# COMMUNITY DETECTION IN SOCIAL NETWORKS

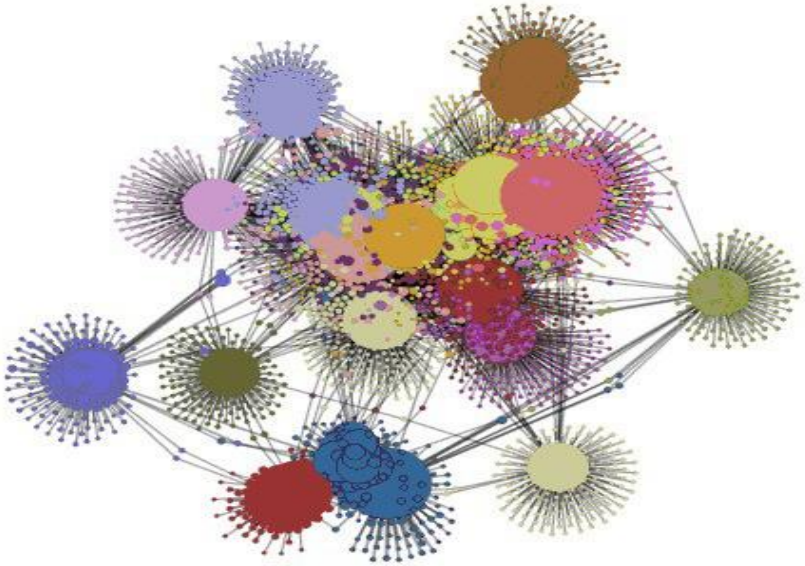
Chovardas Doxakis, Liapikos Theodoros, Serderidis Konstantinos, Vasdekis Dimitrios

# INTRODUCTION

- Most networks display a community structure where vertices are organized into groups called communities.
- Defining these communities is crucial in understanding the features and properties of a network
- In this presentation a review of eight algorithms where each of them proposes a novel approach in community detection is attempted

✓ **ABCD** ✓ **AMKM** ✓ **HICODE** ✓ **ASOCCA** ✓ **EADP** ✓ **DSCAN** ✓ **DCK** ✓ **CC - GA**

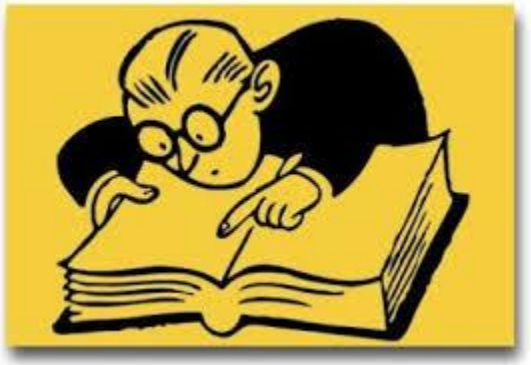
# **A**TTRACTIVENESS-**B**ASED **C**OMMUNITY **D**ETECTION (**ABCD**)



# ABCD

- An agglomerative algorithm which introduces the weighted concept
- Aimed for large-scale networks, to represent the users' relationships on social networks on Internet

# DEFINITIONS



Given a Graph G (V, E, $W_V$, $S_E$) , V node set , E edge set, $W_V$ the weight of the node set, $S_E$ the weight of edge set, communities are identified as the Graph G clusters.

Given that the No of edges between clusters i and j is **q**, the edges weight **Se**, e ∈ 1, 2, …, q, communities i, j have **Qi**, $\mathbf{Q_j}$ nodes, the node weight is $\mathbf{W_a}$, a set of definitions is constructed:

- **Cluster density:** $W_i = \frac{\sum_{a=1}^{Qi} Wa}{Qi}$
- **Attractiveness** between clusters i and j: $S_{ij} = \frac{\sum_{e=1}^{q} Se}{QixQj}$ which feeds the attractiveness matrix
- **Inter-interested** clusters: Clusters i, j are inter-interested, if $q \geq Q_i$, $q \geq Q_j$
- **Community**: A cluster i is classified as community if $S_{ij} < W_i + W_j \ _{\forall} j$

# ABCD METHODOLOGY

Start with each node as a single cluster

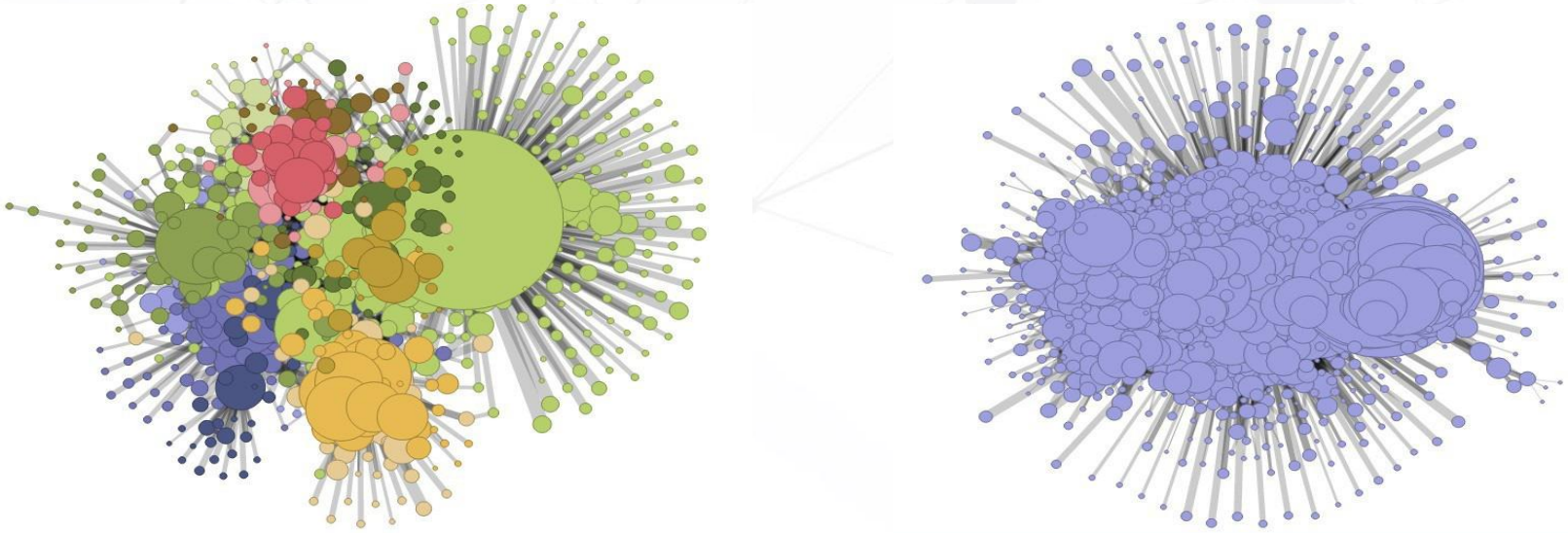
- Two main steps
  1. Merge the pair of clusters with the highest value of attractiveness. The clusters are merged if the following condition applies: $Sij \geq \mathrm{Wi} + \mathrm{Wj}$
  2. Calculate/update the cluster density and cluster attractiveness matrix and then iterate between the two steps ……. ……. up to the point where the cluster structure remains unchangeable

- Prerequisites:
  - inter-interested clusters do exist
  - candidate clusters have weight higher than the attractiveness with other inter-interested clusters
  - there is no preference among possible inter-interested clusters.

# EVALUATION PROCESS

- **Metrics**
  - ✓ Number of Identified communities for three given datasets:

    Sina Weibo, a Chinese micro-blogging site, College Football dataset, Renren: a Chinese social network service (like Facebook).



Communities identified

ABCD vs CNM - Sina Weibo micro-blog

source: Ruifang Liu et al. / Procedia Computer Science 31 ( 2014 ) 85 – 94

# CRITIQUE

**Pros**

√ Adopts a simple methodological approach

√ Easy to assess based on a single metric

√ Time Complexity addressed, indicated performance $O(nk + tm^2)$

(n: nodes No, k: average No of inter-interested clusters, m: average No of clusters at the beginning of iterations, t: max No of iterations)

√ No requirement to specify the number of clusters in advance

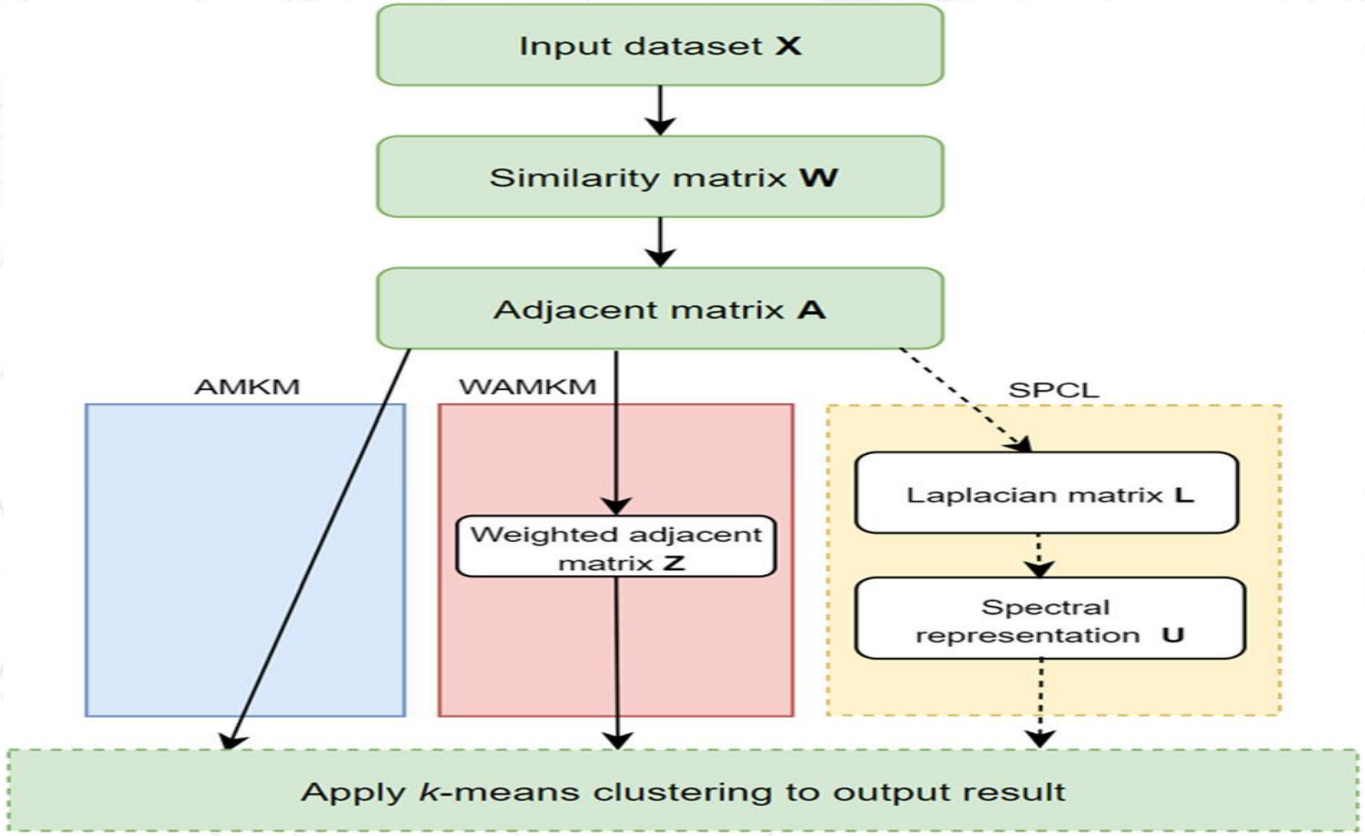
**Cons**

× Assessment only vs a single algorithm, via a single metric -> not adequate

× Experimental tests are simplified, not all attributes used (e.g. in Sina Weibo)

× User profile data is not exploited (future work)

# **W**EIGHTED **A**DJACENT **M**ATRIX FOR **K-M**EANS CLUSTERING (**AMKM-WAMKM**)

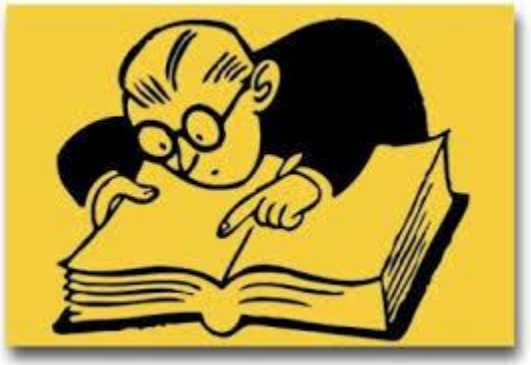



source: Jukai Zhou, Tong Liu, Jingting Zhu, 2019. Weighted adjacent matrix for K-means clustering. # Springer Science + Business Media, LLC, part of Springer Nature 2019, https://doi.org/10.1007/s11042-019-08009-x

# AMKM-WAMKM

- An algorithm building up on the classic K-means clustering
- Aimed to improve the K-means in terms of complexity on large datasets and to address the limitation of the similarity measurements

# DEFINITIONS



- G= (V, E): V = $\{v_1, \ldots, v_n\}$ E = $\{e_1, \ldots, e_m\}$ for a fully connected graph, ($m = n \times (n-1)/2$)
- Input data points X = $\{x_1, \ldots, x_n\}$
- Similarity matrix $W = (wi,_j)_{i,j=1}^{n}$ where $w_{i,j} \geq 0$ is the similarity between $x_i$ and $x_j$

e.g. for the Euclidean distance for feature t :

$$w_{i,j} = \sqrt{\sum_{t=1}^{d} \left(x_{i,t} - x_{j,t}\right)^2} \quad (i, j \in [1, n], t \in [1, d])$$

# METHODOLOGY

- The same approach for both AMKM and WAMKM

  *but in WAMKM the features are weighted as % of all features*

  - Construction of the Similarity matrix W
  - Construction of the adjacency matrix A based on W by determining the distance between two data points via a similarity function, i.e. a kernel function, Polynomial, Gaussian or Sigmoid (introducing a parameter σ). e.g. for Gaussian the corresponding element $\alpha_{i,j}$ is defined by:

$$a_{i,j} = e^{-\left(\frac{\|w_i - w_j\|_2^2}{2*\sigma^2}\right)} \qquad (i, j \in [1, n])$$

  - Run the K-means on the adjacency matrix A

# EVALUATION PROCESS

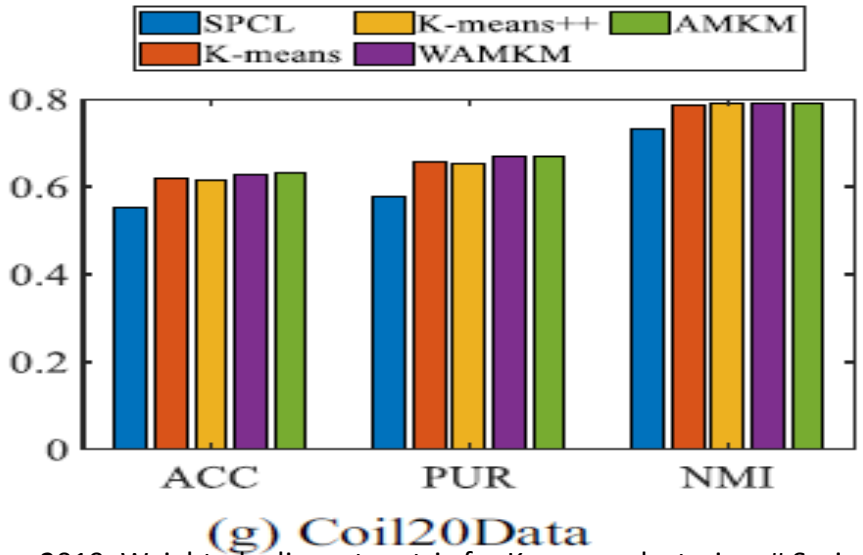
## Metrics

Accuracy ACC, NMI, PUR*

against three n clustering algorithms

(k-means, k-means++, and normalised spectral clustering algorithms)

Indicative Results for a given dataset:




(g) Coil20Data

*12 datasets (UCI Machine Learning Repository and the data mining centre website)

source: Jukai Zhou, Tong Liu, Jingting Zhu, 2019. Weighted adjacent matrix for K-means clustering. # Springer Science + Business Media, LLC, part of Springer Nature 2019, https://doi.org/10.1007/s11042-019-08009-x

# CRITIQUE

**Pros**

√ Based on a widely applied algorithm

√ Good Performance with multiple metrics and datasets

√ Appropriate for multiple features

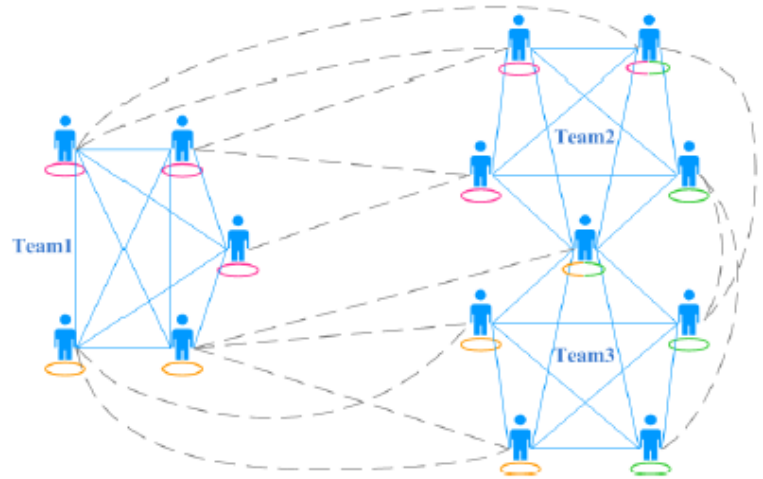
**Cons**

× Time Complexity improvement is not convincingly addressed

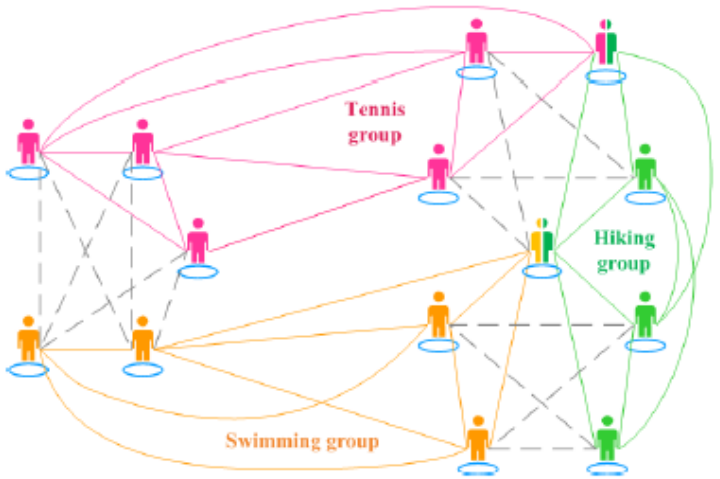
× Results are data-driven, the impact of parameter σ is not clear

× Not clear if it is suitable for sizeable networks, no concrete experimental data

# HIDDEN COMMUNITY DETECTION (HICODE)




(a) Dominant communities




(b) Hidden communities

# WHAT IS HICODE?

- An approach to identify hidden community structure as well as the dominant one.
- By weakening the strength of dominant structure one can uncover the hidden structure beneath
- By reducing the strength of the hidden structure one can more accurately identify the dominant structure

# NECESSARY DEFINITIONS

- **Hiddenness Value of a Community:** The fraction of nodes of a community $C_k$ belonging to another community with higher community quality score.
- **Layer:** A set of communities that partitions the nodes of a network
- **Hiddenness Value of a Layer:** The weighted average hiddenness values of the communities in the layer.

# HOW DOES HICODE WORK?

- **Identification**
  - ✓ Identify an initial layer using a base algorithm
  - ✓ Weaken the structure of previous (stronger) detected layers
  - ✓ Repeat until the appropriate number of layers are found

- **Refinement** (in each iteration we consider each layer)
  - ✓ Weaken the structure of all other layers from the original network
  - ✓ Apply the base algorithm to the resulting network

# WEAKENING? HOW?

- **Remove Edge:** Weakens a detected community by removing all intra-community edges
- **Reduce Edge:** Each community is a block; Randomly remove edges within the community to match the background edge probability
- **Reduce Weight:** Like Reduce Edge but reduces the weight of each edge within community to match average weighted background probability

# EVALUATION PROCESS

- **Metrics**
  - ✓ Modularity
  - ✓ NMI
  - ✓ Jaccard-based Precision, Recall, $F_1$ Score
- **Datasets**
  - ✓ 11 real-world networks
  - ✓ 3 of them with ground-truth communities
  - ✓ Small – medium- large scale
- Performance strongly related to the base algorithm used. HICODE improved the precision of the base algorithms tested

# CRITIQUE

- HICODE is an algorithm for hidden community identification with possible use in various fields such as security, counter-terrorism services and more.
- Uses a base algorithm so it’s performance is strongly depended by this choice.
- Not only detects hidden communities but also leads to higher quality community structures than the base algorithm without HICODE.
- The authors experiment with networks of various types and sizes but do not test directed or weighted  graphs.
- No serious assumptions are made.

# ADJACENT NODE SIMILARITY OPTIMIZATION COMBINATION CONNECTIVITY ALGORITHM (ASOCCA)

(a) Highest similarity neighbor node list

| node | 0 | 0 | 1 | 1 | 2 | 2 | 3 | 4 | 4 | 5 | 6 | 7 | 8 | 8 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Highest similarity neighbor | 4 | 8 | 4 | 8 | 9 | 10 | 6 | 1 | 0 | 8 | 5 | 6 | 1 | 0 | 5 | 8 | 2 |

(b) Combination 1

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 4 | 10 | 6 | 1 | 8 | 5 | 6 | 5 | 8 | 2 |

(c) Combination 2

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 4 | 10 | 6 | 1 | 8 | 5 | 6 | 5 | 8 | 2 |

(d) Combination 3

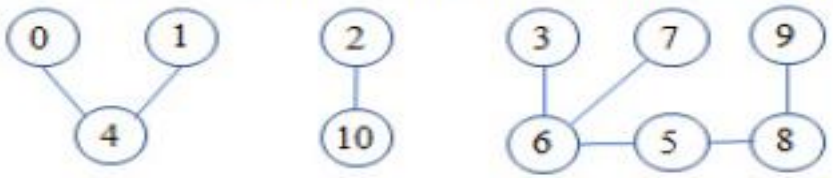
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 8 | 10 | 6 | 1 | 8 | 5 | 6 | 5 | 8 | 2 |

(e) Combination 4

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 8 | 10 | 6 | 1 | 8 | 5 | 6 | 5 | 8 | 2 |

(f) .....

(g) a fixed set of connected components generated by combination 1



# WHAT IS ASOCCA?

- An algorithm for accurate community detection utilizing a local similarity measure based on clustering coefficient.
- Focuses on non-overlapping community detection for undirected and unweighted graphs.

# NECESSARY DEFINITIONS

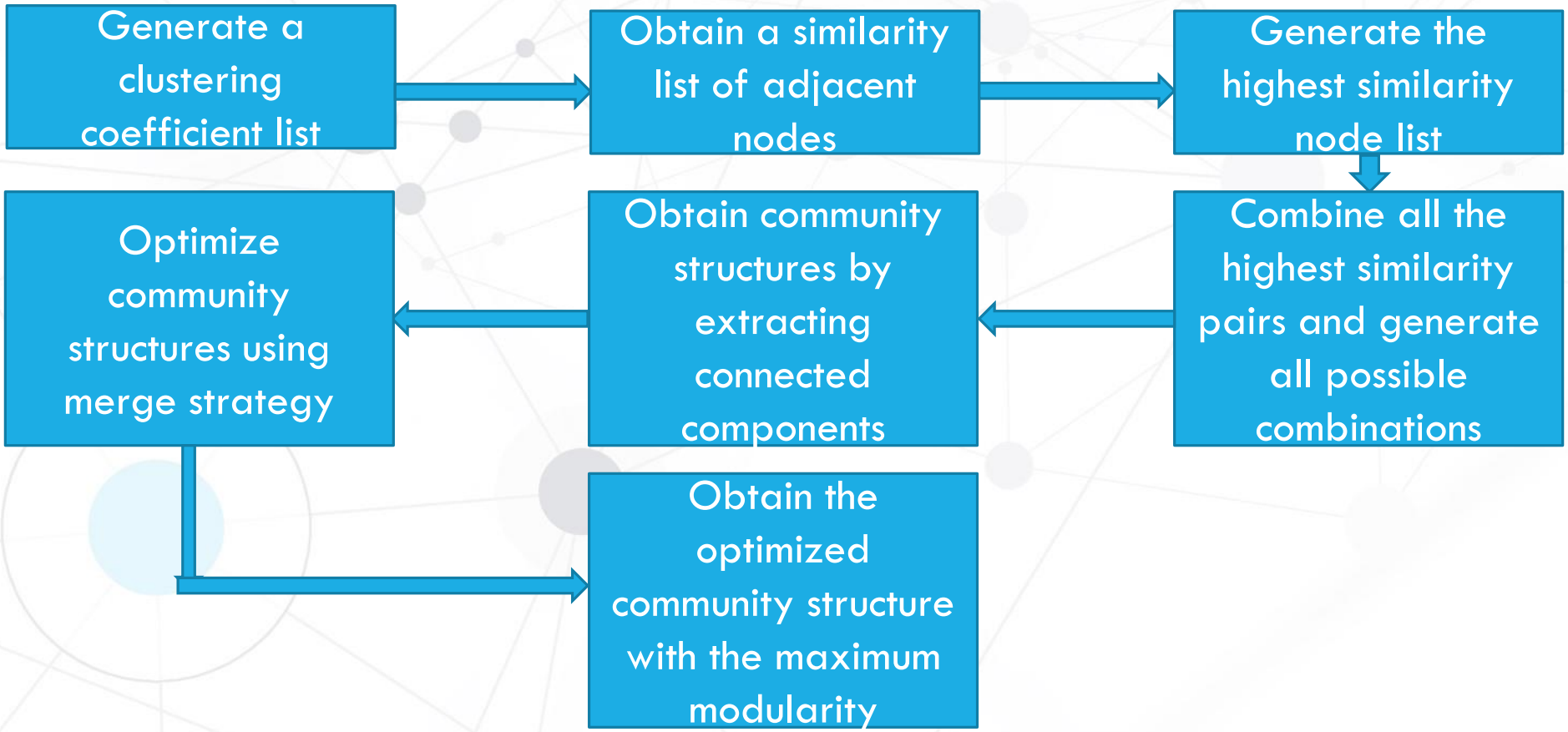
The authors propose a new similarity index based on local similarity, defined as follows:

$$Sim_{ij} = \sum_{t \in \Gamma_{(i)} \cap \Gamma_{(j)}} CC_t,$$

Instead of any of the common used similarity indexes:

| similarity index | Formula |
|---|---|
| Salton index | $\frac{\lvert\Gamma(i)\cap\Gamma(j)\rvert}{\sqrt{k_i \cdot k_j}}$ |
| Jaccard index | $\frac{\lvert\Gamma(i)\cap\Gamma(j)\rvert}{\lvert\Gamma(i)\cup\Gamma(j)\rvert}$ |
| Sørensen Index | $\frac{2\lvert\Gamma(i)\cap\Gamma(j)\rvert}{k_i+k_j}$ |
| Resource Allocation (RA) index | $\sum_{t\in\Gamma(i)\cap\Gamma(j)} \frac{1}{k_t}$ |

# ASOCCA's WORKFLOW



# EVALUATION PROCESS

- **Metrics**
  - ✓ Modularity
  - ✓ NMI

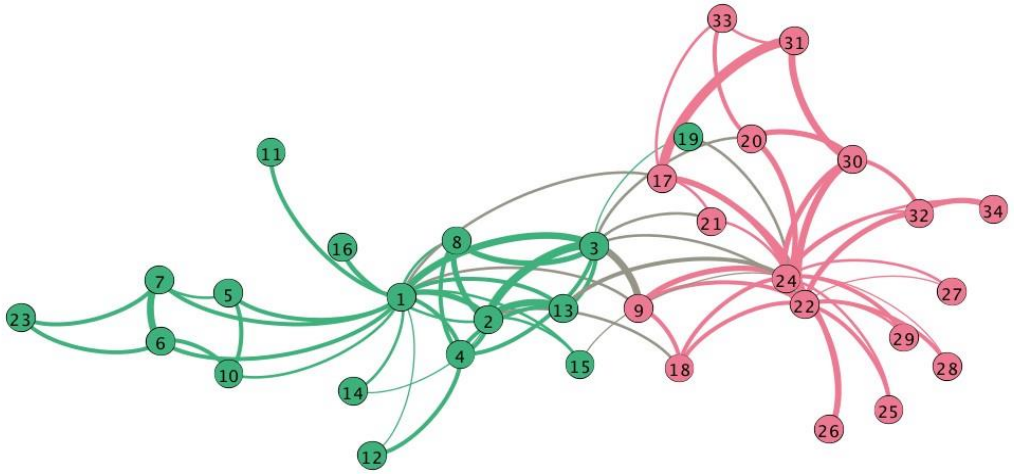
- **Datasets**
  - ✓ 6 real-world and 2 synthetic networks
  - ✓ 3 of them with ground-truth communities
  - ✓ Small – medium and large scale

# CRITIQUE

- The metrics used for performance evaluation are modularity and NMI. Neither of them alone is considered enough for strongly documented results.
- A new Similarity Index is defined to measure similarity.
- The datasets used are considered adequate regarding their types and sizes, although there is no implementation or testing for weighted, directed or overlapping community networks.
- An important assumption is made by limiting the maximum combination number to 100.
- Another crucial assumption is the minimum number of nodes in a community (λ) that has to be found by trial.
- The whole merging strategy is based on experimental results and is not strongly documented.
- The complexity of the algorithm is O(nm) a complexity acceptable even for large scale networks.

# EXTENDED ADAPTIVE DENSITY PEAKS (EADP)

- An approach to identify overlapping communities in social networks
- Is based in Density Peaks Clustering (DPC) and extends it



# HOW DOES EADP WORK?

- In the calculation of the **Distance** between nodes takes into consideration the weights of the edges and the common nodes

- Automatically finds the **Cluster Centers** by calculating the **Local Density** and the **Separation Distance** of each node

- Allocates the remaining nodes to the Clusters in two steps

# EVALUATION PROCESS

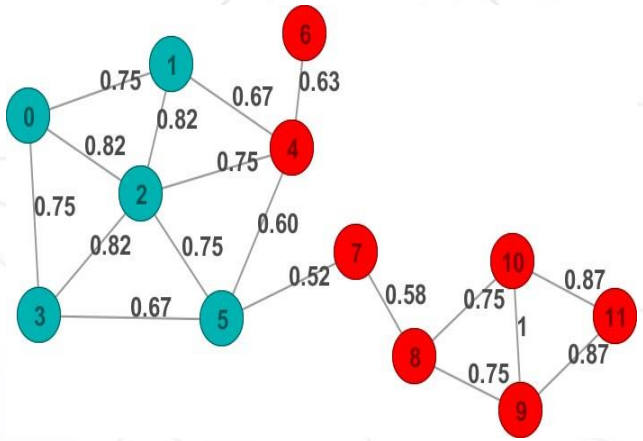
- **Baseline Algorithms**
  - ✓ MOSES, SPLA, HOCTracker, OCDDP, SMFRW
- **Metrics**
  - ✓ ONMI, Ω Index, F1 Score for ground truth communities
  - ✓ Overlapping Modularity for unknown ground truth communities
- **Datasets**
  - ✓ 12 real-world networks and synthetic networks
- Better performance, in most cases
- Average speed

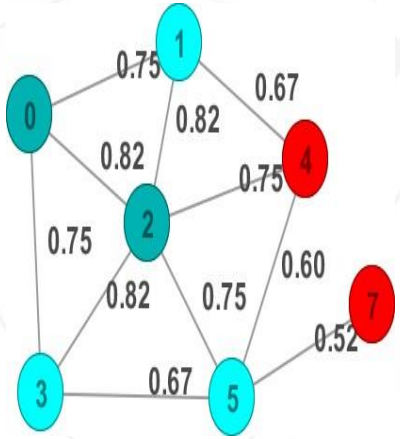
# CRITIQUE

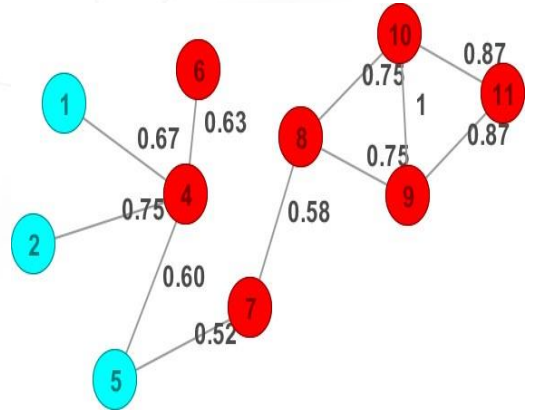
- The method is generally better than the baseline algorithms and can be used for finding overlapping communities in social networks
- Two parameters related to its performance must be appointed manually
- It is not recommended in large networks because of its average speed

# DISTRIBUTED STRUCTURAL CLUSTERING ALGORITHM FOR NETWORKS(DSCAN)

- An approach for finding communities in very large networks
- A distributed implementation of SCAN method



# SCAN OVERVIEW

- Determines the **Core Nodes** of the network by finding their strong connections and forming their **ε-Neighborhood**

- Allocates the nodes with strong connections to the same **Cluster**

- Identifies **Bridges** and **Outliers**

# HOW DOES DSCAN WORK?

- Is based on a distributed Master/Slave architecture
- Divides the graph file to partitions and send it to the workers
- Finds the **Frontier** nodes from neighboring partitions
- Copies the Frontier nodes to the partitions

# HOW DOES DSCAN WORK?

- Performs the SCAN algorithm
- Merges the results from all workers
- Solves  Disputes

# EVALUATION PROCESS

- **Baseline Algorithms**
  - ✓ SCAN, pSCAN, AnySCAN, ppSCAN
- **Datasets**
  - ✓ 5 large real-world networks
- Only time-related tests
- Slowest of all but horizontally scalable
- Only one able to compute the largest network

# CRITIQUE

- The method can compute  very large networks because of its distributed nature
- Time complexity was not given
- Very slow, especially in the merging step
- No information for its accuracy compared to related methods

# COMMUNITY DETECTION AND VISUALIZATION IN COMPLEX NETWORK BY THE DENSITY-CANOPY-KMEANS ALGORITHM AND MDS EMBEDDING

# OBJECTIVES OF THE STUDY

- Improve basic *K*-means algorithm
  - Improve accuracy
  - Improve stability
  - Improve time complexity, $O(n^{Kd+1})$

    *n*: number of nodes in network, *d*: dimensionality, *K*: number of clusters
- Acceleration of algorithm by dimensionality reduction
  - MDS (MultiDimensional Scaling) algorithm

# DEFINING NODE DISTANCE

- Plenty of metrics can be used
- *Jaccard* similarity: $s_{ij} = \frac{L_{11}}{L_{01} + L_{10} + L_{11}}$
  - Exploit of adjacency matrix *A*
  - $L_{11}$: Number of common neighbors of nodes *i* and *j*
  - $L_{01} + L_{10} + L_{11}$: total number of neighbors of nodes *i* and *j*
- Distance between nodes: $dist_{ij} = 1 - s_{ij} \Rightarrow dist_{ij} = \frac{L_{01} + L_{10}}{L_{01} + L_{10} + L_{11}}$
- Problem when applying MDS on zero distance nodes
  - New distance metric: $d_{ij} = \gamma \cdot dist_{ij} + dist\_random_{ij}$
  - γ: community structure coefficient, $dist\_random_{ij}$ in [0,1]
  - Creation of distance matrix *D*

# DIMENSIONALITY REDUCTION

- MDS algorithm, reduction to *m* (<*d*) dimensions
  - Use of dimension matrix *D*
  - Computation of a new matrix *B*

$$b_{ij} = -\frac{1}{2}\left( d_{ij}^2 - \frac{\sum_{j=1}^{n} d_{ij}^2 - \sum_{i=1}^{n} d_{ij}^2}{n} + \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2}{n^2} \right)$$

  - Decomposition of matrix *D* to eigenvalues and eigenvectors matrices
    - Selection of *m* largest eignevalues, creation of diagonal matrix **Λ**
    - Selection of the corresponding *m* eigenvectors, creation of matrix **Ψ**
- Projection of nodes to new coordination system

$$X = \Psi\sqrt{\Lambda}$$

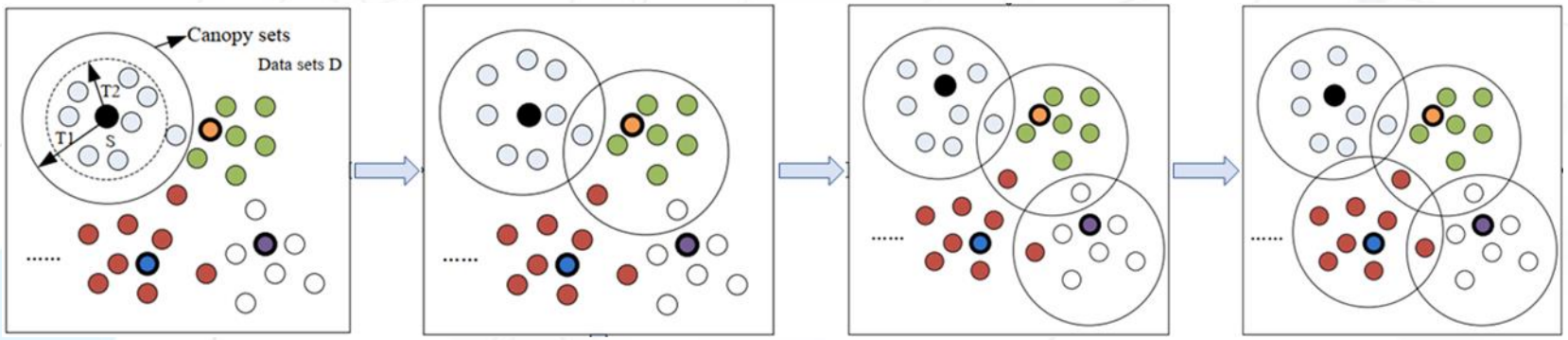
# DENSITY–CANOPY–***K*-MEANS** ALGORITHM

- *K*-means algorithm
  - Reliable algorithm for unsupervised clustering
  - Has a high time complexity $O(n^{dK+1})$

    *d*: node dimensionality, *K*: number of clusters (communities)
  - Finding an optimum solution to *K*-means is an *NP*-hard problem
    - Optimum number *K* of clusters,
    - Optimum initial seed

# DENSITY–**CANOPY**–*K*-MEANS ALGORITHM

- Canopy algorithm
  - Simple, fast and efficient algorithm for unsupervised clustering
  - Uses a random initial node and two predefined radial distances $T_1$ and $T_2$ ($T_1 > T_2$), to assign every node to a specific canopy (cluster)




  - The number of detected clusters and their centers are entered to the *K*-means algorithm as value of hyperparameter *K* and as initial seed respectively
  - How to set optimum values for hyperparameters $T_1$ and $T_2$?

# **DENSITY**–CANOPY–*K*-MEANS ALGORITHM

- Density clustering algorithm
  - To improve the stability and complexity of Canopy algorithm, the principle of density clustering is applied
    - Two new hyperparameters *r* and *t* are defined in order to compute variables *R* and *T*
      - $R = d_{\min} + r \cdot (d_{\max} - d_{\min})$
      - $T = d_{\min} + t \cdot (d_{\max} - d_{\min})$

      $d_{min}$: the average value of the minimum distance between each node and other nodes

      $d_{max}$: the average value of the maximum distance between each node and other nodes

      *r*: the density radius coefficient

      *t*: the distance threshold coefficient
    - Variables *R* and *T* correspond to hyperparameters $T_1$ and $T_2$ of Canopy algorithm

# DCK ALGORITHM, FINAL RESULTS

- The hyperparameters values of each individual algorithm are determined by the immediately preceding algorithm
- In total 3 new hyperparameters are introduced to system
  - $\gamma$: is used in distance metric
  - $r$ and $t$: are used in density clustering
  - Their optimal values are determined by maximizing the *modularity* metric, which estimates the quality of the algorithm detected communities
    - $r = t = 1/6$ and $\gamma = 100$
- Reduction of overall algorithm complexity to $O(n^3)$

# DCK ALGORITHM, EVALUATION

- The efficiency of the DCK algorithm was evaluated in a series of artificial and real-world networks and compared to other known algorithms (in real-world networks only)
  - Various network sizes, from small to medium (~5K)
  - Known community structure
- Accuracy in identifying the number of communities were assessed
- The quality of the identified communities was assessed through the modularity metric.
- Other metrics used:
  - Accuracy, F-measure, Rand Index (RI), Adjusted Rand Index (ARI), Normalized Mutual Information (NMI)

# DCK ALGORITHM, EVALUATION

- Results in artificial networks:
  - High accuracy in identifying the number of communities, which is decreasing as the network size increases (~2K nodes)
- Results compared to simple *K*-means algorithm in real-world networks:
  - Higher accuracy in identifying the number of communities

# DCK ALGORITHM, CRITIQUE

**Pros**

√ Simple and easy-to-implement algorithm

√ As input it only requires the network adjacency matrix

√ No additional prior knowledge about community structure is needed

√ Reduced time complexity compared to simple *K*-means algorithm

**Cons**

× Requires the optimization of additional hyperparameters

× Very slow on large scale networks

# CC-GA. A CLUSTERING COEFFICIENT BASED GENETIC ALGORITHM FOR DETECTING COMMUNITIES IN SOCIAL NETWORKS

# OBJECTIVES OF THE STUDY

- GAs are used in a wide variety of real-world applications
  - ✓ Optimization problems
  - ✓ Search problems
- GAs limitations
  - ✓ Random generation of initial population
    - Trapped in local optimum
    - Takes much longer to converge to a near-global optimum
- Find a better technique to generate initial population
  - ✓ Faster convergence to near-global optimum
  - ✓ Improve efficiency and accuracy of community detection

# TYPICAL GENETIC ALGORITHM STEPS

- Generation of initial population
  - ✓ Finite number of (random) chromosomes
- Fitness Function
  - ✓ Evaluation of chromosome/population performance
  - ✓ Select best performing chromosomes
- Crossover operator
  - ✓ Combines the gene structures of two chromosomes to generate the gene of the next generation's chromosomes
- Mutation operator
  - ✓ Alters a random gene of a random chromosome
- Population update
  - ✓ Replaces the least performing chromosomes with new best performing ones
  - ✓ Creates new generation

# CC-GA ALGORITHM STEPS

- Generation of initial population
  - ✓ Algorithm connects each node to only one of its neighbors in the original network
  - ✓ Instead of random selection of the neighbor, algorithm exploit the *Clustering Coefficient* (CC)

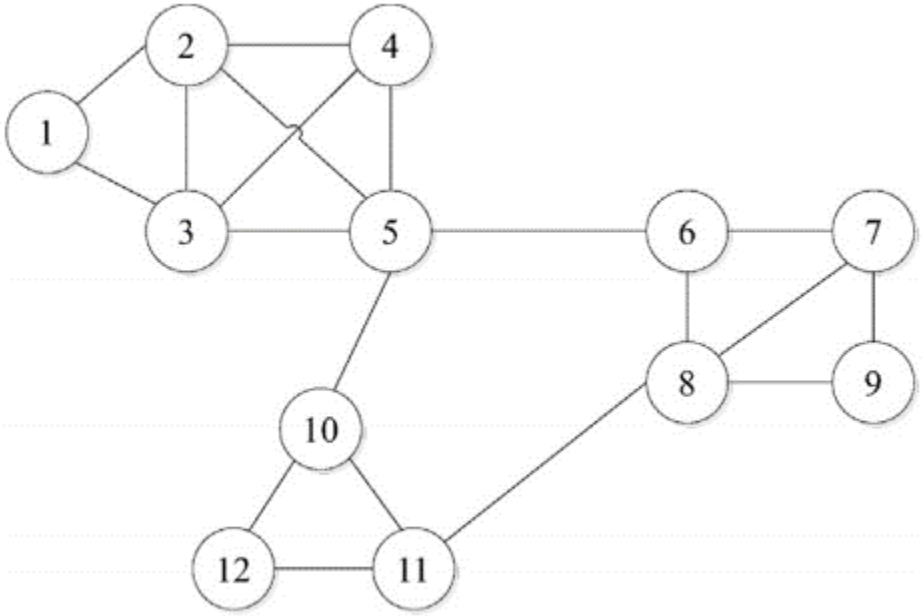
$$C_i = 2 \times \frac{L_i}{k_i\left(k_i - 1\right)}$$

$L_1$: the total number of edges among $v_i$'s neighbors, excluding the node itself, $k_i$: node's degree

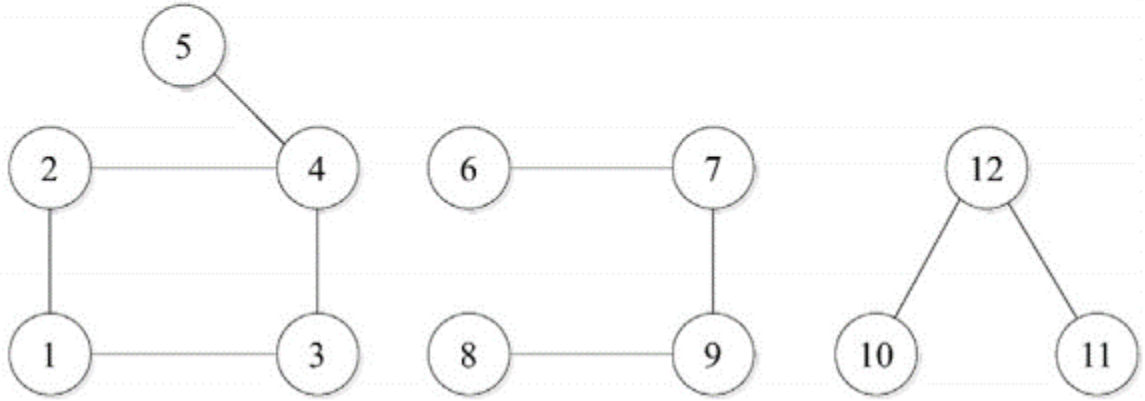
  - ✓ CC quantifies how close are the neighbors of a specific node to being a clique
  - ✓ If a node has multiple neighbors, the algorithm links it to only one neighbor, the one with the *highest* value of CC
  - ✓ Nodes connected with a *bridge* edge in original network, have lower values of CC, and are *less* possible to be selected by algorithm
  - ✓ As a final result:
    - Algorithm splits the original network to simpler *connected components*, by removing preferably the edge-local bridges
    - Each connected component is a different *community*
    - Local bridges are the *natural cut points* separating various communities

# CC-GA ALGORITHM STEPS

- Generation of initial population




| Nodes | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Genes | 3 | 1 | 4 | 2 | 4 | 7 | 9 | 9 | 8 | 12 | 12 | 11 |

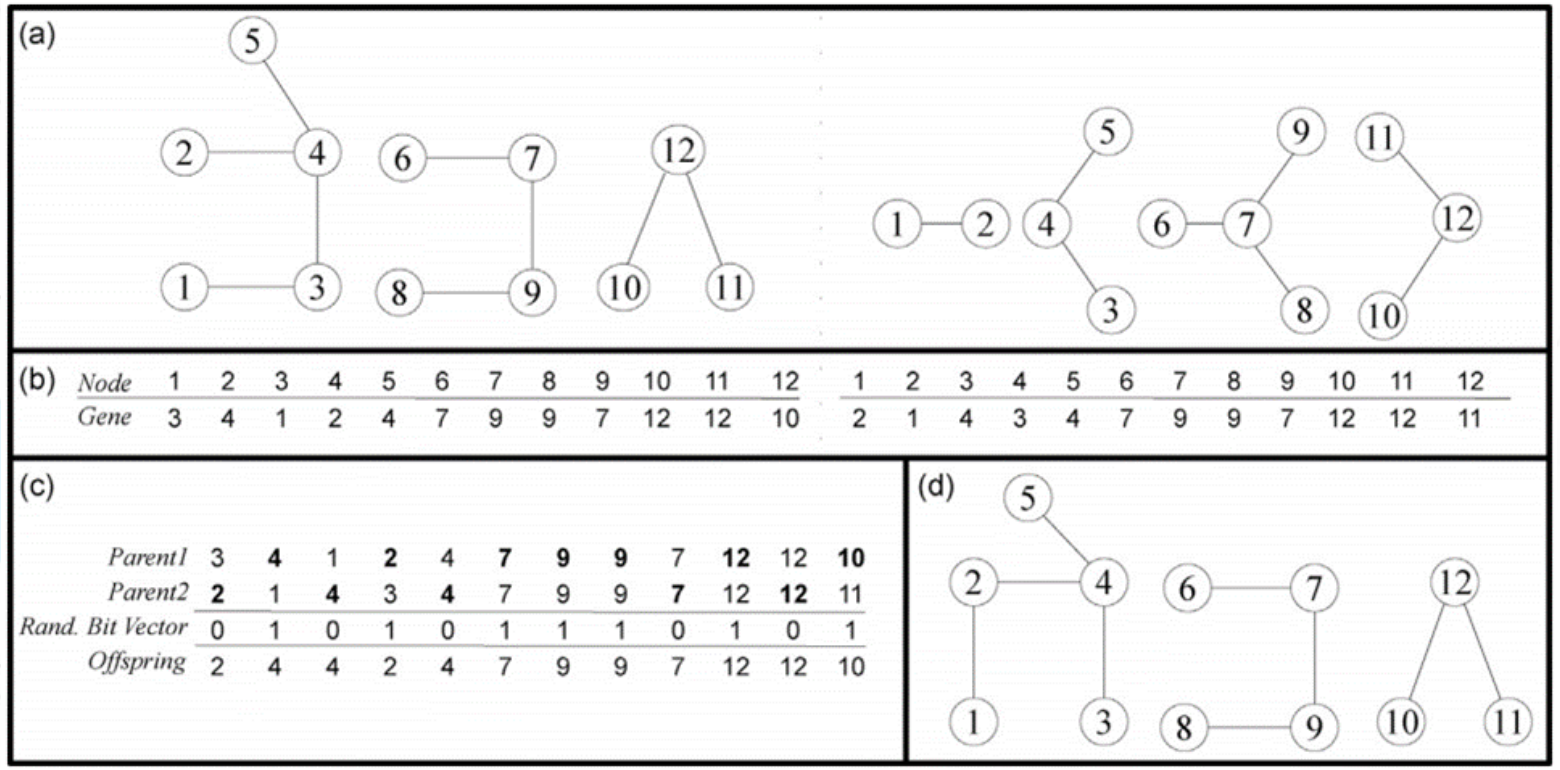


# CC-GA ALGORITHM STEPS

- Fitness Function
  - ✓ *Modularity* is used to evaluate the quality of community structure in each chromosome
  - ✓ Select best performing chromosomes
- Crossover operator
  - ✓ Uniform random crossover of two chromosomes' genes
- Mutation operator
  - ✓ Introduced a novel mutation operator
  - ✓ Algorithm selects random chromosomes from different communities and joins them
  - ✓ Ensures that smaller communities are merged into larger ones, and increases the community structure quality (modularity value) at the same time
- Population update
  - ✓ Replaces the least performing chromosomes with new best performing ones
  - ✓ Creates new generation

# CC-GA ALGORITHM STEPS

- Crossover operator



# CC-GA ALGORITHM HYPERPARAMETERS

- Must be carefully set before execution
  - ✓ Population size
  - ✓ Crossover rate
  - ✓ Mutation rate
  - ✓ Mutation extension rate
  - ✓ Percentage of chromosomes from the current population to be involved in the next iteration (generation) of genetic operators
  - ✓ Maximum number of iterations without any improvement (termination criterion)

# CC-GA ALGORITHM, EVALUATION

- ✓The efficiency of the CC-GA algorithm was evaluated using 13 artificial and 11 real-world networks and compared to 3 other GA and 6 other community detection algorithms
  - Various network sizes, from small to medium (~23K)
  - Known community structure
  - Modularity measure used to compare results of CC-GA with other techniques
- ✓ Compared to other GA approaches
  - CC-GA provides a better community structure than existing GA-based approaches
- ✓Compared to other non-GA-based community detection approaches
  - CC-GA delivers competitive results for various real-world networks
  - CC-GA is capable of detecting the community structure within a few iterations

# CC-GA ALGORITHM, CRITIQUE

**Pros**

√ Clustering Coefficient provides high quality initial population

√ No prior knowledge of the size, number or structure of the communities

√ Extensive evaluation of the algorithm against many state of-the-art algorithms, using many real or artificial networks of different sizes

**Cons**

× A relatively large number of hyperparameters are required to be set as algorithm's input

× Authors don't report overall algorithm time complexity

× Calculating Clustering Coefficient process has polynomial time complexity $O(n^k)$, where k<2.376

# CONCLUSION

- Community detection is a problem that can be addressed by a variety of methodological approaches.
- The algorithms reviewed in this presentation cover a wide range of these approaches.
- The majority of them are structural (except HICODE which extends any base algorithm).
- For their evaluation networks of various types and sizes are used.
- Two very common evaluation metrics are modularity and NMI. In some cases other metrics are used adding credibility to the process.
- Parameters and thresholds are factors that can seriously affect the performance of algorithms depending on them.