Link Prediction via Entropy Reduction

Applications of MIDER (Villaverde et al., 2014)

Walker Harrison, Duke University

Introduction

For most graph implementations, the practice of link prediction, or identifying potential edges between nodes, has practical importance. In biological networks, a link might be an interaction between proteins, in social networks, it might be a friend request between acquaintances.

Various algorithms based on information theory fundamentals, such as ARACNE (Margolin, 2004) and MRNET (Meyer, 2007), have found success for specific types of chemical reaction networks. In this poster we show the more general efficacy of MIDER (Villaverde et al., 2014), for various cellular networks, and also apply it to a time series of CPU usage by process on a laptop.

Definitions

Entropy

$$H(X) = -\sum_x p(x) \log p(x)$$

Conditional Entropy

$$H(X|Y) = -\sum_x \sum_y p(x,y) \log p(x|y)$$

Mutual Information

$$I(X,Y) = H(X) - H(X|Y)$$

Transfer Entropy

$$T_{X \to Y} = I(Y_t; X_{t-1:t-L} \mid Y_{t-1:t-L})$$

Datasets

The first dataset consists of the concentrations of ten different compounds measured during the process of glycolysis. They were recorded at 57 distinct times.

The second dataset borrows the idea (Wiggins at al., 2003) of applying bioinformatic algorithms to CPU usage time series. Their paper substitutes relative %CPU for concentrations and process types (emacs, latex, etc.) for genes across a multi-user network.

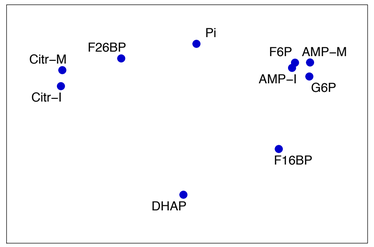
My dataset tracks the CPU usage of each process of a single user (myself) on a laptop over the course of an hour by using the Python package Syrupy (System Resource Usage Profiler) to track activity.

Algorithm

- **Step 1:**
  - Estimate distance between each pair of variables as:

    $$d(X,Y) = e^{-I(X,Y)}$$

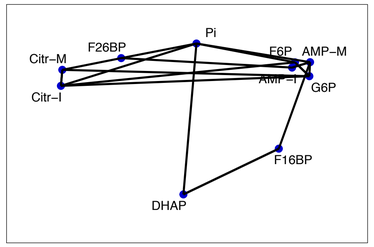
  - Project distance matrix onto two dimensions for graphing
  - *Note: Calculate mutual information by discretizing data with an adaptive partitioning algorithm (e.g. Fraser-Swinney)*



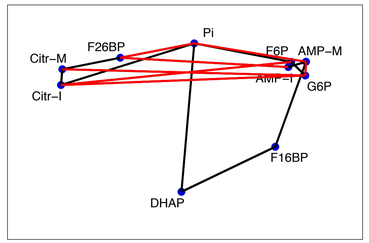
- **Step 2:**
  - For each variable Y, consider edges to the three variables X_1, X_2, and X_3 that successively maximize entropy reduction:

    $$H(Y) - H(Y|X_1)$$
    $$H(Y|X_1) - H(Y|X_1, X_2)$$
    $$H(Y|X_1, X_2) - H(Y|X_1, X_2, X_3)$$

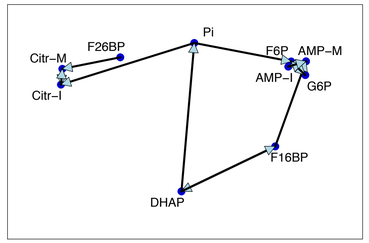
  - *Note: in theory, mutual information for any marginally independent variable would be 0 and this step would terminate; in practice, small, spurious entropy reductions occur so we cap potential edges at 3 (for computational reasons as well)*



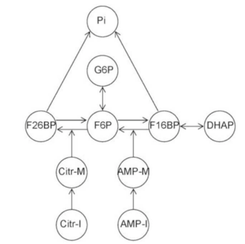
- **Step 3:**
  - Eliminate any edges where the entropy reduction is < K
  - Suggested K: 0 – 0.2(H(Y))

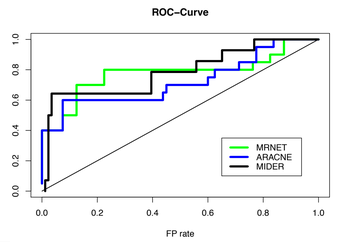


- **Step 4:**
  - For each ordered pair of variables, calculate the time lag L that maximizes mutual information and calculate the transfer entropy at that lag
  - Assign direction to remaining edges based on higher transfer entropy between the two nodes

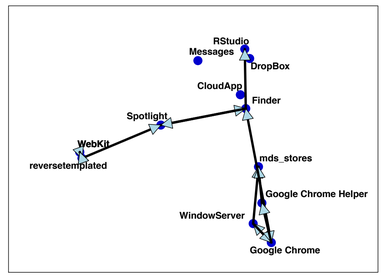


Performance





As is evident once the true glycolytic pathway is revealed, the network that MIDER came up with is far from a perfect reproduction. However, such graphs are famously hard to reverse engineer, especially with such little data, and the algorithm outperforms the two aforementioned procedures according to AUC.



While we can't compare the graph predicted for my laptop's processes with a ground truth, we do see that some of the connections make sense. Incoming messages and occasional Dropbox syncs are active on a generally random basis, and thus have no inferred edges. Google Chrome is highly interconnected with its helper app and also the process that controls displayed windows. Spotlight and Finder activity, both contingent on my directory, share an edge.

Reference

Villaverde AF, Ross J, Mora F, Banga JR (2014) MIDER: Network Inference with Mutual Information Distance and Entropy Reduction. PLoS ONE 9(5): e96732. doi:10.1371/journal.pone.0096732

H. Wiggins, C & Nemenman, Ilya. (2003). Process Pathway Inference via Time Series Analysis. Experimental Mechanics. 43. 361-370. 10.1007/BF02410536.

All implementations of MIDER and relevant graphics were done in R. Code can be found at:
https://github.com/WalkerHarrison/STA563_FinalProject