V7 – Biological PPI Networks

- graph bisection (-> communities)
- are biological networks really scale-free?
  - network growth
- functional annotation in the network

Mon, Nov 14, 2016
Modularity: an example of graph partitioning

The simplest graph partitioning problem is the division of a network into just 2 parts. This is called graph bisection.

If we can divide a network into 2 parts, we can also divide it further by dividing one or both of these parts ...

**graph bisection problem:** divide the vertices of a network into 2 non-overlapping groups of given sizes such that the number of edges running between vertices in different groups is minimized.

The number of edges between groups is called the cut size.

In principle, one could simply look through all possible divisions of the network into 2 parts and choose the one with smallest cut size.
Algorithms for graph partitioning

But this exhaustive search is prohibitively expensive!

Given a network of \( n \) vertices. There are \( \frac{n!}{n_1!n_2!} \) different ways of dividing it into 2 groups of \( n_1 \) and \( n_2 \) vertices.

The amount of time to look through all these divisions will go up roughly exponentially with the size of the system.

Only values of up to \( n = 30 \) are feasible with today’s computers.

In computer science, either an algorithm can be clever and run quickly, but will fail to provide the optimal answer in some (or perhaps in many) cases, or it will always find the optimal answer, but takes an impractical length of time to do so.
The Kernighan-Lin algorithm

This algorithm proposed by Brian Kernighan and Shen Lin in 1970 is one of the simplest and best known heuristic algorithms for the graph bisection problem. (Kernighan is also one of the developers of the C language).

(a) The algorithm starts with any division of the vertices of a network into two groups (shaded) and then searches for pairs of vertices, such as the pair highlighted here, whose interchange would reduce the cut size between the groups.

(b) The same network after interchange of the 2 vertices.
The Kernighan-Lin algorithm

(1) Divide the vertices of a given network into 2 groups (e.g. randomly).

(2) For each pair \((i, j)\) of vertices, where \(i\) belongs to the first group and \(j\) to the second group, calculate how much the cut size between the groups would change if \(i\) and \(j\) were interchanged between the groups.

(3) Find the pair that reduces the cut size by the largest amount and swap the vertices.

If no pair reduces it, find the pair that increases it by the smallest amount.

Repeat this process, but with the important restriction that each vertex in the network can only be moved once.

Stop when there is no pair of vertices left that can be swapped.
The Kernighan-Lin algorithm (II)

(3) Go back through every state that the network passed through during the swapping procedure and choose among them the state in which the cut size takes its smallest value.

(4) Perform this entire process repeatedly, starting each time with the best division of the network found in the last round.

(5) Stop when no improvement on the cut size occurs.

Note that if the initial assignment of vertices to groups is done randomly, the Kernighan-Lin algorithm may give (slightly) different answers when it is run twice on the same network.
The Kernighan-Lin algorithm (II)

(a) A mesh network of 547 vertices of the kind commonly used in finite element analysis.
(b) The best division found by the Kernighan-Lin algorithm when the task is to split the network into 2 groups of almost equal size. This division involves cutting 40 edges in this mesh network and gives parts of 273 and 274 vertices.
(c) The best division found by spectral partitioning (alternative method).
Runtime of the Kernighan-Lin algorithm

The number of swaps performed during one round of the algorithm is equal to the smaller of the sizes of the two groups $\in [0, n/2]$.

→ in the worst case, there are $O(n)$ swaps.

For each swap, we have to examine all pairs of vertices in different groups to determine how the cut size would be affected if the pair was swapped.

At most (if both groups have the same size), there are $n/2 \times n/2 = n^2 / 4$ such pairs, which is $O(n^2)$. 
Runtime of the Kernighan-Lin algorithm (ii)

When a vertex $i$ moves from one group to the other group, any edges connecting it to vertices in its current group become edges between groups after the swap.

Let us suppose that there are $k_i^{\text{same}}$ such edges.

Similarly, any edges that $i$ has to vertices in the other group, (say $k_i^{\text{other}}$ ones) become within-group edges after the swap.

There is one exception. If $i$ is being swapped with vertex $j$ and they are connected by an edge, then the edge is still between the groups after the swap

→ the change in the cut size due to the movement of $i$ is $-(k_i^{\text{other}} - k_i^{\text{same}} - A_{ij})$

A similar expression applies for vertex $j$.

→ the total change in cut size due to the swap is

$$-(k_i^{\text{other}} - k_i^{\text{same}} + k_j^{\text{other}} - k_j^{\text{same}} - 2A_{ij})$$
Runtime of the Kernighan-Lin algorithm (iii)

For a network stored in adjacency list form, the evaluation of this expression involves running through all the neighbors of $i$ and $j$ in turn, and hence takes time on the order of the average degree in the network, or $O (m/n)$ with $m$ edges in the network.

→ the total running time is $O ( n \times n^2 \times m/n ) = O(mn^2)$.

For a sparse network with $m \propto n$, this is $O(n^3)$.

For a dense network (with $m \rightarrow \frac{n(n-1)}{2}$), this is $O(n^4)$.

This time still needs to be multiplied by the number of rounds the algorithm is run before the cut size stops decreasing.

For networks up to a few 1000 of vertices, this number may be between 5 and 10.
Lethality and centrality in protein networks

The most highly connected proteins in the cell are the most important for its survival.


→ "PPI networks apparently are scale-free…"

"Are" they scale-free or "Do they look like" scale-free???
Generate networks of various types, 
**sample sparsely** from them  
→ determine degree distribution

- Random (ER / Erdös-Renyi) → $P(k) = \text{Poisson}$  
- Exponential (EX) → $P(k) \sim \exp[-k]$  
- scale-free / power-law (PL) → $P(k) \sim k^{-\gamma}$  
- $P(k) = \text{truncated normal distribution (TN)}$
Partial Sampling

**Estimated for yeast:** 6000 proteins, 30000 interactions

Y2H experiments *detected* only 3...9% of the complete interactome!
**R square**

Given: a data set with \( n \) values \( y_1,...,y_n \) and a set of fitted / predicted / modelled) values \( f_1,...,f_n \) e.g. from linear regression.

We call their difference **residuals** \( e_i = y_i - f_i \)

and the mean value

\[
\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i
\]

The total sum of squares (proportional to the variance of the data) is:

\[
SS_{tot} = \sum_{i} (y_i - \bar{y})^2,
\]

The sum of squares of residuals is:

\[
SS_{res} = \sum_{i} (y_i - f_i)^2 = \sum_{i} e_i^2
\]

The **coefficient of determination**, \( R^2 \) or \( r^2 \) is often defined as:

\[
R^2 \equiv 1 - \frac{SS_{res}}{SS_{tot}}.
\]
Sparsely Sampled random (ER) Network

resulting $P(k)$ for different coverages

(c) Shows linearity (R square) between detected $P(k)$ and ideal power law; good agreement (red; $R \approx 1$ for low edge coverage)

for sparse sampling (10-20%), even an ER network "looks" scale-free (when only $P(k)$ is considered)

Anything Goes – different topologies

Compare to Uetz et al. data

Uetz et al. data (solid line) is compared to sampled networks of similar size.

Sampling density affects observed degree distribution → true underlying network cannot be identified from available data
Network Growth Mechanisms

Given: an observed PPI network → how did it grow (evolve)?

Inferring network mechanisms: The *Drosophila melanogaster* protein interaction network

Manuel Middendorf\(^1\), Eytay Ziv\(^6\), and Chris H. Wiggins\(^9\)

\(^1\)Department of Physics, \(^2\)College of Physicians and Surgeons, \(^4\)Department of Applied Physics and Applied Mathematics, and \(^6\)Center for Computational Biology and Bioinformatics, Columbia University, New York, NY 10027

Communicated by Barry H. Henin, Columbia University, New York, NY. December 20, 2004 (received for review September 7, 2004)

*PNAS* **102** (2005) 3192

Look at **network motifs** (local connectivity):
compare motif distributions from various network prototypes to fly network

**Idea:** each growth **mechanism** leads to a typical motif **distribution**,
even if global measures are comparable
The Fly Network

Y2H PPI network for *D. melanogaster* from Giot et al. [Science 302 (2003) 1727]

Giot *et al.* assigned a confidence score [0, 1] for every observed interaction.

→ use only data with \( p > 0.65 \) (0.5) because ...

→ remove self-interactions and isolated nodes

High confidence network with 3359 (4625) nodes and 2795 (4683) edges.

Use prototype networks of same size for training.

Size of largest components. At \( p = 0.65 \), there is one large component with 1433 nodes and the other 703 components contain at most 15 nodes.

Network subgraphs -> motives

All non-isomorphic subgraphs that can be generated with a walk of length 8

Middendorf et al, PNAS 102 (2005) 3192
Growth Mechanisms

Generate 1000 networks, each, of the following 7 types
(same size as fly network, undefined parameters were scanned)

DMC Duplication-mutation, preserving complementarity
DMR Duplication with random mutations
RDS Random static networks
RDG Random growing network
LPA Linear preferential attachment network (Albert-Barabasi)
AGV Aging vertices network
SMW Small world network
Growth Type 1: DMC

"Duplication – mutation with preserved complementarity"

**Evolutionary idea:** gene **duplication**, followed by a partial **loss** of function of one of the copies, making the other copy essential

**Algorithm:**

Start from two connected nodes

- duplicate existing node with all interactions

- for all neighbors: delete with probability $q_{del}$ either link from original node **or** from copy

Repeat these steps many (e.g. $N - 2$) times
Growth Type 2: DMR

"Duplication with random mutations"

Gene duplication, but no correlation between original and copy (original unaffected by copy)

**Algorithm:**

Start growth from five-vertex cycle, repeat $N - 5$ times:

- duplicate existing node with all interactions
- for all neighbors: delete with probability $q_{\text{del}}$ link from copy
- add **new links** to non-neighbors with probability $q_{\text{new}}/n$
Growth Types 3–5: RDS, RDG, and LPA

**RDS** = static random network

Start from $N$ nodes, add $L$ links randomly

**RDG** = growing random network

Start from small random network, add nodes, then edges between all existing nodes

**LPA** = linear preferential attachment

Add new nodes similar to Barabási-Albert algorithm, but with preference according to $(k_i + \alpha)$, $\alpha = 0 \ldots 5$ (BA for $\alpha = 0$)
Growth Types 6-7: AGV and SMW

**AGV** = aging vertices network

Like growing random network, but preference decreases with age of the node → citation network: more recent publications are cited more likely


Randomly rewire regular ring lattice
Alternating Decision Tree Classifier

Trained with the motif counts from 1000 networks of each of the 7 types → prototypes are well separated and can be reliably classified

Prediction accuracy for networks similar to fly network with $p = 0.5$:

<table>
<thead>
<tr>
<th>Truth</th>
<th>DMR</th>
<th>DMC</th>
<th>AGV</th>
<th>LPA</th>
<th>SMW</th>
<th>RDS</th>
<th>RDG</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMR</td>
<td>99.3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.6</td>
</tr>
<tr>
<td>DMC</td>
<td>0.0</td>
<td>99.7</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>AGV</td>
<td>0.0</td>
<td>0.1</td>
<td>84.7</td>
<td>13.5</td>
<td>1.2</td>
<td>0.5</td>
<td>0.0</td>
</tr>
<tr>
<td>LPA</td>
<td>0.0</td>
<td>0.0</td>
<td>10.3</td>
<td>89.6</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>SMW</td>
<td>0.0</td>
<td>0.0</td>
<td>0.6</td>
<td>0.0</td>
<td>99.0</td>
<td>0.4</td>
<td>0.0</td>
</tr>
<tr>
<td>RDS</td>
<td>0.0</td>
<td>0.0</td>
<td>0.2</td>
<td>0.0</td>
<td>0.8</td>
<td>99.0</td>
<td>0.0</td>
</tr>
<tr>
<td>RDG</td>
<td>0.9</td>
<td>0.0</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>99.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Middendorf et al, PNAS 102 (2005) 3192
Are the generated networks different?

Example: DMR vs. RDG: Similar global parameters $\langle C \rangle$ and $\langle l \rangle$ (left), but different counts of the network motifs (right)

$\rightarrow$ networks can (only) be perfectly separated by motif-based classifier

How Did the Fly Evolve?

<table>
<thead>
<tr>
<th>Rank</th>
<th>Class</th>
<th>Score</th>
<th>Rank</th>
<th>Class</th>
<th>Score</th>
<th>Rank</th>
<th>Class</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DMC</td>
<td>8.2 ± 1.0</td>
<td>1</td>
<td>DMC</td>
<td>8.6 ± 1.1</td>
<td>1</td>
<td>DMC</td>
<td>0.8 ± 2.9</td>
</tr>
<tr>
<td>2</td>
<td>DMR</td>
<td>-6.8 ± 0.9</td>
<td>2</td>
<td>DMR</td>
<td>-6.1 ± 1.7</td>
<td>2</td>
<td>DMR</td>
<td>-2.1 ± 2.0</td>
</tr>
<tr>
<td>3</td>
<td>RDG</td>
<td>-9.5 ± 2.3</td>
<td>3</td>
<td>RDG</td>
<td>-9.3 ± 1.6</td>
<td>3</td>
<td>AGV</td>
<td>-3.1 ± 2.2</td>
</tr>
<tr>
<td>4</td>
<td>AGV</td>
<td>-10.6 ± 4.2</td>
<td>4</td>
<td>AGV</td>
<td>-11.5 ± 4.1</td>
<td>4</td>
<td>LPA</td>
<td>-10.1 ± 3.1</td>
</tr>
<tr>
<td>5</td>
<td>LPA</td>
<td>-16.5 ± 3.4</td>
<td>5</td>
<td>LPA</td>
<td>-14.3 ± 3.2</td>
<td>5</td>
<td>SMW</td>
<td>-20.6 ± 1.9</td>
</tr>
<tr>
<td>6</td>
<td>SMW</td>
<td>-18.9 ± 0.7</td>
<td>6</td>
<td>SMW</td>
<td>-18.3 ± 1.9</td>
<td>6</td>
<td>RDS</td>
<td>-22.3 ± 1.7</td>
</tr>
<tr>
<td>7</td>
<td>RDS</td>
<td>-19.1 ± 2.3</td>
<td>7</td>
<td>RDS</td>
<td>-19.9 ± 1.5</td>
<td>7</td>
<td>RDG</td>
<td>-22.5 ± 4.7</td>
</tr>
</tbody>
</table>

*Drosophila* is consistently (independently of the cut-off in subgraph size) classified as a DMC network, with an especially strong prediction for a confidence threshold of $p^* = 0.65$.

→ Best overlap with DMC (Duplication-mutation, preserved complementarity)
→ Scale-free (LPA) or random networks (RDS/RDG) are very unlikely

Motif Count Frequencies

-> DMC and DMR networks contain most subgraphs in similar amount as fly network (top).

rank score: fraction of test networks with a higher count than Drosophila (50% = same count as fly on avg.)

Middendorf et al, PNAS 102 (2005) 3192
Experimental Errors?

**Randomly** replace edges in fly network and **classify** again:

→ Classification **unchanged** for ≤ 30% incorrect edges, at higher values RDS takes over (as to be expected)
Summary (I)

Sampling matters!

→ "Scale-free" $P(k)$ is obtained by sparse sampling from many network types

Test different hypotheses for

• **global** features
  → depends on unknown parameters and sampling
    → no clear statement possible

• **local** features (motifs)
  → are better preserved
    → DMC best among tested prototypes
What Does a Protein Do?

**Enzyme Classification scheme**
(from http://www.brenda-enzymes.org/)
What about Un-Classified Proteins?

Many unclassified proteins:
→ estimate: \(\sim 1/3\) of the yeast proteome not annotated functionally
→ BioGRID: 4495 proteins in the largest cluster of the yeast physical interaction map.

only 2946 have a MIPS functional annotation
Partition the Graph

Large PPI networks can be built from (see V3, V4, V5):
• HT experiments (Y2H, TAP, synthetic lethality, coexpression, coregulation, …)
• predictions (gene profiling, gene neighborhood, phylogenetic profiles, …)
→ proteins that are functionally linked

Identify unknown functions from clustering of these networks by, e.g.:
• shared interactions (similar neighborhood)
• membership in a community
• similarity of shortest path vectors to all other proteins (= similar path into the rest of the network)
Protein Interactions

Nabieva et al used the *S. cerevisiae* dataset from GRID of 2005 (now BioGRID) → 4495 proteins and 12 531 physical interactions in the largest cluster

http://www.thebiogrid.org/about.php
Function Annotation

**Task:** predict function (= functional annotation) for an unlabeled protein from the available annotations of other proteins in the network

Similar task:
How to assign colors to the white nodes?

Use information on:
- distance to colored nodes
- local connectivity
- reliability of the links
- …
Algorithm I: Majority

This concept was presented in Schwikowski, Uetz, and Fields, "A network of protein–protein interactions in yeast" Nat. Biotechnol. 18 (2000) 1257

Consider all direct neighbors and **sum up** how often a certain **annotation occurs**

→ score for an annotation = count among the direct neighbors
  → take the 3 most frequent functions

Majority makes only limited use of the local connectivity

→ cannot assign function to next-neighbors

For weighted graphs:

→ use weighted sum
Extended Majority: Neighborhood

This concept was presented in Hishigaki, Nakai, Ono, Tanigami, and Takagi, "Assessment of prediction accuracy of protein function from protein–protein interaction data", *Yeast* **18** (2001) 523

Look for *overrepresented* functions within a given *radius* of 1, 2, or 3 links → use as function score the value of a $\chi^2$–test

Neighborhood algorithm does not consider local network topology

Both examples (left) are treated *identically* with $r = 2$

although the right situation feels more certain (2 direct neighbors of ? are labeled)
Minimize Changes: GenMultiCut

This concept was presented in Karaoz, Murali, Letovsky, Zheng, Ding, Cantor, and Kasif, "Whole-genome annotation by using evidence integration in functional-linkage networks" PNAS 101 (2004) 2888

"Annotate proteins so as to **minimize** the number of times that **different** functions are associated to **neighboring** (i.e. interacting) proteins"

→ generalization of the multiway $k$-cut problem for weighted edges, can be stated as an integer linear program (ILP)

Multiple possible solutions → scores from **frequency** of annotations
Nabieva et al: FunctionalFlow

Extend the idea of "guilty by association"
→ each annotated protein is considered as a source of "function"-flow
   → propagate/simulate for a few time steps
   → choose the annotation \( a \) with the highest accumulated flow

Each node \( u \) has a reservoir \( R_t(u) \), each edge a capacity constraint (weight) \( w_{u,v} \)

**Initially:**
\[
R_0^a(u) = \begin{cases} 
\infty, & \text{if } u \text{ is annotated with } a, \\
0, & \text{otherwise.}
\end{cases}
\]

and \( g_0^a(u, v) = 0 \)

Then: **downhill flow** from node \( u \) to neighbor node \( v \):

\[
g_t^a(u, v) = \begin{cases} 
0, & \text{if } R_{t-1}^a(u) < R_{t-1}^a(v) \\
\min \left( w_{u,v}, \frac{w_{u,v}}{\sum_{(u,y) \in E} w_{u,y}} \right), & \text{otherwise.}
\end{cases}
\]

**Score** from accumulated in-flow:

\[
f_a(u) = \sum_{t=1}^{d} \sum_{v: (u,v) \in E} g_t^a(v, u)
\]
An Example

accumulated flow

thickness = current flow

.....
Comparison

For FunctionalFlow:
six propagation steps were simulated; this is comparable
to the diameter of the yeast network $\approx 12$

Majority results are initially very good, but has limited coverage.

Results with neighborhood get more imprecise for larger radii $r$

Change **score threshold** for accepting annotations $\rightarrow$ ratio $\frac{TP}{FP}$
$\rightarrow$ **FunctionalFlow** performs **best** in the high-confidence region
$\rightarrow$ but generates still many false predictions!!!
Going the Distance for Protein Function Prediction: A New Distance Metric for Protein Interaction Networks

Citation: Cao M, Zhang H, Park J, Daniels NM, Crovella ME, et al. (2013) Going the Distance for Protein Function Prediction: A New Distance Metric for Protein Interaction Networks. PLoS ONE 8(10): e76339. doi:10.1371/journal.pone.0076339

Relying on the ordinary shortest-path distance metric in PPI networks is problematic because PPI networks are “small world” networks. Most nodes are “close” to all other nodes.

→ any method that infers similarity based on proximity will find that a large fraction of the network is proximate to any typical node.

Largest connected component of *S. cerevisiae* PPI network (BioGRID) has 4990 nodes and 74,310 edges (physical interactions).

Right figure shows the histogram of shortest-path lengths in this network. Over 95% of all pairs of nodes are either 2 hops or 3 hops apart.
What nodes mediate short contacts?

The 2-hop neighborhood of a typical node probably includes around half of all nodes in the graph.

One of the reasons that paths are typically short in biological networks like the PPI network is due to the presence of hubs.

But hub proteins often represent proteins with different functional roles than their neighbors.

Hub proteins likely also have multiple, distinct functions.

→ not all short paths provide equally strong evidence of similar function in PPI networks.
DSD Distance Metric

Given some fixed $k > 0$, we define $He^{[k]}(A,B)$ to be the expected number of times that a random walk starting at $A$ and proceeding for $k$ steps, will visit $B$. If there is no ambiguity about $k$, we can drop $k$.

$$He(v_i) = (He(v_i,v_1), He(v_i,v_2), ..., He(v_i,v_n))$$

$He(v_i)$ is a „random walk distance vector“ of node $v_i$ from all other nodes.

$$DSD(u,v) = \|He(u) - He(v)\|_1$$

where

$$\|He(u) - He(v)\|_1$$

denotes the $L_1$ norm of the $He$ vectors

Two nodes $u$ and $v$ have small DSD if they have similar distance from all other nodes.

Explanation: The one-norm (also known as the $L_1$-norm, $\ell_1$ norm, or mean norm) of a vector $\vec{v}$ is denoted $\|\vec{v}\|_1$ and is defined as the sum of the absolute values of its components:

$$\|\vec{v}\|_1 = \sum_{i=1}^{n} |v_i|$$

(1)

for example, given the vector $\vec{v} = (1, -4, 5)$, we calculate the one-norm:

$$\|(1, -4, 5)\|_1 = |1| + |-4| + |5| = 10$$
DSD clearly improves functional predictions

**MIPS Top Level, Accuracy**

**MIPS Second Level, Accuracy**

**MIPS Third Level, Accuracy**

**F1 Score on GO term Prediction for S. cerevisiae**

Figure 6. Improvement on F1 Score for DSD using three evaluation methods: exact match, overlap depth and overlap counting, on informative GO terms for the four algorithms for *S. cerevisiae* in 10 runs of 2-fold cross validation.
Summary

V8: wrap up protein interaction networks

Then next block of the lecture: gene-regulatory networks