CSC 226 Lab 3: Graph Summary using algs4.jar

Joe Howie

September 22, 2023

Contents

List of Figures

---

1 Objective: make a summary of a graph

Disclaimer: Last week's lab led us through the installation process of the algs4.jar package. This lab assumes you have working access to these classes.

Motivation: Sometimes in data science, it is useful to reduce the number of nodes in a digraph G by creating super nodes consisting of the strongly connected components (SCC) from G. If there was an edge between two nodes u, v where u, v are not apart of the same SCC, then that edge exists between the super nodes in the summarized graph (what we called the reduced graph in lecture).

Your job: The file SummarizedGraph.java has two methods that you need to fill in, BuildSG() then printTopOrder(). Your TA can assist you during the lab.

1.1 Download

Save all files to the same directory as algs4.jar

- tinyDG.txt
- mediumDG.txt
- largeDG.txt

2 Algs4 Classes

To efficiently implement the two methods in SummarizedGraph.java, BuildSG() and printTopOrder(), you will need methods from the following classes in algs4.jar.

Understanding the basics To use these classes we do not need to know everything about how the code works (that would defeat the point entirely). We do need to know what inputs are expected and how to extract the output correctly. All these java file are located at edu/princeton/cs/algs4/.

2.1 KosarajuSharirSCC.java

Useful methods For your implementation, you will find the count() and id() methods helpful.

2.2 Digraph.java

Useful methods For your implementation, you will find the V() and adj() methods helpful.

2.3 Topological.java

Useful method For your implementation, you will find the order() method useful.

3 Coding Exercise

3.1 Starting material

Given: You are provided with the following scripts:

- SummarizedGraph.java
- Pair.java

The class Pair.java is a helper object of comparing edges (surprisingly there was not one in algs4.jar).

3.2 Finish the java program SummarizedGraph.java

Most of the script has been written for you there are two spots labelled "ADD YOUR code here". The first portion to implement finds the edges that are present in the summary graph from the original edge list. The second portion to implement prints out the topological sorted order. Note that the summary will always be a DAG.

3.3 Run time

Worst case The SummarizedGraph algorithm can be implemented in $O(m + n)$, this comes from the KosarajuSharirSCC and Topological algorithms. The code you write for BuildSG() should take $\Theta(m)$ to run.

3.4 Psuedocode

Algorithm 1 Summarize Graph Algorithm

```
1: procedure SummarizedGraph(G)
2:     scc ← KosarajuSharirSCC(G)
3:     edge_list ← ∅
4:     for (u, v) ∈ E do
5:         if u and v belong to different clusters in scc then
6:             Add edge (cluster(u), cluster(v)) to edge_list
7:
8:     Print edge_list to file
9:     DAG ←load edge_list from file
10:    top ← Topological(DAG)
11:    Print topological order of DAG
```