582653 Computational Methods of Systems Biology

Lecture 3: Network motifs

27.1.2011
Network motifs

- Besides by global network properties, networks can be analyzed in terms of their local properties
  - What kind of local substructures (motifs, graphlets) does the network have
  - Assessing the statistical significance of these substructures

Fig. 2. An example graph (a), a pattern (b) and all different matches of the pattern (c, $M_1 - M_5$). The vertices of the graph and of the matches are numbered consecutively for identification purposes.
What is a motif?

- A *motif* is a statistically overrepresented pattern of local interactions in the network.
- Overrepresentation = occurring more frequently than expected by chance.
- The rationale is that overrepresentation may denote possible function that has been conserved in the evolution of the network.

![Diagram of motifs](image-url)
What is a motif?

- A *motif* is a small connected subgraph $G' = (V', E')$
- Size of motif is the number of vertices it contains

![Diagram](image)

*Fig. 2.* An example graph (a), a pattern (b) and all different matches of the pattern (c, $M_1 - M_5$). The vertices of the graph and of the matches are numbered consecutively for identification purposes.
Types of motifs

- Motifs can be
  - Directed or undirected
  - Cyclic (loopy) or acyclic

matching the type of underlying network to be analyzed, e.g.
- Protein-protein interactions: undirected
- Gene regulatory interactions: directed, cyclic
Matching motifs

- A match of a motif $G'$ in the target graph $G = (V, E)$ is a subgraph $G'' = (V'', E'')$ which is *isomorphic* to motif $G'$.
- Two graphs $G'$ and $G''$ are isomorphic if there is a bijective mapping between the edge and vertex identities.
  - i.e. $G'$ is transformed to $G''$ by changing the vertex and edge identities.

![Graphs](image)

**Fig. 2.** An example graph (a), a pattern (b) and all different matches of the pattern (c, $M_1 - M_5$). The vertices of the graph and of the matches are numbered consecutively for identification purposes.
Alternative definition: Induced subgraph

- Isomorphic induced subgraph (graphlet): a subgraph \( G'' = (V'', E'') \) in \( G = (V, E) \) is accepted as a match only if it contains all edges of the original graph between the nodes in \( V'' \): mathematically we require that if \( e = (n_i, n_j) \in E \) and \( n_i, n_j \in V'' \) then \( e \in E'' \)

- Motivation: leaving out interactions from the motif may give false ideas of the biological function
The complexity of graph isomorphism is in the 'grey area' of complexity, it is known to belong to NP class of problems (problems where solution is easy to verify once found) but it is not known whether it is NP-complete (problems that are believed to be hard to solve, but easy to verify).

Subgraph isomorphism, checking if a subgraph $G''$ that is isomorphic to given graph $G'$ exists in a larger graph $G$, is known to be NP-complete.

No hope for really fast algorithms for finding motifs.
Motif frequency

- How many times a motif occurs in the network to be analyzed?
- Depends on
  - Definition of a match (subgraph or induced subgraph)
  - Counting schemes for matches

Fig. 2. An example graph (a), a pattern (b) and all different matches of the pattern (c, M_1 – M_5). The vertices of the graph and of the matches are numbered consecutively for identification purposes.
Counting schemes

- Possible schemes for counting matches
  - $\mathcal{F}_1$: Two matches may overlap so that they share vertices and edges
  - $\mathcal{F}_2$: Two matches may overlap so that they share vertices but not edges
  - $\mathcal{F}_3$: Two matches may not overlap, they need to have disjoint sets of vertices

![Diagram](image-url)
Counting schemes

<table>
<thead>
<tr>
<th>Concept</th>
<th>Graph elements shared by different matches</th>
<th>Values for the example in Fig. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{F}_1$</td>
<td>yes yes</td>
<td>Frequency: 5, Selected matches: ${M_1, M_2, M_3, M_4, M_5}$</td>
</tr>
<tr>
<td>$\mathcal{F}_2$</td>
<td>yes no</td>
<td>Frequency: 2, Selected matches: ${M_1, M_4}$ or ${M_3, M_4}$</td>
</tr>
<tr>
<td>$\mathcal{F}^*$</td>
<td>no yes</td>
<td>Frequency: --</td>
</tr>
<tr>
<td>$\mathcal{F}_3$</td>
<td>no no</td>
<td>Frequency: 1, Selected matches: one of ${M_1, M_2, M_3, M_4, M_5}$</td>
</tr>
</tbody>
</table>

**Fig. 2.** An example graph (a), a pattern (b) and all different matches of the pattern (c, $M_1 - M_5$). The vertices of the graph and of the matches are numbered consecutively for identification purposes.
The frequency of a motif in some network does not directly tell us its importance.

Testing for statistical significance is more informative:
- How often we would expect to see this motif by chance in a similar random network.

Need to formulate a *null hypothesis* and check the probability of the motif occurring as frequently under the null hypothesis.
Null hypothesis from random networks

- In traditional hypothesis testing, one typically analytically formulates a probability distribution for the values of the random variable of interest (here frequency of a motif).
- In network analysis, analytically determining a suitable probability distribution may be difficult.
- Instead, randomization tests are being used: a large set of random networks of appropriate structure are generated and the average frequency of the motif together its variance is recorded.
- Computationally demanding process if the networks are large.
Randomization algorithm for Null model networks

- Typical method for null model generation is to take the original network being analyzed and make large number of randomized versions of it by modifying the network by a large number random edit operations.
- Commonly used edit operation is to rewire the network locally:
  - Take two edges \((A, B)\) and \((C, D)\) and replace them with edges \((A, D)\) and \((C, B)\).
  - Preserves degree distribution of nodes.
  - If the nodes are chosen from a small neighborhood, also keeps average path length close to original.
Randomization algorithm for Null model networks

Additional criteria to be preserved can be set, e.g.

- Preserve number of bidirectional edges
- Preserve number of motif of size $n - 1$ when searching for motifs of size $n$
- ...

As a guideline, the null model should as close to the original as possible, but to randomize the property of interest.
Measures of motif significance: Z-score

- Denote by $\mathcal{F}(m)$ the frequency of motif $m$ and by $\overline{\mathcal{F}_r}(m)$ and $\sigma_r(m)$ the average and standard deviation of the motif frequency among the randomized networks.
- Z-score: “how far above the mean of the random networks”

$$Z(m) = \frac{\mathcal{F}(m) - \overline{\mathcal{F}_r}(m)}{\sigma_r(m)}$$

- Z-score above 2.0 is generally considered significant (“two standard deviations””)
Measures of motif significance: P-value

- P-value: “how often a random network has more motif occurrences”

\[ P(m) = \frac{1}{N} \sum_{r=1}^{N} 1\{F_r(m) \geq F(m)\} \]

- \( 1\{A\} \) denotes the indicator function, \( F_r(m) \) denotes the motif’s frequency in \( r \)'th randomized network

- Requires a large number of randomized networks (≈ 1000) to be accurate
  - Estimating the tail of the distribution is harder than estimating its mean (as in Z-score)
Motif significance profile

- Motif significance profile $SP$ is a vector of normalized $Z$-scores for a particular set of motifs

$$SP = ((SP(m_1), \ldots, SP(m_2)),$$

where $SP(m) = Z(m_i)/\sqrt{\sum_j Z(m_j)^2}$.

- Motif significance profile allows comparing different size networks in terms of the motifs they contain

- Typically, the set of motifs contains all motifs of particular size
Hardness of motif discovery

Several challenging subproblems:

- Graph isomorphism testing: required to check if two motifs are in fact the same. No polynomial time algorithm is known for this problem.
- Number of motifs: grows exponentially in the size of the motif. Especially with directed motifs grows very fast.
- Number of matches: theoretically the worst case number of potential matches is $O(|E_t||E_m|)$ where $E_t$ and $E_m$ are the number of edges in the target and motif, respectively.
- Size of analyzed networks affects the above steps via the number of different patterns and matches that can be found.
- Calculation of statistical significance via randomization calls for generation and motif discovery from a large number of networks, multiplying the computation time of all the above points.
Algorithm for finding network motifs


- The method is implemented in the MAVisto tool (http://mavisto.ipk-gatersleben.de/)

- We focus on directed motifs but the method works in principle for undirected motifs as well
Flexible pattern finder (FPF) algorithm

- A frequent graph mining algorithm, specialized for finding motifs from biological networks
- Finds maximally frequent motifs of given size
- Basic approach is to grow a pattern tree (pattern \( \sim \) motif), containing all small subgraphs of that occur in the network with high enough frequency
Downward closure property

- Downward closure property (Called anti-monotonicity in data mining jargon) ensures that
  - The frequency of a pattern can only decrease when it is extended, or equivalently
  - Subgraphs of a graph are at least frequent as the whole graph
- Downward closure property holds true for the counting schemes $\mathcal{F}_2$ (no edges shared) and $\mathcal{F}_3$ (no vertices or edges shared)
- Can be used to stop growing the tree early
Downward closure property

- Downward closure property does not hold true for counting scheme $\mathcal{F}_1$ (edge overlaps allowed)
- This can be seen from the example in the picture
  - Subgraph (b) is found once (3 edges in the center of (a)), but subgraph (c) is found 6 times (4 different edge sets), which are partially overlapping
  - We cannot stop extending the pattern tree at pattern (b) based on the downward closure

*Fig. 3.* A graph of size 9 (a) and two patterns of size 3 (b) and size 4 (c) illustrating that frequency concept $\mathcal{F}_1$ is not downward closed.
Pattern tree

- The root of the tree contains the simplest possible pattern (one edge)
- The children of a node contain subgraphs that are extensions of the pattern by one edge
- With each pattern, the set of matches is stored and updated
Overview of the FPF algorithm

- To avoid generating too many patterns, FPF only includes patterns that are supported by the target network to the tree.
- Only promising branches are explored.
- Each pattern has a *generating parent* that is the only allowed way of generating the pattern (hence we have a pattern tree and not a pattern lattice).
Overview of the FPF algorithm

- The algorithm keeps track of maximum frequency of discovered target motif
- Once the frequency of an intermediate size motif falls below the frequency threshold the growing of the branch can be stopped by the downward closure property
- To find the maximally frequent motifs early, the algorithm uses best-first search in extending the branches (highest frequency motif extended first)
Pattern extension and traversal

- Search starts with patterns of size 1 consisting of one edge and two vertices.
- All matches of this pattern in the target network are generated.
- Matches are represented as sets of edges (at first singleton sets).
- To extend a pattern $p$, its match set $M_p$ is explored for all possible one edge extensions $p'$ for which $p$ is generating parent.
  - A match $m$ is extended to a new match $m'$ by adding an edge that is incident to $m$ in the target graph.
  - If the resulting extension pattern $p'$ does not have $p$ as the generating parent is it discarded.
  - Otherwise, if the pattern was not created already, it is stored as the extension pattern in the tree.
  - The match $m'$ is joined to the match set $M_{p'}$ of pattern $p'$. 
Canonical labels and Generating parent

- The patterns are uniquely identified by their canonical label.
- Isomorphic graphs have the same canonical label.
- Canonical label of a graph can be computed by organizing the adjacency matrix of the graph in a defined way and representing that matrix as a string.
- Graphs can also be (lexicographically) ordered by their canonical labels.
- Generating parent of size $k$ pattern $p'$ is the pattern of size $k - 1$ that has the lexicographically smallest canonical label.
Counting non-overlapping matches

- Given a match set \( M_p = \{m_1, \ldots, m_t\} \) we need to compute the maximum number of non-overlapping matches that set can realize
- This is done by converting the matches into a overlap graph as follows:
  - Each match corresponds to a node
  - If two matches overlap (share edges) they are connected by an edge in the overlap graph
  - Finding the maximum number of non-overlapping matches corresponds to finding maximum independent set in the overlap graph (set of nodes that are not adjacent in the graph).
Enhancements and extensions

The algorithm can be made

- to return all patterns up to given size above given frequency threshold
- to return $n$ most frequent patterns of given size
- to make use of parallel processing (extension of different branches can be made independently)
- to define pattern size in vertices instead of edges