Community Detection in Social Networks with Attribute and Relationship Data

The Anh Dang, Emmanuel Viennet*

*L2TI - Institut Galilée - Université Paris-Nord
99, avenue Jean-Baptiste Clément - 93430 Villetaneuse - France
{theanh.dang,emmanuel.viennet}@univ-paris13.fr

1 Community Detection Algorithms

We present two methods to discover communities in an attributed graph, given a similarity measure. An attributed graph is denoted as $G = (V, E, X)$, where V is the set of nodes, E is set of edges, $X = X^1, ..., X^d$ is the set of d attributes associated with the nodes in V. Each node v_i is associated with an attribute vector $(x_i^1, ..., x_i^d)$.

a)Algorithm SAC1

In this algorithm, we first define a composite modularity Q as an extension of Newman's well-known modularity function (Clauset et al. (2004))

$$Q = \sum_C \sum_{i,j \in C} \left(\alpha \cdot \frac{1}{2m} \cdot \left(G_{i,j} - \frac{d_i \cdot d_j}{2m} \right) + (1 - \alpha) \cdot simA(i, j) \right)$$

α is the weighting factor, $0 \leq \alpha \leq 1$, $simA(i, j)$ is the attribute similarity of the nodes (i, j). To find an approximate optimization of Q, we follow an approach directly inspired by the Louvain algorithm (Blondel et al. (2008)). The algorithm starts with each node belonging to a separated community. A node is then chosen randomly. The algorithm tries to move this node from its current community. If a positive gain is found, the node is then placed to the community with the maximum gain. Otherwise, it stays in its original community. This step is applied repeatedly until no more improvement is achieved. The first phase is completed when there is no more positive gain by moving of nodes. Next, we can reapply this phase by grouping the nodes in the same communities to a new community-node. To determine the attribute similarity between two communities, we propose two approaches. The first is to sum up the similarity of their members, the second way is to set to the similarity of their centroids.

b)Algorithm SAC2

Our first algorithm SAC1 repetitively checks all nodes, leading to $O(n^2)$ complexity. To reduce the computational cost, we propose another approach that only makes use of a node's nearest neighbors. Given an attributed graph : $G = (V, E, X)$, we define a k-nearest neighbor graph (k-NN) $G_k = (V, E_k)$ as a graph in which each node has exactly k edges, connecting to its k most similar neighbors in G. The similarity measure between 2 nodes i and j is defined as

$$S(i, j) = \alpha \cdot G_{i,j} + (1 - \alpha) \cdot simA(i, j)$$