Dynamics of Influence Networks Subject to Regular Shocks

Nan Ding, Class of 2019

Network models are useful for representing real-life situations such as social networks and adoption of certain devices or strategies. In this project we implemented a cascade model to represent a network of individual nodes influencing each other, with each node carrying random initial behavior and a threshold over which a change of behavior occurs. The specific cascade model in this project is the linear threshold model, in which each node i is randomly assigned a threshold between 0 and 1, and each edge coming into node i from its neighbors (nodes in $S(i)$) are randomly assigned a weight w_{ji} between 0 and 1, such that the sum of all incoming weights to node i is less than or equal to 1. At each iteration, each node will take in the behaviors x_j of all its neighbors, and the node i will only adopt behavior 1 if $\sum_{j \in S(i)} w_{ji} x_j \geq b_i$. In order to better represent real-life situations with this model, we applied a shock to the cascade model after it reached its first equilibrium (at which all nodes stop changing behavior) and then let it converge, and this process is carried on repeatedly. By doing this, we aim to study the resilience of the network against regular shocks, which in this case means the ability of the model to reach its equilibrium under shocks. Thanks to Son D. Ngo and Mingo Sanchez's previous work (both Class of 2017), we already had the skeleton of a computer program that simulated a cascade model with regular shocks applied to it. During the project, we looked at four topologies of network models: star, random, barabasi-albert, watts-strogatz models.

To simulate real-life behaviors, the algorithm first randomly picks a shock value, and then use a probability distribution related to the shock value to apply different actual effects on each node (i.e., updating the threshold of each node) while keeping the new thresholds in the range. A large part of the work we did this semester focused on finding the appropriate and computationally simple probability distribution for the shock value and the actual shock effect on each node. To achieve this goal, we tested on normal, uniform, and custom distributions from which shock effects on the nodes are chosen, and we abandoned the use of normal and uniform distribution for updating the thresholds. Instead, we decided to choose two shock values $s_1; s_2$ from a uniform distribution from 0 to 1 and then define an intermediate value shock effect x_i