(Board 53/Work in Progress) Engaging the Next-Generation of IC Designers with Puzzle-Solving Competitions

Prof. Daniel Limbrick, North Carolina A&T State University

Dr. Daniel Limbrick is an associate professor in the Electrical and Computer Engineering Department at North Carolina Agricultural and Technical State University (NC A&T). As director of the Automated Design for Emerging Process Technologies (ADEPT) laboratory, Dr. Limbrick investigates ways to make microprocessors more reliable and secure through cross-layer design.

Laura Marcela Garcia Suarez
Deriech Cummings II, North Carolina A&T State University

WIP: Engaging the next generation of IC designers with puzzle-solving competitions

Daniel Limbrick, Laura Marcela Garcia Suarez, Deriech Cummings
Department of Electrical and Computer Engineering
North Carolina Agricultural and Technical State University, Greensboro, NC 27411
Email: daniel.limbrick@ncat.edu, lmgarciasuarez,dlcummings@aggies.ncat.edu

1 Introduction

The traditional path to learning how to design integrated circuits (ICs) with very-large-scale integration (VLSI) requires undergraduate students to take courses in digital logic, introductory circuits, electronics, and introductory VLSI design. If they are lucky, by senior year, they have mastered the art of designing a logic primitive, e.g., a CMOS inverter or a simple circuit, e.g., a full adder. With this knowledge, they have the foundation to pursue graduate studies in VLSI design, which, depending on the university, includes a course that introduces them to logic synthesis and physical design. This approach to teaching VLSI has a lot of shortcomings: (1) students with the desire to design a microprocessor from scratch have to delay gratification for several years, (2) students must appreciate and have a strong aptitude for each step of the VLSI design process in sequential order, (3) most universities do not offer the full sequence of courses needed, and (4) exposure to skills needed is usually not available until the students have already committed to a career path.

This paper describes an approach to expose students to advanced VLSI concepts and algorithms needed to design a complex IC by having them compete in a puzzle solving competition. The game requires a player to connect dots with lines while satisfying certain constraints. The objectives of the game can be achieved using VLSI physical design concepts, creating a pathway for students of all ages to discover VLSI design.

The remainder of the paper is organized as follows. Section 2 provides a background on Flow Free. The objectives of the game are related to VLSI concepts in Section 3. Section 4 provides a description of graph-based approaches to solving the game and explains how these approaches relate to VLSI. The authors' experience with using Flow Free as a teaching tool and a proposed competition format is presented in Section 5. Section 6 summarizes the work.

2 Puzzle-Solving with Algorithms

There are many puzzles today that are in a classes of problems called NP-complete and NP-hard. NP-complete problems are hard to solve efficiently, especially as the input size of the problem

increases. It is, however, easy to verify whether a solution is right or wrong. NP-hardness refers to the complexity of decision problems for which no known polynomial-time algorithm exists to solve them. The most popular versions of puzzles that fall in this class provide opportunities to teach programming skills and engineering problems to students without needing to provide context for the problem.

Minesweeper is one such game [1]. This game involves finding landmines in a grid using the hints given at a particular tile. The hint given is the amount of mines around a certain tile. A player must then select tiles using the hints uncovered after its selection, all while avoiding selecting a landmine instead. Minesweeper-like solutions can be seen in some applications. Richard Kaye proved that minesweeper in NP-complete by reducing the game to SAT. SAT is an NP-complete problem that involves Boolean circuits and their respective inputs and outputs. The problem is correctly guessing a combination of inputs that, when fed into a Boolean circuit, outputs TRUE. Kaye decided to show that minesweeper is NP-complete by making Boolean circuits with minesweeper components. These "computers" resemble minesweeper grids, but had properties of logic gates (NOT, AND, OR, etc). This way, any SAT problem could be converted into a minesweeper counterpart. Becker used Minesweeper to teach first year computer science students about implementing random numbers, 2-D arrays, and functions in C++. It was noted that the interest level of the students was higher than with games used in previous years because they had strong familiarity with the game prior to taking the course. Ben-Ari [2] showed solutions to minesweeper which included step-by-step simulation of digital circuit elements required for proving NP-completeness.

Sudoku is also an NP-complete problem [3, 4]. There are papers detailing various aspects of using Sudoku for testing algorithms and solutions, from recreating Sudoku puzzles from a picture of them (Image recognition) to ways to efficiently solve them (beta-hill climbing, genetic algorithms, Monte-Carlo Tree Search). The ubiquity of games like Sudoku makes it a good starting point for introducing students to problems that are solved by trying out different solutions and comparing them to the already realized ones. This also means that finding a solution for Sudoku that solves it efficiently can provide a gateway to solving other NP-complete problems, such as path finding through the traveling salesman problem, cryptography with prime factorization, and checking circuits with SAT.

This background motivates the use of Flow Free to introduce students to VLSI physical design problems. Flow Free is a game that requires a player to connect dots with lines while satisfying certain constraints. Such a game would instill VLSI physical design concepts and be accessible to students of all ages. Flow Free (shown in Figure 1), runs on Android/iOS platforms. The game's objective is similar to the wire-routing problem in integrated circuit design. Flow Free has three constraints: (1) dots of the same color must connect, (2) lines cannot overlap, and (3) every space must be filled with a line. Flow Free, under certain constraints is also an NP-complete problem; This was found by James F Lynch in his paper, "The Equivalence of Theorem Proving and the Interconnection Problem," where he reduces a series of boolean logic (similar to SAT, that we know is NP-complete) into an interconnection problem (similar to flow free) [5]. This way, a solution to SAT can also be a viable solution to flow problems. Flow free can be used as a foundation for VLSI routing problems, where processors need proper and efficient connection points from one transistor to another. Because of its NP-completeness, a solution for any of these problems leads to more efficient connections, which leads to better and more efficient computers, something that would drastically change the way computers are made.

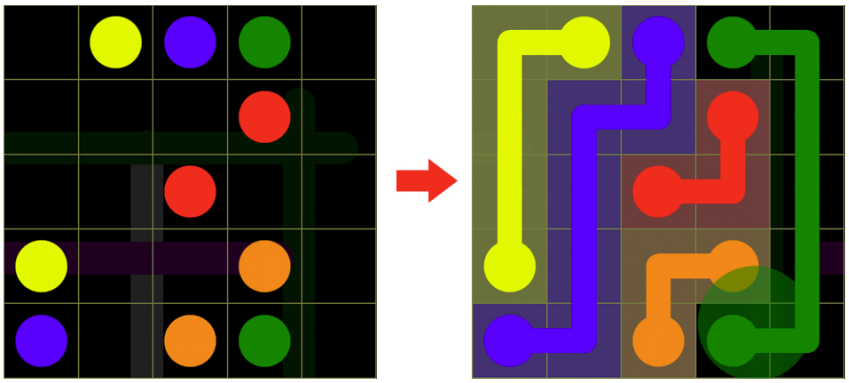

Figure 1: Flow Free

3 Connections to VLSI Education

The evolution of semiconductor technology has reached a critical point where traditional scaling methods alone are no longer enough to meet the increasing demands for overall system performance. While trying to make transistors faster by shrinking the dimensions of the circuit, the performance of the system suffers because of cross-section scaling of on-chip and off-chip interconnects [6].

To solve this, new technologies have emerged and with them a large amount of routing challenges. This literature review explores the impact of emerging technologies on routing techniques, and how these constraints map to versions of Flow Free. The focus will be in the following technologies: 3-D integration of CMOS-based ICs and graphene nanoribbons, each one impacting routing methodologies and algorithms differently, based on their constraints and characteristics.

3.1 Planar CMOS Wire Routing

Routing is the process of selecting a path that connects two points and satisfies a set of constraints. There are multiple areas of circuit design that incorporate routing algorithms, including the physical design of the interconnecting wires for transistor devices and mapping droplet paths for biochips.

In electronic design automation, there are several steps that translate the high-level functional description of a circuit design into a physical implementation of the circuit. In a typical EDA design flow, design creation begins with a written hardware description at the register-transfer level (RTL). This RTL description is then mapped to a specific technology library as a netlist in a process known as logic synthesis. Exact locations for the circuit components in the netlist are assigned during the placement step. Finally, the routing step adds the wires needed to properly connect the placed components while obeying all design rules for the circuit.

An example of this design step can be seen in Figure 2. An active area of research is to find ways to connect modern devices in a given space with wires. The length of the wire must be minimized to reduce power and delay. Additionally, unrelated wires cannot intersect in order to prevent shorted signals. Finally, the ability to route a design is influenced by the resources

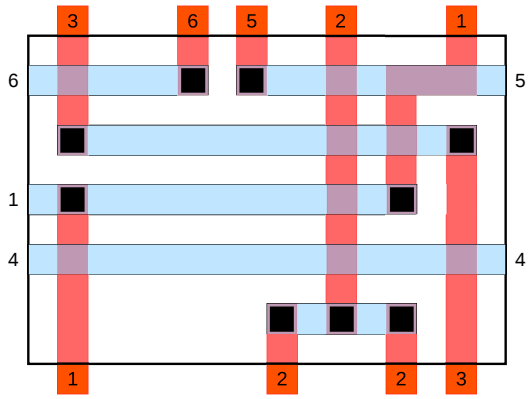

Figure 2: Example of wire routing.

(open spaces) available. When not enough resources are available, *routing congestion* can occur, preventing a solution. Routing congestion has already been established as a growing bottleneck to transistor scaling in deep sub-micron transistor technologies due to the reverse scaling of wires [7]. The routing design problem is generally similar to classic puzzles that involve connecting similar dots and specifically similar to FlowFree.

FlowFree is related to the wire routing problem in the following ways:

- **Specific connections:** Each puzzle contains one set of connections for each color. This color-coded scheme is similar to a cell-level netlist in wire routing, which is also a fixed input.
- **Finite Resources**: FlowFree can be played on a square grid of at least 5 squares by 5 squares dimension (larger grids are available). Based on the number of nodes, the number of available spaces is fixed. This is similar to a placed chip, where the nodes and the channel resources are given.

FlowFree is dissimilar to the wire routing problem because it requires that all empty spaces must be filled in order to move to the next level. In traditional wire routing problems, the goal is to minimize wirelength, not maximize it! For the competition, we relax this constraint in FlowFree in order to more closely resemble wire routing.

3.2 3-D Integrated Circuits

3-D die stacking integration has been one of the most optimistic alternatives to the limitations encountered in 2-D integration and the constant approximation to the ceiling of Dennard scaling law. This technology consists of stacking layers of logic and connecting them through vertical links. These vertical connections are most commonly Through-Silicon Vias (TSV). TSVs, however, come with a significant number of limitations that can be summarized in the presence of a limited vertical bandwidth [8].

The idea of routing in 3-D systems began by tweaking the usual ways of directing traffic in 2D setups. The initial approach was stretching the existing 2D routing tools to handle the complexities of 3D layouts, or in other words, solving in a 2-D manner each layer and then performing the

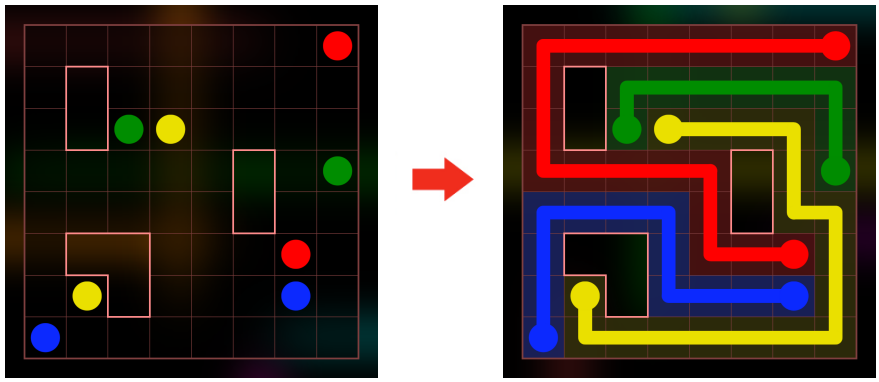

Figure 3: Scattered Pack in Flow Free

vertical connections within the layers. This pseudo-3-D routing approach helped to get a basic grasp of how 3D routing might work.

One example of pseudo-3-D routing algorithms was proposed in [8] where the problem of limited number of vertical links available is addressed. Given the high area consumption of TSVs, and fabrication limitations, the bandwidth of vertical links is relatively smaller than that of horizontal links. This constraint also applies for other vertical link methods. This approach is considered to be one of the first ones to solve the before mentioned problem with a routing algorithm instead of focusing on the hardware design of the chip.

The bandwidth of vertical links is low compared to that of horizontal links. [8] proposes a traffic distributing routing algorithm that attempts to create efficient vertical links without changing the structure of the TSVs, different to what was proposed by earlier algorithms like AdaptiveXYZ. This approach innovated compared to other routing algorithms by taking highly probable congestions in vertical links into consideration.

The algorithms needed to route the connections in 3D IC wire routing relates to the ”Scattered Pack” in Flow Free (shown in Figure 3).

3.3 Graphene nanoribbon

Graphene nanoribbons (GNR) are one of the most promising forms of graphene for use in electronics because of their special electrical properties [9]. One of the most relevant properties is its ability to change its electrical behavior by changing the shape of its edges: zigzag and armchair. Both structures are formed with hexagonal shapes, but with different edge orientation. In order to keep the same structure, thus same electrical properties, it is only allowed to bend 0°, 60°, and 120°. Bendings of 30°, 90°, and 150° would change the chirality of the GNR, and its behavior from metallic to semiconductor, or vice versa. For that reason, it is necessary to use a routing grid that only allows those three angles. Therefore, instead of using the traditional routing grid for Steiner trees, [10] proposes a triangular grid where the before mentioned bendings are the only ones allowed.

Like in traditional routing problems, the resistance depends on the length of the interconnect.

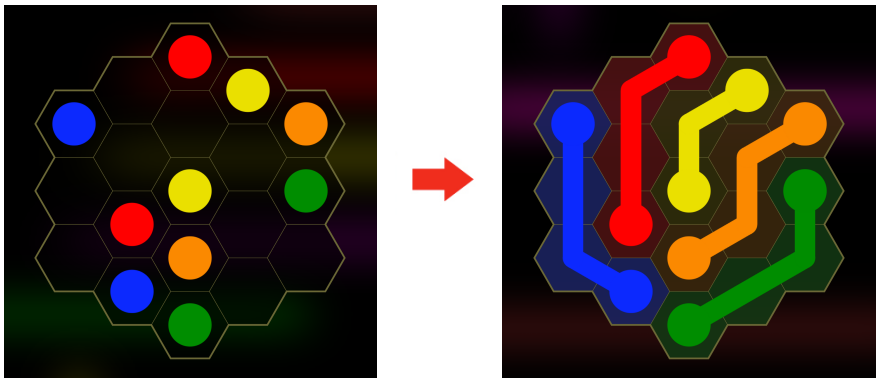

Figure 4: Hexes Pack in Flow Free

In this routing problem the bending angle of the interconnect also causes resistance. The resistance cost varies depending on the number of bendings and their respective angles. The total resistance cost due to interconnect length and bending is known as hybrid cost [10]. The goal of this algorithm is then to find a minimal hybrid cost and replacing 120° bendings when possible.

The algorithm starts by selecting the sink terminal with the farthest distance from the source and calculating its possible hybrid costs. Then the appropriate bending path is chosen based on the average distances of the remaining sink terminals after selecting one of the bending paths. After this, the grid is separated into clusters, which are formed depending on the position of the sink terminals. The idea of clusters is expanded in [9], where a total of 4 clusters are formed. Then the possible Steiner points of each cluster are found and selected.

In addition to the expansion of the clusters, [10] modifies this algorithm by adding a rerouting function for hexagonal obstacles placed in the grid. By using the coordinates of the hexagon's vertices, it is possible to calculate an alternative route that surrounds the obstacle. However, both of these algorithms do not consider crosstalk.

The algorithms needed to route the connections of graphene nanoribbons relates to the "Hexes Pack" in Flow Free (shown in Figure 4).

4 Solutions from the Electronic Design Automation Domain

4.1 Solutions Based on Graph Theory

The definition of the game, provided a grid with empty cells (with potential to be used as paths for connections) and a set of cells of different colors make it possible to define it as a graph structure. By providing attributes to each color, this game becomes a very versatile representation of VLSI concepts. For example, recognizing available vertical and horizontal cells for a specific dot would allow us to represent a connectivity graph. Figure 5 shows an example representation.

In the connectivity graphs the routing regions are represented with vertices that contain vertical and horizontal weights. These weights show the routing availability of that region. This step is important to identify high congestion regions and be able to rearrange the design as required.

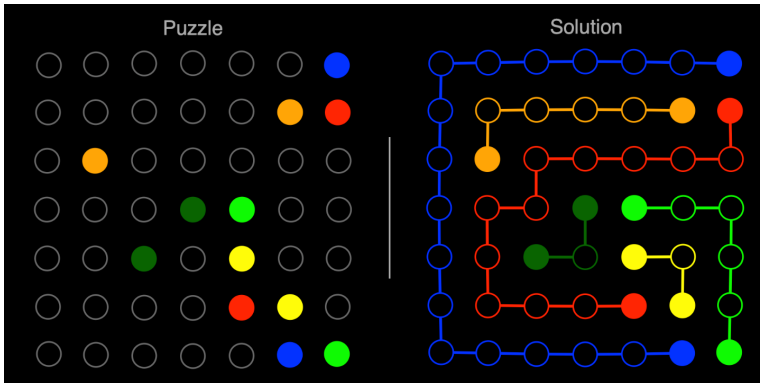

Figure 5: Graph-based representation of Flow Free

Shortest path algorithms are valuable tools for determining the shortest routes between two nodes within a routing graph. Also, these algorithms are characterized for having optimization capabilities. Djikstra's algorithm is well known for its low-cost updates. This means that the addition of a new node requires little computation, because information previously obtained can be used to find the traversal cost of the new node. Also, Djikstra's weights can not only represent paths but also a variety of objectives like electrical properties and routing congestion, as long as they are represented by positive weights.

The A* algorithm performs similarly to Djikstra's, except that only the most promising nodes are calculated. Given an estimated distance for two nodes, only those that seem to be able to provide the shortest distance are calculated. It reduces computation by eliminating large part of the search space.

4.2 Limitations

It is also important to recognize the characteristics and limitations of the game to properly connect these concepts together. VLSI wire routing aims to minimize the wires needed to connect components in order to reduce the delay of the traveling signal. It is evident that wirelength minimization is not a goal in Flow Free. One of the rules of the game consists in occupying every single cell of the grid, meaning that "snaking" or de-routing to longer paths may be necessary if the provided solution has empty cells in it.

Additionally, modern VLSI circuits are routed using multiple metal layers that build in the z direction. However, there is no multi-layer representation of the game. Normally in a 2-D circuit, each cell has a vertical and horizontal capacity. In Flow Free, however, each cell can only be occupied once ignoring whether it is part of a vertical or horizontal connection. This means that 3-D ICs are also a difficult concept to visualize using Flow Free. The closest alternative to this is the "Bridges Pack," which allows for connections to be made over other connections. There are no multi-pin connections. Each puzzle consists of a set of paired colored dots, meaning that only two pin sub-nets can be visualized.

Finally, in the game there are no prioritized nets. Starting with any connection does not alter the final result. However, this diverges from the realistic constraints of timing-driven wire routing, where nets have specific timing and length requirements.

5 Competitions and Teaching Exercises

5.1 High School Group Implementation

The algorithm design competition was implemented in the high school outreach program, STEM Scholars. STEM Scholars was an academic year-long program composed of bi-weekly hour-long seminars that teach students (see Figure 6) through hands-on training of Ubuntu Linux, bash shell programming/scripting, and gnuplot data plotting software. Additionally, the students compete in teams to design an algorithm that solves a puzzle in Flow Free.

Figure 6: High school students developing their routing algorithm

This program aimed to engage high school students by using example problems that are relevant to their current studies. Therefore, math problems were taken from a Scholastic Aptitude Test (SAT) workbook as well as a Calculus textbook. An additional goal is that the students continue the exercises independent of the seminar. To this end, the puzzle game can be accessed from a smart phone and the tools that are used in the seminar are freely available.

The students use the knowledge they acquired (e.g., algorithms, technical presentation) from the bi-weekly seminar to develop an algorithm to solve the puzzles in the game. The algorithms consist of a series of written steps to solve the puzzle without knowing the layout of the puzzle in advance. The students compete to create the best algorithm based on the following criteria: (1) the highest number of puzzles solved, (2) the fewest number of steps in the algorithm, and (3) the best presentation of the algorithm. The team with the best overall algorithm will received a $25 gift certificate for each team member.

5.1.1 Interactive Exercises

In order to teach the students the value of giving explicit instructions the students were required to navigate their peers across the room. One student volunteered to be blindfolded and two students volunteered to give instructions. The blindfolded student had the objective of traveling from one side of the room to the other side based purely on the instructions of his/her peers. This exercise emphasized the level of precision necessary to describe a sequence of steps.

The students were asked to stand in two rows, forming walls. Six students volunteered to stand at specific locations between these walls. Their position was meant to mimic the colored dots from Flow Free. Another student volunteered to solve the puzzle blindly (i.e., without looking at the other students). The information for the position of the students was given to the “blind” student

Figure 7: Example of terms given to participants

through an oral description by the other students. After solving the puzzle, the students discussed what they learned about defining a problem for a computer to understand (e.g., specifying positions relatively).

5.2 High School Individual Implementation

A group of 6 high school students were invited to compete in a Flow Free solving competition individually. Each student was tasked with creating a flowchart that solved the puzzles generally. To facilitate common language, they were given a glossary of terms and coached on it's usage (shown in Figure 7).

The flow charts were evaluated on their ability to solve 10 puzzles at random based on three criteria:

- **Accuracy:** How many puzzles can it solve? If the algorithm cannot find the final solution of a puzzle, then it will count as not solved. The algorithm must provide a "legal solution".
- **Efficiency:** How many steps does it take for the algorithm to complete a puzzle? These steps are divided in search and traversal. Search is the action of checking a group of cells until the evaluators found one that meets the requirements of the instruction. The traversal is making the step from one cell to another.
- **Clarity:** In how many ways can we interpret each instruction? If the instructions were unclear, the evaluators attempted to choose the least convenient way to follow them.

Of the six students, only two were able to submit a flow chart. Feedback from the participants was that in-person activities was preferred to individual effort. Additionally, the evaluation criteria proved to be too ambiguous to provide deterministic scores. Future iterations will require a

graph-based representation of the board to connect solutions to graph algorithms with well-known performance.

6 Summary

This paper outlined the connections between VLSI physical design algorithms and a popular puzzle game, Flow Free. By using a popular game that is accessible on computers and smart phones, students can engage with a familiar problem to learn unfamiliar concepts in VLSI that would not be seen until the graduate level. The experiences incorporating this game in the classroom provide evidence that Flow Free would be useful in many settings.

References

[1] R. Kaye, "Minesweeper is np-complete," *The Mathematical Intelligencer*, vol. 22, pp. 9–15, Mar 2000.

[2] M. M. Ben-Ari, "Minesweeper as an np-complete problem," *SIGCSE Bull.*, vol. 37, p. 39–40, dec 2005.

[3] G. Kendall, A. Parkes, and K. Spoerer, "A survey of np-complete puzzles," *ICGA Journal*, vol. 31, pp. 13–34, 03 2008.

[4] T. YATO and T. SETA, "Complexity and completeness of finding another solution and its application to puzzles," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E86-A, 05 2003.

[5] J. F. Lynch, "The equivalence of theorem proving and the interconnection problem," *SIGDA Newsl.*, vol. 5, p. 31–36, sep 1975.

[6] E. Beyne, "The 3-d interconnect technology landscape," *IEEE Design & Test*, vol. 33, no. 3, pp. 8–20, 2016.

[7] D. Sylvester and K. Keutzer, "Rethinking deep-submicron circuit design," *Computer*, vol. 32, pp. 25–33, 1999.

[8] M. Zhu, J. Lee, and K. Choi, "An adaptive routing algorithm for 3d mesh noc with limited vertical bandwidth," in *2012 IEEE/IFIP 20th International Conference on VLSI and System-on-Chip (VLSI-SoC)*, pp. 18–23, 2012.

[9] S. Das and D. K. Das, "Steiner tree construction for graphene nanoribbon based circuits in presence of obstacles," in *2018 International Symposium on Devices, Circuits and Systems (ISDCS)*, pp. 1–6, 2018.

[10] S. Das, S. Das, A. Majumder, P. Dasgupta, and D. K. Das, "Delay estimates for graphene nanoribbons: A novel measure of fidelity and experiments with global routing trees," in *2016 International Great Lakes Symposium on VLSI (GLSVLSI)*, pp. 263–268, 2016.