Optimum Community Detection Through Fusion of Constant Communities

Aditya Karan
14CS60D05

under the supervision of

Dr Rajib Mall
and
Dr Animesh Mukherjee

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

05 May 16

Table of Contents

Table of Contents

What is a Community?

Definition

Community within a large collection of individuals refers to a group within the collection such that members of that group interact more frequently with each other than with others in the collection.

Edges have:

- Inhomogenous in distribution.
- High concentration within communities.
- Low concentration between communities.

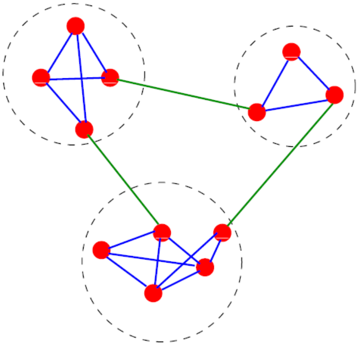


Figure: Communities in a graph

Random Graphs

Assumption

Probability that a pair of vertices has an edge is same for all possible pairs of vertices(Erdos and Rényi 1961).

Assuming:

- n, number of vertices;and
- p, probability of connection between a pair of vertices

then:

- Expected number of edges in the graph

$$e = \frac{pn(n-1)}{2}$$

- Expected mean degree

$$k = p(n-1)$$

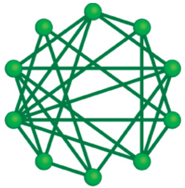


Figure: A random graph with 10 nodes and $p = 0.5$

Real Networks

However

Real networks are not random i.e. probability of connection is not same

- High level of order and organisation.
- Degree distribution follows a power-law:

$$P(d) = cd^{-\gamma}$$

or

$$\log(P(d)) = \log c - \gamma \log d$$

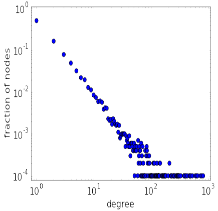


Figure: Scatter plot of a power law degree distribution on a log – log scale(*Scale-free networks - Math Insight*).

Communities are Everywhere

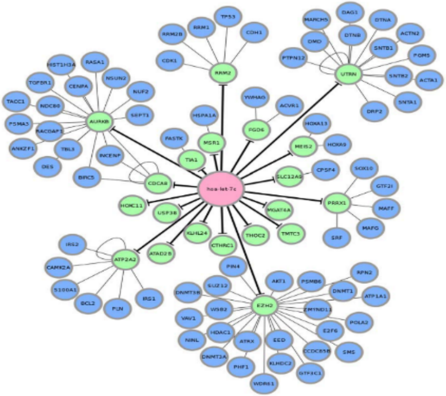


Figure: Protein-Protein Interaction Network

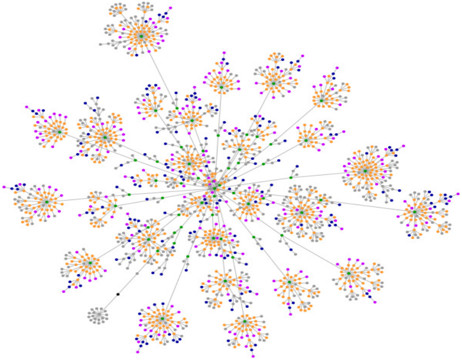


Figure: Pages of a website and their mutual hyperlinks

Communities are Everywhere

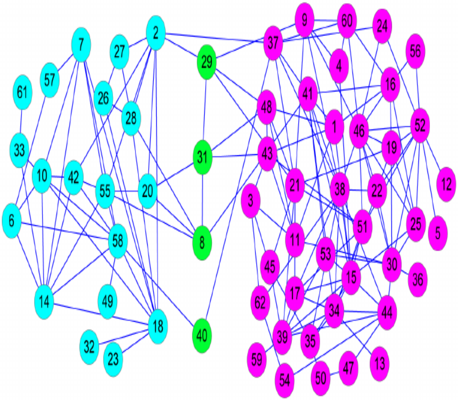


Figure: Lusseau's Network of Bottlenose Dolphins

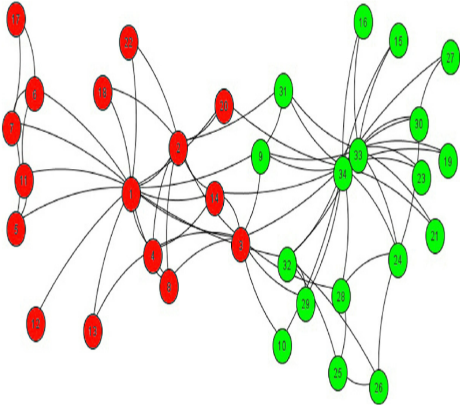


Figure: Zachary's Karate Club Communities

Applications

- Social Networks:Role detection, popularity estimation from communities based on common interests,locations,occupation etc.

Applications

- Social Networks:Role detection, popularity estimation from communities based on common interests,locations,occupation etc.
- Biological Networks: Drug response estimation based on functional groupings.

Applications

- Social Networks:Role detection, popularity estimation from communities based on common interests,locations,occupation etc.
- Biological Networks: Drug response estimation based on functional groupings.
- Citation Networks: Estimation of common research interests, reserach trends etc from communities based on co-cited papers.

Applications

- Social Networks:Role detection, popularity estimation from communities based on common interests,locations,occupation etc.
- Biological Networks: Drug response estimation based on functional groupings.
- Citation Networks: Estimation of common research interests, reserach trends etc from communities based on co-cited papers.
- Website Mirrors: Website mirror assignment to clients from communities based on geographical location.

Applications

- Social Networks:Role detection, popularity estimation from communities based on common interests,locations,occupation etc.
- Biological Networks: Drug response estimation based on functional groupings.
- Citation Networks: Estimation of common research interests, reserach trends etc from communities based on co-cited papers.
- Website Mirrors: Website mirror assignment to clients from communities based on geographical location.
- Graph Coarsening: Summarization or mapping a graph onto a similar smaller graph.

Applications

- Social Networks:Role detection, popularity estimation from communities based on common interests,locations,occupation etc.
- Biological Networks: Drug response estimation based on functional groupings.
- Citation Networks: Estimation of common research interests, reserach trends etc from communities based on co-cited papers.
- Website Mirrors: Website mirror assignment to clients from communities based on geographical location.
- Graph Coarsening: Summarization or mapping a graph onto a similar smaller graph.
- E-Commerce: Recommendation systems for communities of customers based on similar interests.

Project Motivation and Objectives

Motivation:

- Different technologies allow us to probe different aspects of a system e.g.:
  - Identification of cancer subtypes using fusion of similarity networks from DNA methylation, mRNA expression and miRNA expression datasets.
  - Different independent networks of 9/11 aircraft terrorists available with multiple US agencies.
- Combining these complementary perspectives can yield a greater insight.

Main objectives are as follows:

- Propose a method to fuse different observations of a dataset for improving community detection accuracy.
- Propose a method to reduce variations in results due to heuristics.
- Propose a method to increase partition accuracy by refining detected communities.

Project Objectives

Potential Applications:

- Entity Resolution e.g. detection of common social communities from networks on different social platforms(Facebook acquired Whatsapp!)
- Detection of customer interest groups by observing purchasing history from different e-commerce sites.
- Detection of demographically important communities by observing data from diverse sources like culture, education, income, consumption trends etc.
- Detect functional microbiological groupings from different responses to various stimuli.
- Graph compression/summarisation by agglomerating communities.
- Detection of communities of malicious websites (most such websites tend to have links to each other).
- Detection of core terrorist organisations by fusing information from different sources.
- Combining information from RADARs, infrared seekers and COMINT appliances to detect naval fleets.

Project Objectives

Implementation:

- Obtain different observations(networks) from a dataset.
- Detect and agglomerate invariant groups of vertices.
- Obtain predictions(graph partitions) from observations(networks).
- Carry out fusion of individual sets of prediction.
- Refine fused prediction set.

Table of Contents

Finding Community Structure

Intuition

Given a graph G with η vertices.Let C be a subgraph of G with η_C vertices and:

- Link Density of G, $\delta(G) = \frac{\text{Total edges in } G}{\eta(\eta-1)/2}$
- Internal degree of $\nu \in C$(Total edges connecting ν to $C - \nu$) $= K_\nu^{int}$
- External degree of $\nu \in C$(Total edges connecting ν to $G - C$) $= K_\nu^{ext}$

Then

- $K_\nu^{ext} = 0 \Rightarrow C$ is a good community for ν.
- $K_\nu^{int} = 0 \Rightarrow \nu$ is a disjoint vertex, assign ν to a separate community.

Finding Community Structure

Intuition

- Intra-cluster link density of C, $\delta_{int}(C) = \frac{\text{Total edges among nodes in } C}{\eta_C(\eta_C - 1)/2}$
- Inter-cluster link density of C, $\delta_{ext}(C) = \frac{\text{Total edges between nodes in } C \text{ and } G - C}{\eta_C(\eta - \eta_C)}$

Now, for C to be a community,

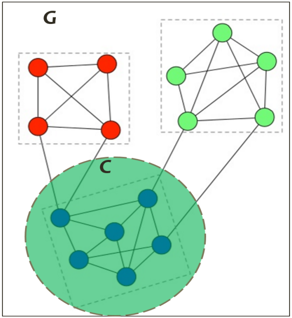
- $\delta_{int}(C) \gg \delta(G)$; and
- $\delta_{ext}(C) \ll \delta(G)$

Or

$$\sum_{\substack{\text{Maximise over}\\ \text{all communities}}} \delta_{int}(C) - \delta_{ext}(C) \tag{1}$$

Finding Community Structure

Intuition



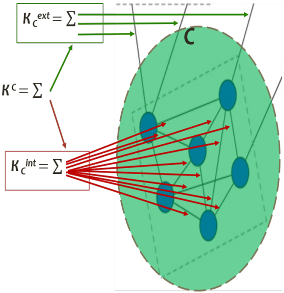


Figure: Intuitive Example of a Community

$\delta(G) = \frac{30}{105} = 0.29$, $\delta_{int}(C) = \frac{11}{15} = 0.73$, $\delta_{ext}(C) = \frac{4}{54} = 0.07$
Therefore, C can be a good community in G

Definitions of A Community

Local Definition

Central Idea

Focusses on sub-graphs under consideration(possibly studying their immediate neighbourhood also) only.

Four criteria :complete mutuality, reachability, vertex degree, comparison of internal vs external cohesion.
Leads to mostly the maximal subgraphs$\sim$ cliques.
But:

- **Too strict definition**: What to do if just one link is missing?
- **No hierarchy**: All vertices are symmetric within community.
- **Hard to find**: Exponential complexity in the graph size.

Definitions of A Community

Local Definition:Internal vs External Cohesion

- **Strong Community**: Sub-graph $C \subseteq G$ such that for $\nu \in C$,

$$K_i^{int}(\nu) > K_i^{ext}(\nu) \; \forall i \in \nu \tag{2}$$

Internal degree of every vertex in C should be greater than its external degree.

- **Weak Community**: Sub-graph $C \subseteq G$ such that for $\nu \in C$,

$$\sum_{i \in \nu} K_i^{int}(\nu) > \sum_{i \in \nu} K_i^{ext}(\nu) \; \forall i \in \nu \tag{3}$$

Total internal degree of sub-graph C should be greater than its total external degree.

Definitions of a Community

Global Definition

Central Idea

A graph has a community structure if it is different from a random graph.

- Communities are defined with respect to the whole graph.
- The graph is compared to a **random graph**.
- A random graph is not expected to have community structure.
    - Any pair of vertices has same probability of being adjacent.

Gives rise to the idea of **modularity**.

Definitions of a Community

Vertex-based Definitions

Central Idea

Communities are sub-graphs of vertices similar to each other.

Various similarity measures like:

- **Structural equivalence**.
- **Neighbourhood overlap**.

Goodness of Partitions

Partition

A division of graph into communities such that each vertex belongs to exactly one community.

What is a good partiton of the given graph?

- A **Quality Function** assigns a number to each partition of a graph.
- We can rank partitions based on the score assigned by the quality function.

A quality function Q is additive if there is an elementary function q such that, for any partition $\mathcal{P}$ of a graph:

$$Q(\mathcal{P}) = \sum_{C \in \mathcal{P}} q(C)$$

Quality Functions

Modularity

More edges are present in a community as compared to the equivalent sub-graph in a random graph(Newman and Girvan 2004).

$$Q = \frac{1}{2m}\sum_{i,j}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j)$$

where,

- $A \leftarrow$ Adjacency matrix.
- $k_i \leftarrow$ Degree of vertex i.
- $\frac{k_i k_j}{2m} \leftarrow$ Expected number of edges between i and j
- $$\delta(C_i, C_j) = \begin{cases} 1 & \text{if } C_i = C_j \\ 0 & \text{otherwise} \end{cases}$$

Quality Functions

Based on Ground Truth(Meilă 2007)

Rand Index(Rand 1971):

- Ratio of number of vertex pairs correctly classified in detected partition as compared to ground truth partition to the total number of pairs.

Adjusted Rand Index(Meilă 2007):

- Adjusted version of Rand Index, utilising a random graph expectation value.
- Null model assumes independence of two partitions.

Quality Functions

Based on Ground Truth

Given:

- $\mathcal{X} = (X_1, X_2, \cdots, X_{n_X})$ and $\mathcal{Y} = (Y_1, Y_2, \cdots, Y_{n_Y})$ are partitions of a graph G.
- $a \leftarrow$ Total pairs of vertices that are in the same community in both partitions.
- $b \leftarrow$ Total pairs of vertices that are in different communities in both partitions.
- $c \leftarrow$ Total pairs of vertices that are in the same community in $\mathcal{X}$ and in different communities in $\mathcal{Y}$.
- $d \leftarrow$ Total pairs of vertices that are in the same community in $\mathcal{Y}$ and in different communities in $\mathcal{X}$.

Quality Functions

Based on Ground Truth

- **Rand Index**:

$$R(\mathcal{X},\mathcal{Y}) = \frac{a+b}{a+b+c+d}$$

- **Adjusted Rand Index**:

$$\frac{Index - Expected\ Index}{Max\ Index - Expected\ Index}$$

Table: Contingency Table

X / Y	Y_1	Y_2	$\cdots$	Y_{n_Y}	Sums
X_1	n_{11}	n_{12}	$\cdots$	n_{1n_Y}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2n_Y}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_{n_X}	$n_{n_X 1}$	$n_{n_X 2}$	$\cdots$	$n_{n_X n_Y}$	a_{n_X}
Sums	b_1	b_2	$\cdots$	b_{n_Y}	

$$ARI = \frac{\sum_{i,j}\binom{n_{i,j}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

Quality Functions

Based on Ground Truth

Mutual Information(MacKay 2003):

- If two partitions are similar, very little information is required to infer one partition given the other. This extra information is used as measure of (dis)similarity.
- Disadvantage is that given a partition $\mathcal{X}$, all partitions derived from $\mathcal{X}$ by further partitioning (some of) its clusters would all have the same mutual information with $\mathcal{X}$, even though they could be very different from each other.

Normalised Mutual Information(Danon et al. 2005):

- Normalises the mutual Information by sum of Shannon Entropies of two given partitions.

Quality Functions

Based on Ground Truth

Given:

- $x_i \leftarrow$ community label of vertex x.
- $\{x_i\} \leftarrow$ community assignment in $\mathcal{X}$.
- $\{y_i\} \leftarrow$ community assignment in $\mathcal{Y}$.
- $n_x^X, n_y^Y \leftarrow$ number of vertices in communities X_i and Y_j.
- $P(x,y) = P(X=x, Y=y) = \frac{n_{xy}}{n}$,
  $P(x) = P(X=x) = \frac{n_x^X}{n}$,$P(y) = P(Y=y) = \frac{n_y^Y}{n}$
- $H(X) = -\sum_x P(x) log P(x)$ is the Shannon entropy of X
- $H(X|Y) = -\sum_{x,y} P(x,y) \log P(x|y)$ is the conditional entropy of X given Y.

Quality Functions

Based on Ground Truth

- **Mutual Information**:

$$I(X, Y) = \sum_{x} \sum_{y} P(x, y) log \frac{P(x, y)}{P(x)P(y)}$$

$$I(X, Y) = H(X) - H(X|Y)$$

- **Normalised Mutual Information**:

$$I_{norm}(\mathcal{X}, \mathcal{Y}) = \frac{2I(X, Y)}{H(X) + H(Y)}$$

Table of Contents

Community Detection Methods

Graph Partitioning

Central Idea

Divide vertices in k groups of predefined size such that number of edges lying between the groups is minimal.

- Number of edges between communities is called cut size.
- However, required to be specified *a priori*:
  - Number of communities.
  - Size of communities.

  Not known before hand.
- Kernighan-Lin algorithm(Kernighan and Lin 1970),

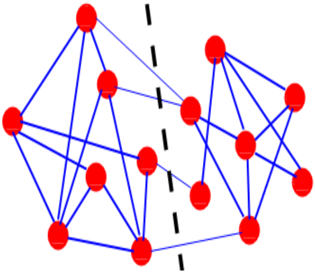


Figure: Graph Partitioning (Castellano, Fortunato, and Loreto 2009)

Community Detection Methods

Hierarchical Clustering

Central Idea

Identify groups with high similarity(not focussed on connectedness)(Friedman, Hastie, and Tibshirani 2001).

- A graph may have a hierarchical structure.
- Hierarchical Clustering:
  - Define a similarity measure between vertices and compute $n * n$ similarity matrix.
  - **Agglomerative** algorithms:(bottom up) Communities are merged if their similarity is sufficiently high.
  - **Divisive** algorithms:(top down) Communities are iteratively split removing edges between vertices with low similarity.
- Stopping condition like optimization of Quality Function (e.g. Modularity) may be applied.
- Girvan Newman divisive method etc.

Community Detection Methods

Hierarchical Clustering

- Number and size of communities not required to be specified.
- Disadvantages:
  - Does not provide a way to conclude which level is better.
  - Results depend on specific similarity measure adopted.
  - Vertices with just one neighbour classified as separate communities.

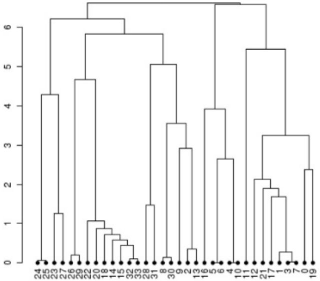


Figure: Hierarchy in a Graph

Community Detection Methods

Modularity Optimisation

Motivation

If high value of modularity indicates a good partition, optimize modularity.

- Exhaustive search impossible since:
  - Search-space is exponential in $|V|$.
  - Modularity optimization is NP-complete(Brandes et al. 2006).
- Various approximation techniques:
  - Greedy optimisation -Newman algorithm, Louvain algorithm etc.
  - Heuristic search - External optimization etc.
  - Probabilistic approximation - Simulated annealing etc.

Community Detection Methods

Modularity Optimisation

Newman's Algorithm(Newman and Girvan 2004):

- Agglomerative Clustering: Repeatedly join communities together in pairs, choosing at each step the join that results in the greatest increase in modularity.
- Tentative joins limited to m - the number of edges since joining a pair of disconnected communities cannot increase modularity.

$$Q = 0.4687, Q_{c1Uc2} = 0.4757,$$

$$Q_{c1Uc3} = 0.3246, Q_{c3Uc4} = 0.4079$$

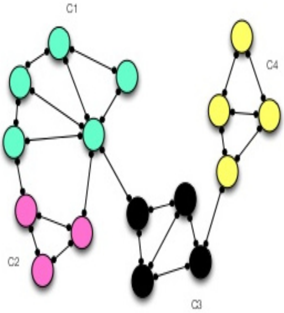


Figure: Newman's Algorithm

Community Detection Methods

Modularity Optimisation

Louvain Method(Blondel et al. 2008):
It consists of two phases:

- **Phase 1**(Modularity Optimisation)
  (Starts with every vertex in its own separate community)
  1. For all neighbours j of vertex i
  Check that by placing vertex i in which of the j communities increases modularity Q.
  2. Place i in that community for which ΔQ is maximum.
- **Phase 2** (Community Aggregation)
  1. Collapse all communities obtained from Phase 1 as single vertices.
  2. Multiple edges of communities are replaced by a single edge of weight equal to sum of weights of edges connecting them previously.

Repeat the Steps until $\Delta Q = 0$ (no more change in modularity).

Community Detection Methods

Modularity Optimisation

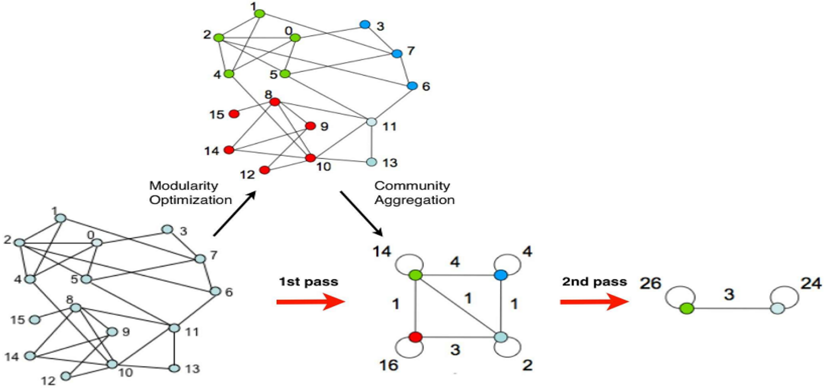


Figure: Louvain Method(Blondel et al. 2008):(a) Modularity optimisation by community reassignment (b) Community aggregation by folding communities.

Community Detection Methods

Constant Communities

- Community detection is an NP-complete problem.
  - Perturbing the order of vertices results in different local maxima of modularity.
- But, certain vertices always stay together in all such iterations - **Constant Communities** (Chakraborty et al. 2013).

Key Idea

Club these constant communities into super-vertices prior community detection and break down such super-vertices post community detection.

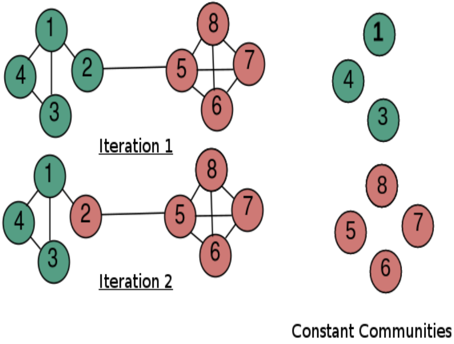


Figure: Illustration: Constant Communities

Community Detection Methods

Constant Communities

- Formation of super-vertices from **constant communities**:
  - Super-vertices replace original constituents.
  - Original edges replaced with single edge of weight equal to sum of these edges.
  - Edges of original vertices within constant communities replaced with a self-loop on respective super-vertices.
  - Edges between original vertices from constant communities and other vertices replaced with a single edge between super-vertices and other vertices.

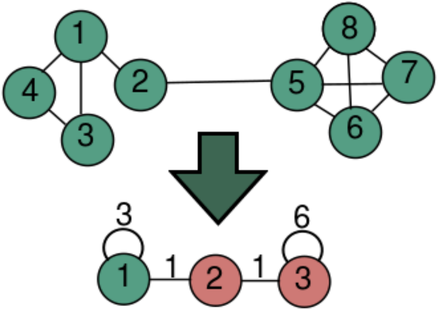


Figure: Illustration: Formation of Super-vertices

Community Detection Methods

Constant Communities

- Replacement of super- vertices by constituent vertices:
  - Post community detection, super- vertices replaced by constituent vertices.
  - Constituent vertices assigned same community as their respective super- vertices.
  - Number of detected communities remains same.
  - Goodness checks carried out on partition after replacement.
- Shown to increase NMI, ARI and modularity.

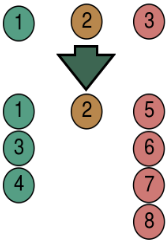


Figure: Illustration: Replacement of Super-vertices

Table of Contents

Graph Theoretic Approach

A recent work describes a Graph Theoretic Approach to data fusion with two basic steps(Žurauskienė, Kirk, and Stumpf 2015) :

- Predictions are obtained from several networks modelled from the dataset representing its structure and dependencies.
- Prediction sets from all network are compared for similarities and common-consensus prediction-set is reported as final.

Given:

- $G^{(x)} \leftarrow x$-th network construction represented as an adjaceny matrix from given dataset.
- $q^{(x)} \leftarrow$ Set of predictions from $G^{(x)}$ (for Graph Theoretic Approach, $q^{(x)}$ is $c^{(x)}$, the partition of network $G^{(x)}$).
- $l_i^{(x)} \leftarrow$ Label of vertex i in partition $q^{(x)}$.
- $n \leftarrow$ Total number of different network constructions.
- $m \leftarrow$ Total number of vertices in the network(constant for a given dataset).

Graph Theoretic Approach

Aim is to model the following joint distribution:

$$p(q^{(1)}, \cdots, q^{(n)} | G^{(1)}, \cdots, G^{(n)})$$

- **Data Fusion Approach**: Following independence is assumed(Balasubramanian et al. 2004):

$$p(q^{(1)}, \cdots, q^{(n)} | G^{(1)}, \cdots, G^{(n)}) \approx p(q^{(1)} | G^{(1)}) * \cdots * p(q^{(n)} | G^{(n)})$$

  Leads to H-Product fusion.
- **Sampling from** $p(q^{(x)} | G^{(x)})$: Equivalent to choosing value g_{ij} from $G^{(x)}$ for all vertex-pairs $(i, j) \forall i, j \in m$.
- **Fusion Function**: Represent partitions from $G^{(x)}$ and $G^{(y)}$ as adjacency matrix $A^{(x)}$ and $B^{(y)}$ where $a_{i,j}^{(x)} \in A^{(x)} = 1$ iff $l_i^{(x)} = l_j^{(x)}$, else 0. Then, fusion function:

$$f(a_{i,j}^{(x)}, b_{i,j}^{(y)}) = a_{i,j}^{(x)} * b_{i,j}^{(y)}$$

Experimental Setup

- GTA appears promising for data fusion.
- Utilisation of H-Product focuses on *minimum-common consensus* among the partitions.
- Using datasets with availability of "Ground-truth" communities i.e. actual community labels for each node.
- Five different networks created from given dataset as fusion candidates.

Datasets

- **Wine Dataset**(Lichman 2013) Chemical analysis results of wines from three cultivars.
  - Nodes:178, Features:13, Original Communities:3
- **Wine Quality Dataset**(Cortez et al. 2009) Physicochemical and sensory variables of red variant of the Portuguese "Vinho Verde" wine.
  - Nodes:1599, Features:11, Original Communities:6
- **Breast Cancer Dataset**(Lichman 2013)(Mangasarian and Wolberg 1990)(*UCI Machine Learning Repository: Breast Cancer Wisconsin (Original) Data Set*) characteristics of the cell nuclei present in the digitized image of a fine needle aspirate (FNA) of a breast mass.
  - Nodes:599, Features:31, Original Communities:2

Generation of Networks

Similarity/Distance Measures

- **Cosine Similarity**:

$$a_{i,j} = \frac{\sum_{k \neq i,j}^{m} A_{i,k} * A_{k,j}}{\sqrt{\sum_{k \neq i,j}^{m} A_{i,k}^2} \sqrt{\sum_{k \neq i,j}^{m} A_{k,j}^2}}$$

- **Pearson Coefficient**:

$$a_{i,j} = \frac{Cov(A_i, A_j)}{\sqrt{Var(A_i) * Var(A_j)}}$$

- **Jaccard Distance**:

$$d_{i,j} = 1 - \frac{\sum_{k \neq i,j}^{m} |minarg(A_{i,k}, A_{k,j})|}{\sum_{k \neq i,j}^{m} |maxarg(A_{i,k}, A_{k,j})|}$$

- **Euclidean Distance**:

$$d_{i,j} = \sqrt{\sum_{k \neq i,j}^{m} (A_{i,k} - Ak, j)^2}$$

- **Manhattan Distance**:

$$d_{i,j} = \sum_{k \neq i,j}^{m} |A_{i,k} - A_{k,j}|$$

- **Bray Curtis Distance**:

$$d_{i,j} = \frac{\sum_{k \neq i,j}^{m} |A_{i,k} - A_{k,j}|}{\sum_{k \neq i,j}^{m} |A_{i,k} + A_{k,j}|}$$

Experimental Setup

Algorithms

Algorithm 1: CONSTRUCT GRAPHS

Data: Set of vertcies S, Threshold τ
Result: Network Graph G
$n \leftarrow$number of vertices in set S
$G \leftarrow n * n$ matrix initialised with zeros
$A \leftarrow$ *relationship value*,$a_{ij} \forall$ vertices
$i, j \in S$
$m \leftarrow 0$ **foreach** i, j *in* A **do**
    if $a_{ij} > \tau$ **then**
        $G(i, j) \leftarrow 1$
        $m \leftarrow m + 1$
    end
end

Algorithm 2: GTA

Data: Networks $G^{(1)}, G^{(2)}, \cdots, G^{(n)}$
Result: Fused partition $c^{(fused)}$
foreach i *in* n **do**
    Find partitions $c^{(i)}$ of network $G^{(i)}$ using *Louvain* Algorithm
    Create Adjacency Matrix $A^{(i)}$ for each partition $c^{(i)}$
end
Set $H = A^{(1)} . * \cdots . * A^{(n)}$
Let $c^{(fused)}$ be the corresponding partition

Experimental Setup

Algorithms

Algorithm 3: GTA APPROACH

Data: Dataset D with m nodes, Types of similarity/distance metrics n, Thresholds $\tau^{(1)}, \cdots, \tau^{(n)}$

Result: Fused partition $c^{(fused)}$

foreach $k \in n$ *similarity metrics* **do**

  $G^{(k)} \leftarrow$ *Algorithm* CONSTRUCT GRAPHS $(D, \tau^{(k)})$

end

$c^{(fused)} \leftarrow$ *Algorithm* GTA$(G^{(1)}, G^{(2)}, \cdots, G^{(n)})$

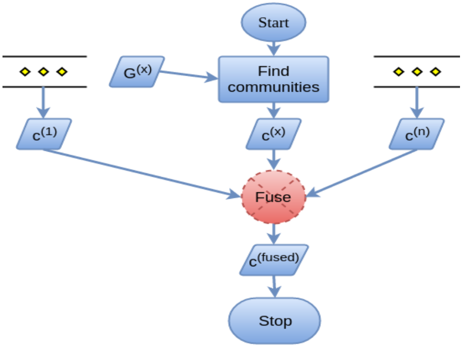


Figure: GTA Approach

Experimental Setup

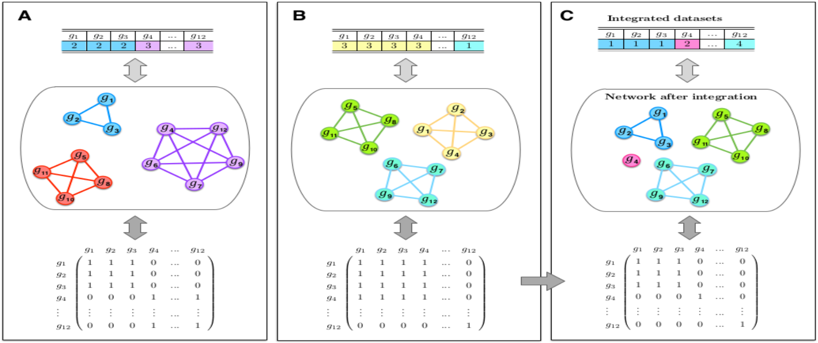


Figure: Illustration of Graph Theoretic Approach: (a) An example network partition of 12 nodes arranged in three communities in the first candidate network realisation for fusion (b) Network partition of same 12 nodes arranged in different three communities in the second candidate for fusion (c) Fused network (and community labels) after performing data fusion.

Results of GTA

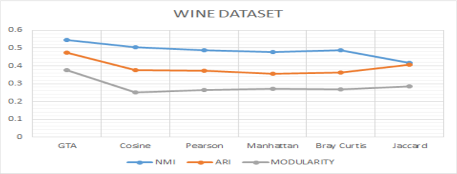
Four parameters observed:

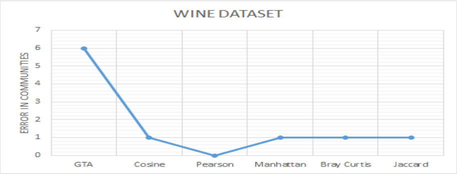
- NMI:Higher the better.
- ARI:Higher the better.
- Modularity:Higher the better.
- Error in number of communities:

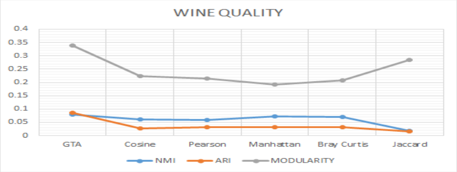
$$\xi = \left| M^{(actual)} - M^{(detected)} \right|$$

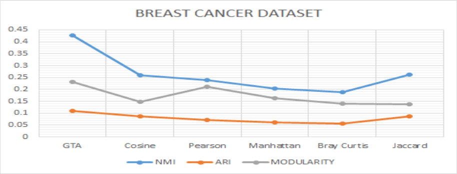
where $M^{(actual)} \leftarrow$ number of communities in "ground truth"
$M^{(detected)} \leftarrow$ number of communities detected
Lower the value, better is the performance.

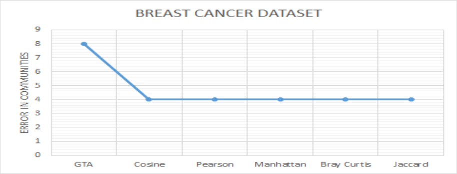
Results of GTA











Results of GTA

Observations:

- **GTA performs better than individual networks on NMI, ARI and modularity but worse on ξ.**
- GTA employs H-Product fusion of communities.
- Very strict fusion - communities that do not have consensus in **all fusion candidate networks** are split up.
- Fusion is happening at last stage of community detection process - no means to remedy the splits.
- Can we bring the invariant groups together before community detection?
  - Utilise the idea of constant communities as a pre-processing step.
- Fusion process can then be moved to different levels of the process.
  - Carry out fusion of data at different levels of community detection process.
- How to remedy the large splits in the end?
  - Carry out a post-processing step to merge communities in final partition based on various community parameters.

Scope of Improvement

- **Constant Communities**:
  - Identify the invariant groups and form super-vertices.
  - Carry out community detection.
  - Replace super-vertices by constituent vertices in final partition.
- **Fusion at Different Levels**:
  - Fuse individual networks.
  - Fuse constant communities.
  - Fuse detected partitions.
- **Post Detection Unionisation**:Join communities whose union performs better than individual communities on following parameters:
  - Link density.
  - $\frac{\text{internal edges of the community}}{\text{external edges of the community}}$.

Table of Contents

Our Proposal

- **Approach 1 (<u>Fuse Detected Partitions</u>)**:

  (1) **Generate graphs** from different similarity measures.
  (2) Perturb the order of vertices k times in a degree preserving order and find vertices that are always in the same community, i.e. **constant communities**.
  (3) **Club** the constant communities.
  (4) Detect communities in this graph using **Louvain Method**.
  (5) **Unclub** to get actual communities.
  (6) **Fuse** the communities to get the final partition.

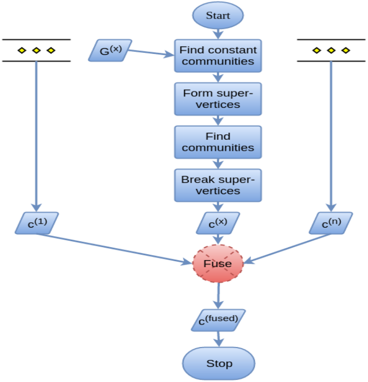


Figure: Approach 1

Our Proposal

- **Approach 2 (<u>Fuse Constant Communities</u>)**:
  1. **Generate graphs** from different similarity measures.
  2. Perturb the order of vertices k times in a degree preserving order and find vertices that are always in the same community, i.e. **constant communities**.
  3. **Fuse** the constant communities using fusion function.
  4. **Club** the constant communities.
  5. Detect communities in this graph using **Louvain Method**.
  6. **Unclub** to get final partition.

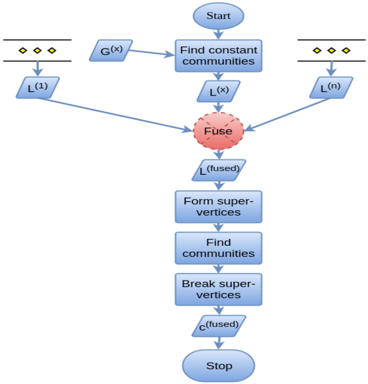


Figure: Approach 2

Our Proposal

- **Approach 3 (Fuse Graph Structures)**:
  (1) **Generate graphs** from different similarity measures.
  (2) **Fuse** the networks using different fusion function as defined in Eq. 4, 5 , 6 and 7.
  (3) Perturb the order of vertices k times in a degree preserving order and find vertices that are always in the same community, i.e. **constant communities**.
  (4) **Club** the constant communities.
  (5) Detect communities in this graph using **Louvain Method**.
  (6) **Unclub** to get final partition.

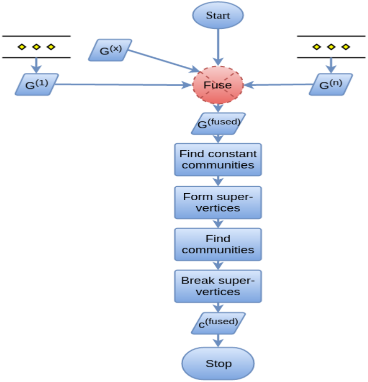


Figure: Approach 3

Our Proposal

- **Graph Fusion**:
  - **H-Product**

$$G_{(ij)}^{(fused)} = \begin{cases} 1 & \text{if} \quad g_{ij} = 1 \forall G \in (G^{(1)}, \cdots, G^{(n)}) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

  - **Majority Consensus**

$$G_{(ij)}^{(fused)} = \begin{cases} 1 & \text{if} \quad g_{ij} = 1 \text{ in } \geq \frac{n}{2} \text{ networks} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

  - **ORing**

$$G_{(ij)}^{(fused)} = \begin{cases} 1 & \text{if } g_{ij} = 1 \text{ in any network} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

  - **Weighted Fusion**

$$G_{(ij)}^{(fused)} = \begin{cases} x & \text{if} \quad g_{ij} = 1 \text{ in } x (\leq n) \text{ number of networks} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Work Accomplished

- **Details of work accomplished**:
  - Experiments carried out using MATLAB. Toolboxes used:
    - Matlab Tools for Network Analysis (2006-2011)(*MIT Strategic Engineering Research Group: Olivier L. de Weck*).
    - Information Theory Toolbox(*Information Theory Toolbox - File Exchange - MATLAB Central*).
    - NCT Toolbox (*(simple) Tool for estimating the number of clusters - File Exchange - MATLAB Central*).
    - Louvain Algorithm from author's website(*Louvain method for community detection*).
  - Following algorithms implemented:
    - Generation of graphs.
    - Obtain constant communities, club constant communities, unclub constant communities.
    - Fusion function.
    - Approaches 1, 2 and 3.
    - Unionising communities.

Algorithms

Algorithm 4: LOUVAIN ALGORITHM

Data: Graph $G^{(x)}$ in Adjacency Matrix Form M
Result: Community Partition $c^{(x)}$
while $\Delta Q > 0$ **do**

Phase 1 (Modularity Optimisation)
(Starts with every node i in a separate community, $C_i \in c^{(k)}$)
foreach *row* $i \in M(i,:)$ **do**
  foreach *Node* j *with* $M(i,j) > 0$ **do**
    $j_{max} \leftarrow$ Node resulting in maximum change in Modularity ΔQ
  end
  Place i in community of $C_{j_{max}}$
end

Phase 2 (Community Aggregation)
foreach *Community* $C_i \in c^{(k)}$ *obtained in Phase 1* **do**
  Collapse C_i as single node
  Replace multiple edges to different nodes as a single edge with edge weight equal to sum of weights of all such edges.
end

end

Algorithms

Algorithm 5: OBTAIN CONSTANT COMMUNITES

Data: Graph $G^{(x)}$ in Adjacency Matrix Form $M^{(x)}$
Result: Constant Communities $L^{(x)}$
Sort $M^{(x)}$ in degree-preserving order $k \leftarrow$ Number of possible degree-preserving permutations of $M^{(x)}$ **for** $i \leq |k|$ **do**

  $M_i^{(x)} \leftarrow M^{(x)}$ sorted according to order k_i $C_i^{(x)} \leftarrow$ LOUVAIN ALGORITHM($M_i^{(x)}$)

end
$L^{(x)} \leftarrow$ all constant communities in $C^{(x)}$

Algorithms

Algorithm 6: CLUB COMMUNITES

Data: Graph $G^{(x)}$ in Adjacency Matrix Form $M^{(x)}$ and Constant Communities $L^{(x)}$

Result: Adjacency Matrix $N^{(x)}$

$N^{(x)} \leftarrow M^{(x)}$ **for** $i \in |L^{(x)}|$ **do**

  $W^i_{self} \leftarrow$ Sum of weight of edges between the vertices the constant community $L^{(x)}_i$ **for** $j \in G^{(x)} - L^{(x)}_i$ **do**

    $W^i_{ext_j} \leftarrow$ Sum of weight of edges between the vertices the constant community $L^{(x)}_i$ and nodes in rest of the graph

  end

  Replace nodes $\in L^{(x)}_i$ in $N^{(x)}$ with a super-vertex with self-loop of weight W^i_{self} and an edge with vertex j with weight $W^i_{ext_j}$

end

Algorithms

Algorithm 7: UNCLUB COMMUNITES

Data: Community Partition $D^{(x)}$ and Constant Communities $L^{(x)}$

Result: Unclubbed Community Partition $c^{(x)}$

$c^{(x)} \leftarrow D^{(x)}$ **for** $i \leq \|c^{(x)}\|$ **do**

  Replace node $c_i^{(x)}$ with all nodes in $L_i^{(x)}$ Label of unclubbed nodes $\leftarrow c_i^{(x)}$

end

Algorithm 8: FUSION FUNCTION

Data: Graphs $G_1^{(x)}, \cdots, G_1^{(n)}$ in Adjacency Matrix Form $M_1^{(x)}, \cdots, M_1^{(n)}$ of equal number of nodes m, Type of fusion f

Result: Fused Adjacency Matrix $M^{(fused)}$

for $i, j \leq m$ **do**

  $M^{(fused)}(i,j) \leftarrow f(M_1^{(x)}, \cdots, M_1^{(n)})$

end

Algorithms

Algorithm 9: OUR APPROACH 1

Data: Dataset D Thresholds $\tau^1, \cdots, \tau^n$, Fusion Function f

Result: NMI, ARI, Modularity and ξ

for $i \leq n$ **do**

  $G^{(i)} \leftarrow$ CONSTRUCT GRAPHS(D,τ^i)

  $L^{(i)} \leftarrow$ OBTAIN CONSTANT COMMUNITIES($G^{(i)}$)

  $N^{(i)} \leftarrow$ CLUB COMMUNITIES($G^{(i)}$,($L^{(i)}$,)

  $D^{(i)} \leftarrow$ LOUVAIN ALGORITHM($N^{(i)}$)

  $c^{(i)} \leftarrow$ UNCLUB COMMUNITIES($D^{(i)}$, $L^{(i)}$)

  $A^{(i)} \leftarrow$ Adjacency Matrix corresponding to $c^{(i)}$ s.t. $A^{(i)}_{x,y} = 1$ if $x, y \in$ same community

end

$A^{(fused)} \leftarrow$ FUSION FUNCTION($A_1^{(x)}, \cdots, A_1^{(n)}$, $f_{HProduct}$)

$c^{(fused)} \leftarrow$ Final Communities corresponding to $A^{(fused)}$

$G^{(fused)} \leftarrow$ FUSION FUNCTION($G_1^{(x)}, \cdots, G_1^{(n)}$, f)

Calculate NMI and ARI from $c^{(fused)}$

Calculate Modularity and ξ from $c^{(fused)}$, $G^{(fused)}$

Algorithms

Algorithm 10: OUR APPROACH 2

Data: Dataset D Thresholds $\tau^1, \cdots, \tau^n$, Fusion Function f
Result: NMI, ARI, Modularity and ξ

for $i \leq n$ **do**
    $G^{(i)} \leftarrow$ CONSTRUCT GRAPHS(D,τ^i)
    $L^{(i)} \leftarrow$ OBTAIN CONSTANT COMMUNITIES($G^{(i)}$)
    $LA^{(i)} \leftarrow$ Adjacency Matrix corresponding to $L^{(i)}$ s.t. $LA^{(i)}_{x,y} = 1$ if $x, y \in$ same constant community
end
$L^{(fused)} \leftarrow$ FUSION FUNCTION($LA_1^{(x)}, \cdots, LA_1^{(n)}, f_{HProduct}$)
$G^{(fused)} \leftarrow$ FUSION FUNCTION($G_1^{(x)}, \cdots, G_1^{(n)}, f$)
$N^{(fused)} \leftarrow$ CLUB COMMUNITIES($G^{(fused}$,($L^{(fused)}$,)
$D^{(fused)} \leftarrow$ LOUVAIN ALGORITHM($N^{(fused)}$)
$c^{(fused)} \leftarrow$ UNCLUB COMMUNITIES($D^{(fused)}$, $L^{(fused)}$)
Calculate NMI and ARI from $c^{(fused)}$
Calculate Modularity and ξ from $c^{(fused)}$, $G^{(fused)}$

Algorithms

Algorithm 11: OUR APPROACH 3

Data: Dataset D Thresholds $\tau^1, \cdots, \tau^n$, Fusion Function f
Result: NMI, ARI, Modularity and ξ

for $i \leq n$ **do**
    $G^{(i)} \leftarrow$ CONSTRUCT GRAPHS(D,τ^i)
end
$G^{(fused)} \leftarrow$ FUSION FUNCTION $(G_1^{(x)}, \cdots, G_1^{(n)}, f)$
$L^{(fused)} \leftarrow$ OBTAIN CONSTANT COMMUNITIES($G^{(fused)}$)
$N^{(fused)} \leftarrow$ CLUB COMMUNITIES($G^{(fused}$,($L^{(fused)}$)
$D^{(fused)} \leftarrow$ LOUVAIN ALGORITHM($N^{(fused)}$)
$c^{(fused)} \leftarrow$ UNCLUB COMMUNITIES($D^{(fused)}$, $L^{(fused)}$)
Calculate NMI and ARI from $c^{(fused)}$
Calculate Modularity and ξ from $c^{(fused)}$, $G^{(fused)}$

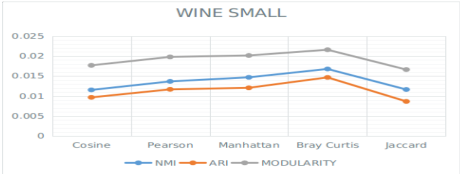
Table of Contents

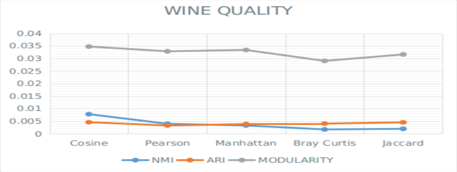
Results and Analysis

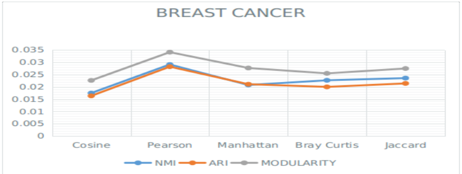
We discuss following results from our experiments:

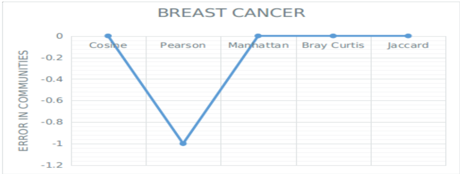
- **Effect of constant communities as a pre-processing step**: Increase in NMI, ARI, modularity and decrease in ξ values.
- **Our approaches against GTA**: Comparison of NMI, ARI, modularity and ξ values.
- **Effect of graph fusion methods**: Increase in NMI, ARI, modularity and decrease in ξ values of winning approach.
- **Effect of unionising communities as a post-processing step for fusion**: Increase in NMI, ARI, modularity and decrease in ξ values due to(for H-product fusion method only):
  - Unionisation of communities in Approach 1.
  - Unionisation of constant communities in Approach 2.

NMI, ARI, Modularity and ξ: Constant Communities





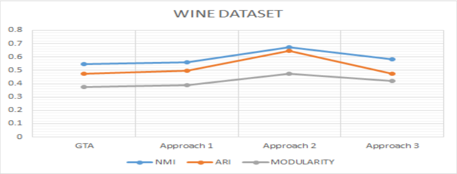


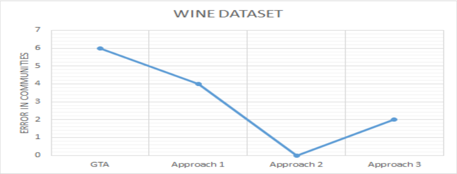


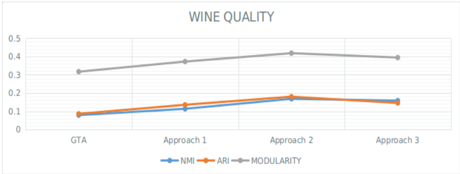
NMI, ARI, Modularity and ξ: Constant Communities

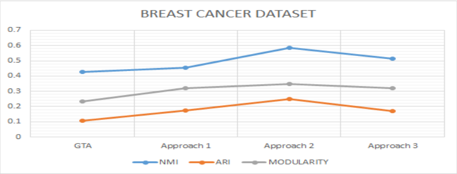
- Using constant communities as a pre-processing step leads to:
  - Increase in NMI, ARI, modularity.
  - Decrease in ξ values.

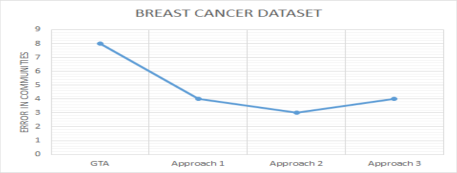
Our Proposed Approaches and GTA: H-Product Fusion



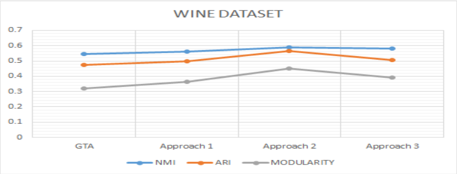


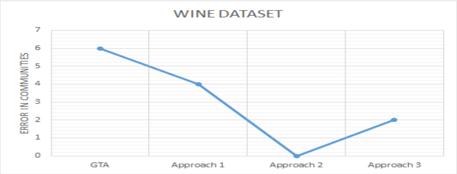


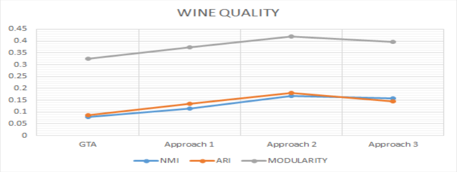


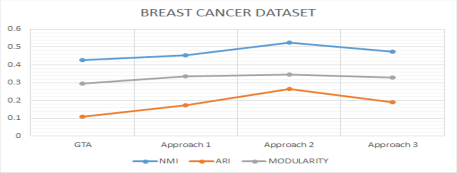


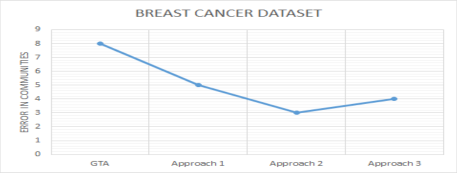
Our Proposed Approaches and GTA: Majority Fusion



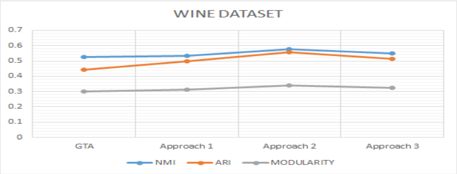


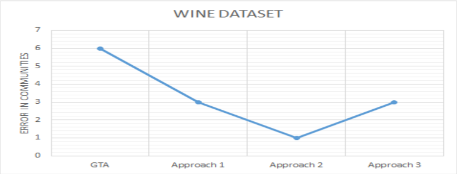


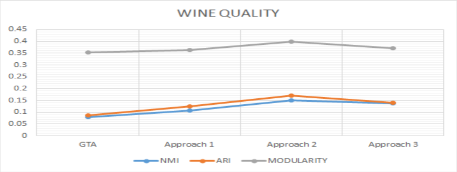


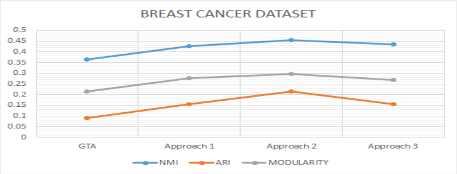


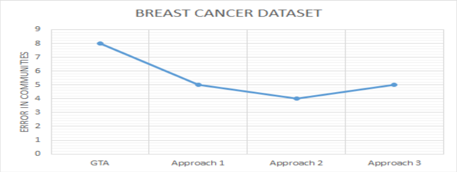
Our Proposed Approaches and GTA: OR Fusion



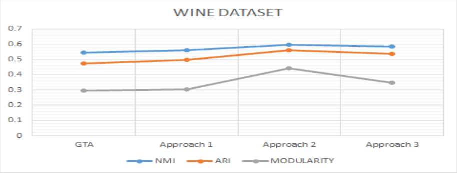


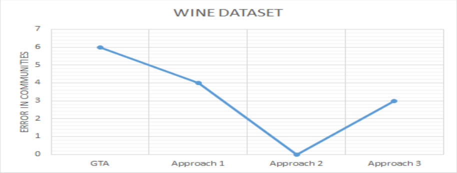


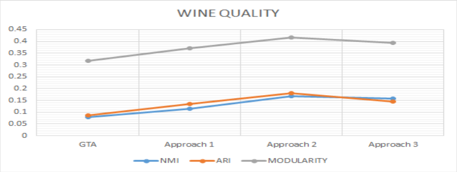


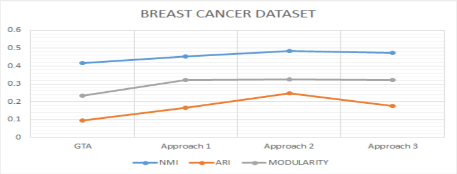


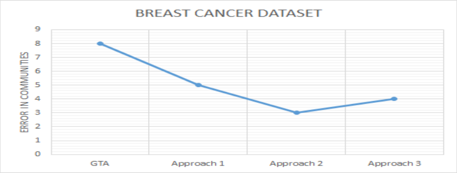
Our Proposed Approaches and GTA: Weighted Fusion







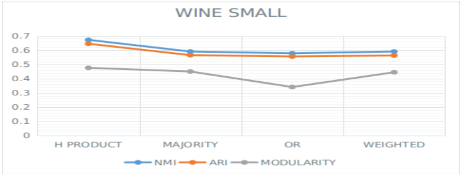


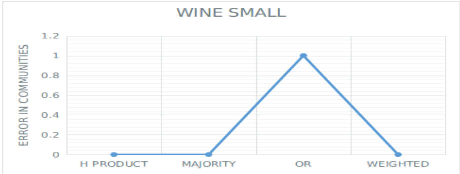


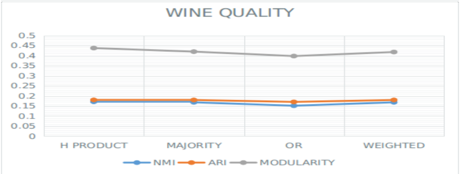
Our Approaches and GTA

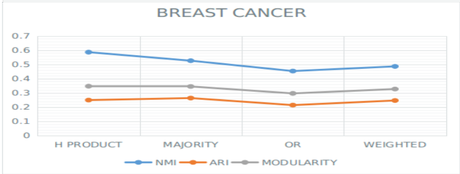
- Approach 2 performs better than the rest of the approaches.

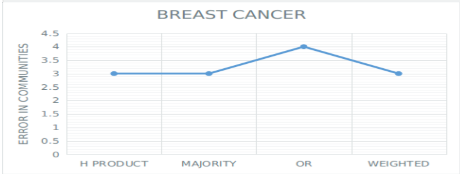
Performance of Our Approach 2 for Different Graph Fusion Methods







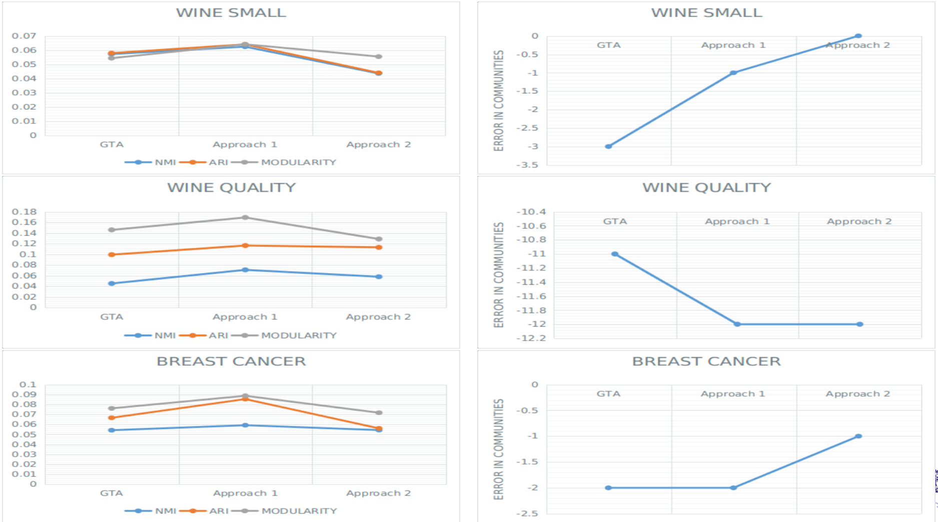




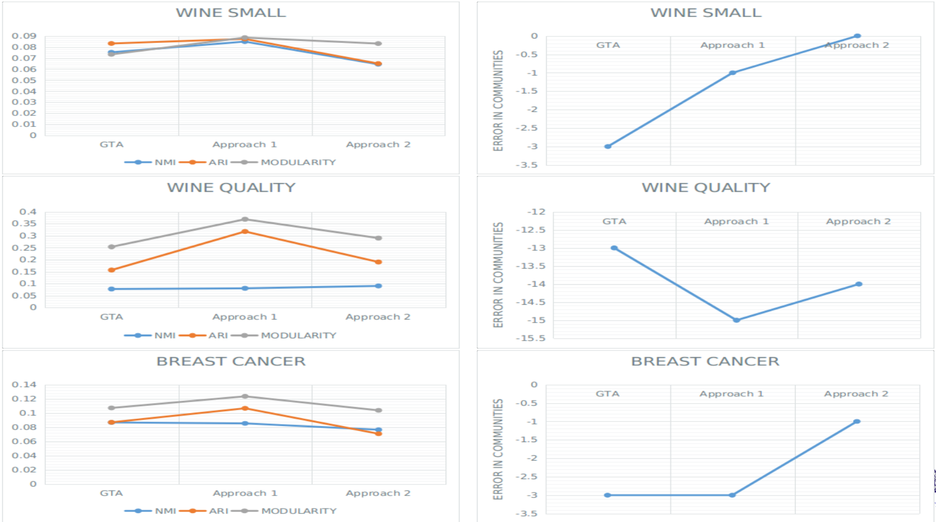
Performance of Our Approach 2 for Different Graph Fusion Methods

- H-Product Fusion Method is performing best on NMI, ARI and modularity.

NMI, ARI, Modularity and ξ due to Unionisation: Using Link Density



NMI, ARI, Modularity and ξ due to Unionisation: Using Ratio of Edges



NMI, ARI, Modularity and ξ due to Unionisation

- Unionisation leads to increase in NMI, ARI and modularity and decrease in ξ. Unionising using ratio of edges performs better than unionising using link density.

Comparative Results: Percentage Increase Over GTA

Table: Performance Increase Over GTA:Wine Dataset

NETWORKS	NMI	ARI	MODULARITY	ξ
		WITHOUT UNIONISING		
Our Approach 1	3.053	5.21	3.31	-33.33
Our Approach 2	23.28	36.00	26.54	-100
Our Approach 3	7.10	6.15	11.88	-66.67
		UNIONISING USING LINK DENSITY		
GTA	10.49	12.21	14.45	-50
Our Approach 1	14.50	18.72	20.35	-50
Our Approach 2	31.28	45.29	41.30	-100
		UNIONISING USING RATIO OF EDGES		
GTA	13.78	17.53	19.50	-50
Our Approach 1	18.61	23.61	26.84	-50
Our Approach 2	**35.09**	**49.71**	**48.64**	**-100**

Comparative Results: Percentage Increase Over GTA

Table: Performance Increase Over GTA:Wine Quality Dataset

NETWORKS	NMI	ARI	MODULARITY	ξ
WITHOUT UNIONISING				
Our Approach 1	57.57	74.18	16.38	-11.36
Our Approach 2	117.06	109.32	29.21	-25
Our Approach 3	103.51	76.14	18.77	-13.64
UNIONISING USING LINK DENSITY				
GTA	57.36	62.92	13.74	-25
Our Approach 1	147.45	127.26	31.93	-38.64
Our Approach 2	190.53	173.48	33.84	-52.27
UNIONISING USING RATIO OF EDGES				
GTA	98.23	91.95	28.62	-29.55
Our Approach 1	172.19	175.60	31.46	-45.45
Our Approach 2	**231.59**	**225.30**	**58.73**	**-56.82**

Comparative Results: Percentage Increase Over GTA

Table: Performance Increase Over GTA:Breast Cancer Dataset

NETWORKS	NMI	ARI	MODULARITY	Error
WITHOUT UNIONISING				
Our Approach 1	6.47	60.06	38.34	-50
Our Approach 2	37.31	130.49	49.17	-62.5
Our Approach 3	20.29	55.60	37.29	-50
UNIONISING USING LINK DENSITY				
GTA	12.70	61.53	32.71	-25
Our Approach 1	20.34	138.91	76.47	-75
Our Approach 2	50.03	182.22	79.99	-75
UNIONISING USING RATIO OF EDGES				
GTA	20.34	80.28	46.05	-37.5
Our Approach 1	26.49	158.51	91.40	-87.5
Our Approach 2	**55.24**	**195.90**	**93.77**	**-75**

Analysis

- Using constant communities as a pre-processing step leads to appreciable improvement in NMI, ARI, modularity and ξ values.
- Approach 2 betters the rest of the approaches:
  - Fusion at constant communities level - Greater level of invariance is incorporated than GTA despite H-product fusion pruning the constant communities.
  - Lowest ξ value - Final community partition avoids the H-product splits.
- Approach 3 is behind Approach 2 but better than rest:
  - Fusion at graph level - Change in graph structure itself.
  - Higher ξ value - Reduction in distinguished community structures in graph leading to higher number of communities.
- Approach 1 is better than GTA but worst of proposed approaches:
  - Fusion at community partitions stage - Large number of splits, no further chance of agglomeration.
- Winning approach performs best with H-product graph fusion method.
- Unionisation helps reduce splits due to h-product fusion and leads to improvement in NMI, ARI, modularity and ξ values.

Table of Contents

Conclusion and Future Direction

- **Novelty of proposed approach**:
  - Incorporates community invariance prior to community detection process:
    - Reduction in graph size.
    - Increase in NMI, ARI, modularity etc.
  - Employing fusion at level of constant communities leads to fewer splits than GTA.
  - Unionising communities as a post-processing step further undoes the splits due to H-product fusion.
- **Future Direction**:
  - Fusion of communities and constant communities by methods other than H-product.
  - Unionising communities as a post-processing step using better graph theoretic parameters.

Table of Contents

References I

L Euler. *Commentarii Academiae Scientiarum Imperialis Petropolitanae 8, 128*. 1736.

Lester R Ford and Delbert R Fulkerson. "Maximal flow through a network". In: *Canadian journal of Mathematics* 8.3 (1956), pp. 399–404.

Paul Erdos and Alfréd Rényi. "On the evolution of random graphs". In: *Bull. Inst. Internat. Statist* 38.4 (1961), pp. 343–347.

James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

Brian W Kernighan and Shen Lin. "An efficient heuristic procedure for partitioning graphs". In: *Bell system technical journal* 49.2 (1970), pp. 291–307.

References II

William M Rand. “Objective criteria for the evaluation of clustering methods”. In: *Journal of the American Statistical association* 66.336 (1971), pp. 846–850.

Earl R Barnes. “An algorithm for partitioning the nodes of a graph”. In: *SIAM Journal on Algebraic Discrete Methods* 3.4 (1982), pp. 541–550.

OL Mangasarian and WH Wolberg. “Cancer diagnosis via linear programming. University of Wisconsin-Madison”. In: *Computer Sciences Department* (1990).

T Łuczak. “Proceedings of the Symposium on Random Graphs, Poznań 1989”. In: (1992).

Michael Molloy and Bruce Reed. “A critical point for random graphs with a given degree sequence”. In: *Random structures & algorithms* 6.2-3 (1995), pp. 161–180.

References III

Alex Pothen. "Graph partitioning algorithms with applications to scientific computing". In: *Parallel Numerical Algorithms*. Springer, 1997, pp. 323–368.

Béla Bollobás. *Modern graph theory, volume 184 of Graduate Texts in Mathematics*. 1998.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. Springer series in statistics Springer, Berlin, 2001.

Réka Albert and Albert-László Barabási. "Statistical mechanics of complex networks". In: *Reviews of modern physics* 74.1 (2002), p. 47.

Jon Kleinberg. "An impossibility theorem for clustering". In: *Advances in neural information processing systems* (2003), pp. 463–470.

References IV

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Mark EJ Newman. "The structure and function of complex networks". In: *SIAM review* 45.2 (2003), pp. 167–256.

Raji Balasubramanian et al. "A graph-theoretic approach to testing associations between disparate sources of functional genomics data". In: *Bioinformatics* 20.18 (2004), pp. 3353–3362.

Aaron Clauset, Mark EJ Newman, and Cristopher Moore. "Finding community structure in very large networks". In: *Physical review E* 70.6 (2004), p. 066111.

Mark EJ Newman and Michelle Girvan. "Finding and evaluating community structure in networks". In: *Physical review E* 69.2 (2004), p. 026113.

References V

Leon Danon et al. "Comparing community structure identification". In: *Journal of Statistical Mechanics: Theory and Experiment* 2005.09 (2005), P09008.

Stefano Boccaletti et al. "Complex networks: Structure and dynamics". In: *Physics reports* 424.4 (2006), pp. 175–308.

Ulrik Brandes et al. *On modularity-np-completeness and beyond*. Citeseer, 2006.

Leon Danon, Albert Díaz-Guilera, and Alex Arenas. "The effect of size heterogeneity on community identification in complex networks". In: *Journal of Statistical Mechanics: Theory and Experiment* 2006.11 (2006), P11010.

Hristo N Djidjev. "A scalable multilevel algorithm for graph clustering and community structure detection". In: *Algorithms and models for the web-graph*. Springer, 2006, pp. 117–128.

References VI

Mark EJ Newman. "Finding community structure in networks using the eigenvectors of matrices". In: *Physical review E* 74.3 (2006), p. 036104.

Josep M Pujol, Javier Béjar, and Jordi Delgado. "Clustering algorithm for determining community structure in large networks". In: *Physical Review E* 74.1 (2006), p. 016107.

Haifeng Du et al. "An algorithm for detecting community structure of social networks based on prior knowledge and modularity". In: *Complexity* 12.3 (2007), pp. 53–60.

Marina Meilă. "Comparing clusterings—an information based distance". In: *Journal of multivariate analysis* 98.5 (2007), pp. 873–895.

Satu Elisa Schaeffer. "Graph clustering". In: *Computer Science Review* 1.1 (2007), pp. 27–64.

References VII

Ken Wakita and Toshiyuki Tsurumi. "Finding community structure in mega-scale social networks:[extended abstract]". In: *Proceedings of the 16th international conference on World Wide Web*. ACM. 2007, pp. 1275–1276.

Vincent D Blondel et al. "Fast unfolding of communities in large networks". In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

Philipp Schuetz and Amedeo Caflisch. "Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement". In: *Physical Review E* 77.4 (2008), p. 046112.

Zhenqing Ye, Songnian Hu, and Jun Yu. "Adaptive clustering algorithm for community detection in complex networks". In: *physical review E* 78.4 (2008), p. 046115.

References VIII

Claudio Castellano, Santo Fortunato, and Vittorio Loreto. "Statistical physics of social dynamics". In: *Reviews of modern physics* 81.2 (2009), p. 591.

Paulo Cortez et al. "Modeling wine preferences by data mining from physicochemical properties". In: *Decision Support Systems* 47.4 (2009), pp. 547–553.

Erik Holmström, Nicolas Bock, and Johan Brännlund. "Modularity density of network community divisions". In: *Physica D: Nonlinear Phenomena* 238.14 (2009), pp. 1161–1167.

Jure Leskovec et al. "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters". In: *Internet Mathematics* 6.1 (2009), pp. 29–123.

Andreas Noack and Randolf Rotta. "Multi-level algorithms for modularity clustering". In: *Experimental Algorithms*. Springer, 2009, pp. 257–268.

References IX

Biao Xiang, En-Hong Chen, and Tao Zhou. "Finding community structure based on subgraph similarity". In: *Complex Networks*. Springer, 2009, pp. 73–81.

Santo Fortunato. "Community detection in graphs". In: *Physics reports* 486.3 (2010), pp. 75–174.

Benjamin H Good, Yves-Alexandre de Montjoye, and Aaron Clauset. "Performance of modularity maximization in practical contexts". In: *Physical Review E* 81.4 (2010), p. 046106.

Michele Tumminello, Fabrizio Lillo, and Rosario N Mantegna. "Correlation, hierarchies, and networks in financial markets". In: *Journal of Economic Behavior & Organization* 75.1 (2010), pp. 40–58.

Tanmoy Chakraborty et al. "Constant communities in complex networks". In: *Scientific reports* 3 (2013).

References X

M. Lichman. *UCI Machine Learning Repository*. 2013. URL: http://archive.ics.uci.edu/ml.

Justina Žurauskienė, Paul DW Kirk, and Michael PH Stumpf. "A Graph Theoretical Approach to Data Fusion". In: *bioRxiv* (2015), p. 025262.

Vincent D Blondel. *Louvain method for community detection*. URL: https://perso.uclouvain.be/vincent.blondel/research/louvain.html.

Information Theory Toolbox - File Exchange - MATLAB Central. URL: http://in.mathworks.com/matlabcentral/fileexchange/35625-information-theory-toolbox.

MIT Strategic Engineering Research Group: Olivier L. de Weck. URL: http://strategic.mit.edu/downloads.php?page=matlab_networks.

References XI

Network clusterization algorithm–Group Prof. Amedeo Caflisch. URL: http://www.biochem-caflisch.uzh.ch/public/5/network-clusterization-algorithm.html.

Scale-free networks - Math Insight. URL: http://mathinsight.org/scale_free_network.

Seven Bridges of Königsberg - Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Seven_Bridges_of_K{\"{o}}nigsberg.

(simple) Tool for estimating the number of clusters - File Exchange - MATLAB Central. URL: http://www.mathworks.com/matlabcentral/fileexchange/13916--simple--tool-for-estimating-the-number-of-clusters.

Structure and Strangeness. URL: http://cs.unm.edu/~aaron/research/fastmodularity.htm.

References XII

UCI Machine Learning Repository: Breast Cancer Wisconsin (Original) Data Set. URL: http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original).

THANK YOU

Multiple Observations of a Dataset

- A dataset can be modelled in different ways or have different observations.
- Combining these complementary perspectives can yield benefits like:
  - Enhancing knowledge about overall dataset dependencies.
  - Different predictions can be derived independently in parallel.
- Data fusion categories(unsupervised):
  - **Bayesian techniques**.
  - **Non-Bayesian techniques**.

Different Observations of a Dataset

Bayesian techniques:

- Dependencies between networks modelled using similarity coefficients or a consensus structure.
- Prior beliefs and uncertainty formally expressed as probability densities.
- High computational cost.

Hence, we follow a graph -theoretic approach for the fusion of homogeneous networks:

- Multivariate joint modelling not required- low computational cost.

Implementation of Key Algorithms

- **Detection of Constant Communities**:
  (1) Calculate number of permutations of vertex-ordering (maintaining degree-preserved order), t.
  (2) Initialize C, an $n * t$ matrix where n is the number of vertices.
  (3) Repeat for all t permutations:
    i. Perturb the order of vertices.
    ii. Run **Louvain algorithm** and detect community labels.
    iii. Store community labels in $C(:, y)$ for y-th run.
  (4) Compare rows pairwise and identify rows that have same community labels in all columns. Groups of rows that have same community labels are identified as constant communities.

	Run 1	Run 2	⋯	Run t
V1	1	1	⋯	2
V2	1	2	⋯	2
V3	1	1	⋯	2
V4	2	2	⋯	1
V5	2	2	⋯	1
V6	2	2	⋯	1

Constant communities:

1. V1, V3.
2. V2.
3. V4,V5,V6.

Implementation of Key Algorithms

- **Unionisation of Communities**: Depending upon whether link density or ratio of internal edges to external edges is chosen as parameter p to be observed:
  (1) Generate sub-graphs for all communities in final partition $c^{(fused)}$ from original graph and **find their p values**.
  (2) Generate sub-graphs for all pairwise unions of communities in $c^{(fused)}$ and **find p values for all unions**.
  (3) Sort the set of p values of all communities and unions in descending order.
  (4) Start with empty partition $c^{(final)}$. Repeat till all vertices get selected in final partition:
     i. Select the community or union corresponding to highest p value in the stack and add to final partition $c^{(final)}$.
     ii. If a community was selected above:
        Remove all p-values of unions of which the selected community is a constituent.
        Else:
        Remove all p-values of constituent communities and unions of which the constituent communities are independent constituents.
  (5) Report $c^{(final)}$ as final partition.