Attributed Graphs:
Market Segmentation and Influence Propagation

Nagiza F. Samatova
Professor, Department of Computer Science
North Carolina State University
and
Senior Scientist, Computer Science and Mathematics Division
Oak Ridge National Laboratory

Outline

- ***Market Segmentation***
- ***Graphs***
    - Graph Terminology
    - Types of Graphs
- ***C0mmunity Detection***
    - Goodness metrics
    - Objective function
    - Algorithms
    - Community Detection for Attributed Graphs
- ***Influence Propagation***
    - Virus Propagation Models
    - Effective Strength of a Virus
    - Immunization Policies

Market Segmentation

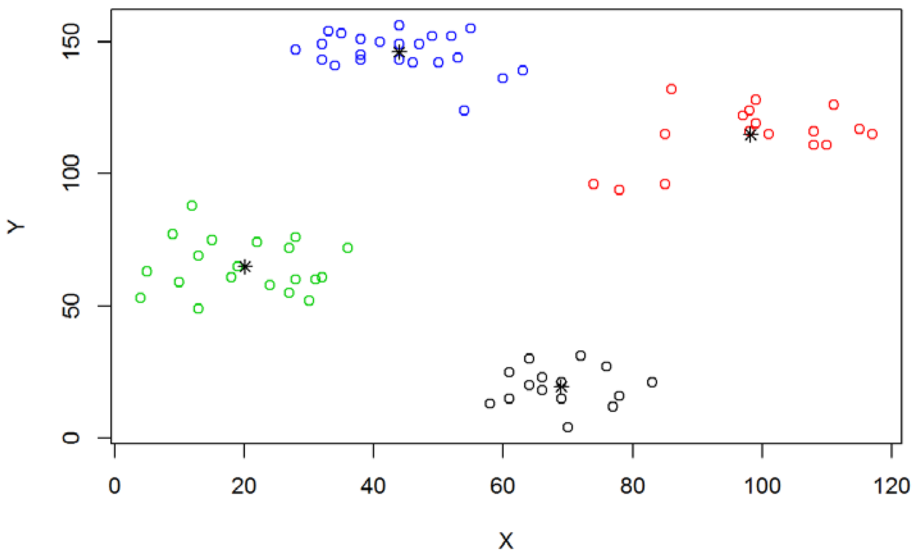
- ***Market segmentation*** is the task of dividing a market into ***groups of customers*** with homogenous needs, such that marketing firms can target groups and allocate resources efficiently, as customers in the same segment are likely to respond similarly to a given marketing strategy [1].

- Traditional ***market segmentation*** methods are based on ***clustering*** attribute data, such as demographics (e.g., age, gender, ethnicity) and psychographic (e.g., lifestyle, personality) profiles, using traditional ***clustering*** algorithms (e.g., ***k-means***).

- Nowadays, ***social networks*** have become important in marketing, as social relationships can also impact the formation of market segments.

k-Means Clustering

- ***k-means clustering*** algorithm:
  1. Randomly select ***k*** initial ***centroids***.
     - ***k*** is a user-specified parameter (number of clusters).
  2. Assign each point to the cluster with the closest ***centroid***.
  3. Update the ***centroid*** of each cluster based on the points assigned to the cluster.
  4. Repeat (2) and (3) until no point changes clusters.



Influence in Social Networks

- *Why is social network information important for market segmentation?*
  - A marketing firm can use this information to design marketing campaigns that target ***influential*** users in the network.
  - ***Influence*** in social networks refers to the phenomenon where the actions of a user induces his/her friends to behave in a similar way [2].
  - For example, a user buys a product because one of his/her friends bought the same product.

- Market segmentation in social networks can be formulated as a problem of ***community detection*** in ***graphs***.
  - ***Communities*** represent market segments.

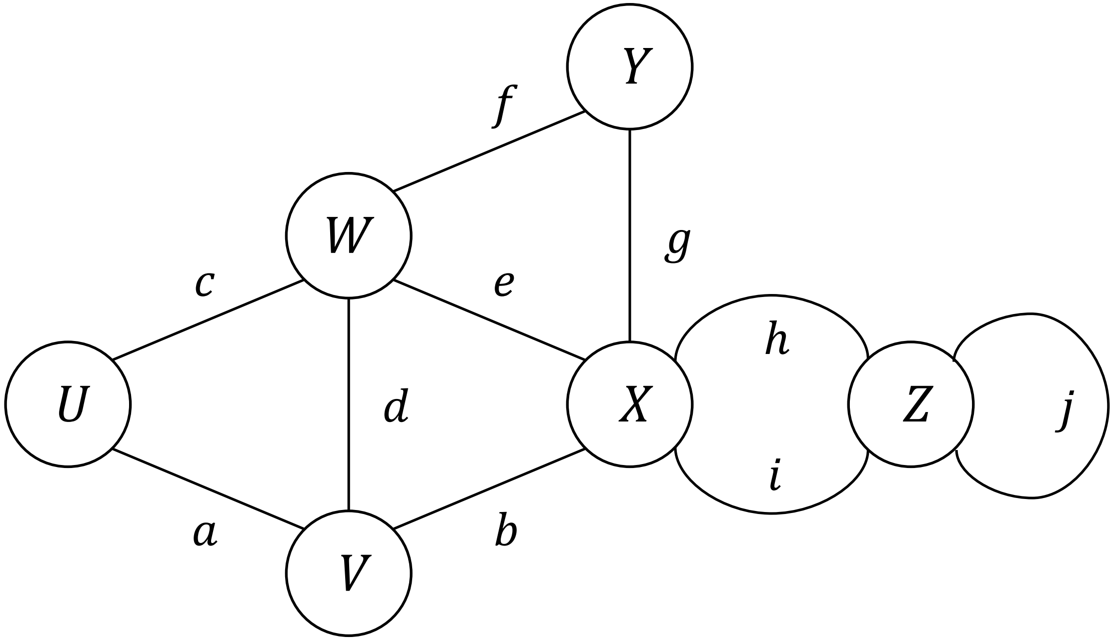
Graphs

- A ***graph*** is a representation of a set of objects and the relationships between them.
- We denote a ***graph*** as $G = (V, E)$, where
  - V is a set of ***vertices*** (i.e., objects).
  - E is a set of ***edges*** (i.e., relationships between objects).

- Social networks can be represented as ***graphs***.
  - ***Vertices*** represent users.
  - ***Edges*** represent relationships (e.g., “friendship”) between a pair of users.

Graph Terminology

- Vertices U and V are the ***endpoints*** of edge a.
- Edges a, b, and d are **incident** on vertex V.
- Vertices U and V are **adjacent**.
- The ***degree*** of vertex X (i.e., ***number of edges incident on vertex*** X) is 5.
- Edges h and i are ***parallel edges***.
- Edge j is a ***self-loop***.



Types of Graphs

- ***Directed*** vs. ***Undirected***
  - ***Directed*** graphs are those where edges have orientations.

- ***Weighted*** vs. ***Unweighted***
  - ***Weighted*** graphs are those where a value (***weight***) is assigned to each edge.

- ***Attributed*** vs. ***Unattributed***
  - ***Attributed*** graphs are those where vertices (or edges) contain additional information (***attributes***).
  - We denote an ***attributed*** graph as $G = (V, E, X)$, where
    - $X = X^1, \ldots, X^d$ is a set of d ***attributes*** associated with the vertices in V.
  - Vertex ***attributes*** in social networks may include name, age, gender, occupation, etc.

Community Detection

- A ***community*** is a set of vertices in a graph that are densely connected within each other and sparsely connected with the rest of the graph.
  - ***Communities*** in social network represent ***social groups***.

- ***Community detection*** is the problem of partitioning a given graph into communities. Solving this problem involves:
  - Defining an ***objective function*** to partition the graph into communities.
    - ***Modularity***.
  - Defining ***"goodness" metrics*** to evaluate the quality of the communities.
    - ***Density***.
    - ***Conductance***.
    - ***Clustering coefficient***.

Goodness Metrics for Community Detection (I)

- ***Goodness metrics*** quantitatively measure different attributes of community structures [3].
  - ***Density***: the ratio of edges to the number of possible edges, given by

$$\frac{2E_s}{|S|(|S|-1)}$$

  - ***Conductance***: the fraction of edges that point outside the community, given by

$$\frac{O_s}{2E_s + O_s}$$

where S is a community (i.e., set of vertices), E_s is the number of edges between vertices in S, and O_s is the number of edges between vertices in S and any vertex outside of S.

Goodness Metrics for Community Detection (II)

- ***Goodness metrics*** quantitatively measure different attributes of community structures [3].
  - ***Clustering coefficient***: the ratio of closed triplets to all triplets, given by

$$\frac{|T_c|}{|T_c| + |T_o|}$$

    where T_c is the set of closed triplets, and T_o is the set of open triplets.
    - A ***triplet*** is defined a tuple of three vertices (u, v, w) where (u, v), $(v, w) \in E$. If $(u, w) \in E$, then the triplet is set to be ***closed***, otherwise the triplet is ***open***.

Objective Function for Community Detection

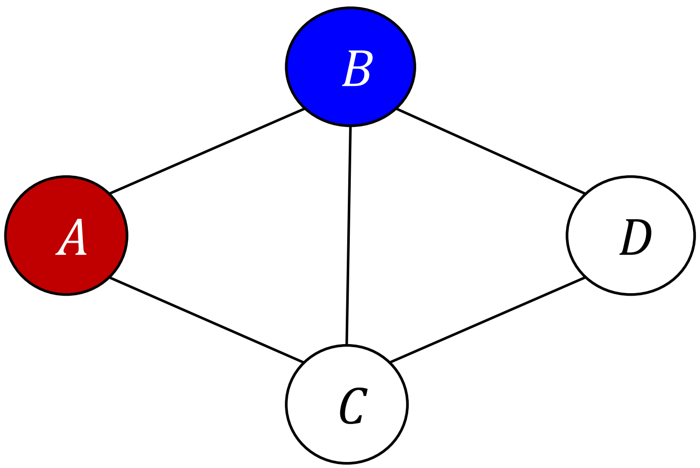
- The most widely used ***objective function*** for community detection is ***modularity*** [4], which is defined as the difference between the number of edges within the communities and the expected number of these edges in a random graph with the same degree distribution. For a simple graph G, the ***modularity*** is given by

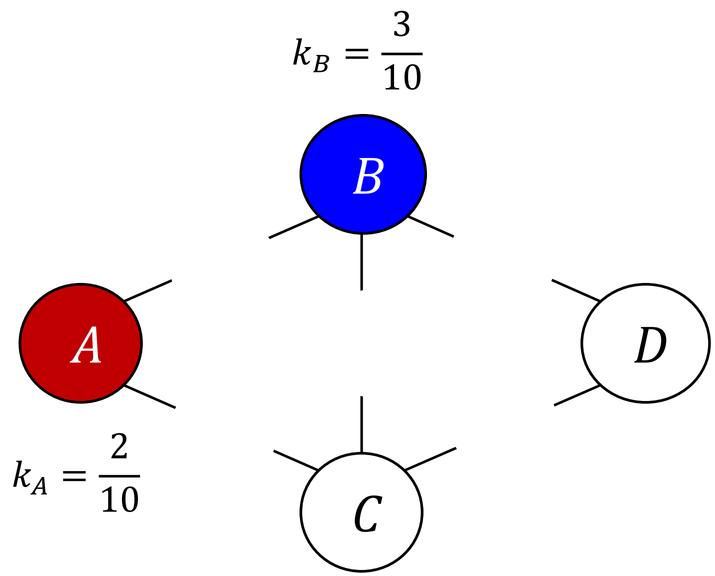
$$Q = \frac{1}{2m} \sum_{v,w \in V} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(v,w)$$

  where A is the adjacency matrix of the graph, m is the number of edges, k_v is the degree of vertex v and $\delta(i,j)$ is 1 if i and j belong to the same community and 0 otherwise.

- Optimizing modularity is an ***NP-complete problem***, but greedy algorithms for this problem have been proposed (e.g., the ***Louvain method*** [5]).

Understanding Modularity (I)





Number of edges $= m = 5$

Number of ***stubs*** $= 2m = 10$

Probability of an edge between A and B $\approx \left(\frac{2}{10}\right)\left(\frac{3}{10}\right) + \left(\frac{3}{10}\right)\left(\frac{2}{10}\right) = 2\left(\frac{2}{10}\right)\left(\frac{3}{10}\right)$

Expected number of edges between A and B $\approx 2\left(\frac{2}{10}\right)\left(\frac{3}{10}\right) \cdot 5 = \frac{k_A k_B}{2m}$

Understanding Modularity (II)

Number of edges between v and w

Expected number of edges between v and w

$$Q = \frac{1}{2m} \sum_{v,w \in V} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(v,w)$$

Normalize by dividing by number of ***stubs***

Consider only pairs of nodes in the ***same*** community

Louvain Method [5] (I)

Input: graph $G = (V, E)$
Output: set of communities C

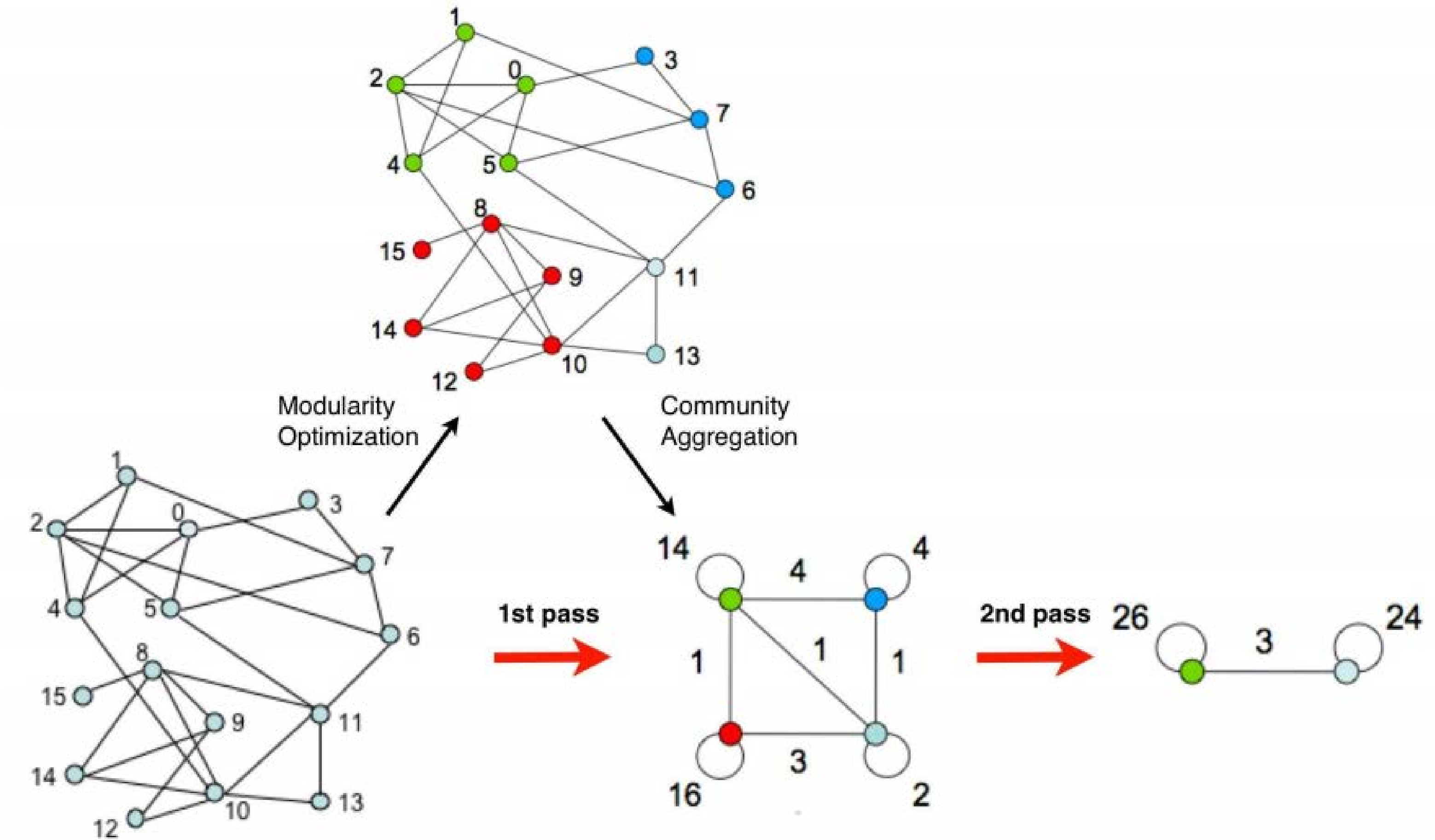
1. Initialize a community for each vertex $v \in V$.

2. For each vertex $v \in V$, assign v to the community that yields the ***highest positive gain in modularity***.
   - Repeat (2) until no further improvement can be achieved.
   - Obtain a set of communities

3. Construct a new graph by aggregating the vertices in each community into a single meta-vertex.

4. Repeat (2) and (3) until no further improvement can be achieved.

Louvain Method [5] (II)



Source: Blondel, Guillaume, Lambiotte, and Lefebvre, 2008

Community Detection in Attributed Graphs

- Traditional community detection methods take into account only the ***structural*** information of the graph. However, many real-world networks, such as social networks, are ***attributed***. Considering both the ***structural*** and ***attribute*** information of the graph may allow us to detect more meaningful communities.

- We redefine the ***community detection*** problem for ***attributed*** graphs. Solving this problem involves:
  - Defining an ***objective function*** to partition the ***attributed*** graph into communities.
    - ***Composite modularity***.
  - Defining ***"goodness" metrics*** to evaluate the quality of the ***attributed*** communities.
    - ***Similarity***.

Goodness Metrics for Community Detection in Attributed Graphs

- ***Goodness metrics*** for attributed graphs need to consider the degree of closeness of the vertices in terms of their attributes. Vertices in the same community are expected to have similar attributes.
    - ***Similarity***:
        - For binary attributes, use **simple matching coefficient** (i.e., ratio of matching attributes to all attributes).
        - For continuous attributes, use similarity metric based on the **Euclidean distance**, given by

$$sim(v,w) = \frac{1}{1 + \sqrt{\sum_d \left(x_v^d - x_w^d\right)^2}}$$

where x_v^d is the value of attribute x^d for vertex v.

Objective Function for Community Detection in Attributed Graphs

- Traditional ***modularity*** does not take into account the attribute similarity between vertices. Thus, we use as ***objective function*** for community detection in attributed graphs the ***composite modularity*** [6], a weighted combination of modularity and similarity given by

$$Q = \sum_{C} \sum_{v,w \in C} \left(\alpha \cdot \left(\frac{1}{2m} \cdot \left(A_{vw} - \frac{k_v k_w}{2m} \right) \right) + (1 - \alpha) \cdot sim(v, w) \right)$$

  where α is the weighting factor, $0 \leq \alpha \leq 1$.

- Optimizing composite modularity is an ***NP-complete problem***, but greedy algorithms for this problem have been proposed (e.g., the ***Structure-Attribute Clustering SAC1 algorithm*** [6] based on the Louvain method).

$SAC1$ Algorithm [6]

Input: attributed graph $G = (V, E, X)$
Output: set of communities C

1. Initialize a community for each vertex $v \in V$.

2. For each vertex $v \in V$, assign v to the community that yields the ***highest positive gain in composite modularity***.
   - Repeat (2) until no further improvement can be achieved.
   - Obtain a set of communities

3. Construct a new graph by aggregating the vertices in each community into a single meta-vertex.

4. Repeat (2) and (3) until no further improvement can be achieved.

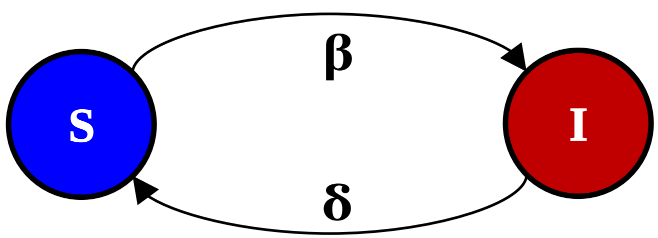
Influence Propagation in Graphs

- Modelling the market segmentation problem using graphs also allows us to analyze the ***propagation*** of a marketing campaign across the social network.
    - *Will the marketing campaign spread across the network?*
    - *Which influential users should we target in order to maximize the spread of the marketing campaign?*

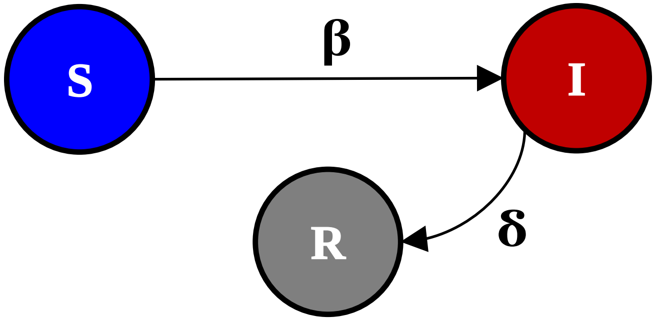
- These questions may be answered by introducing fundamental concepts from epidemiology.
    - A ***virus propagation model*** (VPM) is a simplified model of disease spread that provides general information about the behavior of a disease.
        - *How virulent is the disease?*
        - *How quickly does the host recover (if ever)?*
        - *Does the host obtain immunity?*

Virus Propagation Models (I)

- Example of ***VPMs***:
    - ***SIS VPM*** ("*susceptible, infected, susceptible*").
        - Two states: Susceptible (S), Infected (I).
        - Transition probabilities: transmission probability β, healing probability δ.
    - ***SIR VPM*** ("*susceptible, infected, recovered*").
        - Three states: Susceptible (S), Infected (I), Recovered (R).
        - Transition probabilities: transmission probability β, healing probability δ.



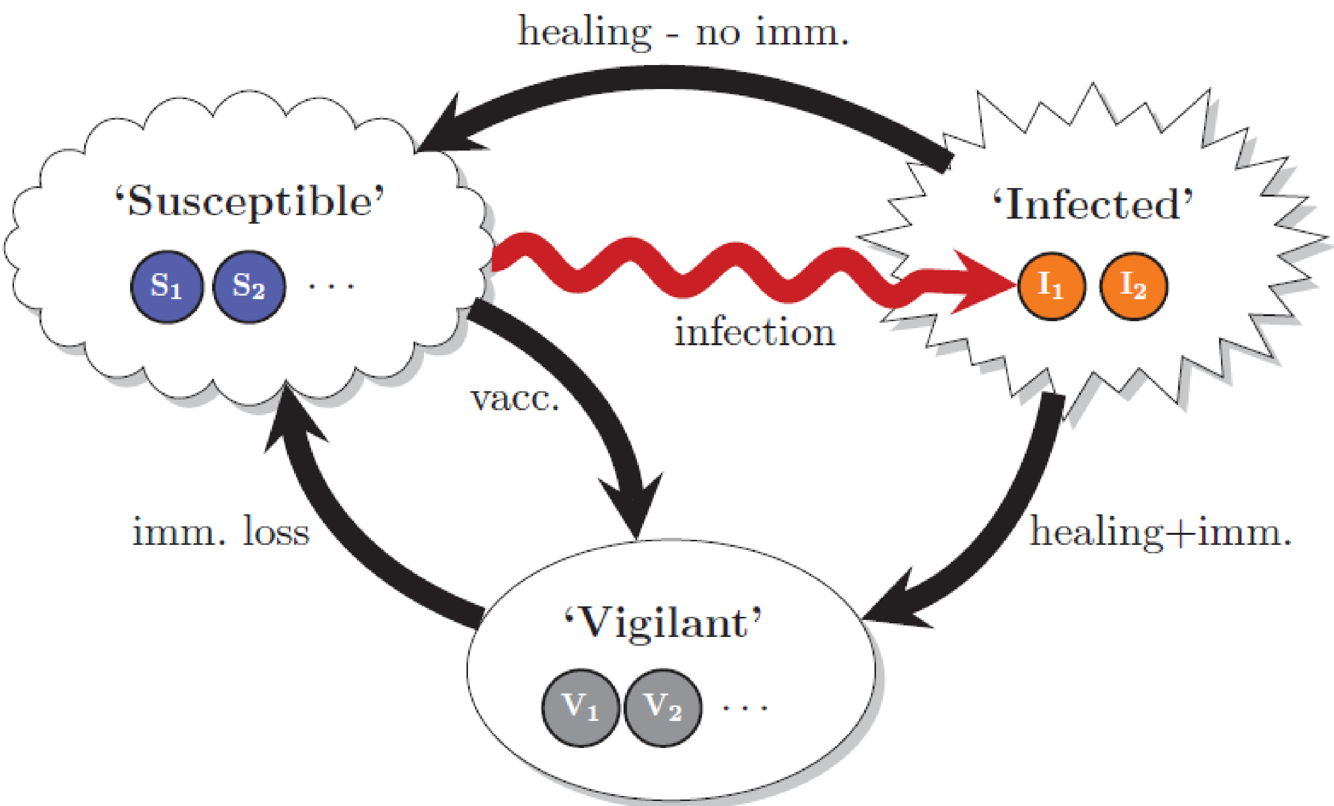
SIS **VPM**



SIR **VPM**

Virus Propagation Models (II)

- All existing ***VPMs*** can be generalized to the $S^{*}I^{2}V^{*}$ ***VPM*** [7].



Source: Chakrabarti and Faloutsos, 2012

Effective Strength of a Virus

- For any VPM that follows the $S^*I^2V^*$ model and for any arbitrary network with adjacency matrix A, the ***effective strength*** [8] of a virus is given by

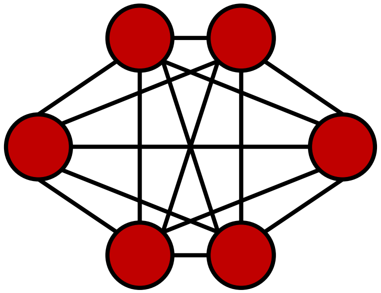
$$s = \lambda_1 \cdot C_{VPM}$$

  where λ_1 is the largest eigenvalue (or ***spectral radius***) of A, which measures the connectivity of the network, and C_{VPM} is a constant that depends on the VPM. For the SIS and SIR VPMs, $C_{VPM} = \beta/\delta$.

- The ***epidemic threshold*** that captures the transition in the behavior of the system is reached when $s = 1$.
  - Above the threshold, the virus can spread across the network and result in a network-wide epidemic.
  - Below the threshold, the virus can't spread across the network.

Spectral Radius

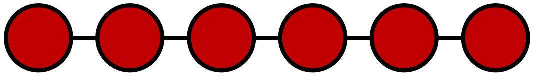
- A better connected network facilitates the spread of the virus.
- How to measure the ***connectivity*** of the network?



Clique

Avg. degree = 5

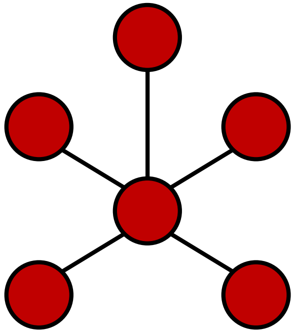
$\lambda_1 = n - 1 = 5$



Chain

Avg. degree = 1.67

$\lambda_1 = 1.80$



Star

Avg. degree = 1.67

$\lambda_1 = \sqrt{n-1} = \sqrt{5} = 2.24$

- The ***spectral radius*** (i.e., ***largest eigenvalue***) of the adjacency matrix of the network (λ_1) measures the ***connectivity*** better than the average degree because it takes into account paths of all lengths.

Immunization Policies

- Given a network and a number of available vaccines k, an ***immunization policy*** determines which are the k best vertices to immunize in order to prevent an epidemic.
    - In marketing, we want to determine the k best vertices to target in order to maximize the spread of the campaign.

- The optimal policy is to find the subset of k nodes with the largest drop in λ_1. This policy is computationally intractable. Instead, the drop in λ_1 caused by a set of nodes S can be approximated by calculating the ***Shield-value score*** [9] given by

$$Sv(S) = \sum_{i \in S} 2\lambda_1 u_1(i)^2 - \sum_{i,j, \in S} A(i,j) u_1(i) u_1(j)$$

where A is the adjacency matrix of the network, λ_1 is the largest eigenvalue of A and u_1 is the corresponding eigenvector.

NetShield Algorithm [9]

Input: graph $G = (V, E)$
Output: k best nodes to immunize/target

1. Compute the largest eigenvalue λ_1 and the corresponding eigenvector u_1 of the network's adjacency matrix A.

2. For each node i in the contact network, calculate the ***Shield-value score*** $Sv(i)$.

3. Initialize an empty subset S.

4. For each node i in the network, compute:

$$score(i) = Sv(i) - 2 \cdot A(:, S) \cdot u_1\,(S) \cdot u_1\,(i)$$

5. Add the node i with the maximum $score(i)$ to the subset S.

6. Repeat (4) and (5) until S contains k nodes.

References

1. R. Ge, M. Ester, B. J. Gao, Z. Hu, B. Bhattacharya, B. Ben-Moshe. *Joint cluster analysis of attribute data and relationship data: The connected k-center problem, algorithms and applications*. TKDD, 2008.
2. A. Anagnostopoulous, R. Kumar, and M. Mahdian. *Influence and Correlation in Social Networks*. KDD, 2008.
3. S. Harenberg, G. Bello, L. Gjeltema, *et al*. *Community detection in large-scale networks: a survey and empirical evaluation*. WIREs Computational Statistics, 2014.
4. M. E. J. Newman. *Analysis of weighted networks*. Physical Review E, 2004.
5. V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre. *Fast unfolding of communities in large networks*. Journal of Statistical Mechanics: Theory and Experiment, 2008.
6. T. A. Dang, and E. Viennet. *Community detection based on structural and attribute similarities*. ICDS 2012.
7. D. Chakrabarti, C. Faloutsos. *Graph Mining: Laws, Tools, and Case Studies*. Synthesis Lectures on Data Mining and Knowledge Discovery, 2012.
8. B. A. Prakash, D. Chakrabarti, M. Faloutsos, N. Valler, C. Faloutsos. *Got the Flu (or Mumps)? Check the Eigenvalue!* arXiv, 2010.
9. H. Tong, B. A. Prakash, C. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, D. H. Chau. *On the Vulnerability of Large Graphs*. ICDM, 2010.