6.854 Final Project
Routing in Control Layer of Microfluidic Chips

Nada Amin
namin@mit.edu

December 12, 2007

1 Introduction

Programmable microfluidic chips are “lab-on-a-chip” systems, that can automate biological computations or experiments by integrating a diverse set of biological sensors and by manipulating fluids at the picoliter scale [7]. For my final project, I investigate the problem of routing in the control layer of microfluidic chips. We identified this novel problem after detailed conversations with microfluidic researchers at MIT, who often spend hours at a time designing the control layers for their chips. Concretely, the control layer is composed of v valves and p punches, with $v \leq p$, and each valve needs to be connected to a punch. A valve is the fundamental design primitive (analogous to a gate), while a punch is the external control port (analogous to a pin). The routing needs to respect minimum distances between connection lines, between a valve and another connection line, and between a punch and another connection line. In addition, the routing must avoid some obstacles and minimize crossings of flow lines. In the final routing, we would like to minimize the total length of the connection lines, and also the number of corners, where a corner is defined as a point where a line changes direction (from vertical to horizontal, or horizontal to vertical, assuming a manhattan routing). We minimize corners primarily for aesthetic purposes, as microfluidic designers have shown hesitancy to accept a tool that produces unduly complex layouts. Minimizing the number of corners may also improve the robustness of chip operations.

Outline In section 2, I formulate the routing problem as a min-cost max-flow problem. This formulation doesn’t minimize the number of corners, but only the total length of the connection lines. In section 3, I show how to iteratively improve the min-cost max-flow solution to greedily minimize the number of corners. However, this iterative improvement doesn’t provide any provable guarantees on the minimization of corners. In section 4, I formulate the routing problem as an integer linear program, with one variable for each acceptable

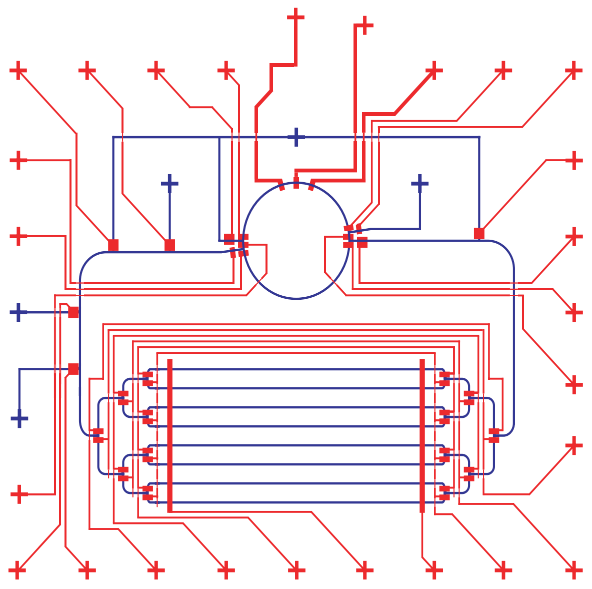

Figure 1: Layout of a microfluidic chip. The flow layer is in blue and the control layer is in red. The valves are rectangles and the punches are pluses.

route between a valve and a punch, and I solve it using the technique of randomized rounding. I explore various formulations of the integer program, which allow me to simultaneously minimize edge capacity, wire length and number of corners. I prove that my solution obtained via randomized rounding will be close to optimum with high probability. In section 5, I formulate the routing problem as a different integer linear program, based on multi-commodity flow, which frees the integer linear program of the dependence between number of variables and possible routes. This formulation allows me to minimize the total wire length and total number of corners at once. Like before, I use the technique of randomized rounding, selecting the final routes using random walks in the network. Again, I prove that my solution will be close to optimum with high probability. Finally, in section 6, I conclude by summarizing and contrasting my results.

2 Min-Cost Max-Flow Formulation

Let me first ignore the issue of minimizing corners. I can represent the problem of routing each valve to a punch, while minimizing total wire length, as a min-cost max-flow problem as follows [8]. I impose on my chip a manhattan grid with resolution equal to the minimum distance between any two connection lines. A grid point is a node, with double directed edges to its upper, lower, left and right neighbors. Each edge has capacity 1 and a cost of 1. We also want each node to have capacity 1, so, as seen in class, I duplicate each node into two nodes, an incoming node, which will take the links to all incoming edges, and an outgoing node, which will take the links to all outgoing edges, and I link the incoming node to the outcoming node with an edge of capacity 1 and cost 0. Then, I create a source node which has edges of capacity 1 and cost 0 to all valves and a sink node which has edges of capacity 1 and cost 0 from all punches. I then simply look for the min-cost max-flow from the source to the sink. The value of the flow is equal to the number of valves connected and the cost of the flow is equal to the total wiring length.

I can easily extend this model to minimize flow lines crossing and support obstacles. In order to minimize crossing of flow lines, I add a certain cost to the edges between incoming and outgoing nodes, for all grid points that are on top of a flow line. In order to avoid obstacles, I simply omit the neighbor edges where one of the grid point involved is on top of an obstacle.

However, it is not so straightforward to extend this model to also minimize number of corners. In [8], the number of corners is minimized by using a 2-grid instead of a simple grid. The idea is to split each grid point into a vertical grid point, which takes the edges to the upper and lower neighbors, and a horizontal grid point, which takes the edges to the left and right neighbors, and, then, to add a more costly edge between each pair of vertical and horizontal grid points. So, this extension uses two layers: one for vertical lines, one for horizontal lines. In microfluidic chips, we cannot afford to have two layers for just the control. In particular, this model allows a horizontal connection line to intersect with a vertical line, because each pair of vertical and horizontal grid points have separate capacities of 1 each. Therefore, this extension as it stands is unsuitable for our purpose. In section 5, we will use it as a starting point for an integer linear program, adding constraints to restrict the flow coming into a grid point pair.

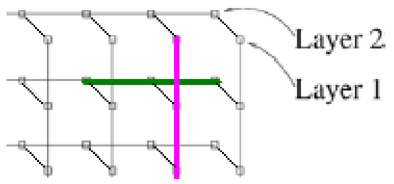

Figure 2: Example of a 2-grid with a vertical line (in purple) and a horizontal line (in green) intersecting.

3 Greedy Corner Minimization

In this section, I describe an ad-hoc method that I devised to iteratively improve the min-cost max-flow solution by greedily avoiding corners.

The basic operation, "rip and redo", is to rip a connection line and, while maintaing all the other connection lines as obstacles, redo the ripped connection using Lee's algorithm [2], and avoiding corners while tracing back. Lee's algorithm is straightforward: I maintain a matrix, with one entry per grid point, and a queue of grid points. I start by setting the matrix entry for the valve grid point to 0 and putting that grid point in the queue. While the queue is not empty, I pop the next grid point out of the queue, and explore the neighbors of this grid point when they are not on obstacles. For each neighbor, if it hasn't been explored yet, I set its matrix entry to the matrix entry of the popped grid point plus 1, and add it to the queue. Once I reach the punch, I am done. The matrix entry of the punch gives me the length of the connection line. I can use the matrix to trace back the actual connection line. I can choose to start the tracing back vertically or horizontally. Then, I maintain the same direction if possible, so as to greedily minimize changes in the directions, which correspond to corners. I choose the tracing back starting vertically or horizontally depending on which gives me the least number of corners.

It is straightforward to take minimization of flow lines crossing into account when doing rip and redo. I simply use a priority queue instead of a FIFO queue, and, when I set a matrix entry, I use a penalty increment if the grid point is on a flow line.

So the strategy to greedily minimize corners is to rip and redo each connection line, many times (ideally, until all the connection lines stop changing – the strategy eventually must reach a fixed point, as the connection lines only changes if they are improved).

Let n be the number of grid points in the grid. Let v be the number of valves. Let k be the number of iterations of ripping and redoing each connection. Then, the running time of the rip and redo phase is $O(kvn \log n)$ (using the priority queue to also minimize flow line crossing). Given that the min-cost max-flow algorithm [1] runs in $O(n^3)$, the running time of the rip and redo phase is acceptable.

In practice, this iterative algorithm performs rather well: I implemented this algorithm as an extension to AutoCAD and evaluated the result on half a dozen real microfluidic designs. However, the corner minimization is only local: it is possible, that even for this particular assignment of valves to punches and this particular wiring length, there exist a solution that has less corners, if only because the connection lines are ripped and redone separately. Indeed, choosing a particular connection line for one pair of valve and punch might block another connection line from realizing a smaller number of corners. So there could be many stable situations that are far from optimum.

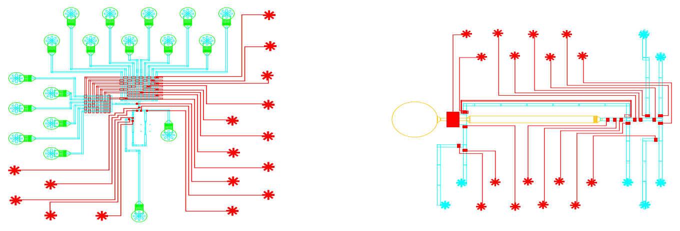

Figure 3: Chips where the connection lines in the control layer were routed by the iterative algorithm.

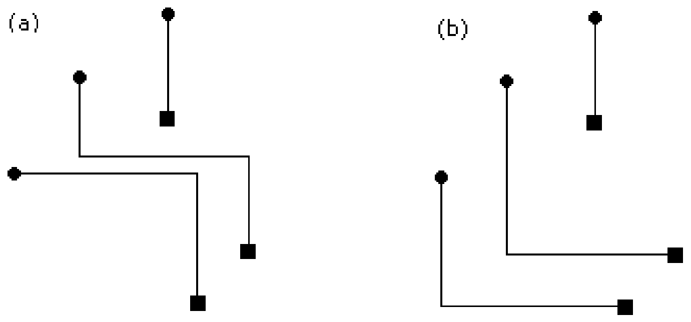

Figure 4: Two stable set of routes, which could conceivably be chosen by the iterative algorithm. Yet, (a) has 3 corners while (b) has 2 corners.

4 Integer Programming Formulation

Now, I would like to explore solutions to the routing problem that provide some guarantee as to how close to optimum they are. In this section and the next, I will formulate the routing problem as an integer linear program, solve the linear program relaxation and use some form of randomized rounding [6, 5, 4, 3] to obtain an integer solution that is provably close to the optimum of the linear relaxation.

In this section, I explore integer linear programs, in which each $0, 1$ variable corresponds to a possible route connecting a valve to a punch. To keep things simple to start with, the first program simply attempts to find a feasible solution among all the possible routes (without optimizing wire length or number of corners further). So let's start by assuming that we find the possible routes by running the min-cost max-flow algorithm of section 2: this gives us an assignment of valves to punches, then, for each valve, we consider all routes going to its punch that are of the same length and have less than a fixed number of corners.

Each valve v has possible routes T_v^j for $1 \leq j \leq I_v$. Let $x_{v,j}$ be the indicator variable that denotes whether valve v is connected using route T_v^j. Then, the traffic on an edge is $U(x, e) = \sum_{(v,j)|e \in T_v^j} x_{v,j}$. Let x_L be the max load of any edge (if we have a feasible solution, then $x_L = 1$). The integer linear program that finds a feasible set of routes is:

$$\begin{aligned}
&\text{minimize } x_L \\
&\quad \text{subject to} \\
&x_{v,j} \in \{0,1\}, \forall \text{ valves } v, \forall j, 1 \leq j \leq I_v \\
&\sum_{j=1}^{I_v} x_{v,j} = 1, \forall \text{ valves } v \\
&U(e) \leq x_L, \forall e \in E
\end{aligned}$$

This ILP has $v + m$ constraints (where v is the number of valves and m the number of edges), and one variable per possible route, in addition to the variable x_L.

Now, I solve the linear relaxation of this ILP, requiring $x_L \geq 0$ and all other variables to come from the interval $[0, 1]$. Call the solution of the resulting LP α. I use randomized rounding to find a solution x to the ILP from the LP with solution α as follows. For each valve v, I have to set one of $x_{v,j}$ to 1 and all the others to 0. Notice the LP constraint $\sum_{j=1}^{I_v} x_{v,j} = 1$. This means that, for each valve v, I can roll a I_v-sided die, where side j has probability $\alpha_{v,j}$, allowing me to pick which $x_{v,j}$ wins in a way such that $E[x_{v,j}] = \alpha_{v,j}$. Suppose I attempt the randomized rounding k times.

First, recall the Chernoff bound.

Chernoff Bound Let the function $\Psi = \sum_{i=1}^{n} c_i X_i$ be the weighted sum of n independent Bernouilli trials. Here, $c_i \in [0, 1], \forall i = 1, \ldots, n$. Let α_i be the probability that trial X_i turns up 1. Assume that $\chi = \mathbb{E}[\Psi]$. Let $\delta > 0$. Then

$$\mathbb{P}[\Psi > (1+\delta)\chi] < \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}} \right)^{\chi}$$

Let's call the event, that the value of Ψ exceeds the expected value χ by more than a factor of $(1 + \delta)$ a δ-failure. The Chernoff bound allows us to compute the probability of a δ-failure given the deviation δ. When we analyze an approximation algorithm, we want the opposite: given a probability ϵ, we want to find some small deviation $\delta = \delta(\chi, \epsilon)$, such that the probability of a δ-failure is smaller than ϵ. Let $B(\chi, \delta)$ denote the right-hand side of the Chernoff

Bound, i.e. $B(\chi, \delta) = \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^\chi$. We define $\delta(\chi, \epsilon)$ formally by the equation

$$B(\chi, \delta(\chi, \epsilon)) = \epsilon$$

Now, we can reverse the Chernoff Bound.

Reverse Chernoff Bound Assume that $\chi > \ln(1/\epsilon)$. Then,

$$\delta(\chi, \epsilon) \leq (e-1)\sqrt{\frac{\ln(1/\epsilon)}{\chi}}$$

Assume that $\chi \leq \ln(1/\epsilon)$. Then,

$$\delta(\chi, \epsilon) \leq \frac{e \ln(1/\epsilon)}{\chi \ln \frac{e \ln(1/\epsilon)}{\chi}}$$

I will use the Reverse Chernoff Bound to analyze the integer solution obtained by randomized rounding.

Consider the constraint for an edge e:

$$U(e) = \sum_{(v,j) | e \in T_v^j} x_{v,j} \leq x_L$$

The left-hand side is a sum of the indepedent Bernouilli trials: for each valve v, the probability that we use edge e is the sum of the $\alpha_{v,j}$ that occur on the left-hand side. By the constraint of the LP, $\mathbb{E}[U(e)] \leq \alpha_L$. In terms of the Chernoff Bound, $\Psi = U(e)$ and $\mathbb{E}[U(e)] = \chi$. Let δ denote $\delta(\alpha_L, \epsilon^{1/k}/m)$, where m is the number of edges. Let us call the event that the randomized rounding tried k times computes a value for x_L that is no greater than $(1+\delta)\alpha_L$ a success, and the opposite event a failure. We will show that the probability of a failure does not exceed ϵ. Since the k tries are independent, we need to show that the probability that one randomized rounding attempt fails is $\leq \epsilon^{1/k}$. Since there are m edges, by the union bound, we need to show that the probability of failure at each edge $\leq \epsilon^{1/k}/m$. Failure at edge e means that the outcome of the Bernouilli trials for this edge yields a sum that exceeds $(1+\delta)\alpha_L$. If $\chi_e = \alpha_L$, then, by definition of δ, this means that the failure happens with a probability smaller than $\epsilon^{1/k}/m$. If $\chi_e < \alpha_L$, then, certainly, the failure happens with an even smaller probability. Thus, expanding the Reverse Chernoff Bound:

Analysis Result Let $\alpha_L > \ln(m/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution whose cost does not exceed

$$\alpha_L + (e-1)\sqrt{\alpha_L \ln(m/\epsilon^{1/k})}$$

with probability greater than $1 - \epsilon$.

Let $\alpha_L \leq \ln(m/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution whose cost does not exceed

$$\alpha_L + \frac{e\ln(m/\epsilon^{1/k})}{\ln\frac{e\ln(m/\epsilon^{1/k})}{\alpha_L}}$$

with probability greater than $1-\epsilon$.

I can extend the algorithm and the analysis to also minimize the total wire length and the total number of corners. These extensions will allow us to include more possible routes as part of the program, and, in particular, use the ILP to also find the assignment of valves to punches. Let x_W be the total wire length of used routes and x_C be the total number of corners of used routes. Let W_v^j be the length and C_v^j be the number of corners of route T_v^j. Then, x_W and x_C can be computed by extending the integer linear program with:

$$\sum_v \sum_{j=1}^{I_v} W_v^j x_v^j \leq x_W$$

$$\sum_v \sum_{j=1}^{I_v} C_v^j x_v^j \leq x_C$$

We also change the objective of the integer linear program to first minimize x_L, then x_W, then x_C (by order of priority). In practice, we can maintain this order of priority by weighting x_W on an upper bound on the value of x_C and weighting x_L on an upper bound on the value of x_W (including its weight) and x_C, or we could just run the linear program to first minimize x_L, add this requirements for this minimization as constraints, then minimize x_W, etc.

Like before, we solve the LP relaxation of this ILP and use the same technique of randomized rounding, with k attemps. We analyze independently the probability of meeting each objective (x_L, x_W, x_C). For x_W and x_C, the analysis is similar to the previous one for x_L, except that we must scale the values so that the coefficients c_i for the Chernoff Bound are between 0 and 1. So let $W = \max W_v^j$ and $C = \max C_v^j$. We simply divide the constraints equations for x_W and x_C by W and C respectively. Here, the independent Bernouilli trials correspond to the (proportioned) wire length and (proportioned) number of corners, respectively, for the route used for each valve. As another difference with the x_L case, we don't use the union bound because we only have one set of independent Bernouilli trials to consider. Therefore, we get:

Analysis Result for Total Length Let $\alpha_W > W\ln(1/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution in which the total wire length does not exceed

$$\alpha_W + W(e-1)\sqrt{\alpha_W \ln(1/\epsilon^{1/k})}$$

with probability greater than $1-\epsilon$.

Let $\alpha_W \leq W \ln(1/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution in which the total wire length not exceed

$$\alpha_W + W \frac{e \ln(1/\epsilon^{1/k})}{\ln \frac{e \ln(1/\epsilon^{1/k})}{\alpha_W}}$$

with probability greater than $1 - \epsilon$.

The result for number of corners is similar, replacing all W's with C's.

Analysis Result for Total Number of Corners Let $\alpha_C > C \ln(1/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution in which the total number of corners not exceed

$$\alpha_C + C(e-1)\sqrt{\alpha_C \ln(1/\epsilon^{1/k})}$$

with probability greater than $1 - \epsilon$.

Let $\alpha_C \leq C \ln(1/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution in which the total number of corners does not exceed

$$\alpha_C + C \frac{e \ln(1/\epsilon^{1/k})}{\ln \frac{e \ln(1/\epsilon^{1/k})}{\alpha_C}}$$

with probability greater than $1 - \epsilon$.

One issue with all these integer linear program is that the number of variables depend on the number of routes, which grows exponentially with the number of corners. Let n be the number of grid points. Then, even if we fix an assignment of valves to punches, if we allow k corners per route, the possible number of routes for each valve is $O(n^{k-1})$. The next section solves this issue by considering routes implicitly as opposed to explicitly.

5 Integer Programming Formulation based on Multi-Commodity Flow

I now propose a different formulation of the routing problem as an integer linear program, which will avoid this exponential increase in variables. My starting point is the 2-grid presented in section 2. I use the min-cost max-flow formualtion, except that I don't need the source anymore, and instead ask to transport v commodities of flow 1 each from each valve to the sink. It is straightforward to translate this multi-commodity min-cost flow into an integer linear program. In order to prevent vertical lines and horizontal lines intersecting, I add the following constraints to the linear program: the total flow coming into each pair of vertical and horizontal nodes must be ≤ 1 (by flow conservation, the total flow coming out of the pair will also be ≤ 1). By minimizing the cost of this modified multi-commodity flow ILP, I automatically minimize the total wiring length including corners (included as more expensive wires).

Like before, I solve the LP relaxation of this ILP. Call the solution to the LP relaxation, α. Now, I use randomized rounding to convert the fractional flow for valve v into a route. For each valve, I perform a random walk from the valve to the sink. The random walks for different valves are probabilistically independent.

Consider a valve v. I convert the routing graph into a directed graph by deleting all edges that have zero flow for this valve viewed as a commodity, then by directing all remaining edges in the direction of positive flow. I perform a random walk on the resulting network. I start at the valve v, the source node. In general, if I am currently at a node ω, I choose an edge e out of ω with probability

$$\frac{\alpha_v(e)}{\sum_{e' \in E_{\omega\rightarrow}} \alpha_v(e')}$$

where $\alpha_v(e)$ denotes the flow of valve v viewed as a commodity along edges e in the optimal solution α.

For the analysis, note that, even though the choices of edges along the same random walks are dependent, the different random walks are independent so the requirements of the Chernoff Bound are met: the Bernouilli trials contribution to the same edge constraint are independent. Edge e is on the random walk for valve v with probability that does not exceed $\alpha_v(e)$ (this can easily be shown by induction). Therefore, the expected value of the sum of all Bernoulli trials contribution to the same edge constraint never exceeds α_L, and so the same result as that of section 4 applies.

Analysis Result for Max Edge Load Let $\alpha_L > \ln(m/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution whose max edge load does not exceed

$$\alpha_L + (e-1)\sqrt{\alpha_L \ln(m/\epsilon^{1/k})}$$

with probability greater than $1-\epsilon$.

Let $\alpha_L \leq \ln(m/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution whose max edge load does not exceed

$$\alpha_L + \frac{e\ln(m/\epsilon^{1/k})}{\ln \frac{e\ln(m/\epsilon^{1/k})}{\alpha_L}}$$

with probability greater than $1-\epsilon$.

Using scaling as in section 4, I can also prove that the total wiring length (including corners penalty) will be close to optimum with high probability, though the result is not very tight because the scaling factor is rather large (here, I have to scale by an upper bound on the length, including corner penalties, of any route). More specifically, let α_W be the total wiring length used by the connection and W an upper bound on the length (including corner penalties) of any route. Then,

Analysis Result for Total Length Let $\alpha_W > W \ln(1/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution in which the total wiring length does not exceed

$$\alpha_W + W(e-1)\sqrt{\alpha_W \ln(1/\epsilon^{1/k})}$$

with probability greater than $1-\epsilon$.

Let $\alpha_W \leq W \ln(1/\epsilon^{1/k})$. Then, the randomized rounding algorithm obtains a solution in which the total wiring length does not exceed

$$\alpha_W + W \frac{e \ln(1/\epsilon^{1/k})}{\ln \frac{e \ln(1/\epsilon^{1/k})}{\alpha_W}}$$

with probability greater than $1-\epsilon$.

6 Conclusion

In this paper, I explore the routing problem for the control layer of microfluidic chips. I have found an algorihtm that works well in practice, optimizing the total wiring lenght while using a heuristic approach to minimize number of corners. Since the practical algorithm didn't have any guarantees regarding the corner minimization, it was interesting for me to explore more theoretical solutions, with provably good results. The theoretical algorithms guarantee that the wiring length and number of corners will be close to optimum with high probability.

The algorithm in section 5 is interesting because, while incorporating the minimization of corners, it remains polynomial, and yet, is provably close to optimal with high probability. Though I will probably not use it in practice, it is very satisfying to have been able to adapt the intuitive 2-grid model of section 2 to obtain this theoretical result.

Through this project, I learned a lot about the technique of randomized rounding. It is a powerful technique, because it can prove that its result will be close to optimum with high probability. In addition, for certain problems including those presented in this paper, it is possible to de-randomize the algorithm so as to deterministically find a good solution. The trick is to construct a probabilistic tree of the rounding options using the solution to the LP to estimate the probability of ending up in a "good" leaf. By using pessimistic estimates, we can guarantee ending up in a "good" leaf. This method, called method of conditional probabilities, is detailed in [5, 4, 3].

Finally, I want to mention that we have produced an AutoCAD plugin, developed iteratively with microfluidic researchers over the last year, to provide a robust framework for deploying these results. We have publicly released the tool, available at http://cag.csail.mit.edu/biostream/cad/, and it is being used in the daily design of microfluidic chips.

7 Acknowledgments

I thank Prof. David Karger for pointing out the technique of randomized rounding and Bill Thies for discussing this project with me and giving me feedback.

References

[1] R. K. Ahuja, A. V. Goldberg, J. B. Orlin, and R. E. Tarjan. Finding minimum-cost flows by double scaling. *Math. Program.*, 53(3):243–266, 1992.

[2] C. Lee. An algorithm for path connections and its applications. *IRE Transactions on Electronic Computers*, EC-10(2):364–265, 1961.

[3] T. Lengauer. *Combinatorial algorithms for integrated circuit layout*. John Wiley & Sons, Inc., New York, NY, USA, 1990.

[4] R. Motwani and P. Raghavan. *Randomized algorithms*. Cambridge University Press, New York, NY, USA, 1995.

[5] P. Raghavan. *Randomized rounding and discrete ham-sandwich theorems: provably good algorithms for routing and packing problems (integer programming)*. PhD thesis, University of California at Berkeley, 1986.

[6] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[7] J. Urbanksi, W. Thies, C. Rhodes, S. Amarasinghe, and T. Thorsen. Digital microfluidics using soft litography. *Lab on a Chip*, 6(1):96–104, 2006.

[8] H. Xiang, X. Tang, and D. F. Wong. Min-cost flow-based algorithm for simultaneous pin assignment and routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 22(7):870–878, July 2003.