Backbone Determination in Wireless Sensor Network

CSE 7350/5350 Algorithm Engineering Class Term Project Documentation

(3/7/2018 Updated)

Zizhen Chen

Southern Methodist University

I. Due Dates, Extra / Penalty Credits and Submission

Total points:

400 = Part I (100 pts.) + Part II (100 pts.) + Part III (200 pts.)

Due dates:

Part I is March 7th, 2018, 5:00 p.m.

Part II is April 11st, 2018, 5:00 p.m.

Part III is May 2nd, 2018, 5:00 p.m.

Late submission penalties:

For Part I, -25 pts. penalty if turned in by due date of Part II.

For Part II, -25 pts. penalty if turned in by due date of Part III.

For Part III, -30 pts. penalty if turned in by May 11th, 5:00 p.m. If you couldn't finish it by May 11th, 5:00 p.m., you still have chance to continue working on the project, but you would receive -40 pts. penalty and perhaps receive 'I' as a final grade for this course in current semester.

Extra credits:

The extra credits listed here are the total credits of all three parts.

- +24 pts. bonus for Disk topology implementation (for CSE 5350 enrolled students only)
- +30 pts. bonus for Part III final report turned in by April 25th, 5:00 p.m.
- +18 pts. bonus for Sphere topology implementation
- +18 pts. bonus for linear time algorithms implementation
- +24 pts. bonus for Algorithm Visualization implementation

Submission:

For each part, submit your report plus any appendices containing source code as a single file to Canvas. For on campus students, you are required to print your report and submit your hard copy to the mailbox of Zizhen Chen's (in Suit 445 of Caruth Hall). Each report can only have a maximum length of 30 pages.

Contact information:

Office hour: Every Wednesday afternoon from 2:00 p.m. to 5:00 p.m. in office located at Computer Teaching Lab 308 in Caruth Hall by appointment.

Email: zizhenc@smu.edu.

II. Introduction and Summary

In this project you are asked to implement an algorithm for determining a coloring, terminal clique, and a selection of bipartite subgraphs that are produced by an algorithm for graph coloring in a random

geometric Graph (RGG). These results model a wireless sensor network (WSN) with each bipartite subgraph providing a communication backbone. We split the project into 3 parts on purpose in order to help students arranging time properly to complete the term project within the semester, so part II and part III are expansions from part I. Part III would be the final complete report of the whole project. You are first asked in Part I to prepare several RGG's as a randomized benchmark set. RGG's on the plane (unit square), RGG's on a disk (unit circle) and RGG's on the globe (unit sphere) model WSN's on geographic regions. In Part I you are further asked to output your graph as an adjacency list. This becomes the input for Part II.

In Part II, you are asked to accept an adjacency list as input and implement the smallest-last coloring algorithm for vertex coloring and terminal clique identification in the RGG's specified by the input. The coloring algorithm is to be applied to the randomized benchmark sets of Part I and is also to be applied to several original graphs you specifically create to exhibit the strengths and weaknesses of the coloring and clique determination algorithms including a few graphs of considerably larger RGG's.

Regarding algorithm efficiency you are asked to verify the running time bound for each algorithm you implement in Part I and Part II with extra credit if you are able to achieve linear time ($\Theta(|V|+|E|)$) implementation. You are asked to document your implementations in the form of a report to a technical manager with substantial use of graphic display of results again with extra credit for interfacing it to a graphics package to show real-time behavior of your solution. Section III provides benchmarks for you to run implementation and collect data. Section IV provides general format requirements and grading details for the project report. Section V provides more details of different requirements for each part.

III. Benchmarks

A randomized case benchmark of 10 particular RGG's output from Part I should be part of your graph set input to your vertex coloring in Part II and your backbone selection program in Part III. Your program must find a smallest-last vertex ordering, terminal clique, vertex coloring and several bipartite backbones for each of these ten benchmark graphs as well as the graphs you create as requested.

Benchmark Data Sets

Benchmark #	N	Avg. Degree	Distribution
1	1000	32	Square
2	8000	64	Square
3	16000	32	Square
4	64000	64	Square
5	64000	128	Square
6	128000	64	Square

7	128000	128	Square
8	8000	64	Disk
9	64000	64	Disk
10	64000	128	Disk

Extra Credit Benchmark Data Sets

Benchmark #	N	Avg. Degree	Distribution
11	16000	64	Sphere
12	32000	128	Sphere
13	64000	128	Sphere

IV. Project Format and Grading

The report should be initiated by a **Executive Summary** containing an Introduction and Summary, and a Programming Environment Description as described in the following. The core of your report should contain the **Reduction to Practice** details and **Result Summary** as outlined in the following. Give clear citations (e.g. [1]) to your sources listed in the final **References** section including downloaded programs and previous student's projects that most influenced your work. In style, the report should be oriented towards a technical manager with a wide perspective, say in Scientific American magazine style.

1. Introduction and Summary (20%)

   Give a brief description, in layman's terms, of the project, including your major result. Consider this section to be a professionally worded marketing effort. That is, would the competent technical manager (your audience) be likely to want to read on and try to understand your claimed good results. Describe the strongest features of your implementation. Summarize your use of illustrations, figures, tables, and graphics employed that will reveal your results effortlessly to the reader. The summary should include a table of results for the benchmarks described in section III. Give your conclusion on the strengths and weaknesses of the algorithms on one hand and your implementation on the other. Indicate applications where your implementation would be strongest and what kind of input would be most difficult to handle. Your analysis and conclusions relevant to your separately created test graphs can be a focal point of this topic.

2. Programming Environment Description (20%)

   Give a description of the environment you used to develop the program including both hardware and software components, i.e., the specific type and brand of computer, the computer's processing speed, the amount of memory, the operating system, the language, any graphics tools, and special libraries, etc. The description should include metrics on the program resource utilization allowing

judgment of "was the total used respectable or an overkill?" The purpose of this description is to allow comparisons and facilitate reproduction. Consider: would a knowledgeable colleague be able to judge the scope of your work from this description of the environment and resource utilization? The graphics package and languages used to present the report and used to create the graph drawings and displays should also both be described. Discuss the interactions between system components you needed to establish to effect both computational efficiency and effective result display.

3. Reduction to Practice (50%)

   This is the core of your report and should cover the following topics appropriately integrated for each part:

   (1) Data structure Design: Describe and illustrate the representation and organization of data employed.

   (2) Algorithm Descriptions: Provide descriptions of each algorithm you implemented. Estimate the time and space complexity of each algorithm. Discuss the input and output of each algorithm including what calculation you made to fulfill the input data requirement of the algorithm.

   (3) Algorithm Engineering: Describe how the algorithm implementations are engineered to achieve efficiencies in time and space and provide arguments for the efficiencies. In particular, discuss your expected running times and carefully substantiate the linear time and appropriate optimal space bounds and include your plots of running time.

   (4) Verification: You are required to attach images and related running data. Using plots and walkthrough if necessary to draw conclusion of your discovery.

   You should provide a standard output for each randomized benchmark graph. These points are awarded on the basis of your overall use of illustrations, figures, tables, bar charts, plots and diagrams in your report. More details of Reduction to Practice for each Part will be described in section V.

4. References (10%)

   Provide an enumerated list of publications, texts, websites, programs and resource packages in typical reference style as a concluding part of the executive summary. You should include references to several papers related to WSN's, backbones in WSN's, RGG's, graph coloring algorithms, the Smallest Last algorithm and to the major tools used in developing the report. For example:

   [1] D.W. Matula, Wireless Sensor Network Project, www.lyle.smu.edu/cse/7350/, 2014

IMPORTANT: If you read previous student projects from the archives in the department office or online regarding the smallest last coloring implementation, backbone determination, and displays, you should include a reference to each of them in your reference list.

V. *Detail Requirements for each Part*

1. Part I: RGG Generation

According to the benchmark table in section III, you are provided the following input data:

- N = number of sensors (vertices)
- A = expected density (average degree) of the sensor network
- T = network topology (unit square, unit disk, or unit sphere).

In this part, you are required to provide the following output through your algorithms:

- r = estimated radius r value
- E = number of distinct pairwise sensor adjacencies (edges)
- Avg. Deg. = real density (average degree) of the generated sensor network
- Max Deg. = maximum degree of generated networks (RGG's)
- Min Deg. = minimum degree of the generated networks (RGG's)

For planar input data generation, we shall pick N points in two dimensions according to some specified distribution using pseudo random number and determine the pairs of points within distance r of each other. Develop the RGG's adjacency list data structure (think about why we use adjacency list instead of other structure?) for the benchmark data sets in the table created as specified.

We have three graph topologies in the benchmark table:

- Unit Square has the vertices uniformly distributed over the square of length 1.
- Unit Disk has the vertices uniformly distributed over the disk (circle) of radius unity.
- Uniform Sphere (Extra Credit) has the vertices uniformly distributed over the surface of the unit radius sphere.

In the Verification part, besides the screenshots of your generated RGG's, you are also required to provide the following two charts:

- Degree distribution: Plot the number of vertices having degree i as a function $f(i)$ for $i = 0,1, \dots, max\ degree$.
- Running Time Graph: Provide two plots of the running time vs the input size.

- In the first plot, use the number of vertices on the x-axis and the running time on the y-axis. You should hold the average degree constant and increase the number of vertices from 1000 to 32000 and plot the running time of your solution. You should include the actual values for 1000, 4000, 8000, 16000 and 32000. Use enough other points as needed to get a good graph.
- In the second plot, you should hold the number of vertices constant at 32000 and increase the average degree. The x-axis will represent the average degree and the y-axis will represent your running time of your algorithm.

For the RGG display: You are required to provide a screenshot image for each RGG benchmark case. For the benchmark cases having graph orders (number of vertices) less or equal to 8000, you have to provide an image depicting the vertices and edges of the RGG's you generated. For the benchmark cases having graph orders larger than 8000 nodes, you can show images of graph vertices only, but highlight the vertices having the minimum degree and the maximum degree.

2. Part II: Graph Coloring

Vertex coloring is the most common graph coloring problem. The problem is, given m colors, find a way of coloring the vertices of a graph such that no two adjacent vertices are colored using same color. The set of vertices of same color is termed as "independent set". Pair up any two independent sets could establish a bipartite subgraph of the original RGG which would be an input for Part III to select final bipartite backbones.

The vertex coloring algorithm we provided for this project is the sequential coloring algorithm based on smallest-last ordering [1], which is termed as "smallest-last coloring" algorithm.

Therefore, to proceed the smallest-last coloring algorithm, we have two steps to do:

Input: A graph

(1) Use smallest-last ordering algorithm to order the vertices of a RGG.

(2) Sequentially color the vertices of the RGG in smallest-last order.

Output: Independent color sets

In the verification part, your report should include the following two charts plotted according to the smallest-last coloring algorithm for each graph, along with any additional information you feel would "sell" your implementation.

(i) Sequential Coloring Plot: For all graphs, provide a plot of the degree-when-deleted function in the smallest-last order. You should also show the plot of corresponding original degree (upper bound) function in the same chart.

(ii) Color Set Size Distribution: For all graphs, provide a plot of the size distribution for the independent sets of vertices colored with color j for $j = 1, 2, \ldots, maxcolor$.

In the Reduction to Practice part, you are required to provide a Walkthrough of the smallest-last coloring algorithm: Present a walkthrough for an RGG of the unit square topology with N=20 vertices and r=0.40. Draw a graph for each step of the walkthrough of the Smallest-last vertex ordering and the graph coloring. Draw the corresponding sequential coloring plot and color set size distribution.

In the Introduction and Summary part, you are also required to expand the Summary Table from your Part I report to include the data generated from the smallest-last coloring algorithm. You need to add maximum degree-when-deleted, number of colors, max color set size, terminal clique order for each benchmark case.

3. Part III: Backbone Selection

Bipartite backbone selection by independent set pairings: For the first four largest independent color sets of each of the benchmark cases, determine which two of the six possible bipartite subgraphs have the largest size (edges) in their maximum connected subgraph (giant component, termed the "backbone").

In the Verification part, you are required to provide screenshots of the two backbones selected from each benchmark case along with some metadata of the backbone graph like the order (number of vertices), the size (number of edges), the colors of the two independent sets, the number of faces (for Sphere topology only) and percentage of sensors covered by the backbone (domination).

In the Reduction to Practice part, you are required to expand the Walkthrough from Part II to include the determination of the backbone for the vertices colored with the first and second colors. Plot the backbone bipartite subgraph illustrating the domination provided by the first color set and the planarity of the bipartite subgraph.

In the Introduction and Summary part, you are also required to expand the Summary Table from your Part II report to include the data generated from the bipartite backbone selection. You need to add the orders, the sizes, the number of faces (for Sphere topology only) and the dominations of the two selected backbones for each benchmark case.