In silico prediction of edges

CMSC 858L
Even high-throughput experiments are expensive

• Look to computation to infer linkages

• Experiments are noisy, so the bar is low :)

• Combination of multiple experiments improves accuracy drastically; stands to reason weak computational predictors can be combined.

• > 500 complete genomes now publicly available; can’t yet run Y2H or TAP-MS on all of them.

• Hopefully exploit complementary kinds of information with different biases
Other “Edge” Predictions

- Predicting other types of linkages:
  1. **Functional linkage**: two proteins involved in the same process.
  2. **Co-complexed**: two proteins involved in the same complex (w/o necessarily directly interacting).

- Machine learning methods that integrate external information.
  2. Decision trees (Zhang et al, 2003)

- Not strictly protein-protein interactions.

- Combining noisy graphs (e.g. Srinivasan et al, 2006).
Functional, Physical, Genetic Linkage

• Physical linkage

• Functional linkage

• Not independent: pairs in the same function are more likely to physically interact.

• “Genetic” linkage - AKA synthetic lethality
## Computationally Predicting Edges

<table>
<thead>
<tr>
<th>Method</th>
<th>Kind of Linkage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phylogenetic Profiles</td>
<td>Functional</td>
</tr>
<tr>
<td>Co-evolution</td>
<td>Physical / Functional</td>
</tr>
<tr>
<td>Rosetta Stone</td>
<td>Physical</td>
</tr>
<tr>
<td>Domain association</td>
<td>Physical</td>
</tr>
<tr>
<td>Gene neighborhood</td>
<td>Functional</td>
</tr>
<tr>
<td>Co-expression</td>
<td>Functional</td>
</tr>
</tbody>
</table>
Similar function

Similar Evolutionary History
Similar function

Similar Evolutionary History
History Representation

Phylogenetic profile:

\[
\begin{bmatrix}
0 & 1 & 0 & 0 & 1 & 1 & 1
\end{bmatrix}
\]

**homolog**: protein of similar sequence

Can use real values instead of 0/1 representing similarity of protein

(Pellegrini+99, Marcotte+99; Date & Marcotte,03;...)
Shared Function

Need way to compare profiles:

\[ Sim(P_1, P_2) \Rightarrow \text{shared function} \]

- Hamming distance (Pellegrini+99)
- Correlation Coefficient (Wu+03)
- Hypergeometric distribution (Wu+03)
- Mutual Information (Date+03, Wu+03)

How much does knowing one profile tell you about the other?

“Coins” have either 2 (binary profiles) or 10 (gradated profiles) sides (Date+03)
Mutual Information

- Entropy $H(X) :=$ amount of uncertainty in a random value:

$$H(X) := \sum_{a \in \text{domain}(X)} -p_a \log p_a$$

- $\approx$ expected minimum # of questions to determine r.v. $X$.
  $\approx$ average length of shortest description of $X$

- Conditional Entropy := expected uncertainty in $Y$, given the value for $X$:

$$H(Y|X) = \sum_{a \in \text{domain}(X)} p_a H(Y|X = a)$$

- Mutual Information := reduction in uncertainty in $Y$ if you know $X$.

$$I(X; Y) := H(Y) - H(Y|X)$$
Date & Marcotte

- Binned real-value profiles
- 0.75 MI score cutoff
- Pathway Similarity = Jaccard Coefficient
Inferred Yeast Network

0.75 MI cutoff
804 proteins
3,875 linkages
Gradated MI vs. Binary MI

Remove ‘Ribosome’ function:

Binary

Gradated
Another bad name :) -- it seems everything that seeks to reveal hidden meanings is called Rosetta something.

- Proteins fused to better couple expression
- Proteins split to increase flexibility

Predicts 6,809 interactions in E. coli, 5% of which are corroborated by phylogenetic profiles; Among pairs with annotations, 68% share a keyword.
Evolution of Interacting Proteins:

Marcotte et al., 1999
Domain association

- Use interacting pairs to infer additional interacting pairs
- 1274 known interactions
- Domain assignments from InterPro database.
- 434 types of domains
- Score:
  \[ \log_2 \left( \frac{P_{ij}}{P_i P_j} \right) \]
  - 1141 pairs score \( \geq 2 \)
  - 40 pairs with score \( \geq 2 \) AND count \( \geq 5 \)
  - Leave-one-out cross validation \( \Rightarrow 94\% \) specificity.

Sprinzak & Margalit, 2001
Co-evolution

- Within a family known to interact generally, find the pairs which actually interact.

- Interacting partners will co-evolve (with, e.g., compensatory mutations)

Goh et al, 2000
Gertz et al, 2003

Gertz et al, 2003
Pairing Proteins

• Represent tree by distance matrix

• Swap rows / columns in one matrix to maximize the correlation coefficient:

\[
    r(X,Y) = \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (X_{ij} - \bar{X})(Y_{ij} - \bar{Y})}{\sqrt{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (X_{ij} - \bar{X})^2} \sqrt{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (Y_{ij} - \bar{Y})^2}}
\]

• Monte Carlo, with Metropolis criterion:
  Choose 2 indices, swap, keep with probability:

\[
P(\sigma S_i | S_i) := \begin{cases} 
1 & \text{if } \Delta E \leq 0, \\
e^{-\Delta E / T} & \text{if } \Delta E > 0,
\end{cases}
\]

Gertz et al, 2003
Predicting Missing Edges

- Completing Defective Cliques (Yu, et al, 2006):
  
  - P, Q both adjacent to all nodes in clique (there are two \((n-1)\)-cliques that overlap by \((n-2)\) nodes) ⇒ likely that P,Q should be adjacent.
Review

- Experimental methods for determining interactions:
  - Yeast 2-hybrid: physical interactions
  - TAP-MS: protein complexes
  - Chip-Chip: Regulatory networks
  - Synthetic Lethal: “redundant” pathways
  - Noisy: Y2H might have 50% FP rate, e.g. but combinations do much better.
  - High-throughput data available in many databases

- Computational methods for determining linkages:
  - Phylogenetic profiles
  - Rosetta stone
  - Domain association
  - Co-evolution

These methods require homology detection, which is a hard problem itself.
Code & Visualization Resources
<table>
<thead>
<tr>
<th>Library</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetworkX</td>
<td>Implemented in Python</td>
</tr>
<tr>
<td></td>
<td>Large collection of analysis functions</td>
</tr>
<tr>
<td>IGraph</td>
<td>Implemented in C</td>
</tr>
<tr>
<td></td>
<td>Can call from R</td>
</tr>
<tr>
<td></td>
<td>Bindings for several other languages (python, ruby)</td>
</tr>
<tr>
<td>C++ BOOST</td>
<td>Graph data structures with some advanced algorithms (e.g. planarity testing)</td>
</tr>
<tr>
<td></td>
<td>Infested with C++isms.</td>
</tr>
</tbody>
</table>
import networkx as nx
import matplotlib.pyplot as plt

T = nx.tutte_graph()
nx.draw(T)
plt.show()

Tutte Graph
(First example of a 3-regular, 3-connected graph without a Hamiltonian cycle)
Osprey

http://biodata.mshri.on.ca/osprey/servlet/Index
Pajek

http://vlado.fmf.uni-lj.si/pub/networks/pajek/
Graph Visualization

Graph visualization is a way of representing structural information as diagrams of abstract graphs and networks. Automatic graph drawing has many important applications in software engineering, database and web design, networking, and in visual interfaces for many other domains.

Graphviz is open source graph visualization software. It has several main graph layout programs. See the gallery for some sample layouts. It also has web and interactive graphical interfaces, and auxiliary tools, libraries, and language bindings.

The Mac OS X edition of Graphviz, by Glen Low, won two 2004 Apple Design Awards.

The Graphviz layout programs take descriptions of graphs in a simple text language, and make diagrams in several useful formats such as images and SVG for web pages, Postscript for inclusion in PDF or other documents; or display in an interactive graph browser. (Graphviz also supports GXL, an XML dialect.)

Graphviz has many useful features for concrete diagrams, such as options for colors, fonts, tabular node layouts, line styles, hyperlinks, and custom shapes.

In practice, graphs are usually generated from an external data sources, but they can also be created and edited

http://www.graphviz.org/
/* courtesy Ian Darwin and Geoff Collyer, Softquad Inc. */
digraph unix {
    size="6,6";
    node [color=lightblue2, style=filled];
    "5th Edition" -> "PWB 1.0";
    "6th Edition" -> "LSX";
    "6th Edition" -> "1 BSD";
    "6th Edition" -> "Mini Unix";
    "6th Edition" -> "Wollongong";
    "6th Edition" -> "Interdata";
    "Interdata" -> "Unix/TS 3.0";
    "Interdata" -> "PWB 2.0";
    "Interdata" -> "7th Edition";
    "7th Edition" -> "8th Edition";
    "7th Edition" -> "32V";
    "7th Edition" -> "V7M";
    "7th Edition" -> "Ultrix-11";
    "7th Edition" -> "Xenix";
    "7th Edition" -> "UniPlus+";
    "V7M" -> "Ultrix-11";
    "8th Edition" -> "9th Edition";
    "1 BSD" -> "2 BSD";
    "2 BSD" -> "2.8 BSD";
    "2.8 BSD" -> "Ultrix-11";
    "2.8 BSD" -> "2.9 BSD";
    "32V" -> "3 BSD";
    "3 BSD" -> "4 BSD";
    "4 BSD" -> "4.1 BSD";
    "4.1 BSD" -> "4.2 BSD";
    "4.1 BSD" -> "2.8 BSD";
    "4.1 BSD" -> "8th Edition";
    "4.2 BSD" -> "4.3 BSD";
    "4.2 BSD" -> "Ultrix-32";
    "PWB 1.0" -> "PWB 1.2";
    "PWB 1.0" -> "USG 1.0";
    "PWB 1.2" -> "PWB 2.0";
    "USG 1.0" -> "CB Unix 1";
    "USG 1.0" -> "USG 2.0";
    "CB Unix 1" -> "CB Unix 2";
    "CB Unix 2" -> "CB Unix 3";
    "CB Unix 3" -> "Unix/TS++";
Cytoscape

http://www.cytoscape.org/

- Written in Java
- Large number of pluggins