# Fast Online Synthesis of Generally Programmable Digital Microfluidic Biochips

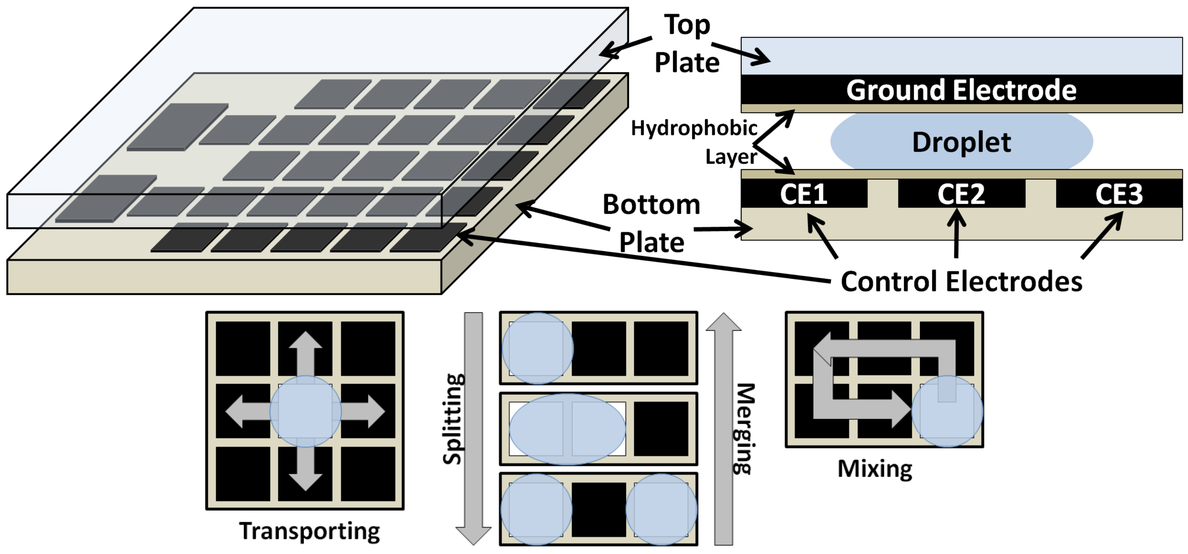
Dan Grissom and Philip Brisk
University of California, Riverside (UCR)

## Digital Microfluidic Technology

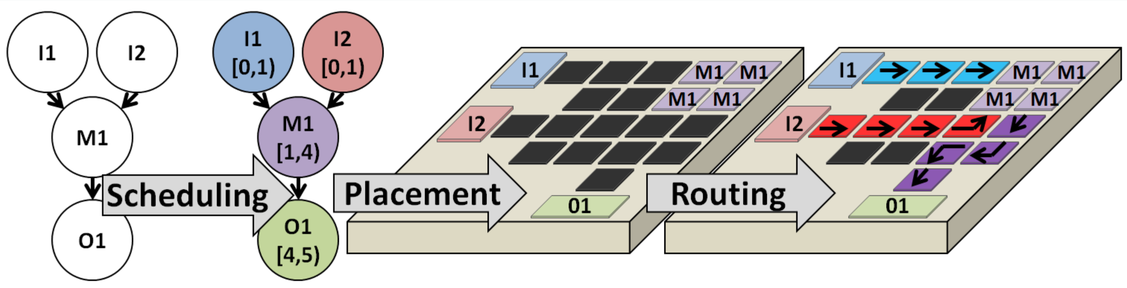
**Digital Microfluidic Biochips (DMFBs) are an emerging "lab-on-a-chip (LoC)" technology that perform biochemical reactions by operating on fluidic droplets on the scale of nano-liters.**

**Applications:**
- Clinical pathology
- Point of care diagnostics
- Drug discovery
- Proteomics, DNA, PCR, etc.
- Real-time detection of biochemical terror attacks

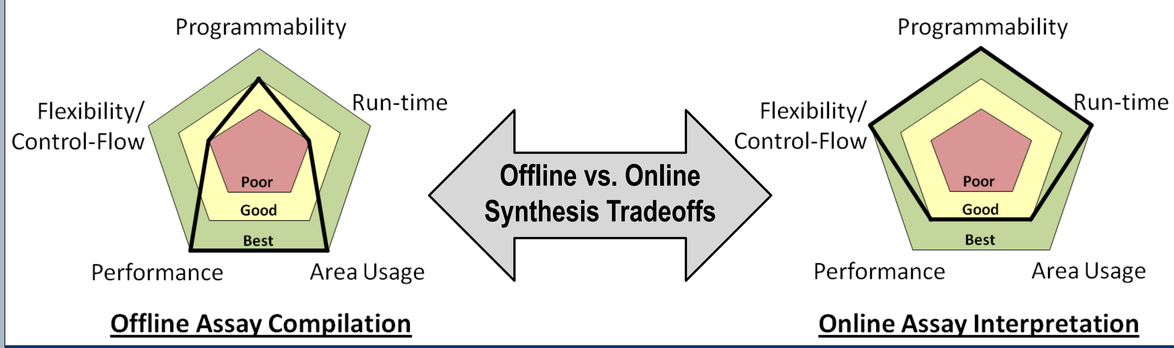
**Key advantages:**
- Reduced cost
- Reduced reagent and sample sizes
- Increased throughput and efficiency
- Increased sensitivity and accuracy
- Automation and miniaturization



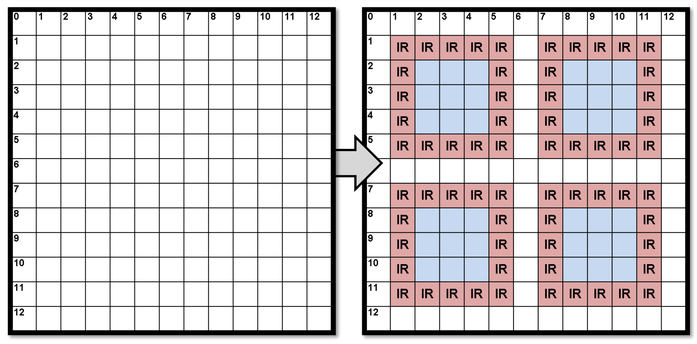

## Microfluidic Synthesis




### Microfluidic Synthesis Flow

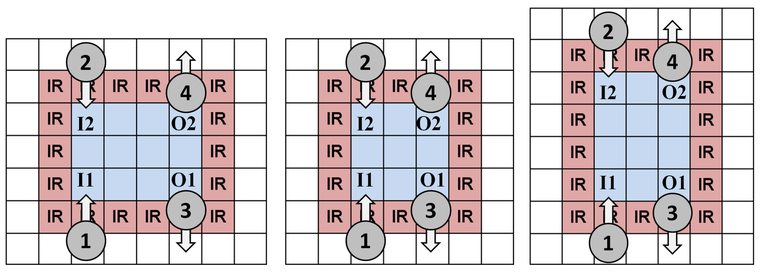



## Module Topology & Synchronization



Physical DMFB  Applied Module Placement

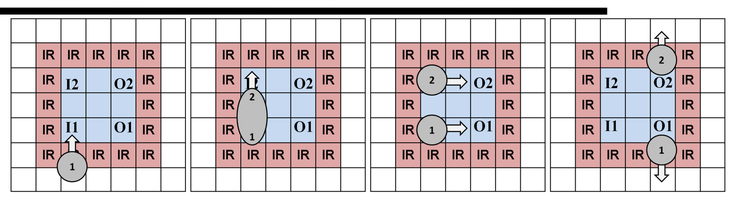
- Modules arranged regularly
- Operations limited to chambers
- Space left between modules for sufficient routing



Input/output cells on modules of different sizes

- Each module has 2 distinct input and output cells
- Allows for deadlock-free routing; a droplet can wait in its output cell as long as necessary until there is a clear path to its destination

**Example of droplet coordination in a module:**

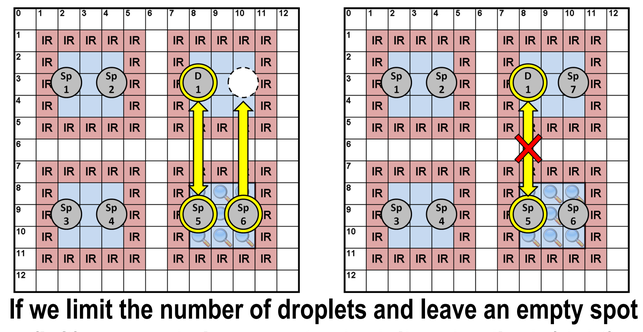


D1 enters and waits for TS to begin | D1 splits into two droplets | D1 and D2 slide over to exits and wait until TS is over | D1 and D2 leave module once TS is complete

## Fast Synthesis

### Scheduling : List Scheduling

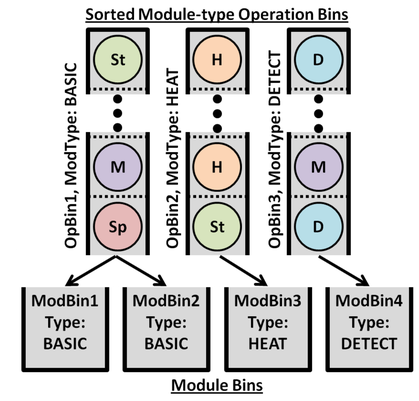
- Greedy constructive algorithm
- Non-iterative
- Limit number of droplets to prevent scheduling deadlock
- Resource availability based on knowledge of placer/binder
- Storage/module limited to number of I/O cells (2 in this case)



If we limit the number of droplets and leave an empty spot (left), we can help prevent scheduling deadlock (right)

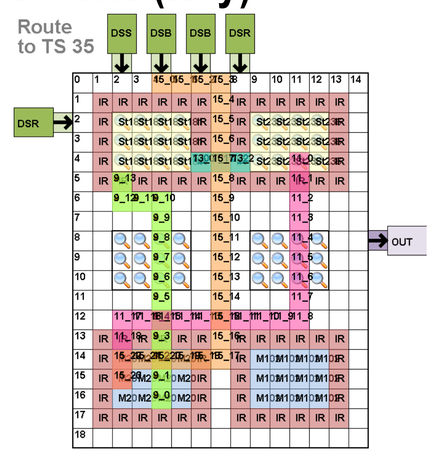
### Placement: Module Binding

- Fixed binding instead of free placement
- Greedy left-edge algorithm
- Non-iterative
- Modules placed at fixed locations
- Space left between modules guarantees a route




### Routing: Simplified Maze Router (Roy)

- Based on Roy's Soukup Maze Router
- Routes generated from source to destination in one pass
- Routes are compacted after computation
  - Stalls added to beginning or middle of routes
- Routes deadlock free because of fixed placer
  - Module spacing guarantees valid path
  - Module I/O cells prevent droplet deadlock




## Evaluation of Synthesis Flow

- Performed comparison against classic offline synthesis flow
- Performed experiments on Intel i7 and low-powered Atom processor

- List scheduling produces comparable schedules in much less time than long-running iterative algorithms

- Fixed placement/binding takes more space, but finds solutions much quicker

- Routing is a quick process on both flows; is guaranteed with our online synthesis flow

**Genetic Scheduling vs. List Scheduling**

| Benchmark | Genetic Scheduling | | | List Scheduling | | |
|---|---|---|---|---|---|---|
| | i7 (ms) | Atom (ms) | Sched (s) | i7 (ms) | Atom (ms) | Sched (s) |
| PCR | 395 | 2,621 | 12 | 1 | 1 | 12 |
| InVitro_1 | 665 | 4,475 | 15 | 0 | 2 | 15 |
| InVitro_2 | 1,293 | 8,122 | 17 | 0 | 5 | 19 |
| InVitro_3 | 1,990 | 13,156 | 19 | 1 | 13 | 23 |
| InVitro_4 | 3,541 | 22,376 | 23 | 1 | 17 | 26 |
| InVitro_5 | 5,744 | 39,410 | 31 | 2 | 27 | 35 |
| Protein | 3,297 | 22,334 | 110 | 3 | 14 | 116 |

**Placement vs. Binding**

| Benchmark | Simulated Annealing Placer | | | Module Binder | | |
|---|---|---|---|---|---|---|
| | i7 (ms) | Atom (ms) | %Cells Used | i7 (ms) | Atom (ms) | %Cells Used |
| PCR | 16 | 200 | 16 | 0 | 0 | 20 |
| InVitro_1 | 621 | 12,843 | 16 | 0 | 0 | 24 |
| InVitro_2 | 105,138 | 141,177 | 21 | 0 | 0 | 28 |
| InVitro_3 | 72,311 | 506,767 | 29 | 0 | 0 | 36 |
| InVitro_4 | 19,789 | 3,317,571 | 32 | 0 | 0 | 45 |
| InVitro_5 | 74,899 | 1,399,936 | 36 | 0 | 0 | 48 |
| Protein | 4,867,220 | 79,531,695 | 29 | 0 | 4 | 46 |

**Simplified Roy Routing**

| Benchmark | GA Scheduling - SA Placement Flow | | | List Scheduling - Module Binding Flow | | |
|---|---|---|---|---|---|---|
| | i7 (ms) | Atom (ms) | # Routing Cycles / # Sub-problems | i7 (ms) | Atom (ms) | # Routing Cycles / # Sub-problems |
| PCR | 0 | 2 | 78 / 4 | 0 | 6 | 56 / 4 |
| InVitro_1 | 0 | 2 | 135 / 9 | 0 | 3 | 111 / 9 |
| InVitro_2 | 0 | 4 | 180 / 11 | 0 | 7 | 167 / 12 |
| InVitro_3 | 0 | 10 | 207 / 15 | 0 | 12 | 209 / 16 |
| InVitro_4 | 0 | 7 | 234 / 15 | 1 | 19 | 299 / 19 |
| InVitro_5 | 1 | 9 | 342 / 22 | 1 | 26 | 351 / 24 |
| Protein | 5 | 32 | 1212 / 71 | 3 | 76 | 638 / 45 |

- Our flow much more feasible for online synthesis because of fast algorithms

**GA Scheduling - SA Placement - Simp. Roy Routing Offline Flow**

| Benchmark | Schedule | | Placement | Routing | | Total |
|---|---|---|---|---|---|---|
| | AT (s) | CT (s) | CT (s) | AT (s) | CT (s) | (AT + CT) (s) |
| PCR | 12 | 3 | 0 | 1 | 0 | 16 |
| InVitro_1 | 15 | 4 | 13 | 1 | 0 | 34 |
| InVitro_2 | 17 | 8 | 141 | 2 | 0 | 168 |
| InVitro_3 | 19 | 13 | 507 | 2 | 0 | 541 |
| InVitro_4 | 23 | 22 | 3,318 | 2 | 0 | 3,365 |
| InVitro_5 | 31 | 39 | 1,400 | 3 | 0 | 1,474 |
| Protein | 110 | 22 | 79,532 | 12 | 0 | 79,676 |

**List Scheduling - Module Binding - Simp. Roy Routing Online Flow**

| Benchmark | Schedule | | Placement | Routing | | Total |
|---|---|---|---|---|---|---|
| | AT (s) | CT (s) | CT (s) | AT (s) | CT (s) | (AT + CT) (s) |
| PCR | 12 | 0 | 0 | 0 | 0 | 12 |
| InVitro_1 | 15 | 0 | 0 | 0 | 0 | 15 |
| InVitro_2 | 19 | 0 | 0 | 0 | 0 | 19 |
| InVitro_3 | 23 | 0 | 0 | 0 | 0 | 23 |
| InVitro_4 | 26 | 0 | 0 | 0 | 0 | 26 |
| InVitro_5 | 35 | 0 | 0 | 0 | 0 | 35 |
| Protein | 116 | 0 | 0 | 1 | 0 | 117 |

## Contact

Dan Grissom (grissomd@cs.ucr.edu)
Philip Brisk (philip@cs.ucr.edu)
Computer Science Department, Bourns College of Engineering
University of California, Riverside