Multi-terminal PCB Escape Routing for Digital Microfluidic Biochips using Negotiated Congestion

Jeffrey McDaniel*, Daniel Grissom†, Philip Brisk*

*Department of Computer Science and Engineering
University of California, Riverside
Riverside, California 92507
Email: jmcda001@ucr.edu, philip@cs.ucr.edu

†Department of Computer Science
Azusa Pacific University
Azusa, California, 91702
Email: dgrissom@apu.edu

***Abstract*—This paper introduces a multi-terminal escape routing algorithm for the design of Printed Circuit Boards (PCBs) that control Digital Microfluidic Biochips (DMFBs). The new algorithm is based on the principle of negotiated congestion, which has been applied in the past to problems including FPGA routing and PCB escape routing for single-terminal nets. PCBs designed for Pin-constrained DMFBs, in which one control pin may drive multiple electrodes, require multi-terminal escape routing solutions. Experimental results indicate that negotiated congestion is more effective for multi-terminal escape routing than existing techniques, which are based on maze routing coupled with rip-up and re-route, yielding an overall reduction in the number of PCB layers in most the test cases that were tried.**

I. Introduction

This paper presents a multi-terminal escape routing algorithm for the design of printed circuit boards (PCBs) for digital microfluidic biochips (DMFBs). A DMFB is an emerging laboratory-on-a-chip (LoC) technology that offers the possibility of miniaturized, automated, software-programmable chemistry. LoCs, and DMFBs in particular, are poised to replace traditional benchtop chemistry methods in the biological sciences. LoCs offer significant benefits including reduced usage of costly reagents on a per-experiment basis, and the elimination of human error due to automation.

As shown in Fig. 1, a DMFB is a 2-dimensional grid of electrodes that offers discrete control over individual droplets

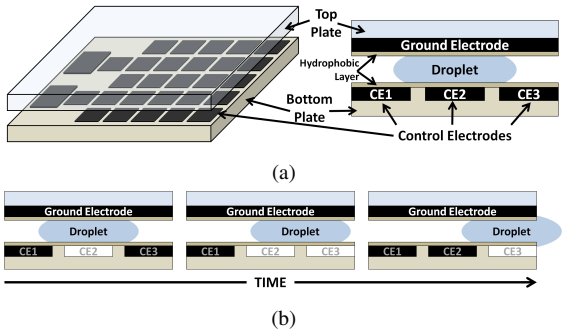


Fig. 1. (a) A DMFB is comprised of a 2D array of control electrodes; (b) a cross-sectional shows that a droplet is sandwiched between a top ground electrode and the array of control electrodes on the bottom; (c) droplet motion is induced by the proper sequence of electrode activations (white) and deactivations (black).

978-1-4799-6016-3/14/$31.00

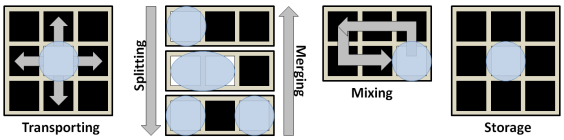


Fig. 2. Droplet transportation, splitting, merging, mixing and storage are the fundamental microfluidic operations and can be combined together in sequence to perform complex assays.

of liquid: activating an electrode underneath a droplet holds it in-place; activating adjacent electrodes induces droplet motion through an electrostatic force, a phenomenon referred to as Electrowetting on Dielectric (EWoD) [12]. Fig. 2 illustrates the basic instruction set of a DMFB; as a DMFB offers significant spatial parallelism, many such operations can be performed concurrently. In addition to the operations shown in Fig. 2, integrated sensors [11], [13], [4] and external devices (e.g., heaters, detectors, etc.) can be placed on pre-specified array locations, which add new operational capabilities to the device.

Fig. 3 illustrates the main stages of DMFB synthesis. The input is an assay, i.e., a step-by-step sequence of biochemical fluidic operations to perform, which is specified as a directed acyclic graph (DAG). A DMFB compiler must schedule, place, and route the DAG onto the device. Due to space limitations, we cannot describe these steps in detail; instead, we refer the interested reader to an appropriate survey paper that comprehensively covers the topic [1].

A DMFB is typically mounted on a PCB. As shown in Fig. 4, a direct-addressing DMFB, each electrode has independent control, which requires a large number of control pins; in a pin-constrained DMFB, several electrodes share a control pin, which reduces the overall PCB cost [17], [6], [16], [8]. Referring back to Fig. 3, pin assignment refers to the algorithmc step of determining the total number of control pins and the subset of electrodes that each pin controls. The final step of DMFB synthesis flow, which is the focus of this paper, is the PCB wire routing step, which connects control pins to electrodes in the DMFB.

As shown in Fig. 5, multiple PCB layers may be needed to realize both direct-addressing and pin-constrained DMFBs, although it has always been presumed that the latter requires fewer layers. In this paper, we demonstrate the that number of PCB layers depends primarily on the quality of the wire

routing, as opposed to the pin assignment. We introduce a multi-terminal PCB router for pin-constrained DMFBs based on the principle of negotiated congestion [10], [9]. Our results demonstrate that this PCB wire router is more effective than the prevailing approach based on maze routing with an integrated rip-up and re-route step [6], [16], which is the prevailing approach used in existing work on pin-constrained DMFB synthesis today.

II. Related Work

A. Single-terminal Escape Routing for PCBs

In PCB design automation, escape routing is the problem of routing all of the terminal pins inside of a pin grid (e.g., the electrodes of a DMFB) to the boundary of the grid [15], [9]. Hence, the pins escape from the component. In this context, each component corresponds to a pin array on the board, which may have multiple routing layers. Area routing [14] is then performed to establish appropriate connections between the pins of components (e.g., between the control electrodes of a DMFB and one or more shift registers that hold the value of the current bit vector in the actuation sequence).

Outside of the context of PCB-mounted DMFBs, most work on escape routing assumes single-terminal connections, i.e., each used pin in the grid must escape, but no pins are wired together [15], [9]. One of the most successful approaches to single-terminal escape routing is based on negotiated congestion [9], which was originally introduced for FPGA routing [10]. This paper extends this idea to encompass multi-terminal escape routing in which multiple pins may be wired together in the context of a pin-constrained DMFB.

B. Multi-terminal Escape Routing for Pin-constrained DMFBs

To date, we are aware of two general approaches for multi-terminal escape routing, which has been applied to the design of pin-constrained DMFBs. The simplest approach is to use a maze router, which can be enhanced by ripping up and re-routing a subset of the nets when routing failures occur [6], [16]. A second approach is to formulate multi-terminal escape routing as an Integer Linear Program (ILP), which provides an optimal solution, but will be unable to scale for large problem instances (as long as P != NP) [2]. The approach presented here achieves better quality results than the former, but without the scalability problems that plague the latter. Although beyond the scope of this work, manually constructed escape routing solutions for specific pin layout schemes have also been proposed [3].

C. Synergistic Pin Assignment and Escape Routing for Application-specific Pin-constrained DMFBs

The papers discussed in the preceding sections, as well as this paper, perform escape routing under the assumption that the pin assignment has been computed in advance. A more comprehensive approach is to simultaneously optimize pin assignment and escape routing together, which can both reduce the total number of PCB layers and the number of control pins, both of which affect the total PCB cost [6], [16]. These approaches repeatedly call an escape router based on maze routing with rip-up and re-route as a sub-routine. The multi-terminal negotiated congestion router presented in this paper could just as easily be called as a sub-routine in this context, and is shown to be more effective than maze routing with rip-up and re-route.

III. Wire Routing

This section introduces a multi-terminal PCB escape routing algorithm for DMFBs based on the principle of negotiated congestion. The input to the algorithm is direct addressing or pin-constrained DMFB with a known pin assignment. The algorithm computes a multi-terminal escape routing solution without modifying the pin assignment. The objective of the algorithm is to minimize the number of PCB layers required to route all nets; often, but not always, a legal routing solution can be obtained with a single PCB layer.

A. Problem Definition

The pin mapping solution is a one-to-many mapping from a set of external control pins P to a set of electrodes E. Control pin $p_i \in P$ maps to a subset of electrodes $E_{p_i} \subset E$, such that $E_{p_i} \cap E_{p_j} = \emptyset$ if $i \neq j$. Under escape routing, the location of pin p_i on the perimeter of the chip is not known; an escape route for p_i is a multi-terminal net that connects all of the electrodes in E_{p_i} and is then routed to the perimeter of the PCB region underneath the DMFB [15], [9], [6], [16], [2], [3]. Establishing a physical connection from the escape point to the control pin, which may be placed anywhere on the PCB, is a separate problem in PCB VLSI/CAD, which is handled by different algorithms[14].

B. Graph Representation

We employ a planar graph to represent the free space on the PCB underneath the DMFB [9]; similar to FPGA routing, we refer to this data structure as the Routing Resource Graph (RRG). The orthogonal capacity of the RRG is the number of wires that can pass between two orthogonally adjacent electrodes, and the diagonal capacity is the number of wires that can pass between diagonally adjacent electrodes; in modern PCB technologies, the diagonal capacity is slightly larger than the orthogonal capacity [15].

The RRG is constructed as a 2D array of tiles, where each tile itself is a planar RRG, as shown in Fig. 6. Each tile contains a set of edge nodes which may be escape nodes on the perimeter of the chip, or an interface node to an adjacent tile. The tile also includes internal nodes, which represent the physical portion of the PCB layer available for wire routing along with nodes that represent control electrodes, which are sinks for the router.

The tile can be generated for different orthogonal capacities: the orthogonal capacity is determined by the PCB technology (wire diameter and spacing rules) and the electrode dimension; the diagonal capacity is derived from the tile and its orthogonal capacity. Without a loss of generality, we use a capacity of three (and resultant diagonal capacity of six) in this paper. A higher orthogonal capacity can reduce the number of layers needed since more wires are able to pass between the electrodes; our solution works and RRG can scale to any orthogonal capacity, but through experiment, we found diminishing returns on orthogonal capacities above three.

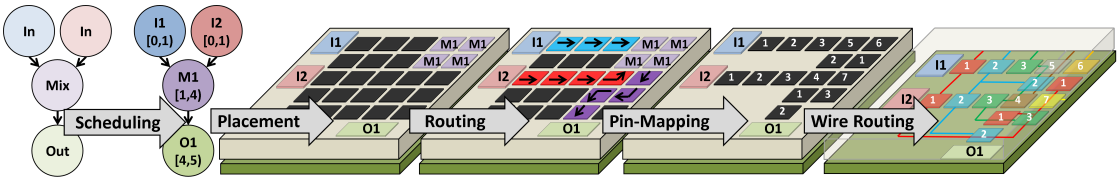
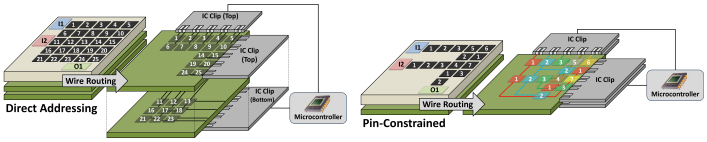


Fig. 3. Microfluidic synthesis maps an assay to a physical DMFB. An assay, represented as a directed acyclic graph (DAG) is scheduled; operations are then placed and routes are computed to transport droplets between operations and I/O ports on the perimeter of the chip. After synthesis, pin mapping and wire routing are performed to reduce the number of external control signals and lay out the PCB that delivers signals to the DMFB.



(a) (b)

Fig. 4. A DMFB typically has one or more PCB layers (green) beneath the electrode array which connect the microcontroller to the control electrodes; (a) a direct addressing DMFB is believed to require many PCB layers, while (b) pin-constrained DMFBs are designed to reduce the number of PCB layers.

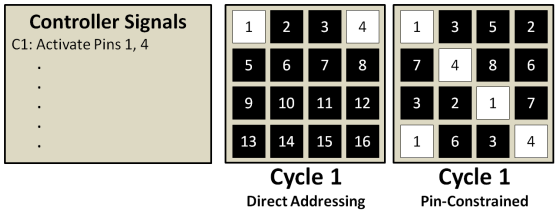


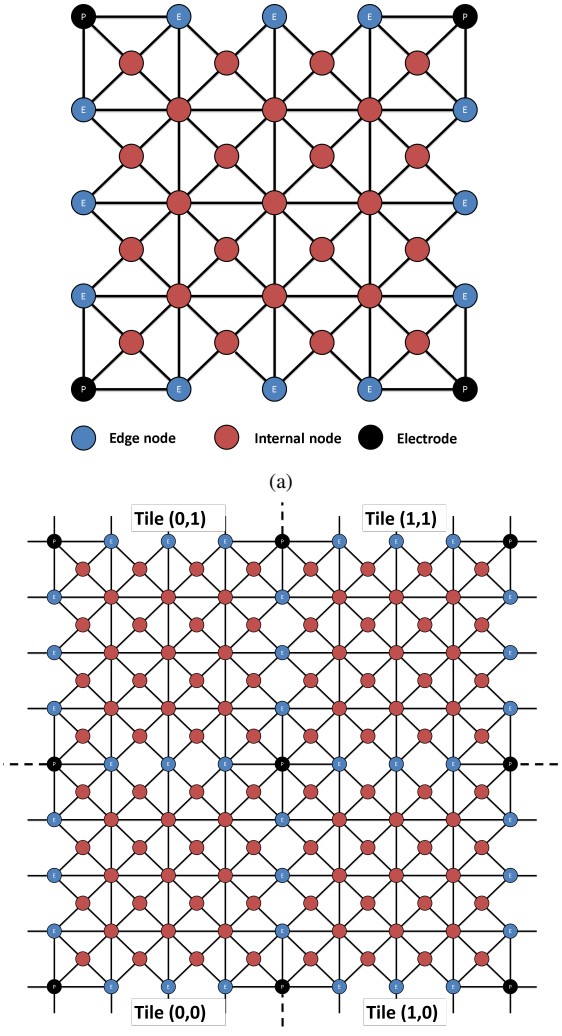
Fig. 5. (a) In a direct-addressing DMFB, activating one control pin activates exactly one electrode. (b) In a pin-constrained DMFB, activating one control pin may activate multiple electrodes concurrently.

C. Single PCB Layer Multi-Terminal Escape Routing

First, we describe a multi-terminal PCB escape routing procedure for a single PCB layer. As shown in Fig. 8, the RRG and pin map are input to the algorithm. The routing phase (lines 5-7) executes a multi-terminal variant of Lee's maze routing algorithm [7] on each pin group p_i. Under the negotiated congestion paradigm, multiple paths corresponding to distinct nets may share RRG nodes and edges [10], [9]; multiple iterations of Lees algorithm may be required before a set of disjoint paths (i.e., a legal solution) is obtained.

Pin groups are routed one at a time. For each pin group, a *supersource* node which connects to each external pin on the RRG perimeter, is used as the initial source. Once the first sink (i.e., an electrode $e_j \in E_{p_i}$ is found, the path obtained is traced back to the current net W_{p_i} corresponding to that electrode group. If e_j is the first electrode to be discovered, then W_{p_i} contains only the *supersource*; otherwise, a path that connects e_j to the existing net W_{p_i}, thereby extending it to connect to e_j.

Lees Algorithm is run repeated until all electrodes (sinks) in the group E_{p_i} are discovered. The current net W_{p_i} is initialized to contain the stack of current nodes on W_{p_i} during each run. Lees Algorithm explores the RRG using a breadth-first search until each subsequent sink is discovered, as shown in Fig. 7.



(a)

(b)

Fig. 6. Routing Resource Graph (RRG) Tile (a), and a 2x2 tiled chip (b). The lines going off chip in (b) inidicate external pins. The black nodes represent electrodes, the red nodes are internal nodes and the blue are edge nodes.

Negotiated congestion assigns an intersection cost to nodes that are in use, or have been used recently, to dissuade sharing them among paths. After routing each pin group $p_i \in P$, the negotiated congestion router increases the *history cost* of each RRG node x that is shared among multiple paths lines (8-10):

$$History_x = old\_history_x + (occupancy - 1) \quad (1)$$

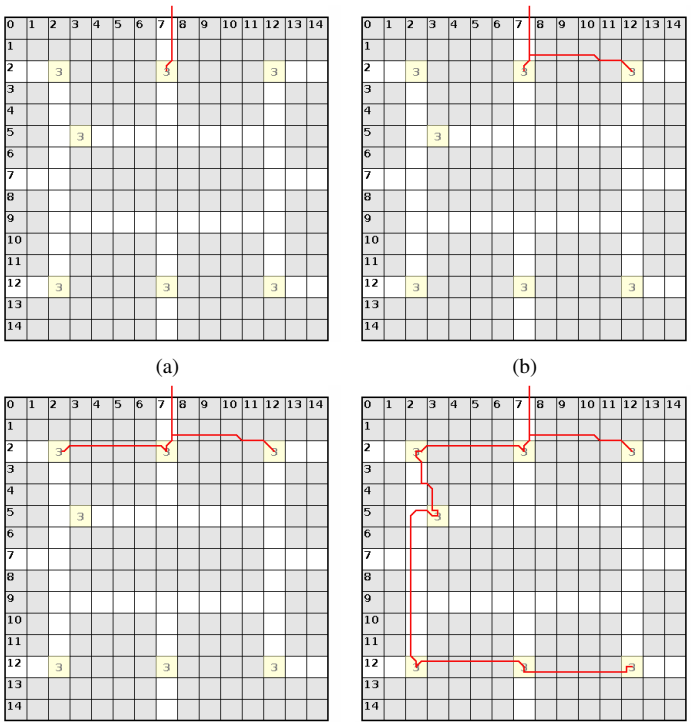


Fig. 7. Routing a single electrode group. (a) The first link establishes a connection between the *supersource* and the first electrode discovered by Lees Algorithm. (b), (c) The second and third links respectively establish a multi-terminal connection to the second and third electrodes discovered. (d) The process continues until all electrodes in the group are discovered.

A legal route is obtained if all nets are routed without any shared nodes; otherwise, all nets are immediately ripped up and re-routed on the RRG using the updated history costs, which dissuades usage of these nodes during subsequent iterations.

Negotiated congestion does not guarantee eventual convergence to a legally routed solution, presuming that one exists. As such, the algorithm could, presumably, loop infinitely. To prevent this from occurring, a maximum number of iterations is established a-priori; if a legal route is not achieved after the maximum number of iterations, then the router quits and reports a failure to the user. Based on previous work [10], we let the router iterate 30 times without declaring a failure; we experimented with different values as well, and did not observe a significant change in success rates.

Fig. 9(a) shows an example of a pin assignment that can be routed in a single PCB layer; Fig. 9(b) shows a more complex pin assignment that cannot be routed in a single PCB layer.

D. Multiple PCB Layer Multi-Terminal Escape Routing

If single-layer routing fails, it is still possible to produce a legal and usable multi-layer PCB, as per the pseudocode in Fig. 11. If the single-layer negotiated congestion algorithm fails, routed nets are processed one-by-one in-order. If the current net shares at least one node with a net that has been previously processed and has been routed successfully, then it is removed from the current layers routing solution; otherwise, it is left in-place. Fig. 10 shows an example. After processing all nets, at least one will be routed successfully on the current

Input : $P :=$ set of unrouted pins
Output : $R :=$ set of routed pins
//$M :=$ max iterations
//$HC :=$ History cost
//$i :=$ iteration

```
1:  i = 0
2:  R_best ← ∅
3:  repeat
4:      Rip up routes
5:      R_curr ← ∅
6:      for all p_i ∈ P do
7:          R_curr ← LeeMazeRoute(p_i)
8:      end for
9:      for all Nodes x ∈ R_curr do
10:         if x_occ > 1 then
11:             x_HC = x_old_HC + (x_occ − 1)
12:         end if
13:     end for//int(R):=#intersections
14:     if R_best == ∅|| int(R_curr)<int(R_best) then
15:         R_best ← R_curr
16:     end if
17:     i += 1;
18: until int(R_best) == 0||i >= M
19: if int(R_best) > 0 then
20:     Failed to route
21: end if
22: return R_best
```

Fig. 8. Single Layer Wire Router

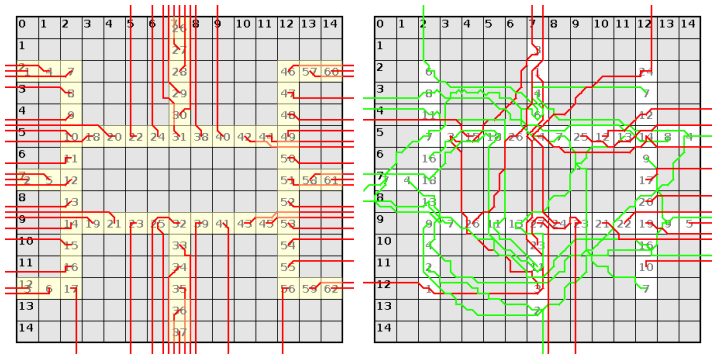


Fig. 9. (a) The PCR_DA benchmark routed successfully using one layer. (b) The Zhao_Protein benchmark failed to route on one layer: red lines indicate successful routes, and green lines indicate routes that failed due to intersections.

layer. The single-layer routing algorithm is then re-run using all unsuccessfully routed nets to generate a subsequent PCB layer.

IV. Experimental Results

We compare the negotiated congestion router introduced in this paper with an existing multi-terminal PCB routing algorithm for DMFBs introduced by Yeh et al. [16]. It is important to note that Yeh et al. interleave the construction of a pin assignment with calls to an escape router, whereas, our experiments focus exclusively on using the escape routing algorithms with existing pin assignments. Specifically, we compare with the escape routing algorithm introduced by

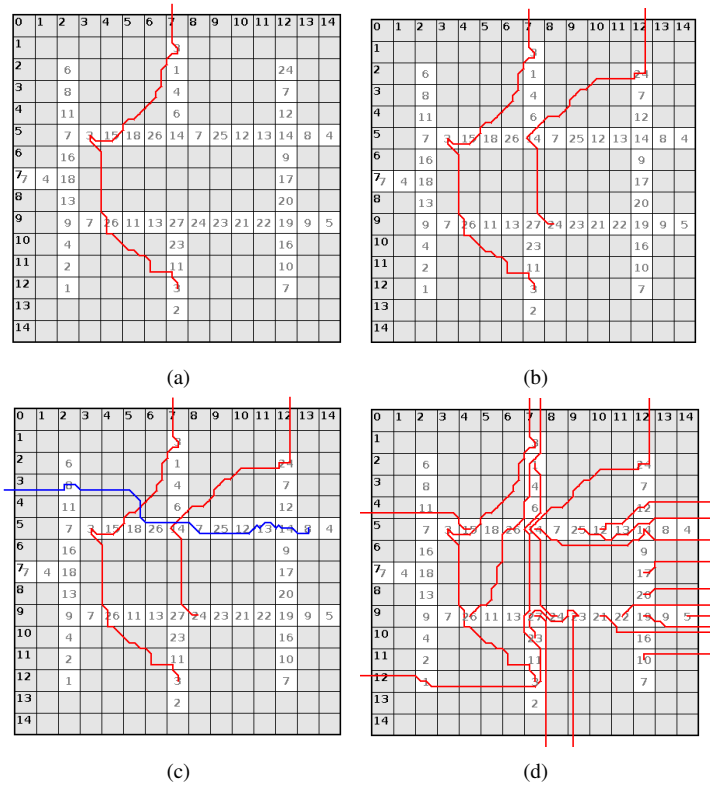


Fig. 10. (a) The net for Pin 3 routes successfully. (b) The net for Pin 24 routes successfully without intersecting the net for Pin 3. (c) The net for Pin 8 routes unsuccessfully, intersecting with the routes computed for Pins 3 and 24; Pin 24 will be removed from the current layer, and eventually routed on a subsequent layer. (d) Escape routes for the current layer are completed; no other nets can route successfully on the current layer.

Input : $P :=$ set of unrouted pins
Output : $L :=$ Routed PCB layers

1: **repeat**
2: singleLayer($\forall p_i \in P$)
3: $L_{curr} \leftarrow \emptyset$
//Add paths to current layer
4: **for all** $p_i \in P$ **do**
5: **if** !intersect(p_i, L_{curr}) **then**
6: $L_{curr} \leftarrow p_i$
7: Remove p_i from P
8: **end if**
9: **end for**
10: $L \leftarrow L_{curr}$
11: **until** $P = \emptyset$

Fig. 11. Multi-Layered Wire Router

Yeh et al., not the highly optimized pin assignment scheme introduced in their paper.

Experiments were performed on a Dell Optiplex 580 PC with an AMD Phenom II X2 B53 Processor, and 4 GB of memory running 32-bit Ubuntu 12.04 LTS. For both algorithms, Table I reports the number of PCB layers required along with the algorithmic runtime. Average, maximum, and/or total wirelength are not metrics of interest, as they do not affect the cost of the PCB and the 200 Hz actuation rates of electrodes is low enough that skew due to wires of unequal length is not a sigificant problem.

TABLE I. WIRE ROUTING

Benchmark	Pin Count	Layered Pathfinder		[16]	
		Run Time(s)	Layers	Run Time(s)	Layers
Zhao_PCR	14	15.338	4	2.162	5
Luo_PCR	22	18.824	5	3.401	6
PCR_DA	62	0.320	1	0.557	2
Zhao_Protein	27	10.655	3	1.688	4
Luo_Protein	21	11.435	4	1.700	4
Protein_DA	54	0.301	1	0.597	2
Zhao_InVitro	25	14.350	4	2.371	5
Luo_InVitro	21	16.358	5	2.278	5
InVitro_DA	59	0.335	1	0.406	2
Zhao_Multi	32	15.995	5	3.181	5
Luo_Multi	27	16.204	6	4.543	6
Multi_DA	81	0.411	1	1.028	2
FPPC 12x15	33	17.941	4	1.777	5
IA 10x10	100	0.183	1	0.363	2
IA 15x15	225	32.842	1	3.936	2
IA 15x19	285	54.558	2	8.412	3
IA 30x30	900	1436.7	5	219.247	6

All benchmarks that were used in this study are either direct addressing pin assignments, or optimized pin assignments taken from previously published papers. The benchmarks labeled Zhao_XXX were taken from Ref. [17] and those labeled Luo_XXX from Ref. [8]; these benchmarks do not use every electrode in the 2D grid. The benchmarks labeled XXX_DA impose a direct addressing scheme on the pattern of used electrodes from the preceding references. The benchmark FPPC 12x15 is taken from [5], and the benchmarks IA_XXX are individually addressing chips, the use the direct addressing scheme, except *all* electrodes are used.

The multi-terminal PCB escape routing algorithm presented by Yeh et al. [16] was designed exclusively for single-layer chips. We extended the algorithm to produce multi-laye routes using the same basic approach as shown in Fig. 11.

Table I reports the results of the experiment. In thirteen cases, negotiated congestion yielded fewer PCB layers than Yeh et al.s algorithm; in the remaining 4 cases, both algorithms yielded an equal number of PCB layers. These results clearly establish the algorithmic superiority of negotiated congestion compared to single-time Maze Routing with rip-up and re-route, which is the approach taken by Yeh et al.s algorithm. On average, negotiated congestion reduced the number of layers by 26.59%, with a maximum savings of 50%.

Negotiated congestion ran considerably slower than Yeh et al.s algorithm. This is because each failed attempt to route all nets on a single PCB layer entailed 30 calls to Lees Maze Routing algorithm, which is called just once per layer by Yeh et al.s algorithm. Since PCB CAD is performed fully offline, we consider this runtime overhead to be tolerable.

As a detailed example, Fig. 12 and 13 show escape routes for the Zhao_Protein benchmark using negotiated congestion (3 layers) and Yeh et al.s algorithm (4 layers).

V. CONCLUSION AND FUTURE WORK

This paper has presented an algorithm for multi-terminal PCB escape routing for direct addressing and pin-constrained DMFBs based on negotiated congestion. Experiments have shown that it is more effective, although also more time-consuming, than existing multi-terminal PCB escape routers

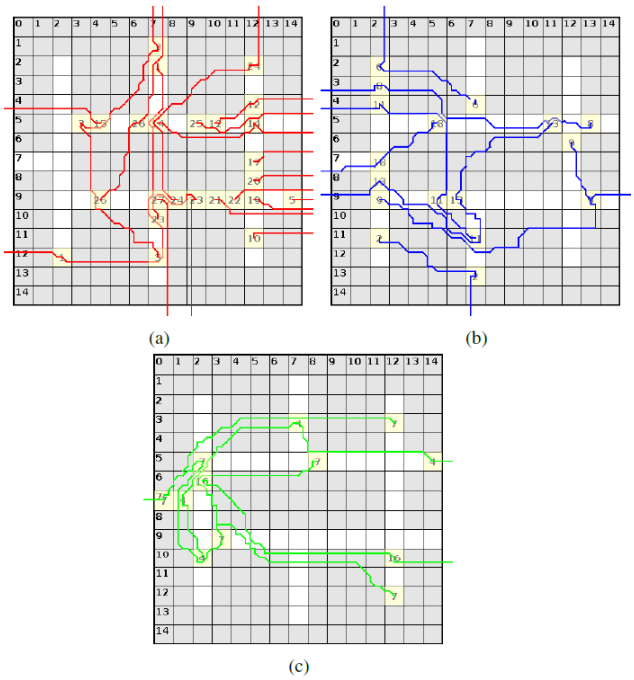


Fig. 12. The Zhao_Protein benchmark [17] routed using the proposed multi-layered iterative negotation method. A 3-layer PCB was obtained.

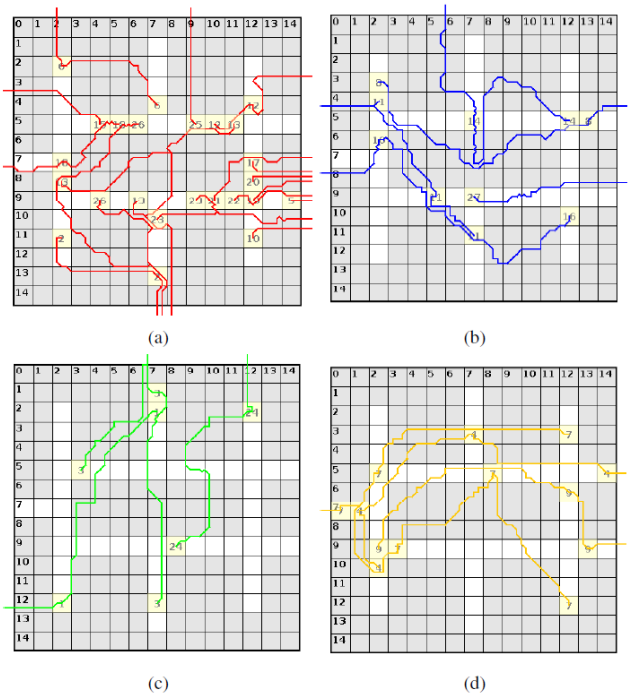


Fig. 13. The Zhao_Protein benchmark [17] routed using an existing multi-terminal PCB escape routing algorithm [16]. A 4-layer PCB was obtained.

that have been applied to the same problem. Future work will attempt to integrate the negotiated congestion router into a larger and more comprehensive algorithm that simultaneously optimized pin assignment in conjunction with escape routing to further reduce PCB cost.

Acknowledgment

This work was supported by NSF Grant 1035603. D. Grissom was supported by an NSF Graduate Research Fellowship and a UC Riverside Graduate Division Year Fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the NSF.

References

[1] K. Chakrabarty. Design Automation and Test Solutions for Digital Microfluidic Biochips. *IEEE Transactions on Circuits and Systems*, 57(1):4–17, 2010.

[2] J.-W. Chang, S.-H. Yeh, T.-W. Huang, and T.-Y. Ho. An ILP-based Routing Algorithm for Pin-Constrained EWOD Chips With Obstacle Avoidance. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(11):1655–1667, 2013.

[3] J.-W. Chang, S.-h. Yeh, T.-w. Huang, and T.-Y. Ho. Integrated Fluidic-Chip Co-Design Methodology for Digital Microfluidic Biochips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(2):216–227, 2013.

[4] K. Choi, A. H. C. Ng, R. Fobel, D. A. Chang-yen, L. E. Yarnell, E. L. Pearson, C. M. Oleksak, A. T. Fischer, R. P. Luoma, J. M. Robinson, J. Audet, and A. R. Wheeler. Automated Digital Microfluidic Platform for Magnetic-Particle-Based Immunoassays with Optimization by Design of Experiments. *Analytical chemistry*, (85):9638–9646, 2013.

[5] D. Grissom and P. Brisk. A field-programmable pin-constrained digital microfluidic biochip. *Proceedings of the 50th Annual Design Automation Conference on - DAC '13*, page 1, 2013.

[6] T.-W. Huang, S.-Y. Yeh, and T.-Y. Ho. A Network-Flow Based Pin-Count Aware Routing Algorithm for Broadcast-Addressing EWOD Chips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(12):1786–1799, Dec. 2011.

[7] C. Y. Lee. An Algorithm for Path Connections and Its Applications. *IRE Transactions on Electronic Computers*, 30:1389–1401, 1959.

[8] Y. Luo and K. Chakrabarty. Design of Pin-Constrained General-Purpose Digital. In *Design Automation Conference*, pages 18–25, 2012.

[9] Q. Ma, T. Yan, and M. D. Wong. A negotiated congestion based router for simultaneous escape routing. *2010 11th International Symposium on Quality Electronic Design (ISQED)*, pages 606–610, Mar. 2010.

[10] L. Mcmurchie and C. Ebeling. PathFinder : A Negotiation-Based Performance-Driven Router for FPGAs.

[11] M. A. Murran and H. Najjaran. Capacitance-based droplet position estimator for digital microfluidic devices. *Lab on a chip*, 12(11):2053–9, May 2012.

[12] M. G. Pollack, a. D. Shenderov, and R. B. Fair. Electrowetting-based actuation of droplets for integrated microfluidics. *Lab on a chip*, 2(2):96–101, May 2002.

[13] S. C. C. Shih, I. Barbulovic-Nad, X. Yang, R. Fobel, and A. R. Wheeler. Digital microfluidics with impedance sensing for integrated cell culture and analysis. *Biosensors & bioelectronics*, 42:314–20, Apr. 2013.

[14] T. Yan and M. D. Wong. BSG-Route: A length-matching router for general topology. *2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 499–505, Nov. 2008.

[15] T. Yan and M. D. Wong. Correctly Modeling the Diagonal Capacity in Escape Routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(2):285–293, Feb. 2012.

[16] S.-H. Yeh, J.-W. Chang, T.-W. Huang, and T.-Y. Ho. Voltage-aware chip-level design for reliability-driven pin-constrained EWOD chips. *Proceedings of the International Conference on Computer-Aided Design - ICCAD '12*, page 353, 2012.

[17] Y. Zhao, T. Xu, and K. Chakrabarty. Broadcast Electrode-Addressing and Scheduling Methods for Pin-Constrained Digital Microfluidic Biochips. *IEEE Transactions on Circuits and Systems*, 30(7):986–999, 2011.