1998.

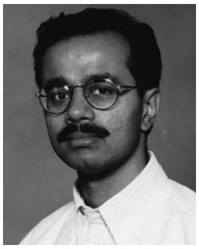
[45] D. B. West, *Introduction to Graph Theory*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2001.

[46] A. R. Wheeler, H. Moon, C.-J. C. Kim, J. A. Loo, and R. L. Garrell, "Electrowetting-based microfluidics for analysis of peptides and proteins by matrix-assisted laser desorption/ionization mass spectrometry," *Analytical Chem.*, vol. 76, no. 16, pp. 4833–4838, Aug. 2004.

[47] T. Zhang, K. Chakrabarty, and R. B. Fair, *Microelectrofluidic Systems: Modeling and Simulation*. Boca Raton, FL: CRC Press, 2002.

Eric J. Griffith received the M.S. degree in computer science from Rensselaer Polytechnic Institute, Troy, NY, in 2004. He is currently pursuing the Ph.D. degree in computer science in the Data Visualization Group, Delft University of Technology, Delft, The Netherlands.

He is interested in research on visualization of large time-dependent data sets in virtual reality. He is also interested in motion-planning and control algorithms for digital microfluidic systems (DMFS).



Srinivas Akella (S'90–A'95–M'96) received the B.Tech. degree in mechanical engineering from the Indian Institute of Technology, Madras, India, in 1989 and the Ph.D. degree in robotics from the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, in 1996.

He was a Beckman Fellow at the Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign, from 1996 to 1999. He is currently an Assistant Professor in the Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY. His research interests are in robotic manipulation and motion planning, and in particular, developing algorithms for automation, design, and bioinformatics applications.

Dr. Akella is a recipient of the NSF Career Award, and the National Talent Search Scholarship from the Government of India.

Mark K. Goldberg received the Ph.D. degree in theoretical computer science from the Institute of Mathematics, Novosibirsk, Russia, in 1969.

He worked as an Applied Mathematician till his emigration to the USA in 1979. He held positions at the University of Waterloo, ON, Canada, University of Toronto, Canada, State University of South Carolina, and Clarkson University before he joined the faculty of the Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY, in 1985. He has authored more that 70 publications. His research interests involve applications of graph theory and algorithm design.