COL 719 – Synthesis of Digital Systems
I Semester 2016-17
Assignment 1: Netlist and Graphs
Submission Deadline: 11 August 2016, 23:55 PM

The objective of this assignment is to learn about the Graph data structure, and use it to model a gate-level netlist.

1. **Read a netlist into a graph**. The first task is to write a C++ program for reading a netlist into a graph data structure. Assume that the netlist is given by the user in a file that you will read into your program. The format of the netlist is as follows:

   ```
   A X Y Z
   X P Q
   B Y P
   ```

   The file consists of a set of lines, each line specifying a net/wire connecting one *source* and an set of *destinations* as follows.
   <source> <dest1> <dest2> <dest3>...
   In the netlist above, "A X Y Z" means that the output of gate A is connected to the inputs of gates X, Y, and Z. The presence of Y in two different destination lists (lines 1 and 3) means that Y has two inputs – one connected to A, and the other to B. Primary inputs are modelled as just a dummy gate with no input. Primary outputs are represented by gates that are not connected to any other input. Note that we are assuming that each gate has only one output.
2. **Print netlist properties**. For the netlist that is now stored in your graph, print the following:
   a. the set of primary inputs
   b. the set of primary outputs
   c. the set of internal gates (which is just the set of gates that are neither primary inputs nor primary outputs).
   d. the gate count (exclude primary inputs).
3. **Perform Timing Analysis**. A timing analysis of the netlist should report the critical path, consisting of the maximum number of gates on any path between a primary input and primary output (including both). Traverse your graph and report the critical path.

NOTES:

1. Make your own assumptions if you see ambiguities in the specification anywhere.
2. Remember that the objective here is to familiarise yourself with the software and library packages. Efficiency of the timing analysis is of lower priority. However, the algorithm needs to be correct.
3. Use the Moodle system to post queries so that we answer the same question only once. This mode of interaction is preferred to email. We will answer general installation related queries only within 5 days of the assignment being posted. We won't answer any queries in the last 3 days before the assignment deadline (remember: make your own reasonable assumptions).

Instructions for using the OGDF Graph Package

Lokesh Siddhu, Teaching Assistant

1. Installation Instructions for OGDF:
   Link - http://www.ogdf.net/doku.php/tech:installgcc
2. Startup Examples:
   Link - http://www.ogdf.net/doku.php/tech:howto:manual
   In this example instead of using GML format output, use SVG format. Use GraphIO::drawSVG(GA, “manual_graph.svg”).

3. How to run a program with OGDF libraries included:
   (Assuming OGDF installation location to be */home/siddhulokesh/sem4/TA/*OGDF)
   g++ -I*/home/siddhulokesh/sem4/TA/OGDF*/include -O2 test.cpp -o test -L*/home/siddhulokesh/sem4/TA/OGDF*/_release -lOGDF -lCOIN -pthread