CS 220 Synthesis of Digital Systems

University of California, Riverside

Fall 2014

Project: Line Routing Algorithms

In this project, the student will implement a collection of line routing algorithms in C++. These algorithms will be integrated into a framework for the design of microfluidic chips based on integrated microvalve technology that is under development in Dr. Brisk’s laboratory. Line routing is implemented as a standalone module within the framework.

Course TA Brian Crites ([bcrit001@ucr.edu](mailto:bcrit001@ucr.edu)) will provide students with access to the framework. Please contact Brian via email to initiate your project. This project is organized as a series of mini-assignments, 1-3 weeks in length. Students should consult with Brian after completing each mini-assignment to verify its correctness and coding style. Brian will provide a document describing coding standards to the students.

The microfluidic chip architecture is represented as a graph called a netlist. Vertices correspond to components (valves, mixers, etc.) and edges (called “nets”) correspond to fluid channels between components. Prior to routing, the microfluidic chip is placed (either algorithmically or by hand). Placement determines the location of each component and each component’s I/O interface ports in the 2D plane. Grid or line routing is then invoked to establish connections between components, allowing for the transfer of fluid.

The student is expected to implement two versions of each line routing algorithm:

1. No intersections are allowed between routed nets.
2. Unlimited intersections are allowed between routed nets (although nets may intersect at crosspoints, two nets are not allowed to route on top of one another).

For any given algorithm, the code for both versions should be similar, and maximal code sharing (within reason) for each version is expected.

All of the algorithms to be implemented in this project route one net a time. The results obtained are highly sensitive to the order in which nets are routed. Some algorithms attempt to select intelligent orderings of nets; others do not. If the algorithm does not explicitly state how a net ordering is computed, the nets should be routed in the order that they are specified in the input file to the framework.

Mini-project #1

Duration: Two weeks

1. Implement the Mikami-Tabuchi [1] line routing algorithm

Mini-project #2

Duration: Two weeks

1. Implement Hightower’s [2] line routing algorithm

Mini-project #3

Duration: Two weeks

1. Implement Heyns’ [3] line routing algorithm

Mini-project #4

Duration: Two weeks

1. Implement a bi-directional [4] variant of the Mikami-Tabuchi algorithm [1].
2. Implement a bi-directional [4] variant of Hightower’s algorithm [2].
3. Implement a bi-directional [4] variant of Heyns’ algorithm [3].

References

[1] Koichi Mikami, Kinya Tabuchi: A computer program for optimal routing of printed circuit conductors. IFIP Congress (2) 1968: 1475-1478

[2] David W. Hightower: A solution to line-routing problems on the continuous plane. DAC 1969: 1-24

[3] Walter Heyns, Willy Sansen, Herman Beke: A line-expansion algorithm for the general routing problem with a guaranteed solution. DAC 1980: 243-249

[4] Ira Pohl: Bi-directional Search, in Meltzer, Bernard; Michie, Donald, Machine Intelligence 6, Edingburgh University Press: 127-140 (1971)