# 1 Information

Student : Ziyang Jin (f4a0b@ugrad.cs.ubc.ca)
Supervisor : William S. Evans (will@cs.ubc.ca)
Time : 2018 - 2019 Winter Term 1 [Sep - Dec]
Topic : Graph Theory, Graph Drawing, and Graph Algorithms

# 2 Topics

1. Basics in graph theory
2. Planar graphs
   - Definition and characterization of planar graphs
   - The left-right planarity test
   - Algorithms in planar graphs
3. Graph drawing
   - Draw a planar graph on a grid
   - Schnyder's algorithm
   - Tutte's algorithm
   - Force-directed Methods
4. Graph Algorithms
   - Karger's algorithm
   - Dijkstra's algorithm using Fibonacci heaps
   - Kosaraju's algorithm
5. Spectral graph theory
6. Random walk on graphs
7. Ramsey's theory

# 3 Materials

## Papers

- *The Left-Right Planarity Test* by Ulrik Brandes
- *How to Draw a Planar Graph on a Grid* by Hubert de Fraysseix, János Pach, and Richard Pollack
- *Embedding Planar Graphs on the Grid* by Walter Schnyder
- *How to Draw a Graph* by William Thomas Tutte
- *Maximum Flows and Parametric Shortest Paths in Planar Graphs* by Jeff Erickson

## Textbooks

- *Algorithm Design* by Jon Kleinberg and Éva Tardos
- *Introduction to Algorithms* by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein
- *Graph Theory* by John Adrian Bondy and Uppaluri Siva Ramachandra Murty
- *Graph Drawing - Algorithms for the Visualization of Graphs* by Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis

## Lecture notes and online resources

- *UBC CPSC 536E* course website: `https://www.cs.ubc.ca/~will/536E/`
- *Spectral Graph Theory* lecture notes by Daniel A. Spielman

# 4 Notes

## Week 0

`Meeting: 2018-09-11, Tue, 11:30 - 12:40, at Will's office`

### What we did

We started to figure out the topics for the direct study and search out for materials like websites and textbooks related to these topics. We also discussed Ramsey number and proofs for $R(3,3) = 6$. Unfortunately the argument seems cannot be applied to bigger Ramsey numbers. We also discussed some examples of random walk.

### What I learned from discussion

- Matrix representation of graph is quite useful. It is normally used in computing paths. Also, one question we want to ask about a graph is how connected the graph is. That has something to do with the eigenvalues of the matrix representation of graph. Google's page rank is an example of applying the use of eigenvalues in graph representation. Also, matrix representation of graphs can also be used in graph drawing.
- A clique in graph is a complete graph with n nodes.
- Ramsey number $R(x, y)$ is defined as the smallest complete graph s.t. for all choices of red/blue colorings, there (unavoidably) exists a red x-clique or a blue y-clique. $R(3,3) = 6$, and $R(4,4) = 18$, and $R(5,5)$ has not be confirmed yet, but it is $\geq 43$ and $\leq 48$. Based on the computing power human beings currently have, brute force will not for finding Ramsey numbers.

### To-do before next meeting

- Borrow "Bondy and Murty" and "Di Battista et al." from UBC CS reading room.
- Read "Bondy and Murty" about graph theory and planar graphs.
- Find 4 things I am interested in graph drawing, from "Di Battista et al."

## Week 1

`Meeting: 2018-09-18, Tue, 11:30 - 12:40, at Will's office`

### What we did

We proved that $K_{3,3}$ is non-planar using the Jordan curve theorem. I explained the idea of stereographic projection, and its interesting corollaries. I explained the idea of dual, and we also did a proof that the dual of a plane graph is also a planar graph by countering each other's arguments. In the end, as a practice, we found a $K_{3,3}$ subdivision from the Petersen graph.

### What I learned from discussion

- A nonseparable graph is a graph that has no cutvertex.
- When asked why in a proof, never say why not. Proof techniques of making the statement more and more precise by looking at more and more complicated examples.
- The circular ordering of edges at a vertex (clockwise and counter clockwise).

### To-do before next meeting

- Read "The Left-Right Planarity Test" by Ulrik Brandes.
- Download "P.I.G.A.L.E", play with it, and read the algorithm implementations.
- Find papers to read about how to test whether a graph is planar and how to draw a planar graph.

## Week 2

`Meeting: 2018-09-25, Tue, 11:30 - 13:00, at Will's office`

### What we did

I explained what a LR partition is and how the left-right planarity test works in general. Then Will asked me how to do the LR partition in linear time. It is equivalent to a 2SAT problem: for each pair of back edges, it is either the same-side or different-side. Then Will showed me the implication graph of 2SAT, and explained to me the concept of strongly connected components. I was unable to come up with a linear-time algorithm quickly. Then we discussed how to solve 2SAT in linear time using Kosaraju's algorithm. In the end, we had a brief discussion in zero-knowledge proofs.

### What I learned from discussion

- Never give an exponential algorithm to solve 2SAT outside of Will's office. In general, do not get satisfied with a non-polynomial algorithm unless it can be proven NP-hard.
- In order to solve a 2SAT, we can use implications from graph. To decide whether the Boolean formula can be satisfied, we just need to find whether a variable $a$ and its complement $\overline{a}$ are strongly connected in the equivalence graph. Kosaraju's algorithm performs 2 depth-first searches. The second search is on the graph $G^T$ with reversed edges, and we follow the reverse order of the time leaving the vertex in the first depth-first search.
- Zero-knowledge proofs: person A can make person B believe that A knows something in a way such that B cannot reproduce what person A did to convince others that B knows the same thing. For example, Will knows a Hamiltonian cycle in a complicated graph. He convince me by letting me have a choice of showing the Hamiltonian cycle with the whole graph blocked, or the whole graph (in order to verify that it is the original graph - Will isn't cheating). By doing this, it make me have a $1-2^k$ (where $k$ can be efficiently large) chance of believing that Will knows the Hamiltonian cycle, but I cannot reproduce it to show others. Zero-knowledge proofs were first conceived in 1985 by Shafi Goldwasser, Silvio Micali, and Charles Rackoff.

### To-do before next meeting

- Read *How to draw a planar graph on a grid* by H. de Fraysseix, J. Pach, R. Pollack, 1990
- Read *Embedding planar graphs on the grid* by Walter Schnyder, 1990
- Read *How to draw a graph* by W.T. Tutte, 1963

## Week 3

`Meeting: 2018-10-02, Tue, 11:30 - 12:20, at Will's office`

### What we did

We discussed the the paper by de Fraysseix, Pach, and Pollach, and drew how the algorithm works on board for cases with 3, 4, 5 vertices. Then we discussed the paper by Schnyder, and I explained how the normal labeling of a maximal planar graph works and how to derive a realizer, and how we can build the barycentric coordinates of the graph given its realizer.

### What I learned from discussion

- LEMMA 2.1. Let $v \in V(G) \to (v_1, v_2, v_3)$ be a barycentric representation of a graph G. Then given any three noncolinaer points $\alpha, \beta, \gamma$. The mapping $f : v \in G \to v_1\alpha + v_2\beta + v_3\gamma$ is a straight line embedding of G in the plane spanned by $\alpha, \beta, \gamma$. The proof is a slick.
- When reading an algorithm on graph drawing, it is better to simulate it by drawing it with a hand to see how it really works. Otherwise, I may miss some important details in the algorithm.
- What is interesting in the paper by de Fraysseix, Pach, Pollack is that how they establish the normal ordering of vertices, and use that to gradually build the graph step by step.
- What is interesting in the paper by Schnyder is that how he came up with the labeling of the graph, which is from an implication of something else Schnyder is researching. Will liked the paper so much when he was a graduate student that he put the paper in the bag until it was broken.

### To-do before next meeting

- Read *How to draw a graph* by W.T. Tutte, 1963
- Explore materials on MIT 6.889 course website:
  `http://courses.csail.mit.edu/6.889/fall11/`

## Week 4

`Meeting: 2018-10-09, Tue, 11:30 - 12:30, at Will's office`

### What we did

We discussed the the paper by W.T.Tutte. I drew an example graph and explained how to establish the linear system. The coordinates of an "inner" vertex is the average of the coordinates of all its neighbours. Then we discussed how can it make sure that by performing Tutte's algorithm there is no edge crossing. Tutte made a complicated argument on proving the correctness of his algorithm, i.e., ensure no edges crossing. In the end, we looked at the website of MIT 6.889 to figure out what to do next week.

### What I learned from discussion

- There are two major things Tutte is trying to prove. First, by averaging the coordinates of vertices, there cannot be a concave inner face; every face has to be convex. Second, if we draw a line between shared edges of two faces, it must be true that one of the face is on one side of the line and the other face has to be on the other side of the line.
- Why we cannot write the paper more intuitively? Because your intuition might be different from others. So other people may not understand your intuition. Therefore, you have to establish your theorem by formal proofs so that everybody can understand.
- A $K_{3,3}$ can be drawn in different ways. Some embedding is not easy to be recognized as $K_{3,3}$. If the polygon graph is even-cycle, we can recognize a bipartite graph from it.

### To-do before next meeting

- Read chapter for randomized algorithms in *Algorithm Design* by Jon Kleinberg and Éva Tardos.

## Week 5

`Meeting: 2018-10-16, Tue, 11:30 - 12:25, at Will's office`

### What we did

I explained what I studied for randomized algorithms, and showed some basic results on randomized algorithms. I described a simple version of Karger's Algorithm (Contraction Algorithm), which runs in $\mathcal{O}(n^4)$ to get a constant bound $(1/e)$ in failing to find the global minimum cut. Then Will searched for a more efficient version (the Karger-Stein algorithm), which runs in $\mathcal{O}(n^2 \log n)$. It uses recursion on smaller subsets. Then we showed another randomized algorithm for Max 3-SAT, which if we run $8k$ times, where $k$ is the number of clauses.

### What I learned from discussion

- Uniform circuit set is another computational model other than Turing machine. There is a Turing machine that given some problem it can output the uniform circuit set to solve the problem.
- Empirical algorithms are algorithms that are valued by their practice performance, like SAT-solvers. SAT is NP-Complete, but many SAT solvers have good performance on most cases.
- The intuition is that when the graph is huge, it is usually non-planar.
- It is not the case that most real world problems are planar problems. However, if it is known to be planar, take advantage of the planarity is good.
- No NP-hard problem has a probabilistic algorithm that has a success probability $\geq 1/2 + \epsilon$ for a positive constant $\epsilon$. Such algorithm would imply NP is contained in BPP, which is not believed to be true.

### To-do before next meeting

- Read chapter 19: Fibonacci Heaps in *Introduction to Algorithms* by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.
- Read *Maximum Flows and Parametric Shortest Paths in Planar Graphs* by Jeff Erickson.

## Week 6

`Meeting: 2018-10-17, Tue, 11:32 - 12:33, at Will's office`

### What we did

I explained what I studied in Fibonacci heaps, and we looked at other data structure that uses disjoint heaps. We looked at binary heaps, binomial heaps, and strict Fibonacci heaps, and we compared their performance on different operations. In particular, strict Fibonacci heaps get us interested because the performance of strict Fibonacci heaps is $\Theta(1)$ (not amortized!) except for the *delete-min* operation.[1] Then we discussed on Jeff Erickson's paper and his $\mathcal{O}(n \log n)$ algorithm on maximum flow in planar graphs. In the end, Will showed me his implementations of Tutte's algorithm and (incomplete) Forced-directed drawing algorithms in Icon.

### What I learned from discussion

- The intuition of amortized analysis is to balance the work of each operation and the change in data structure. The change in data structure can potentially "save" some work in other operations, which leads to the idea of potential function.
- It is incredibly hard to find the potential function to do amortized analysis. You have to understand the change in data structure very well in order to come up with the correct potential function.
- Icon is a high-level, general-purpose programming language with novel features including string scanning and goal-directed evaluation.[2]
- Other than graph theory and graph drawing, Will used to do researches in complexity theory, compression, and programming languages.

### To-do before next meeting

- Read chapter 10: Force-Directed Methods in *Graph Drawing: Algorithms for the visualization of graphs* by Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis.

[1] *Strict Fibonacci Heaps* by Gerth S. Brodal, George Lagogiannis, Robert E. Tarjan in STOC 2012 `http://tildeweb.au.dk/au121/papers/stoc12.pdf`

[2] The Icon Programming Language: `https://www2.cs.arizona.edu/icon/index.htm`