TESC: a Two-event Structural Correlation estimator on graphs

Oct. 27, 2013

1 Overview

This package implements the graph correlation estimation framework of two events occurring on the same graph. The basic idea is to sample reference nodes in the graph which serve as "observers" as to how the densities of the two events are in their neighborhoods. Then based on these densities, we can compute the degree of consistency of the two events' density changes among different reference nodes. The measure used is the Kendall's τ rank correlation. Three different reference sampling algorithms are provided in the package, namely, Batch_BFS, Importance Sampling and Whole Graph Sampling. Different sampling algorithms have their respective advantages (please refer to our paper [1] for a detailed analysis). The package also contains a correlated event pair simulator which can simulate positively or negatively correlated event pairs on a graph.

2 Data Requirements

The major input data consist of two files: a graph stored in a .txt file, and events stored in another .txt file.

The format of the file storing the graph is as follows: the first line is the number of nodes in the graph; starting for the second line, each line # i represents the adjacency list of node $i-1$ (node index starts from 1), e.g. line # 2 contains the adjacency list of node 1. The delimiter in the adjacency list is a white space. The adjacency list is sorted so that nodes with smaller indices come first.

The format of the event file is similar: the first line contains the number of events; then starting from the second line, each line gives an event which consists of nodes on which the

event occurred. The delimiter is also a white space.

An example DBLP graph is provided in the data folder of the package.

3 Structure of the Package

The package contains three Java packages: "tesc.eventCorrelation", "tesc.eventSimulation" and "tesc.utils". "tesc.utils" contains utilities used by the main programs. "tesc.eventSimulation" contains the event simulation program. The three correlation estimators (corresponding to the three sampling algorithms) are in "tesc.eventCorrelation". The runnable programs are: BatchBFS.java, ImportanceSamp.java, WholeGraphSamp.java, VicinitySizeCollector.java and SimulateEvents.java.

4 Usage

VicinitySizeCollector

Usage: java -jar VicinitySizeCollector.jar graph_file_path maximum_hop

One should run this program first to pre-compute the neighborhood sizes of each node in the graph stored in graph_file_path up to maximum_hop number of hops. For example, if maximum_hop=3, the program will output three files, each of which contains the vicinity sizes of all nodes for their k-hop neighborhoods, $k = 1, 2, 3$. This sort of information is required by the Importance Sampling method. The generated files will be in the same folder of the graph file.

SimulateEvents

Usage: java -jar SimulateEvents.jar graph_file_path hop_num pORn

This program simulate correlated event pairs on the graph stored in graph_file_path. hop_num indicates the scale of the correlation (i.e. the lower the value, the stronger the correlation). It takes integer values greater than 0 and smaller than the diameter of the graph. pORn is an indicator variable controlling which type of correlations is going to be generated: 1–positive, 2–negative, 0–both.

This program also requires the pre-computed vicinity size files to be in the same folder as the graph file. Please do not change the files names of the vicinity size files and the graph file after the vicinity size files are generated. Otherwise, the program cannot find the vicinity

size files and will report an error. The generated event files will be in the same folder as the graph file.

This program will also add noises in the generated event pairs (please refer to our paper [1] for details) and output those blurred event pairs. The noise levels are hard coded.

BatchBFS

ImportanceSamp

WholeGraphSamp

Usage: java -jar program_name.jar graph_file_path event_file_path hop_num mode event1_index(opt) event2_index(opt)

The three programs evaluate the structural correlations of events and output the results. The command line parameters are the same: the path of the graph file, the path of the event file, the scale of the correlation the user concerns. The fourth parameter mode can take one of the following three string values: (1) "batch": means every 2 lines of the event file are treated as a test pair (event file must contain even number of lines); (2) "all": means computing correlations for all pairs of events in the event file, and the output order is $(0,1),(0,2),(0,3),\ldots,(1,2),(1,3),\ldots(n-2,n-1)$ where n is the number of events in the event file; (3) "single": means just a single pair of events is going to be tested. This is the only case where the last two parameters (event1_index and event2_index) are taken in to consideration, i.e. the 0-based indices of the two target events in the file.

ImportanceSamp also requires the pre-computed vicinity size files to be in the same folder as the graph file. Please do not change the files names of the vicinity size files and the graph file after the vicinity size files are generated. Otherwise, the program cannot find the vicinity size files and will report an error.

Finally, for "batch" and "all" modes, the output will be saved in a file which is in the same folder as the graph file. For "single" mode, the result will be reported in the console directly. All three programs report z-scores of the hypothesis testing where the null hypothesis is that the two events are independent on the graph.

References

[1] Z. Guan, X. Yan, and L. M. Kaplan. Measuring two-event structural correlations on graphs. *Proceedings of the VLDB Endowment*, 5(11):1400–1411, 2012.