CSE 5311 Lab Assignment 2

Due April 5, 2018

Goals:

1. Understanding of network flow techniques as applied to undirected graphs.

2. Understanding of minimum cuts, which are particularly easy to obtain from most Ford-Fulkerson code. For push/relabel code you will need to traverse unsaturated edges from the source in the final residual network to determine S.

3. Understanding of cut-trees, a compact representation for cut information regardless of the chosen source and sink. For small instances, this will be easy to check by inspection.

Requirements:

1. Write a C/C++ program to determine a cut tree for an undirected graph:

    a. The input graph may be treated simply as a symmetric directed graph, i.e. each input edge must be represented in both directions. The provided input graphs are guaranteed to be connected. (If not, then some calls to the network flow routine would have a maxflow of zero.)

    b. In the output cut tree, the smallest capacity on the unique path between vertices *s* and *t* gives the capacity of the minimum cut when *s* and *t* are used as the source and sink for a network flow algorithm.

    c. The cut tree is to be found using the algorithm MGH on p. 152 of `http://epubs.siam.org.ezproxy.uta.edu/doi/abs/10.1137/0219009` . This technique uses a max-flow, min-cut implementation as an "oracle". You may use publicly-available code (or library routines) to get started, but give appropriate credit. Note that Gusfield's pseudocode numbers vertices starting from 1, but that much available graph code numbers vertices starting from 0.

    You will need maxflow code that allows arbitrary vertices as the source and sink.

2. The input will be formatted as:

    a. The first line will have two values, `n` and `m`, giving the numbers of vertices and edges. `n` will not exceed `3000`. `m` will not exceed `10000`.
    b. The next `m` lines will each contain three integers. The first two, the vertices incident to an edge, will be in the range `0 ... n - 1`. They must be different and the file will not contain parallel edges. The third value will be a positive capacity no larger than `10000`. (As an undirected graph, each line corresponds to a pair of symmetric edges.)
    c. The input will be a connected graph.

3. The output from your program should go to standard output, not a file:

    a. The number of output lines will be `n - 1`.
    b. Each line will have the two vertices incident to an an edge in the computed cut tree, followed by its capacity.

4. Your program must compile with GCC and execute on OMEGA. Be sure to provide compilation directions and file interfacing details as code comments.

5. Submit your C/C++ code on Blackboard before 6:45 p.m. on Thursday, April 5.

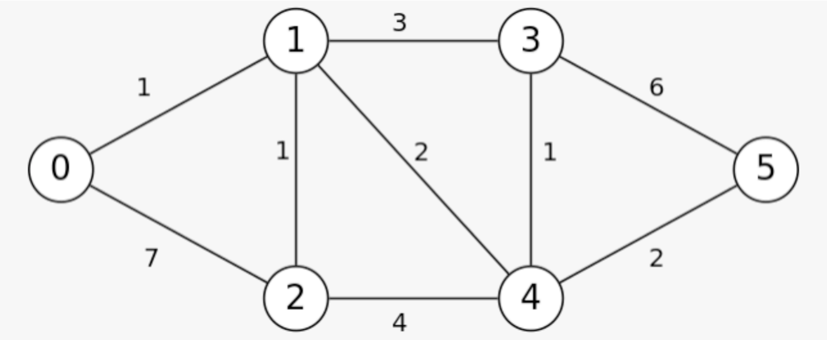
Getting Started:

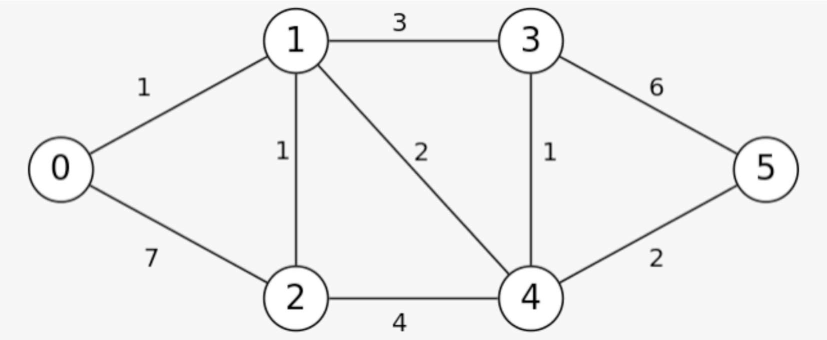
1. Cut trees are also known as Gomory-Hu trees (`https://en.wikipedia.org/wiki/Gomory–Hu_tree`)

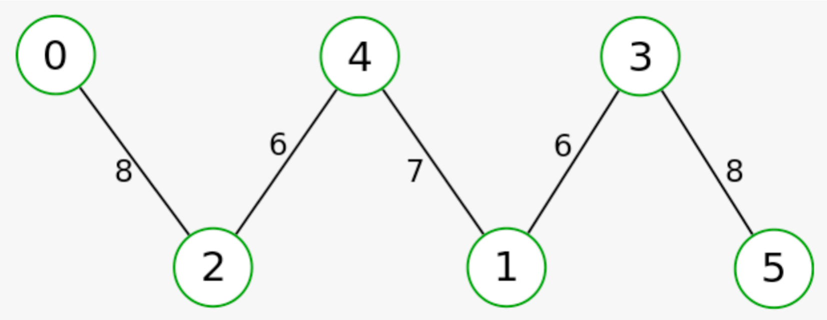
2. Be sure to understand the difference between an *(equivalent) flow tree* and a *cut tree*. Every cut tree is a flow tree, but not vice versa. Algorithm EQ produces a flow tree and is a useful stepping stone for understanding.

3. `http://www.graphviz.org` is an excellent package for graph layouts.

4. Big aside: `http://dl.acm.org.ezproxy.uta.edu/citation.cfm?id=320228`


0
1
2
3
4
5
1
7
3
1
2
4
1
6
2


0
1
2
3
4
5
1
7
3
1
2
4
1
6
2


0
4
3
8
6
7
6
8
2
1
5