Greedy Algorithm for Community Detection

Matthew Mohr
11/20/2017
Basics of Community Detection

- Community (aka cluster): dense subgraph in a network, characterized by several connections between nodes
- It is not graph partitioning: “Graph partitioning divides a network into a predefined number of smaller subgraphs. In contrast, community detection aims to uncover the inherent community structure of a network”
- Communities can be of varying size, not explicitly known ahead of time, density can vary
Modularity

- Modularity: a metric to assess the quality of communities generated by partitioning a network
- Compares whether the community is “real” enough; based on concept known as the “Random Hypothesis”:
  - Randomly wired networks lack an inherent community structure.
  - Because of this, modularity is calculated by comparing real network expectation to random wiring expectations
- Analogy: 20 people total, 10 spend time together in one class, 10 people in another class, who will end up being friends?
Basic Modularity Calculation

\[ M_c = \frac{1}{2L} \sum_{(i,j) \in C_c} (A_{ij} - p_{ij}) \]

\[ p_{ij} = \frac{k_i k_j}{2L} \]

\[ M_c = \frac{L_c}{L} - \left( \frac{k_c}{2L} \right)^2 \]

\[ M = \sum_{c=1}^{n_c} \left[ \frac{L_c}{L} - \left( \frac{k_c}{2L} \right)^2 \right] \]

- \( M_c \) = Modularity of community
- \( M \) = Total modularity of network
- \( A_{ij} \) = Edges between \( ij \)
- \( p_{ij} \) = Expected random wiring of \( ij \)
- \( k_c \) = Total degree of the community
- \( L_c \) = Links in community \( c \)
Maximal Modularity
Hypothesis/Implications

For a given network the partition with maximum modularity corresponds to the optimal community structure.
Greedy Algorithm Steps

- Assign each node to a community of its own, starting with N communities of single nodes.
- Inspect each community pair connected by at least one link and compute the modularity difference $\Delta M$ obtained if we merge them. Identify the community pair for which $\Delta M$ is the largest and merge them. Note that modularity is always calculated for the full network.
- Repeat Step 2 until all nodes merge into a single community, recording $M$ for each step.
- Select the partition for which $M$ is maximal.
Network Modularity:
All nodes: $L_c = 0$

$$M_A = \frac{0}{3} - \left(\frac{2}{2(3)}\right)^2 = -\frac{1}{9} \quad M = -1/3 = -0.333333$$

Iteration 1:
Merge A and B:

$$M_{AB} = \frac{1}{3} - \left(\frac{4}{2(3)}\right)^2 = -\frac{1}{9} \quad M = 2/9 = -0.222222$$

$$M_c = \frac{0}{3} - \left(\frac{2}{2(3)}\right)^2 = -\frac{1}{9}$$

Iteration 2:

$$M_{ABC} = \frac{3}{3} - \left(\frac{6}{2(3)}\right)^2 = 0 \quad M = 0 \text{ (like we said earlier)}$$
Complexity: Pros/Cons

- Modularity difference is constant time; $L$ checks at each iteration; $N$ updates to adjacency matrix to capture new community; $N-1$ iterations = $O((L+N)N)$ or $O(N^2)$ for sparse graphs
- Pros: Polynomial time is good; guarantees we are moving toward more realistic approximations at each stage
- Cons: Smaller communities are destroyed in the process, resolution limit exists, pure version has “empty hits” on sparse graph
- $O(N\log 2N)$ version: [http://cs.unm.edu/~aaron/research/fastmodularity.htm](http://cs.unm.edu/~aaron/research/fastmodularity.htm)
Resolution Limit

\[ \Delta M_{AB} = \frac{l_{AB}}{L} - \frac{k_A k_B}{2L^2} \]

-Advanced topic derivation for modularity change of two communities (A and B) merging

-If A and B are distinct communities, we want them to remain distinct after the merge
  -assume \( k_A \) and \( k_B \) are about equal to \( k \); what happens?
  -modularity change becomes positive

\( l_{AB} = \text{links from nodes in A to nodes in B} \)
Better Approach?

- Link Clustering: Approach that recognizes that links are usually distinct in the network.
- Nodes can be a part of many communities, but the links give a better semantic representation of the community structure.
- Based on the concept of similarity: how many friends do you have in common with someone else?
- In friendship networks, people can be labeled as part of many communities, link clustering helps to determine the best fit based on number of common neighbors.
Link Similarity

\[ S((i, k), (j, k)) = \frac{|n_+(i) \cap n_+(j)|}{|n_+(i) \cup n_+(j)|} \]

- **S:** similarity
- **S((i,k), (j,k)):** node \( i \) and node \( j \) have neighbor \( k \) in common

**Numerator:** numerators \( i \) and \( j \) have in common, including themselves

**Denominator:** everyone that \( i \) and \( j \) know
Hierarchical Clustering

- Assign each node to a community of its own and evaluate $x_{ij}$ for all node pairs.
- Find the community pair or the node pair with the highest similarity and merge them into a single community.
- Calculate the similarity between the new community and all other communities.
- Repeat Steps 2 and 3 until all nodes form a single community.

Single link Hierarchical Clustering: “nearest neighbor” clustering, combine closest nodes at each step
Basic Flow of the Algorithm

- Compute similarity matrix for all pairs in the network using link similarity for link clustering
- Apply single link hierarchical clustering (use similarity matrix instead of centrality or some other similarity formula)
- Merge communities with high similarity
- Repeat until the whole network is one community
Algorithm Results/Similarity Matrix
Complexity: Pros/Cons

- $O(N^{2/(\gamma - 1)}) + O(L^2)$ for scale free networks (gamma indicates level of attachment, overall complexity bounded by maximum degree).
- $O(N^2)$ for sparse graphs.
- Pros: Community structure is more accurately represented, not affected by resolution limit
- Cons: more memory to keep track of, similarity matrix must be factored into calculations, order of iterations matters
Greedy Algorithm applications

- Collaboration networks: (C.M. indicates condensed matter, H.E.P. high-energy physics, and astro astrophysics. These four large communities coexist with 600 smaller communities, resulting in an overall modularity $M=0.713$.)
- Social Network approximations
- Sub-community Detection
Link Clustering Application
References

- [http://barabasi.com/networksciencebook/chapter/9#modularity](http://barabasi.com/networksciencebook/chapter/9#modularity)
- [http://barabasi.com/networksciencebook/chapter/9#hierarchical](http://barabasi.com/networksciencebook/chapter/9#hierarchical)
- [http://barabasi.com/networksciencebook/chapter/9#basics](http://barabasi.com/networksciencebook/chapter/9#basics)
QUESTIONS?