VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

MATHEMATICAL FOUNDATION FOR COMPUTER SCIENCE

Assignment

Community Structure Identification: Book recomendation

Lecturer: **PhD. Nguyễn An Khương**
Students: **2270386 - Ngô Triệu Long**
2270376 - Tăng Quốc Thái
2270174 - Nguyễn Đình Khương
2210176 - Đỗ Võ Hoàng Hưng
2270375 - Nguyễn Trung Thuận

Ho Chi Minh city - November 2023

Contents

List of Figures

I. Introduction

1. Community Structure Identification Problem

The Community Structure Identification Problem (CSIP) is a foundational challenge in network analysis and graph theory. At its core, it grapples with the intricate task of unearthing the inherent groupings, or communities, within a network.

These communities hold the essence of dense internal connections, where nodes readily interact and share information among themselves. Yet, their outward tendrils reach less frequently, creating distinct boundaries with other gatherings within the network.

In a nutshell, the CSIP is about finding hidden groups or cliques within a network. It's like discovering friend groups within a school or social circles in a town.

These groups have tightly connected members but fewer ties to people outside their group. This problem is crucial in many fields because it helps us understand how complex systems are organized and how they function. Some example:

- **Social Networks:** Think about social networking sites like Facebook or Twitter. The challenge is to recognize unique clusters or communities of users who exhibit similar interests, share mutual social connections, or participate in comparable discussions. Identifying these communities enables the platform to understand user behavior better, tailor content suggestions, and identify influential users or potential communities of interest.

- **Biological Networks:** Analyzing communities in protein-protein interaction networks can help us understand cellular functions and identify potential drug targets. For instance, finding tightly connected groups of proteins involved in a disease process could lead to new therapies.

- **Online networks:** In online forums, discussion boards, or social media platforms, the Community Structure Identification Problem involves identifying clusters of users who interact and engage with each other on specific topics or interests. Uncovering these communities can help moderate online discussions, identify influential users, and facilitate content recommendation or targeted advertising.

Solving the Community Structure Identification Problem typically entails applying graph clustering algorithms to partition the network into cohesive communities. These algorithms are techniques that place individuals into their most likely friend groups. They often try to maximize modularity, which measures how well-defined the communities are. It's like checking if the friend groups make sense or if people are randomly scattered.

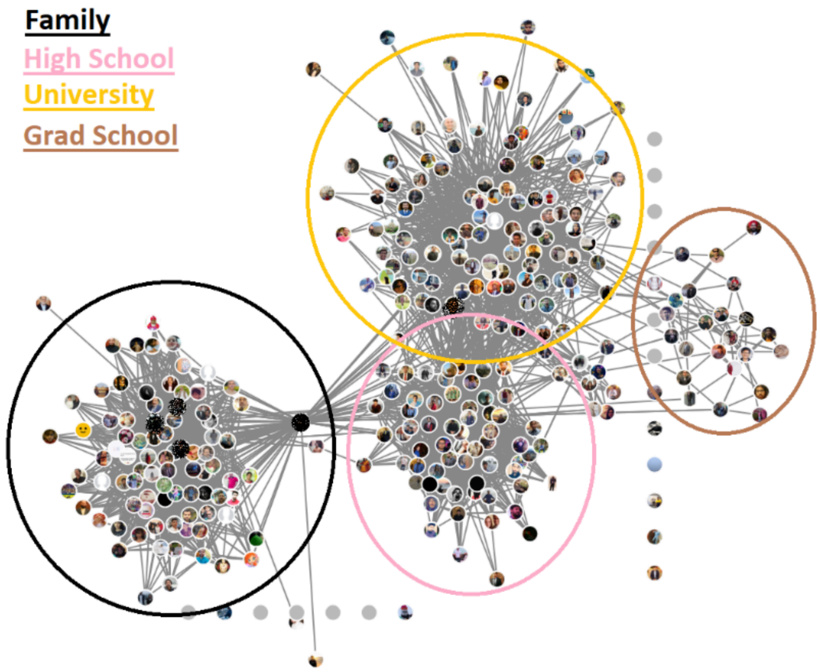


Figure 1: Figure 1.1 Social Network Example

2. CSIP for user cluster and product rating

2.1 Motivation

In the domain of E-commerce, characterized by a multitude of products vying for attention, the implementation of personalization emerges as a powerful strategy to transcend the noise. Imagine a digital storefront equipped with the ability to discern individual preferences even before user engagement, effectively directing customers towards products that evoke feelings of satisfaction and anticipation. This proposition is realized through the utilization of an application that harnesses the potential of user clustering and product rating prediction, thereby augmenting the overall shopping experience within this context.

By exploring the extensive collection of purchase histories and ratings, this technology uncovers interconnected communities of shoppers who share similar

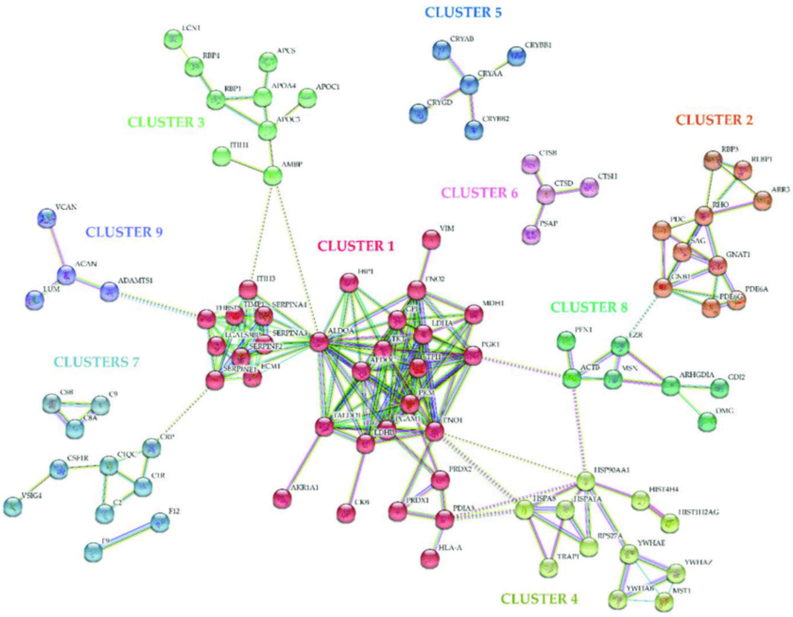


Figure 2: Figure 1.2 Protein Interaction Network example

preferences and aspirations. It can be likened to mapping the lively virtual marketplaces, revealing clusters where like-minded individuals gather around beloved brands, trending styles, or niche interests.

In this scenario, the Community Structure Identification Problem involves identifying clusters of users who share interests in product traits such as price, usability, maintainability, and brand, $\cdots$ and have bought similar products. By uncovering these user communities, we can gain insights into the organization of the buying trends and identify groups of users related to interests, purchase characteristics, or needs.

For businesses, this insight unlocks a treasure trove of opportunities:

- **Targeted Recommendations:** No more sifting through endless aisles. Each customer is greeted with a curated selection of products predicted to resonate with their unique preferences, fostering delight and satisfaction.

- **Personalized Discounts:**Promotions dance perfectly with individual desires, speaking to the heart of each cluster and igniting excitement for brands that truly understand their customers.

- **Strategic Product Development:** By analyzing the rating patterns of different clusters, businesses can uncover hidden gems, identify unmet needs,

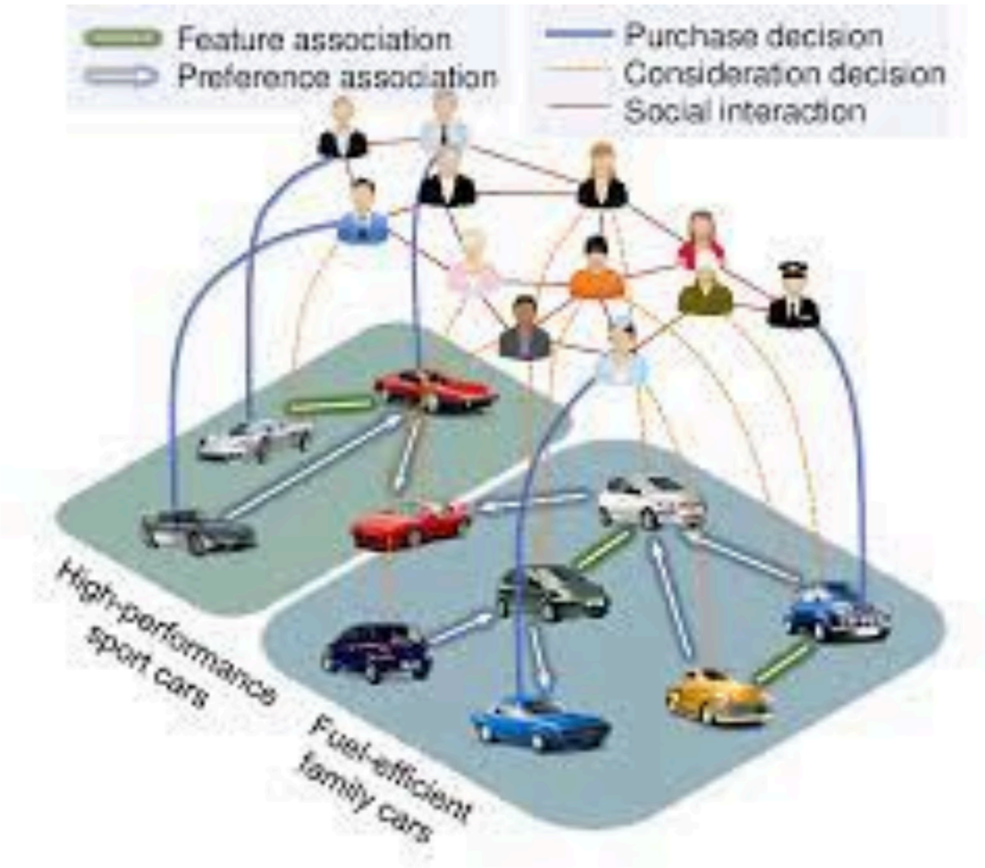


Figure 3: Figure 1.3 An example of how to cluster users based on product

and tailor their product offerings to align with the evolving desires of their customers.

- **Dynamic Pricing:** Prices gracefully adjust to reflect the anticipated value of a product within a specific cluster, optimizing revenue while ensuring a sense of fairness and value for shoppers.

Beside, the benefits would extend beyond immediate sales, shaping a more meaningful and engaging customer experience:

- **Relevant Content and Reviews:** Product descriptions and testimonials resonate more deeply when tailored to the interests of each cluster, fostering trust and connection.

- **Discovering New Delights:** Customers venture beyond their usual browsing habits, guided by recommendations that spark curiosity and introduce them to unexpected treasures they might have missed.

- **Building Brand Loyalty:** Personalized experiences cultivate a sense of belonging and appreciation, transforming customers into ardent advocates who eagerly share their positive experiences with others.

In summary, the Community Structure Identification Problem in the context of user clustering helps to reveal users with similar buying routines, combined with cluster average rating of a product to predict how a user who has never bought a product will rate it, enabling targeted marketing, recommendation systems, and a deeper understanding of market trend.

2.2 Dataset and approach proposal

Dataset Context: customer reviews and ratings of beauty-related products available on Amazon's platform.

Content: The dataset includes valuable information such as:

- Unique User IDs for customer identification.

- Product ASIN (Amazon's distinctive product identifier).

- Ratings, which reflect customer satisfaction on a scale from 1 to 5.

- Timestamps, recorded in UNIX time, indicate when the ratings were submitted.

This dataset is just a fragment of the extensive Amazon product dataset, encompassing a staggering 142.8 million reviews spanning May 1996 to July 2014. The complete dataset provides a wealth of information, including detailed product reviews, metadata, category information, pricing data, brand details, and image features.

Approach proposal

- Step 1: involves constructing a user-user bipartite graph. Edges connect users who share a purchase history for the same product, representing potential affinities between them. This network serves as the foundation for subsequent community detection algorithms.

- Step 2: deploys clustering algorithms to partition the network into cohesive communities. These techniques leverage edge density and topological features to identify groups of users exhibiting stronger internal connections than external linkages.

- Step 3: Step into the realm of predictive modeling. Given a product and a user who has never purchased it, we leverage the power of their assigned community. By aggregating the average rating for that product among the user's community members, we generate an expected rating reflecting the collective sentiment of their peers. This effectively leverages shared preferences within the identified communities to infer potential individual behavior.

- Step 4: concludes the process with rigorous evaluation. We compare the predicted ratings with actual ratings garnered once the user purchases the product. Metrics like mean squared error or absolute error assess the accuracy of our community-based predictions, allowing us to refine our algorithms and improve prediction performance.

In summary, users are divided into groups based on common characteristics, such as similarity of rating patterns or similar demographic data. After that, the prediction of the rating of a user to an item is computed based on the opinions of other users in the same group

CSI presents a compelling application of network analysis and community detection in customer behavior modeling. By uncovering hidden communities and harnessing the power of shared preferences, we can effectively predict user ratings, personalize recommendations, and ultimately foster stronger customer relationships. This approach bridges the gap between individual interactions and collective behavior, revealing the hidden social fabric woven within large-scale online platforms.

II. Preliminaries

1. Networks and Node, Community Structure

In mathematics, graph theory is the study of graphs, which are mathematical structures used to model pairwise relations between objects. A graph (or network) in this context is made up of vertices (also called nodes or points) which are connected by edges (also called links or lines).

A node is a unit of a network that represents an entity, such as a person, a product, a group, or a concept. Edges represent relations between nodes. A graph can be considered as the simplest representation of a complex system, where the vertices are the elementary units of the system and the edges represent their mutual interactions.

The term ” community structure” refers to the composition of a given network, including the number and attributes of nodes, as well as the relationships between them. Community structure is a crucial aspect of complex networks, as it indicates that nodes tend to cluster together into distinct groups or communities.

The term ” community structure” refers to the composition of a given network, including the number and attributes of nodes, as well as the relationships between them. Community structure is a crucial aspect of complex networks, as it indicates that nodes tend to cluster together into distinct groups or communities.

Understanding the community structure of a network can have important implications for many real-world applications, such as predicting the spread of diseases or identifying influential individuals in a social network. Moreover, community detection can also help to reveal underlying patterns and structures in complex systems, providing insights into the organization and behavior of these systems.

Comprehending the community structure of a network carries significant implications for various real-world applications, including forecasting the dissemination of diseases and pinpointing influential individuals within a social network. Furthermore, the process of community detection serves to unveil latent patterns and structures within intricate systems, affording insights into the organizational dynamics and behavioral tendencies inherent in these systems.

2. Community Detection problem

Communities can be implicit or explicit. Explicit communities are those, in which a grouping is predefined and members joining the group form a community. In this case communities are directly visible, for example whatsapp group.

Implicit communities on the other hand do not have any predefined classification. We have to analyze the activities of the individuals to form the community. Community detection is used for implicit communities only Community detection is a fundamental problem in network analysis that aims to identify groups of nodes that are more similar to each other than to the rest of the network.

One can argue that community detection is similar to clustering. Clustering is a machine learning technique in which similar data points are grouped into the same cluster based on their attributes. Even though clustering can be applied to networks, it is a broader field in unsupervised machine learning which deals with multiple attribute types. On the other hand, community detection is specially tailored for network analysis which depends on a single attribute type called edges. Also, clustering algorithms have a tendency to separate single peripheral nodes from the communities it should belong to. However, both clustering and community detection techniques can be applied to many network analysis problems and may raise different pros and cons depending on the domain. Community detection is widely used in various fields such as biology, sociology, and marketing to understand social structures, reveal user data within the network, and develop relevant recommendation systems. There are several approaches to community detection, including hierarchical clustering, spectral clustering, modularity maximization, and random walk methods. However, they can be broadly categorized into two types; Agglomerative Methods and Divisive Methods. In Agglomerative methods, edges are added one by one to a graph which only contains nodes. Edges are added from the stronger edge to the weaker edge. Divisive methods follow the opposite of agglomerative methods. In there, edges are removed one by one from a complete graph. Ever since the discovery of community structure in real-world networks, a plethora of techniques devoted to their detection has been introduced. The challenge is both theoretical, in proposing a good mathematical definition of what constitutes a community, and computational, in developing good heuristics that can detect communities in a reasonable time.

3. Partitioning quality evaluation

A common way of investigating the community structure of networks starts with the definition of a quality function, which assigns a score to any network partition. Larger scores correspond to better partitions, and algorithms are created to find the partition with the largest score.

A good community partition of a network should have fewer edges between communities than one would expect by chance. This indicates significant community structure within the network. On the other hand, if the number of edges between groups is what would be expected by chance, it is not evidence of significant community structure. The measure known as modularity quantifies this idea that true community structure corresponds to a statistically surprising arrangement of edges, with fewer edges between groups and more edges within groups than expected by chance.

3.1 Modularity

Modularity Q is one of the most used measure. The modularity Q is, up to a multiplicative constant, the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random. The modularity can be either positive or negative, with positive values indicating the possible presence of community structure. Thus, one can search for community structure precisely by looking for the divisions of a network that have positive, and preferably large, values of the modularity

The original idea of modularity was given by Newman and Girvan, they have defined modularity Q as:

$$Q = \frac{1}{2}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})\delta(\sigma_i, \sigma j)$$

Here, m is the number of links, k_i is the degree of vertex i, k_j is the degree of vertex j, σ_i is the community to vertex i, σ_j is the community to vertex j, and $\delta(\sigma_i \sigma_j) = 1$ if i and j belong to the same community, otherwise it equals to 0. An alternate formulation of this is as a sum over communities:

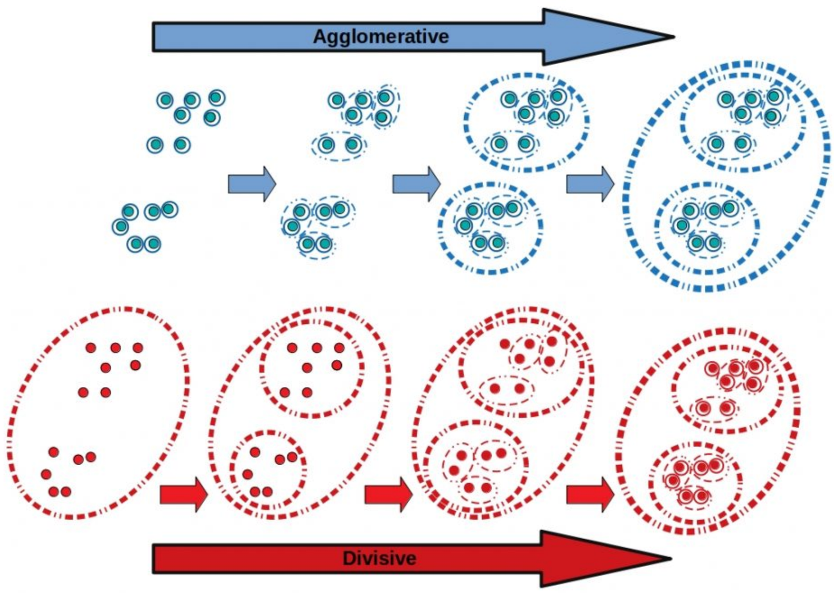
$$Q = \frac{1}{2m}\sum_{c}(A_c - \frac{K_c^2}{4m})$$

where m_c is the number of internal edges (or tatal internal edge weight) of community c and $K_c = \sum_{i|\sigma_i=c} k_i$ is the total (weighted) degree of nodes in community c.

III. Approaches

1. Community Detection Methods in Network Analysis: Agglomerative and Divisive Approaches

Community analytics, an essential task in network analysis, involves the identification of groups or communities within a network structure. Two primary types of methods are commonly employed for community detection: agglomerative methods and divisive methods.



In agglomerative methods (illustrated on the upper Figure 1), the process initiates with individual nodes and gradually merges them based on their similarity or connectivity, leading to hierarchical clustering.

This approach merges nodes from the bottom up, progressively forming larger communities. The agglomerative category includes algorithms such as Louvain and Leiden, which utilize modularity and hierarchical clustering.

Modularity, represented as Q, measures community strength and guides the algorithm. Higher modularity values indicate better communities, while a value below 1 suggests that each node is treated as a separate community.

The dendrogram in Figure 1 represents the hierarchy of clusters generated by hierarchical clustering. On the other hand, divisive methods (depicted on the bottom of Figure 1) start with a single partition containing all nodes and iteratively split it by removing edges with low similarity.

This process, known as the split process, aims to find smaller, more cohesive communities by progressively dividing the network. Although agglomerative and divisive methods employ different approaches, both ultimately aim to unveil meaningful groups of nodes based on their connectivity patterns within the network.

2. Clauset-Newman-Moore greedy modularity maximization

The Clauset-Newman-Moore (CNM) greedy modularity maximization algorithm is a popular method used in community detection, a field of network analysis. It is designed to partition a given network into communities, where members of each community are more densely connected to each other compared to members of different communities. The CNM algorithm aims to maximize the modularity of the network, which is a measure of the quality of the community structure.

The algorithm was proposed by Aaron Clauset, Mark Newman, and Cristopher Moore in 2004, and it has since become widely adopted due to its simplicity and effectiveness. The CNM algorithm is agglomerative method and follows a greedy approach, iteratively merging and splitting communities to optimize the modularity score.

Network representation: Considering an undirected network/graph with N nodes and E edges. We can represent the network as a graph adjacency matrix A of size N x N, where A[i][j] = 1 if there is an edge between nodes i and j, and A[i][j] = 0 otherwise.

$$\mathrm{A}_{vw} = \begin{cases} 1, & \text{if vertices } v \text{ and } w \text{ are connected,} \\ 0, & \text{otherwise.} \end{cases}$$

Community Structure: We represent the community structure of the network using a partition vector s of length N, where s[i] denotes the community membership of node i. Each node belongs to exactly one community.

The fraction of edges that fall within communities is given by:

$$\frac{1}{2m}\sum_{vw} A_{vw}\delta(c_v, c_w)$$

where:

- $\sum_{vw} A_{vw}$ represents the summation over all pairs of vertices v and w, considering the elements of the adjacency matrix A. It computes the total number of edges in the graph, denoted as m.

- $\delta(c_v, c_w)$ is the Kronecker delta function, which takes the value 1 if the community assignments of vertices v and w (cv and cw, respectively) are the same, and 0 otherwise.

- $\frac{1}{2m}$ is a normalization factor, dividing by the total number of edges to ensure the measure falls within the range [0, 1].

However, this measure alone is not sufficient to evaluate the quality of a community structure. It reaches its maximum value of 1 in the trivial case where all vertices belong to a single community. Additional measures like modularity is considered for a comprehensive assessment of community structure.

Modularity: The probability of an edge existing between vertices v and w, assuming random connections while respecting vertex degrees, is given by $\frac{k_v k_w}{2m}$. Here, k_v represents the degree of vertex v, k_w represents the degree of vertex w, and m is the total number of edges in the graph.

Based on this probability, we define the modularity Q as follows:

$$Q = \sum_{s=1} m \left[\frac{l_s}{L} - \left(\frac{d_s}{2L} \right)^2 \right] \quad (1)$$

(m = number of modules, ls = the number of edges inside module s, L = the number of edges in the network, d = total degree of the nodes in module s)

This process of removing an edge and calculating the modularity is iteratively repeated. The algorithm will stop when the new modularity is no longer greater than the modularity from the previous iteration. The ending modularity is usually around 0.6.

2.1 Pseudo Code

Algorithm 1 Clauset-Newman-Moore (CNM) Algorithm

Input: Network graph

Output: Community structure

```
Initialization: Assign each vertex to its own community
Calculate the initial modularity value Q_initial
repeat
    foreach pair of communities C_1 and C_2 do
        Calculate the change in modularity ΔQ by merging C_1 and C_2
    end
until no further improvement in modularity is possible;
Find the pair (C_1, C_2) with the maximum increase in modularity ΔQ_max
Merge communities C_1 and C_2  Update the modularity value Q by adding ΔQ_max
 return Community structure
```

2.2 Complexity

Since the joining of a pair of communities between which there are no edges at all can never result in an increase in Q, we need only consider those pairs between which there are edges, of which there will at any time be at most m, where m is again the number of edges in the graph.

The change in Q upon joining two communities is given by $\Delta Q = e_{ij} + e_{ji} - 2a_i a_j = 2(e_{ij} - a_i a_j)$, which can clearly be calculated in constant time. Following a join, some of the matrix elements e_{ij} must be updated by adding together the rows and columns corresponding to the joined communities, which takes worst-case time $O(n)$. Thus each step of the algorithm takes worst-case time $O(m + n)$.

There are a maximum of $n - 1$ join operations necessary to construct the complete dendrogram and hence the entire algorithm runs in time $O((m+n)n)$, or $O(n^2)$ on a sparse graph. The algorithm has the added advantage of calculating the value of Q as it goes along, making it especially simple to find the optimal community structure.

Each step: worst-case time $O(m + n)$.

A maximum of $(n - 1)$ steps to join n communities.

Thus: $O((m + n)n)$ or $O(n^2)$ for sparse graphs.

2.3 Pros and Cons

Pros

- Ease of implementation: Modularity Maximization (MM) is conceptually simple and can be implemented with relative ease compared to other algorithms. Several open-source libraries and software packages readily implement MM, making it accessible to a wide range of users.
- Scalability: MM efficiently scales to handle large networks with millions of nodes and edges. This makes it suitable for analyzing real-world networks like social media graphs, citation networks, and protein-protein interaction networks.

Cons

- Resolution limit: MM is sensitive to the resolution parameter, which controls the granularity of the detected communities. Choosing an appropriate resolution parameter can be challenging, as it can significantly impact the community structure. Small values tend to result in many small communities, while large values lead to a few large communities, potentially missing finer-grained structures.
- Merging similar clusters: MM can be biased towards merging similar clusters, even if they are not well-connected, to maximize the modularity score. This can lead to communities that are not cohesive or representative of the underlying network structure.

3. Louvain Algorithm

The Louvain algorithm is a widely used community detection algorithm that aims to identify communities or clusters within a network graph. It was developed by Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre in 2008.

The algorithm is known for its efficiency and ability to handle large-scale networks. It follows a greedy optimization approach that iteratively improves the modularity measure of the network by merging and rearranging communities. It can be used to analyze a network of 2 million nodes in only 2 minutes

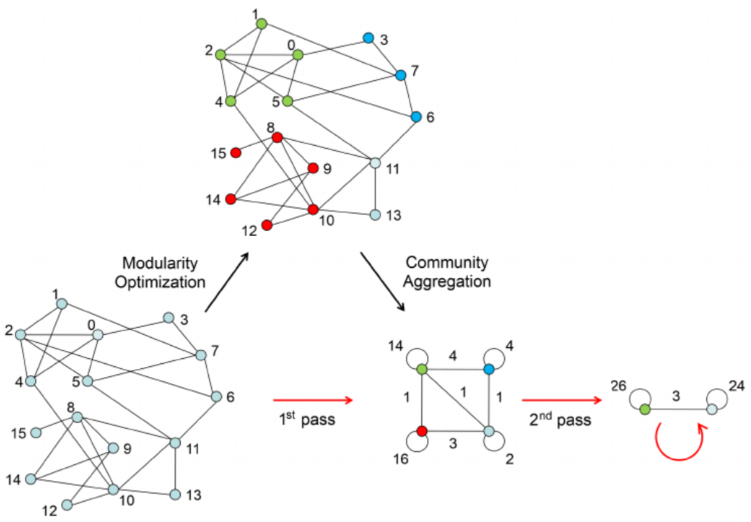


Figure 4: Louvain Algorithm

3.1 Modularity Optimization

As depicted in the figure, the first step is to optimize the modularity of the entire graph. In this example, it splits the nodes into four communities.

To find these clusters, each node is moved into its neighboring community. If the change in modularity (ΔQ) is greater than 0, it is moved into the neighboring community. Otherwise, it remains in its current community. This process is repeated until $\Delta Q = 0$ for all nodes.

3.2 Community Aggregation

After the optimization of modularity, super nodes are introduced to represent each cluster. Following the initial phase of the algorithm, numerous communities are formed.

Nevertheless, the two phases continue to iterate, resulting in the formation of larger and larger communities. The algorithm terminates only when neither of the two operations can further enhance the community structure.

3.3 Pseudo Code

Algorithm 2 Louvain Algorithm

Input: Graph $G = (V, E)$

Output: Community structure of the graph

Initialize each node as a separate community
repeat
    for each node $v \in V$ **do**
        Remove v from its current community
        Calculate the modularity gain ΔQ for each neighboring community by moving v
        Move v to the neighboring community with the highest ΔQ
    end
    Construct the aggregated network where each community is represented by a node
    Update the original graph with the new community assignments
until No further improvement in modularity;

3.4 Pros and Cons

Pros of the Louvain algorithm:

- Fast and scalable for large networks.
- Optimizes modularity, which measures community quality.
- Detects hierarchical community structures.
- Allows flexibility in resolution for different levels of detail.
- Widely used with extensive resources and documentation.

Cons of the Louvain algorithm:

- Resolution limit may merge small communities into larger ones.
- Results can be sensitive to initial community assignments.
- Assumes non-overlapping communities.
- Bias towards detecting large, cohesive communities.
- Lacks theoretical guarantees of optimality.

4. Label Propagation Algorithm

4.1 Intuition

As we will show, the advantage of this algorithm over the other methods is its simplicity and time efficiency. The algorithm uses the network structure to guide its progress and does not optimize any specific chosen measure of community strengths.

4.2 Pseudocode

1. Initialize the labels at all nodes in the network. For a given node x, $C_x(0) = x$.

2. Initialize $t = 1$.

3. Arrange the nodes in the network in a random order and set it to X.

4. For each x in X chosen in that specific order,
   let $C_x(t) = f(C_{x_{i1}}(t), \ldots, C_{x_{im}}(t), C_{x_{i(m+1)}}(t-1), \ldots, C_{x_{ik}}(t-1))$.
   f here returns the label occurring with the highest frequency among neighbors and ties are broken uniformly randomly.

5. If every node has a label that the maximum number of their neighbors have, then stop the algorithm. Else, increment t and go to 3.

4.3 Complexity

It takes a near-linear time for the algorithm to run to its completion Initializing every node with unique labels requires $O(n)$ time. Each iteration of the label propagation algorithm takes linear time in the number of edges $O(m)$.
At each node x, we first group the neighbors according to their labels $O(d_x)$. We then pick the group of maximum size and assign its label to x, requiring a worst-case time of $O(d_x)$. This process is repeated at all nodes and hence an overall time is $O(m)$ for each iteration.

4.4 Pros and Cons

Pros

- Efficiency: Label propagation is computationally efficient, making it suitable for large networks. Its running time is generally faster compared to some other community detection algorithms.

- Low A Priori Information Requirement: One of its notable strengths is its low dependency on prior information about the network structure. You don't need to specify parameters beforehand, which can be advantageous in scenarios where the network characteristics are not well-known.

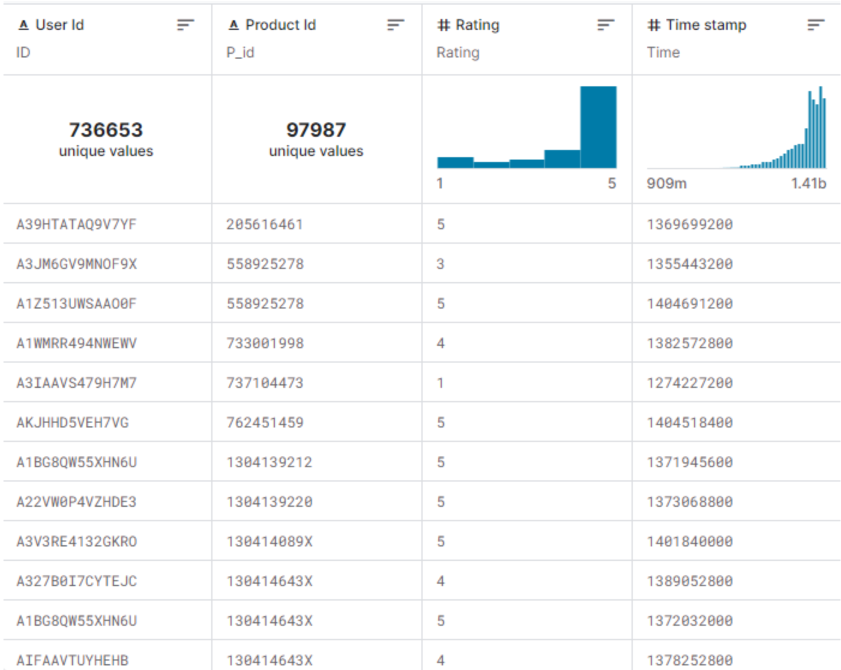
Cons

- Lack of Unique Solutions: As you mentioned, the algorithm doesn't guarantee a unique solution. This can be a drawback if you're looking for a single, definitive community structure. The results can vary across different runs of the algorithm.

- Aggregate Solutions: The algorithm provides an aggregate of multiple solutions, leading to a lack of specificity. This might be undesirable in situations where a precise and unique community structure is crucial.

IV. Experiments

We have proceed to do an evaluation for experiments part.
First, we choose the dataset from Kaggle which is **Customer Rating Data By Amazon**. It can be found via `https://www.kaggle.com/datasets/ahmedaliraja/customer-rating-data-by-amazon/data` .Let's have the overview about dataset:



User Id ID	Product Id P_id	Rating Rating	Time stamp Time
736653 unique values	97987 unique values	1 – 5	909m – 1.41b
A39HTATAQ9V7YF	205616461	5	1369699200
A3JM6GV9MNOF9X	558925278	3	1355443200
A1Z513UWSAAO0F	558925278	5	1404691200
A1WMRR494NWEWV	733001998	4	1382572800
A3IAAVS479H7M7	737104473	1	1274227200
AKJHHD5VEH7VG	762451459	5	1404518400
A1BG8QW55XHN6U	1304139212	5	1371945600
A22VW0P4VZHDE3	1304139220	5	1373068800
A3V3RE4132GKRO	130414089X	5	1401840000
A327B0I7CYTEJC	130414643X	4	1389052800
A1BG8QW55XHN6U	130414643X	5	1372032000
AIFAAVTUYHEHB	130414643X	4	1378252800

It comprises an extensive collection of over 2 million customer reviews and ratings of beauty-related products available on Amazon's platform. The dataset includes valuable information such as:

- Unique User IDs for customer identification.
- Product ASIN (Amazon's distinctive product identifier).
- Ratings, which reflect customer satisfaction on a scale from 1 to 5.
- Timestamps, recorded in UNIX time, indicating when the ratings were submitted.

We did a small test on running 3 different algorithms and have the result as below:

For the Mean Absolute Error (MAE) - The Mean Absolute Error (MAE) values provided for three different algorithms (Louvain, Label prop, and Greedy modularity) represent the average absolute differences between the predicted values and the actual values. :

```python
for algorithm in results.keys():
    gdown.download(
        f"https://drive.google.com/uc?id={results[algorithm]}",
        "algorithm_result_df.parquet",
    )

    predict_result_df = pd.read_parquet("algorithm_result_df.parquet")
    predict_result_df = predict_result_df[
        predict_result_df["predicted_overall"].isna() == False
    ]

    df_evaluate = pd.merge(
        predict_result_df, df_test, on=["reviewerID", "asin"], how="inner"
    )
    df_evaluate["predicted_overall"] = df_evaluate["predicted_overall"].astype(
        int
    )
    df_evaluate.head(2)

    ground_truth = df_evaluate.overall.values
    predicted_values = df_evaluate.predicted_overall.values

    mae = mean_absolute_error(ground_truth, predicted_values)

    print(f"Mean Absolute Error (MAE) from algorithm {algorithm}: {mae}")
```

Figure 5: Evaluation code with 3 different algorithms

+ Louvain Algorithm: 1.3076923076923077
This algorithm has a moderate MAE, suggesting that its predictions have a moderate level of error on average.

+ Label prop Algorithm: 1.75
This algorithm has a higher MAE compared to the Louvain Algorithm, indicating that, on average, its predictions have a larger error.

+ Greedy modularity Algorithm: 1.125984251968504
This algorithm has the lowest MAE among the three, suggesting that it provides more accurate predictions compared to the other two algorithms in your test.

In summary, based on the MAE values, the Greedy modularity Algorithm seems to perform the best among the three algorithms you tested, as it has the lowest MAE.

V. Conclusion

1. Achievement

The algorithms that the group used to solve the community search problem for the artist dataset above accomplished the following:

- Our team have implemented and runned 3 algorithms (Girvan Newman, Louvain, Leiden) on Python, from that, we can see some comparisions and how they works.
- Using the above data clusters, we can suggest similar books to the user while they are seeing the information of the books they want to buy, increasing the diversity of the e-commerce website such as Tiki.
- Analyzing the tastes of the readers community in specific clusters to promote better suggestions and apply them to run ads or PR is another strength that this algorithm brings.
- Besides, we also build a website to demonstrate our work and its application. The demonstration video of the website is here: `https://youtu.be/nnhuGfggjb4`

2. Drawback

Because of the limited research time, the group did not generate many algorithms or make very specific comparisons with the algorithms sought.
However, with the above three algorithms,Girvan-Newmana and Louvain, classical algorithms, and Leiden, an improved algorithm widely used today, the group has also made appropriate comparisons of performance and reliability to generalize them.

3. Future work

The team plans to use more algorithms in the future to get a thorough understanding of the approaches used by businesses and researchers to address problems.

The team also intends to work with more actual data sets in order to fully assess the algorithm’ s efficacy and its potential for use in other contexts. This deals with, for instance, developing techniques for providing suggestions or advertising using these data clusters.

References.

[1] Dataset source: https://www.kaggle.com/datasets/arashnic/book-recommendation-dataset

[2] Lilian Weng, Filippo Menczer Yong-Yeol Ahn (2013). "Virality Prediction and Community Structure in Social Networks". Last access 26/05/2023 https://www.researchgate.net/publication/256188948_Virality_Prediction_and_Community_Structure_in_Social_Networks

[3] Fabian Nguyen (2021). "Leiden-Based Parallel Community Detection". Last access 26/05/2023 https://i11www.iti.kit.edu/_media/teaching/theses/ba-nguyen-21.pdf

[4] How Authors Can Find Their Ideal Reading Audience. Last access 26/05/2023 https://www.janefriedman.com/how-authors-can-find-readers/