Scaling Social Networks and Finding Correlation between Influenza & Weather Variables, Alongside Data Aggregation

Zachery Morris
ztm4qv@virginia.edu

Jiangzhuo Chen
chenj@virginia.edu

Biocomplexity Institute and Initiative, Network Systems Science and Advanced Computing – Charlottesville, Virginia

INTRODUCTION

The first two weeks of the C4GC program, I read research papers on infectious disease models. These models implemented parallel computing, where a main task is split into smaller tasks and sent to be executed on different computing nodes. These models are directly affected by the size of the social contact network. A social contact network, is represented as a graph, where each node is a person. The edges between these nodes hold the contact time between people, the location of contact, and other factors.

TASK ONE

In order to test the efficiency of these infectious disease models and optimize them, I was tasked with creating a program in C++ that scales up a given social network. The social network was given in an extended CSV format file. From left to right, the columns have source ID numbers, source activities, target ID numbers, target activities, and duration of contact. Each ID number represents a person, and thus, a node in the social network graph. Each row in the file represents an edge in the graph. At a high level, the program makes a network that is K times larger than the original. A new network is made consisting of K copies of the given network, and rewiring f fraction of randomly chosen edges, so the copy networks can become connected to the existing network. The program uses command line parameters, which in order are: the CSV file that holds the original network, number of network copies, K, and percentage of rewiring, f.

Step by step, the program first utilizes command line parameters to read in the network edges from a CSV file and stores this undirected edge representation in an "edge" object. These edge objects are then stored in a vector. Each edge object contains both IDs (nodes) and their attributes. As the file is read in, each edge in the network goes from being bidirectional to non-bidirectional. Next, the program makes K copies of the network and adds the copies and original edges to a vector, forming the large network. A helper method is implemented which generates new IDs based on the beginning and ending interval of the original IDs. Both original and copy

IDs are added to a vector. Next, the program iteratively chooses two edges, randomly, and swaps IDs between them, effectively rewiring the edges between nodes in the graph and connecting the copied edges. A helper method generates random integers with an interval equal to the amount of edges—these are used as indices in the large vector. The first two random numbers are used to get the first two IDs in the large vector and swap them. Lastly, the program writes the two header lines and both directions of each edge to an output file.

TASK TWO

Moreover, I was tasked with creating a Python program that extracts and aggregates climate data using Pandas. As background information, a FIPS code is a code that represents locations in the US. A state has a two-digit code. A county has a five-digit code where the first two digits represent the FIPS code of the state where the county belongs. Data in the yyyy.csv file is daily and by location. This file also has eight columns: location_id, date, variable, value, mflag, qflag,sflag, and observation_time. The program ignores the last four columns.

The program extracts data of a specified climatic variable for US counties and states while aggregating the data to spatial or temporal resolution. Going through the steps, first, the program implements command line parameters: *raw_data* which is one year daily observation data (yyyy.csv), *variable* which is a climatic variable (PRCP, TMAX, TMIN, TOBS), *locations* which is a file that maps location ID to FIPS code (county_fips.csv, state_fips.csv), *spatial* which is the spatial resolution of output file (state, county), *temporal* which is the temporal resolution of output file (daily, weekly), and *output* which is the output file (output_data.csv). Next, the program aggregates the weather data to state or county resolution by taking the average of the climatic values over all the locations in the state or county. Next, the program aggregates the weather data to weekly resolution by taking the average of the climatic values over all days from Sunday to Saturday, labeling a week by its Wednesday date. Lastly, the output file is produced based on the inputted command line parameters in CSV format.

TASK THREE

Visualize how influenza indicator variable varies with climatic variable. Using Pandas and Matlibplot, graph activity level of influenza, precipitation, maximum temperature, and minimum temperature of each state in the US against a season (e.g. 2018-2019). It is seen that at peaks of influenza activity level, precipitation and temperature minimum also peak while temperature maximum is lower.