

Coordinating Multiple Droplets in Planar Array Digital Microfluidics Systems

Eric Griffith and Srinivas Akella

Rensselaer Polytechnic Institute, Troy, New York 12180, USA

Abstract. This paper presents an approach to coordinate the motions of droplets in digital microfluidic systems used for biochemical analysis. A digital microfluidic system typically consists of a planar array of cells with electrodes that control the droplets. The primary challenge in using droplet based systems is that they require the simultaneous coordination of a potentially large number of droplets on the array as the droplets move, mix, and split. This paper describes a general-purpose system that uses simple algorithms and yet is versatile. First, a semi-automated approach to generate the array layout in terms of components is explained. Next, simple algorithms to select destination components for the droplets and a decentralized scheme for components to route the droplets on the array are discussed. These are then combined into a reconfigurable system that has been simulated in software to perform DNA polymerase chain reaction and other analyses. The algorithms have been able to successfully coordinate hundreds of droplets simultaneously and perform one or more chemical analyses in parallel. Since it is challenging to analytically characterize the behavior of such systems, methods to detect potential instabilities are proposed.

1 Introduction

The field of microfluidics has been revolutionized recently by the creation of miniature biochemical analysis systems using microfabrication technology. These systems are often termed *micro total analysis systems* or “lab on a chip” systems. These systems offer a number of advantages, including size reduction, power reduction, and increased reliability. However, current systems are typically tailored to a specific task. Therefore an important goal is to create reconfigurable and reprogrammable systems capable of handling a variety of analysis tasks.

Digital microfluidics systems (DMFS) that use techniques such as electrowetting and dielectrophoresis are promising candidates for reconfigurable systems ([21], [14],[19], [7]). We focus on microfluidic systems that manipulate discrete droplets by electrowetting, where the interfacial tension of the droplets is modulated with a voltage [19]. Droplets are microliters in volume, and have been moved at 12–25 cm/sec on planar arrays of 0.15 cm wide electrodes. (See [10] and [7] for details.) The ability to control individual droplets on a planar array enables complex analysis operations to be performed in biochemical “lab-on-a-chip” systems (Figure 1). For example, they can be used to perform polymerase chain reactions for DNA sequence analysis. For many

biochemical analysis operations, no special purpose devices are required aside from the array itself. These systems have the potential to process hundreds of samples quickly. The primary challenge of using droplet based systems is developing algorithms for the simultaneous coordination of a potentially large number of droplets. Planning optimal paths through the array for each droplet would be computationally intractable for a large number of droplets.

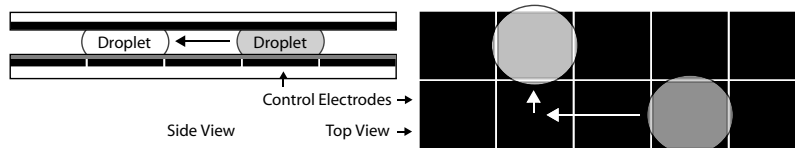


Fig. 1. Droplets on an electrowetting array (side and top views). Droplet motion is controlled by turning the control electrodes on and off. Based on [19].

This paper describes an approach to creating a general-purpose DMFS. First, a semi-automated approach to design the array layout in terms of modular components is explained, along with the motivating design choices. Next, simple algorithms to select destination components for the droplets and a decentralized scheme for components to route the droplets are discussed. These are then combined into a reconfigurable system that has been simulated in software to perform DNA polymerase chain reaction and other analyses. The algorithms have been able to successfully coordinate hundreds of droplets simultaneously and perform one or more chemical analyses in parallel. Since it is challenging to analytically characterize the behavior of such systems, methods to detect potential instabilities due to congestion are proposed.

2 Related Work

Digital Microfluidic Systems: Pollack, Fair, and Shenderov [21] demonstrated rapid manipulation of discrete microdroplets by electrowetting-based actuation. Ding, Chakrabarty, and Fair [8] 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 linear programming formulation to minimize completion time. Droplet paths and areas on the array for storage, mixing and splitting operations are predefined by a human. Fair et al. [10] describe experiments on injection, dispensing, dilution, and mixing of samples in an electrowetting DMFS. Cho, Moon, and Kim [7] 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. Fan, Hashi, and Kim [11] developed a cross-reference grid of single layer electrodes to manipulate droplets with limited row-column addressing.

Paik, Pamula, and Fair [19] studied the effects of droplet aspect ratios and mixing strategies on the rate of droplet mixing. Jones et al. [14] demonstrated dielectrophoresis based liquid actuation and nanodroplet formation.

Multiple robot coordination: The coordination of droplets in a DMFS is closely related to multiple robot motion coordination. However very few motion planning algorithms can guaranteeably coordinate more than 10 or 20 robots. Each droplet in a DMFS can be viewed as a simple robot that moves on a 4-connected array (Bohringer [4,5]). Bohringer outlines 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 ([9]). However note that in a DMFS, the droplets must additionally sometimes combine for different durations to mix, and then be split.

Hopcroft, Schwartz, and Sharir [13] showed that even a simplified two-dimensional case of motion planning for multiple robots is PSPACE-hard. Reif and Sharir [22] and Kant and Zucker [15] developed techniques to plan motions of a single robot among moving obstacles. This can be generalized to obtain a heuristic solution for planning the motions of multiple robots. Erdmann and Lozano-Perez [9] order robots by assigned priority and sequentially search for collision-free paths. Recent efforts have focused on probabilistic approaches. Svestka and Overmars [26] developed a PRM planner for path coordination of multiple car-like robots. Sanchez and Latombe [23] use lazy PRM variants for coordinated path planning of multiple robot arms.

When the paths of the robots are specified, a path coordination problem, first studied by O'Donnell and Lozano-Perez [18] for two robots, arises. LaValle and Hutchinson addressed a similar problem in [16] where each robot was constrained to a C-space roadmap during its motion. Simeon, Leroy, and Laumond [25] coordinate over 100 car-like 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 10–20 robots by changing robot start times. Peng and Akella [20] 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 (Tomlin, Pappas, and Sastry [28], Bicchi and Pallottino [3], Schouwenaars et al. [24]) is also closely related to multiple robot coordination.

3 Components for Array Layout

Many common operations for biochemical analyses can be performed on an array without additional special purpose hardware. These operations include: dispensing droplets onto the array, collecting droplets from the array, transporting droplets around the array, mixing droplets together, and splitting

droplets apart. The array layout design presented here uses a system of “virtual” components. Each component type is responsible for performing one or more types of operation. Multiple instances of the same type of component can be present, and each component instance is allotted an area of the array to perform its operations in. By linking components that can perform all the operations in an analysis, a DMFS can be created to perform that analysis.

In our system, we have defined six component types. The *street*, *connector*, and *intersection* components transport droplets around the array. The *source* component adds droplets to the array, and the *sink* component removes droplets from the array. The *work area* component manages mixing and splitting of droplets. New component types can be defined and integrated into the system for operations that do not require special purpose hardware.

3.1 The Components

This section provides an overview of the functionality of each component type. Each component is responsible for managing all of the droplets that are in its portion of the array. At each clock cycle, components attempt to move all droplets in their section of the array. Each component maintains connections with its neighboring components, which are used to pass control of droplets when they move between components. The connections are entrance/exit pairs where the exit from one component is the entrance into another. Components may have to wait to move their droplets until the components they are connected to have moved theirs. Figure 2 shows an example system with each of the component types present. The details of the layout itself are presented in Section 5. Figure 3 depicts the individual components.

The Street Component: The street component is the general-purpose droplet transportation component. It moves droplets in one direction through at least two array cells. Streets are one-way to prevent two droplets from moving in opposite directions through the component. A street attempts to advance all droplets within it in synchrony at each clock cycle. If moving any droplet would cause it to be adjacent to another, it is not allowed to move.

The Connector Component: The connector component is similar to a street component, except that a droplet only moves through a single cell in it. The distinction is made because droplets in the connector are adjacent to two components simultaneously. Thus, when a droplet attempts to enter a connector, the connector must ensure that there is no other droplet adjacent to the connector.

The Intersection Component: The intersection component is vital to the droplet routing system. Droplets enter an intersection and then move into its middle cell. Once there, the intersection routes the droplet to the appropriate exit based on the algorithm described in Section 4.3.

The Work Area Component: The work area component is where mixing and splitting take place. Each work area has a transit area and multiple *work units*. Each work unit may function as a *mixer* and/or as a *splitter*. Mixers

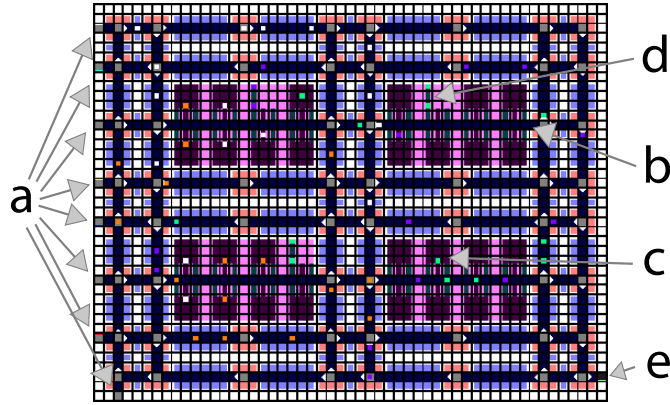


Fig. 2. Array layout for the PCR reaction described in Section 7. On the left side of the array are **(a)** eight sources, which supply the input sample droplets to the system. There are **(b)** four work areas on the array, in which droplets are **(c)** mixed together and **(d)** split apart. In the lower right corner of the array is a **(e)** sink, which collects the droplets of the final products and moves them off the array.

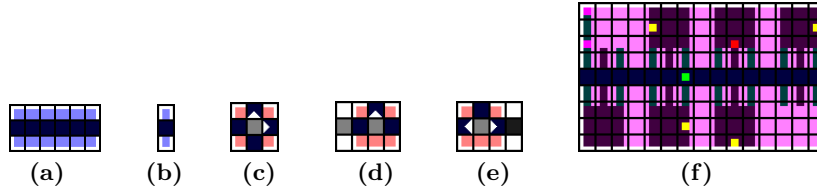


Fig. 3. The components. **(a)** A street component. Droplets move through the dark blue squares. **(b)** A connector component. **(c)** An intersection component with two entrances and two exits. The arrowheads indicate the exits. **(d)** A source component connected to an intersection component. **(e)** An intersection component connected to a sink component. **(f)** An active work area, showing several work units with droplets. Each work unit can function as a mixer and as a splitter.

merge two droplets into a new droplet. The mixer moves the droplet around to speed up mixing, as in [19]. Splitters split the mixed droplet into two droplets. A work unit may be used as a mixer until a mixing operation is completed, and then part of that same area may be used as a splitter. A work area can mix and split multiple droplets at the same time. When a droplet gets to a work area, the work area sends it to a specific work unit. The work units manage the droplets assigned to them. Once a mix and split operation is complete, the resulting droplets are sent out of the work area.

The Source Component: The source component represents droplet entry points into the array. Each source introduces specified droplet types at specified intervals. Droplets entering the array are assigned a goal operation.

The Sink Component: The sink component represents droplet exit points from the array. Each sink removes specified droplet types from the array.

4 Droplet Destination Selection and Routing Algorithms

To be practical, a DMFS should be able to handle a large number of droplets simultaneously. Ideally, it should do this in a way that optimizes some quantity, such as throughput or completion time. We significantly reduce the computational cost of planning the droplet motions at the expense of optimality. By dividing the array into components, restrictions are placed on where operations can take place on the array. The interconnection of the components can be viewed as a network with the intersections as the routing devices and the streets and connectors as the “wires.” This reduces the motion planning problem to a network routing problem on a directed graph.

Selecting destinations for droplets is a two-fold process. First, the droplet must be assigned an operation to perform. Once the droplet has an assigned operation, a component for that operation may be selected. The methods for these processes, as well as the droplet routing, are discussed in this section.

4.1 Assumptions

We assume the following when modeling the operations of the DMFS:

1. Individual cells of the array are addressable, permitting direct control of individual droplets.
2. No two droplets can occupy adjacent cells unless they are to be mixed.
3. Droplets in the array have identical volumes, except during mixing.
4. Every mix operation is followed by a split operation.
5. Mixing operations take longer to perform than transport operations.
6. All the operations can be performed on the array without additional special purpose hardware.

4.2 Droplet Destination Selection

The system is supplied with parameters, described in Section 5, which it uses to maintain a list of components available for certain operations. Work areas can perform a mixing operation with any droplet type, and sinks remove specific types of droplets. Each work area and sink adds itself to a sorted list of components accepting droplets for operations. There is also a sorted list of higher priority containing requests from components for specific droplet types required to complete an operation. Currently, only work areas needing one of two droplet types for a mixing operation place requests in this list.

When a new droplet enters the system, or is created through a mixing operation, the corresponding source or work area assigns it an operation to perform. When the droplet enters an intersection, the intersection tries to find a component to send the droplet to. First, it checks the higher priority list for existing requests for the droplet’s type and operation to perform. If

any exist, then the droplet is assigned to the first requesting component and that request is removed from the list. Failing that, it is assigned to the first component that can accept droplets of its type, and that component is sent to the end of the list (if it is a sink or is a work area with work units available). If no components are available, then the next intersection the droplet enters attempts to assign it a destination. In the worst case scenario, this operation takes $O(w+s)$ time for one droplet at one intersection, where w is the number of work areas and s is the number of sinks in the system.

4.3 Droplet Routing

After a droplet is assigned a destination component, the droplet must be routed to the component. The routing method we use can be viewed as a deflection routing variant of the Open Shortest Path First (OSPF) network protocol ([6], [27]). The routing method relies on each intersection maintaining shortest path information to the other components, computed from the component graph. The *component graph* is a directed graph where each node is a component and each edge is a connection between adjacent components. The directed edge points from the component with the exit to the component with the entrance. The distance along an edge is, generically, taken to be the length of the component containing the exit for the edge.

When the system is initialized, each intersection computes and stores its routing table, which maps the shortest path to each component to a corresponding exit to take from the intersection. The specified exit is the first leg of the shortest path to the component in question. Each intersection constructs its routing table by running Dijkstra's algorithm on the component graph to compute shortest paths from the intersection. The one restriction is that a shortest path should not travel through work areas, sources, or sinks, unless those components are the destination. Dijkstra's algorithm on a sparse graph using a standard implementation has a runtime of $O(n \log n)$ where n is the number of nodes in the graph. Since Dijkstra's algorithm must be run from each of the intersections, the routing system has an initial overhead of $O(i \cdot n \log n)$ where i is the number of intersections in the array.

At each clock cycle, the intersections are processed in a fixed (random) order to select their droplet routing moves. Subsequently, synchronous motion of droplets is executed. The droplet routing is straightforward once the routing table has been constructed. 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. Otherwise, the intersection finds the destination component in its routing table and selects the exit that is the best choice for the droplet. 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. The amount of time required for selecting an exit for a droplet is $O(1)$ because the in-

tersection must check at most three possible exits and the routing table has constant access time.

5 A General-Purpose Digital Microfluidics System

A general-purpose DMFS can be created by combining the component based layout design approach (Section 3) with the accompanying algorithms (Section 4). A general-purpose layout must be capable of handling arbitrary analyses that require the movement, mixing, and splitting of different types of droplets. Further, the layout should contain a sufficient number of work areas and be able to efficiently transport droplets around the array.

Efficient droplet transportation is crucial because the usage of the system is not known in advance. Therefore, all parts of the array should be accessible. We group street components in pairs to simulate two-way streets, allowing droplets to closely follow the same path between two locations in both directions. To allow for intersections between these two-way streets, we group four intersections together in a rotary-like arrangement. See Figure 4.

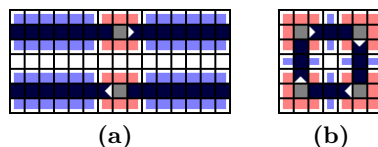


Fig. 4. Simulating two-way transportation: (a) Two-way street (b) Rotary.

We group these two-way streets and rotaries with a work area to form a pattern of components. This pattern, shown in Figure 5, is designed to provide a balance between number of work areas and ease of access. The array layout itself is a periodic tiling of this pattern, completed with an alternating sequence of rotaries and streets along its upper and right edges. The extra intersection components in the two-way streets are for connecting sources and sinks around the perimeter of the layout. The intersections in the vertical streets are connected to provide shorter paths to and from work areas.

To generate the layout, the user must specify a set of parameters dependent on the hardware. These are the physical size of the array and the locations of sources and sinks. To fully define the system, the user specifies parameters based on the chemical analyses to be performed, including the types of droplets introduced at sources, 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 with the droplets on the array.

The size of the array is limited to be of width $9 + 22i$ and height $9 + 16j$, where the layout is i tiles wide and j tiles tall. The pattern tile is 22 cells

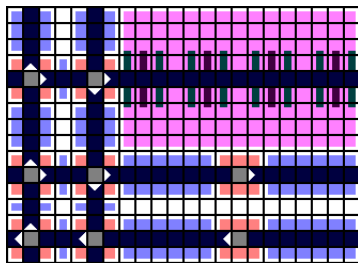


Fig. 5. The layout pattern tile that is a modular building block for the array.

wide and 16 cells tall. The sequence of two-way streets and rotaries that completes the layout adds an additional 7 units. The sources and sinks must be placed around the perimeter of the array such that they can be connected to an intersection. The remaining 2 units come from setting aside one square around all sides of the perimeter of the pattern for sources and sinks. A complete example layout with eight sources and one sink can be seen in Figure 2.

5.1 DMFS Operation

The DMFS can be operated in two modes:

1. **Batch mode:** Here all the droplets necessary for a reaction are input at the source components in one batch. The droplets are coordinated to complete the reaction, and then the next batch of droplets is processed. The droplets are input in a synchronized manner based on when they are required for the reactions. After each mix and split operation, one of the two resulting droplets is sent to a waste output. A reaction performed in batch mode will typically require a smaller number of tiles in the array since the number of droplets in the system is small. One advantage of batch mode is that the droplet routing is almost entirely deterministic, and we can easily analyze the system behavior. Only when no work units are available for a droplet will an intersection component route the droplet randomly.
2. **Continuous mode:** Here the source components input the droplets at a fixed rate. (The rate for each droplet type is specified by the human designer.) One advantage of this mode is that it produces a larger volume of product droplets than the batch mode in the same amount of time, especially when no droplets are discarded as waste droplets. Another advantage is that multiple reactions yielding different products can be processed in parallel to better utilize the array. A potential disadvantage is that system behavior is harder to analyze. We explore this issue, and in particular, the likelihood of system instability, in Section 6.

6 Stability Analysis

The behavior of a general-purpose system changes with the chemical analysis it performs. A system operating in continuous mode may or may not be stable depending on its parameters. 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 ([12]). If a system is not stable, in time it will become heavily congested and may finally become deadlocked. Hence instability is to be avoided.

We have identified a set of conditions for a system to be classified as unstable. At least one of the following two conditions must hold for a system to become congested due to instability. The first is for some droplets to be unable to follow the shortest paths to their destinations. The second condition is for droplets to be unable to be assigned a destination. When either condition is met, it is an indication that the system is not processing droplets fast enough. A system is deadlocked when droplets are unable to reach their destination. Deadlock is a sufficient condition for a system to be unstable.

We treat congestion as an indicator of instability and try to identify conditions that lead to or result from congestion. By selecting system parameters such as input droplet rates and source and sink locations to avoid these conditions, the resulting system will likely be stable. We use two methods to identify these conditions. At the operation level, we analyze a graph of the operations to be performed based on the system parameters. At the component level, we model droplet flow in the system using the component graph.

6.1 Analysis Graph

The (biochemical) *analysis graph* provides an organized representation of the behavior of the system. It is a directed graph, with an input node for each droplet type entering the system, an output node for each droplet type leaving the system, and a mix node for each mixing operation performed in the system. The nodes are connected based on the droplet types they require and produce, and the edges represent transport operations. Each node stores the duration of its operation, and the analysis graph is augmented with additional information (Figure 6).

The first augmentation is the rate at which droplets will enter and leave each node. Droplet rate can be most intuitively described in terms of a source. If a source introduces a particular droplet type once every k cycles, then the rate would be $\frac{1}{k}$ droplets per cycle. Also, there would be $k - 1$ empty cells between each introduced droplet. This rate is propagated through the graph. If these droplets perform a mixing operation, then, on average and assuming no unusual delays, one of these droplets would begin its mixing operation every k cycles. Similarly, one droplet would complete the mixing operation every k cycles.

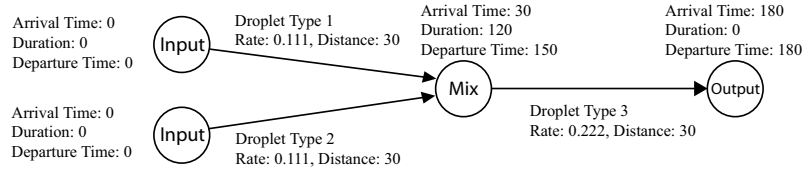


Fig. 6. A simple analysis graph with four nodes: two input nodes, one mix node, and one output node. Droplet types, rates, and expected distances traveled are indicated along the edges. Expected arrival time, duration, and departure time (in units of clock cycles) are indicated at the nodes.

The second augmentation is the best case expected distance to travel between performing operations. Each node's operation can be performed at one or more components on the array; specific components are chosen during destination selection for the droplets. The expected distance is the average of the shortest path between each possible originating component and each possible destination component. In a system with a source component introducing droplets for mixing into a system with two work areas, the distance from the input node to the mix node would be the average of the lengths of the shortest paths from the source to each of the two work areas.

The third augmentation is the best case expected time at which the first droplets will enter each node. There is a separate arrival time from each parent node of a given node, and it is computed as the expected departure time from the parent node plus the best case expected travel time from the parent node to the current node. The expected departure time from a given node is the average of the arrival times, plus the duration of the operation.

The graph is then checked to ensure that the following properties hold:

1. Every path from every source node must lead to a sink node.
2. Droplets must enter a node at the same rate from each parent node.
3. Arrival times to a node should be about the same from each parent node.

If any of these properties is violated, it is likely to result in inefficient processing of droplets. In the event that the properties are violated, the system parameters can be adjusted until they result in a graph without violations.

6.2 Droplet Flow Analysis

The analysis graph provides a reasonable estimate of the overall system stability. However a more detailed component level analysis of system stability may be required. There may be certain bottleneck components that become congested from too many droplets moving through them, and so slow down droplet flow. These bottlenecks can result in an unstable system when they prevent droplets from reaching their destinations. Some simple experiments have demonstrated that this can arise in larger arrays (for example, arrays

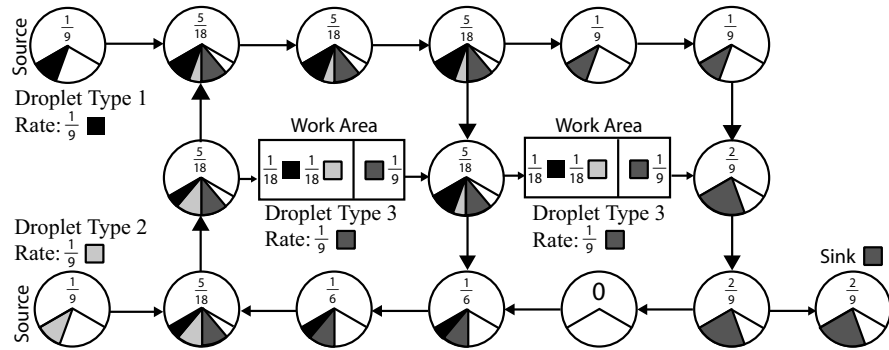


Fig. 7. A component graph for a system with the analysis graph in Figure 6. For clarity, the component graph uses a simplified layout. Each circular node contains a pie chart for flow through the component. The actual flow is the shaded portion of the maximum flow rate of $\frac{1}{3}$ in each node. In this example, inflow is equal to outflow. The total flow through each component is depicted by the fraction in the node. The work areas receive flow from both sources and produce flow destined for the sink. The rate of flow and droplet type (color) produced by each source and work area is indicated by the node.

of size 295×297), where there is a lot of droplet traffic around the perimeter near the sources and sinks but relatively light traffic in the interior. Modeling the droplet flow through the system is an effective tool to identify unstable systems, especially those that are unstable due to these bottleneck, congested components.

The droplet flow modeling attempts to predict the expected flow through the component graph (described in Section 4.3) using the rate information from the analysis graph. (See Figure 7.) The idea is to determine the rate at which droplets enter and leave each component on the array, which is the flow rate through that component, when the system is in a steady state. To approximate the expected flow rate through components, we developed an iterative, network-style analysis. For this analysis, we make the assumption that work areas are uniformly utilized. That is, a droplet being assigned to a work area is equally likely to be assigned to any work area on the array.

Each component is initially assigned inflow and outflow rates of 0. Sources generate a certain amount of flow, dictated by the analysis graph, destined for each work area. For example, if a source produces droplets at a rate of $\frac{1}{4}$ in a system with 2 work areas, it generates a flow of $\frac{1}{8}$ to each work area. Similarly, work areas generate a certain amount of flow destined for each work area and to each sink. At each iteration, the output of each node in the graph is defined as a function of the input to the node. The input of a node is the sum of the outputs, from the previous iteration, of its parent nodes plus any flow it generates at that iteration. For intersections, the input flow is divided amongst the possible exits based on where the flow is destined. If

a particular node becomes congested, nodes sending flow to it try to redirect excess flow away.

Nodes in the graph are assigned a maximum inflow capacity. For all nodes except work areas, this is set as $\frac{1}{3}$, which is the maximum rate that droplets may make a turn through an intersection without overlapping. The maximum inflow rate to a work area is computed based on the duration of its operations in the analysis graph. For example, consider a system where mixing operations take 100 cycles to complete. Each mixing operation requires two droplets and each work area can support 8 simultaneous mixing operations. If the droplets entered a work area at a rate of 1 droplet every 7 cycles, the first mixing operation should complete shortly after, or even before, the eighth operation begins. Droplets would likely not be able to enter the work area at a faster rate than this because there would not be any free work units. To respect the maximum inflow rates, parent nodes must adjust their outflows, which may result in some of the inflow not translating into outflow. A congested node is defined as a node with inflow greater than outflow.

Once the computed flow through the system converges, the component graph is analyzed. If the system is stable, the inflow will equal the outflow at every street, connector, work area, and intersection node. Otherwise, the system is likely unstable. A simplified example system with steady state flow information is depicted in Figure 7.

7 Example Systems

We have simulated example systems operating in batch mode and in continuous mode. We now describe an example system based on the polymerase chain reaction operations outlined in [8]. The reaction involves eight input droplet types and seven mixing operations. See Figure 8 for an analysis graph of the system. Immediately following each mixing operation, the resulting droplet is split into two droplets. The array is set up with four work areas, eight sources, each introducing an input droplet type, and one sink to collect the final product (Figure 2). The size of the array in this example is 53×41 . The system parameters were set with the aid of the stability analysis described above. In continuous mode, the system has an average of 66 droplets on the array. The routing computations are performed at a rate of 73000 cycles a second, enabling rapid simulation of the system to verify stability. Animations of the PCR reaction, in both batch mode and continuous mode, as well as other systems, including those running multiple reactions, are available at www.cs.rpi.edu/~sakella/microfluidics/.

8 Conclusions and Future Work

We described a new approach to creating a general-purpose DMFS by dividing the planar array into a collection of components and coordinating

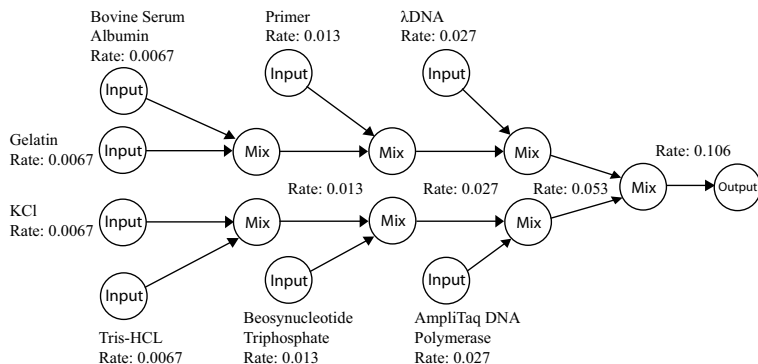


Fig. 8. PCR analysis graph. Input nodes are labeled with the samples they introduce and the rate at which they introduce them. Edges out of mix nodes are labeled with the droplet rate resulting from the operation.

the motions of droplets by implementing a decentralized routing algorithm. We explored techniques to identify potentially unstable systems, and applied them to successfully demonstrate a stable DNA polymerase chain reaction in simulation.

The system described here can semi-automatically generate a layout given a set of system parameters, and then perform real-time droplet manipulation. It has successfully coordinated hundreds of droplets, and is a proof of concept that decentralized network-like motion planning can work for a digital microfluidics system. Further, the software can be easily modified to act as a controller for a physical array. The same array can perform a variety of chemical analyses, and has been demonstrated to even perform multiple analyses in parallel.

The current system can be enhanced in a number of ways for greater flexibility and efficiency. The overall design of the components and the system allows for the introduction of new component types. For example, sensing components that permit sensor monitoring of the reactions, and storage components, where droplets can be temporarily stored, can be incorporated. The system can also support simpler hardware that permits only limited row-column addressing of electrodes ([11]). Automatically sequencing operations to achieve desired droplet concentrations would be a useful extension. Automatically generating the array for a given reaction requires optimizing the number of tiles and their layout, as well as the locations of the sinks and sources on the array. Optimizing tile design to optimize droplet flow rates, and optimizing array layout for a given tile pattern are interesting problems.

Techniques for stability and performance analysis, drawing upon methods from queueing theory and networking ([17], [6], [2]), can be applied to this system to provide insight. Detailed analysis of the system can help identify components where congestion or deadlock may occur, and automatically set

droplet input rates to avoid such problems. More sophisticated dynamic routing techniques can be explored to more effectively load balance the transport sections of the array and reduce bottlenecks at components. The design and control of fully reconfigurable arrays, where any part of the array may be used for any desired operation, pose particularly interesting challenges.

Acknowledgments

Many thanks to Karl Bohringer for introducing us to this problem and providing encouragement and advice along the way. We benefited from discussions on network models with Bulent Yener, Biplab Sikdar, and Jayasri Akella. This work was supported in part by NSF under CAREER Award No. IIS-0093233.

References

1. S. Akella and S. Hutchinson. Coordinating the motions of multiple robots with specified trajectories. In *IEEE International Conference on Robotics and Automation*, pages 624–631, Washington, DC, May 2002.
2. D. P. Bertsekas and R. G. Gallager. *Data Networks*. Prentice-Hall, Englewood Cliffs, N.J., second edition, 1992.
3. A. Bicchi and L. Pallottino. On optimal cooperative conflict resolution for air traffic management systems. *IEEE Transactions on Intelligent Transportation Systems*, 1(4):221–231, Dec. 2000.
4. K.-F. Böhringer. Optimal strategies for moving droplets in digital microfluidic systems. In *Seventh International Conference on Miniaturized Chemical and Biochemical Analysis Systems (MicroTAS '03)*, Squaw Valley, CA, Oct. 2003.
5. K. F. Böhringer. Towards optimal strategies for moving droplets in digital microfluidic systems. In *IEEE International Conference on Robotics and Automation*, New Orleans, LA, Apr. 2004.
6. J. Brassil and R. Cruz. Nonuniform traffic in the Manhattan street network. In *IEEE International Conference on Communications (ICC '91)*, pages 1647–1651, June 1991.
7. S. K. Cho, H. Moon, and C.-J. Kim. Creating, transporting, cutting, and merging liquid droplets by electrowetting-based actuation for digital microfluidic cells. *Journal of Microelectromechanical Systems*, 12(1):70–80, Feb. 2003.
8. J. Ding, K. Chakrabarty, and R. B. Fair. Scheduling of microfluidic operations for reconfigurable two-dimensional electrowetting arrays. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20(12):1463–1468, Dec. 2001.
9. M. Erdmann and T. Lozano-Perez. On multiple moving objects. *Algorithmica*, 2(4):477–521, 1987.
10. 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 *IEEE International Electron Devices Meeting (IEDM)*, 2003.
11. S.-K. Fan, C. Hashi, and C.-J. Kim. Manipulation of multiple droplets on NxM grid by cross-reference EWOD driving scheme and pressure-contact packaging. In *IEEE Conference on MEMS*, pages 694–697, Kyoto, Japan, Jan. 2003.

12. D. Gross and C. M. Harris. *Fundamentals of Queueing Theory*. Wiley, New York, third edition, 1998.
13. 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”. *International Journal of Robotics Research*, 3(4):76–88, 1984.
14. T. B. Jones, M. Gunji, M. Washizu, and M. J. Feldman. Dielectrophoretic liquid actuation and nanodroplet formation. *Journal of Applied Physics*, 89:1441–1448, Jan. 2001.
15. K. Kant and S. W. Zucker. Toward efficient trajectory planning: The path-velocity decomposition. *International Journal of Robotics Research*, 5(3):72–89, Fall 1986.
16. S. M. LaValle and S. A. Hutchinson. Optimal motion planning for multiple robots having independent goals. *IEEE Transactions on Robotics and Automation*, 14(6):912–925, Dec. 1998.
17. N. F. Maxemchuk. Routing in the Manhattan street network. *IEEE Transactions on Communications*, 35(5):503–512, May 1987.
18. P. A. O’Donnell and T. Lozano-Perez. Deadlock-free and collision-free coordination of two robot manipulators. In *IEEE International Conference on Robotics and Automation*, pages 484–489, Scottsdale, AZ, May 1989.
19. P. Paik, V. K. Pamula, and R. B. Fair. Rapid droplet mixers for digital microfluidic systems. *Lab on a Chip*, 3:253–259, 2003.
20. J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. In J.-D. Boissonnat, J. Burdick, K. Goldberg, and S. Hutchinson, editors, *Algorithmic Foundations of Robotics V*, pages 221–237. Springer-Verlag, Heidelberg, Germany, 2003.
21. M. G. Pollack, R. B. Fair, and A. D. Shenderov. Electrowetting-based actuation of liquid droplets for microfluidic applications. *Applied Physics Letters*, 77:1725–1726, 2000.
22. J. Reif and M. Sharir. Motion planning in the presence of moving obstacles. In *Proceedings of the 26th Annual Symposium on the Foundations of Computer Science*, pages 144–154, Portland, Oregon, Oct. 1985.
23. G. Sanchez and J. Latombe. On delaying collision checking in PRM planning — application to multi-robot coordination. *International Journal of Robotics Research*, 21(1):5–26, Jan. 2002.
24. T. Schouwenaars, B. De Moor, E. Feron, and J. How. Mixed integer programming for multi-vehicle path planning. In *European Control Conference 2001*, Porto, Portugal, 2001.
25. T. Simeon, S. Leroy, and J.-P. Laumond. Path coordination for multiple mobile robots: A resolution-complete algorithm. *IEEE Transactions on Robotics and Automation*, 18(1):42–49, Feb. 2002.
26. P. Švestka and M. Overmars. Coordinated path planning for multiple robots. *Robotics and Autonomous Systems*, 23(3):125–152, Apr. 1998.
27. A. S. Tanenbaum. *Computer Networks*. Prentice Hall, Upper Saddle River, NJ, third edition, 1996.
28. C. Tomlin, G. J. Pappas, and S. Sastry. Conflict resolution for air traffic management: A study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, Apr. 1998.