# CPSC 335 Project 2: exhaustive search

Prof. Doina Bein, CSU Fullerton

dbein@fullerton.edu

## Introduction

In this project you will design, implement, and analyze one algorithm for a graph problem. Describe the algorithm using clear pseudocode, analyze it mathematically, implement it in C/C++/Java, and measure its performance in running time.

At the end of this document I have provided you with sample high resolution timing code in C++11, called "Template for a C/C++ program with high resolution timing". If you choose to use Java, you are responsible for figuring out how to do high resolution timing in that language.

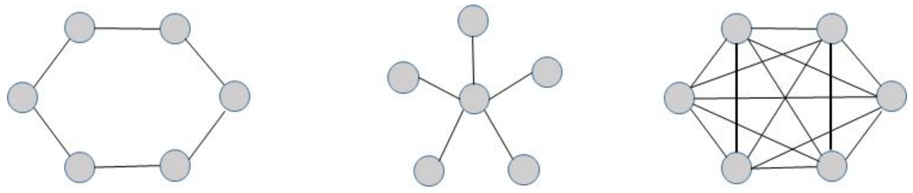
## The hypotheses

This experiment will test the following hypothesis:

*Exhaustive search algorithms are feasible to implement, and produce correct outputs.*

## The problem

A network topology specifies how computers, printers, and other devices are connected over a network. Figure 1 below illustrates three common topologies of networks: the ring, the star, and the fully connected mesh.



You are given a weight matrix W[0..n −1, 0..n −1], where n > 3, which is the weight matrix of a weighted, connected, undirected graph modeling a network with one of these topologies, if any, the matrix represents. Assume the weights to be positive integer values. Instead of infinity, the elements if the matrix will have the value 100. Design a brute-force algorithm (express it in pseudocode) for this problem and indicate its time efficiency class.

The *topology recognition problem* can be formulated as follows:
**input**: n, a positive integer, and a two-dimensional matrix W of integer values
**output**: a message stating either "ring" for ring topology, "star" for star topology, "complete" for fully connected mesh, or "neither" if the weight matrix does not represent either of these three topologies
The topology recognition problem is solvable in polynomial time. You need to find out the exact time complexity and prove it.

## What you need to do

1. Write clear pseudocode for an algorithm solving the problem.

2. Analyze your pseudocode mathematically and write its efficiency class using Big-Oh notation. (You need to compute the total number of steps of the algorithm.)

3. Prove the efficiency using the definition or using limits.

4. Type these notes (electronically or on paper) and submit it as a PDF report.

5. Implement your algorithm in C/C++/Java/Python. You may use the templates provided at the end of this file.

6. Compile and execute the program.

7. Create a file with the output of the program for an input value and submit it together with the program. Note, the output can be redirected to a file (for easy printing). For example, the following command line will create an output file in Linux-based operating system called a1out.txt by re-directing the output from the screen (display) to the file a1out.txt:

```
K:\cpsc335> a.out > a2out.txt
```

## Sample outputs:

<u>Example #1:</u>

```
K:\cpsc335> ast2a
CPSC 335-x - Programming Assignment #2
Topology recognition algorithm
Enter the number of nodes in the topology
5
Enter the weight matrix
0    2    100  100  5
2    0    3    100  100
100  3    0    1    100
100  100  1    0    4
5    100  100  4    0
The topology is
ring
elapsed time: 0.000106 seconds
```

Example #2:

```
K:\cpsc335> ast2a
CPSC 335-x - Programming Assignment #2
Topology recognition algorithm
Enter the number of nodes in the topology
5
Enter the weight matrix
0  2     3     4     5
2  0     100  100  100
3  100  0     100  100
4  100  100  0     100
5  100  100  100  0
The topology is
star
elapsed time: 0.000106 seconds
```

Example #3:

```
CPSC 335-x - Programming Assignment #2
Topology recognition algorithm
Enter the number of nodes in the topology
5
Enter the weight matrix
0  1  2  3  4
1  0  5  2  3
2  5  0  4  1
3  2  4  0  2
4  3  1  2  0
The topology is
```

**Fully connected mesh**

```
elapsed time: 0.000106 seconds
```

Example #4:

```
K:\cpsc335> ast2a
CPSC 335-x - Programming Assignment #2
Topology recognition algorithm
Enter the number of nodes in the topology
5
Enter the weight matrix
0     2     100  4  5
2     0     3     100  100
100  3     0     1     100
4     100  1     0     4
5     100   100  4     0
The topology is
neither
elapsed time: 0.000106 seconds
```

## Template for a C/C++ program with high resolution timing:

```
#include <iostream>
#include <iomanip>
#include <cstdlib>
#include <chrono>

using namespace std;

int main( ) {
  // DECLARE VARIABLES
  // PRINT THE HEADER OF THE PROGRAM
  // READ THE INPUT

  // Start the chronograph to time the execution of the algorithm
  // Reading the input is NOT part of the algorithm so the timer starts after reading the input
  auto start = chrono::high_resolution_clock::now();

  // YOU NEED TO COMPLETE THIS PART OF CODE

  // End the chronograph to time the execution of the algorithm
  auto end = chrono::high_resolution_clock::now();

  // DISPLAY THE OUTPUT

  // Print the elapsed time in seconds and fractions of seconds
  // Displaying the output is NOT part of the algorithm so the timer ends before displaying the output
  int microseconds = chrono::duration_cast<chrono::microseconds>(end - start).count();
  double seconds = microseconds / 1E6;
  cout << "elapsed time: " << seconds << " seconds" << endl;

  return EXIT_SUCCESS;
}
```

## Template for a C/C++ program solving the topology recognition problem

```
// Assignment 2: Topology recognition problem
// Doina Bein
// Given a network with n>3 nodes and a weight matrix W[0..n−1, 0..n−1] of
// positive integers, of a weighted, connected undirected graph modeling
// a network, decide whether the network is one of the topologies, if any:
// ring, star, fully connected mesh. Note: represent infinity by the value 100.
// INPUT: a positive integer n and a list of n^2 positive values
// OUTPUT: message "ring" or "star" or "complete" or "neither"

#include <iostream>
#include <iomanip>
#include <cstdlib>
#include <chrono>

using namespace std;

int main() {
  int n, i, j, count, center;
  int W[100][100];
  bool cond;

  // display the header
  cout << endl << "CPSC 335-x - Programming Assignment #2" << endl;
  cout << "Topology recognition algorithm" << endl;
  cout << "Enter the number of nodes in the topology" << endl;
  // read the number of nodes
  cin >> n;
  // read the weight matrix
  cout << "Enter the positive weights, 100 for infinity" << endl;
  for(i=0; i < n; i++)
   for(j=0; j < n; j++)
     cin >> W[i][j];

  // Start the chronograph to time the execution of the algorithm
  auto start = chrono::high_resolution_clock::now();

  cout <<"The topology is"<< endl;

  cond = true;
  // loop to check whether the topology is a ring
  for (i=0;  i < n ; i++) {
    // YOU NEED TO IMPLEMENT THIS LOOP
  }
```

```
 if (cond) {
   cout <<"ring" << endl;
   return EXIT_SUCCESS;
  }

  cond = true;
  center = 0;
  // loop to check whether the topology is a star
  for (i=0;  i < n ; i++) {
    // YOU NEED TO IMPLEMENT THIS LOOP
  }

  if (cond && (center == 1)) {
    cout <<"star" << endl;
    return EXIT_SUCCESS;
  }

  cond = true;
  // loop to check whether the topology is a fully connected mesh
  for (i=0;  i < n ; i++) {
    // YOU NEED TO IMPLEMENT THIS LOOP
  }

  if (cond)
    cout << "fully connected mesh" << endl;
  else
   cout << "neither" << endl;

  // End the chronograph to time the loop
  auto end = chrono::high_resolution_clock::now();

  return EXIT_SUCCESS;
}
```