Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 569–576
DOI: 10.15439/2022F64
ISSN 2300-5963 ACSIS, Vol. 30

An Integer Programming Approach Reinforced by a Message-passing Procedure for Detecting Dense Attributed Subgraphs

Arman Ferdowsi
Vienna University of Technology-ECS Group
K. N. Toosi University of Technology-Department of Mathematics
Email: aferdowsi@ecs.tuwien.ac.at
armanferdowsi@email.kntu.ac.ir

***Abstract*—One of the recent challenging but vital tasks in graph theory and network analysis, especially when dealing with graphs equipped with a set of nodal attributes, is to discover subgraphs consisting of highly interacting nodes with respect to the number of edges and the attributes' similarities. This paper proposes an approach based on integer programming modeling and the graph neural network message-passing manner for efficiently extracting these subgraphs. The experiments illustrate the proposed method's privilege over some alternative algorithms known so far, utilizing several well-known instances.**

***Index Terms*—Graph partitioning, Network analysis, Integer programming, Message passing, Local search.**

I. Introduction

WHEN studying graphs/networks, we usually face collections consisting of nodes with common topological and nodal attribute characteristics. More specifically, sets of highly interactive vertices are likely to yield and share common relationships and properties. In this sense, in all scientific fields where graphs are somehow implicated, one of the challenging tasks would be to efficiently partition the given graph into a number of dense subgraphs consisting of massively connected vertices that share similar properties. These subgraphs are well known as *communities*, and naturally, the process of identifying them is referred to as the *community detection problem*. Detecting communities has become one of the fundamental subjects in the field of network analysis and graph theory and has numerous applications in a wide range of areas, including the analysis of Social/Biological/Cosmological networks [1], [2], [3], [4] and WEB [5]. It also plays a crucial role in the domain of Network Design problems [6], Signal Processing [7], Image Segmentation [8], Pattern Recognition [9], and Data Mining [10].

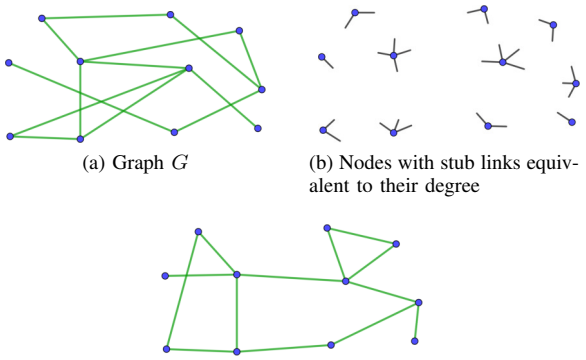
Crucial in this domain is a more formal representation of a graph: A pair $G = (V, E)$ with the set of vertices V and edges E. Subsequently, from the perspective of graph topological structure, a *community* can be contemplated as a subset $C \subseteq V$ with a high density of edges between nodes inside C and a low density of edges connecting C to the other subsets. Accordingly, one can define the community detection problem as *partitioning* V into a set of disjoint communities $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$.

In fact, mining high-quality communities usually coincides with finding a measure that estimates the goodness of communities. In the literature, a large number of such quality measures have been proposed for evaluating the superiority of partitioning from the viewpoint of topological structures. Among them, one of the most widely used and well-known is *Modularity*, introduced by Newman [11]. Intuitively, for a community C, Modularity is the number of edges inside C subtracted by the expected number of such edges, whereat the expected number of edges can be derived by corresponding to G, a randomized graph (called the *Null model*) with exactly the same vertices and the same degree of G, in which edges are placed randomly. More specifically if d_i and m are, respectively, the degree of node i and the number of edges in G, the probability of existing an edge between nodes i and j in such a graph is $\frac{d_i d_j}{2m}$. This is because, first, each node is assigned the number of stub links exactly equal to its edges (Fig. 1a, 1b). Afterward, each of the two stub edges will be joined at random (Fig. 1c). Consequently, the Modularity value for a community C can be defined as the number of edges within C in G minus the number of edges within C in a Null model of G. Thus, Modularity for partitioning $\mathbf{C}$ can then be expressed as

$$Q(\mathbf{C}) = \frac{1}{2m} \sum_{i,j \in V} [a_{i,j} - \frac{d_i d_j}{2m}] \xi(i,j), \tag{1}$$

where $A = (a_{i,j})$ be the adjacency matrix of G, where $a_{i,j}$ is one when there is an edge between node i and node j, and zero otherwise; n is the number of vertices in G. In addition, $\xi(i,j)$ is one if i and j are in the same community and zero otherwise.

As a result, high-quality communities can be determined as the ones with a high value of Modularity. However, despite Modularity's advancements in finding high-quality communities in a wide range of graphs, it is known to suffer from limitations (see [12], [13], for example). In particular, as pointed out in [14], since Modularity only considers the existing edges of the network, it qualifies the goodness of the discovered communities by only measuring how good the partitioning fits the existing edges. This is indeed a drawback



(a) Graph G

(b) Nodes with stub links equivalent to their degree

(c) A null model of G

Fig. 1: A Null model associated with a given Graph G.

because the disconnected nodes (absent links) that lie in the same community are not taken into account.

On the other hand, *Max-Min Modularity* [14], as one of the successful extensions of Modularity, is able to significantly improves the accuracy of the measure by compensating for the Modularity quantity when disconnected nodes are in the same community. More precisely, it is assumed in [14] that (in addition to the graph G) a zero-one *relation matrix* $U = (u_{i,j})$ is given that defines whether every pair of disconnected nodes of the network is related or not: $u_{i,j}$ is one when disconnected nodes i and j are *related*, and zero otherwise. Max-Min Modularity, in fact, tries to take into account the importance of the *indirect* connections between disconnected nodes by penalizing the Modularity measure when *unrelated* nodes are in the same community: Consider a complemented graph $G' = (V, E')$, where E' contains an edge between every pair of disconnected nodes of G that is unrelated; i.e., there is an edge between i and j in G' if there is not such an edge in G and also $u_{i,j}$ is zero. Max-Min Modularity, then, aims at maximizing the Modularity in G as well as minimizing Modularity in G' simultaneously. Mathematically speaking, if $A' = (a'_{i,j})$ is the adjacency matrix of G', d'_i is the degree of node i in G', and m' is the number of the edges in G', then Max-Min Modularity Q_{MM} of a given partition $\mathbf{C}$ of V is defined as follows:

$$Q_{MM}(\mathbf{C}) = \sum_{i,j \in V} [\frac{1}{2m}(a_{i,j} - \frac{d_i d_j}{2m}) - \frac{1}{2m'}(a'_{i,j} - \frac{d'_i d'_j}{2m'})]\xi(i,j). \quad (2)$$

We refer to the problem of finding a partition of the network that maximizes Max-Min Modularity as the *Max-Min Modularity Maximization* problem.

As a crucial remark, we can note that not only is the Max-Min Modularity Maximization problem categorized in the class of NP-hard problems [15], making it hard to find an efficient optimization algorithm, but it also suffers from a major drawback: Max-Min Modularity strongly depends on the accuracy of the given relation matrix, meaning that the quantity of the measure might be heavily affected by the node relationships defined by the user/oracle in the first place. Despite Ferdowsi and Khanteymoori [16] succeeded in proposing a systematic approach for unveiling the relation between non-adjacent vertices, from a more critical point of view, one could argue that both Modularity and Max-Min Modularity, like many other community discovery methods, only utilize topological information of nodes, naively overlooking a rich set of nodal attributes (e.g., user profiles of an online social network or textual contents of a citation network), which is abundant in all real-life networks [17].

In this regard, recently, an emerging domain of deep learning for graphs has ensured the design of more accurate and scalable algorithms. However, despite these approaches achieving outstanding results in graph-related tasks like link prediction and node classification [18], fairly little concentration has been committed to their application on unsupervised learning that encompasses the community detection problem. This is primarily because graph embedding methods can ideally align with (semi-)supervised learning approaches; however, they cannot be naturally generalized to the unsupervised learning manners since it is not very simple to find a proper loss function to govern the back-propagation updating procedure. Despite that, several deep learning-based unsupervised graph algorithms have been proposed [19], [20], [21], but they all suffer from a not accurate choice of a loss function that can authoritatively extend the model to unsupervised vision. The diverse results obtained by these approaches over the same input instances can prove this claim.

Main contribution: In this work, we introduce a refined model for the Max-Min Modularity that empowers us to find high-quality communities with simultaneously taking the graph's topological structure and the nodal attributes into account. In the sequel, we involve the Modularity metric to mine the communities consisting of densely connected vertices. In addition, we provide a technical introduction to *Graph Neural Network (GNN)* formalism, a dominant and fast-growing paradigm for deep learning with graph data, whose ability is to use nodes' features and local structure to generate embeddings. Utilizing the general concept of GNN feed-forward message passing, we devise an efficient mechanism for extracting the information induced by propagating nodes' properties throughout the network, leading to a perfect systematic characterization of the relation matrix. Everything combined, we introduce the advanced Max-Min Modularity scheme and express it with a standard integer programming formulation. Ultimately, by solving the model using a robust approach consisting of a row/column generation technique for solving the model's linear relaxation version and a local search manner for obtaining integer solutions, we identify the final communities.

The paper is organized as follows: the rest of this section focuses on providing a brief literature review. In Section II, we first restate the Modularity Maximization problem in terms of an integer programming model. We then devise a method for providing a refined relation matrix. Afterward, gathering all things together, we extend the primary Max-Min Modularity model and present an integer programming formulation for

it. Next, in Section III, we introduce the employed solution approach that leads to discovering high-quality communities. Section IV eventually focuses on various experiments, confirming the proposed method's high performance.

A. Related Works

Several approaches have been proposed in the literature to detect communities in networks; see, for example, the survey conducted by Souravlas et al. [22]. However, despite a large number of these techniques, relatively little work solves the problem using mathematical programming techniques. Especially in the case of Modularity maximization, few results have been established [23], [24], [25], [26]. On the other side, Chen et al. [14] illustrated that maximizing the Modularity alone does not usually lead to superior communities. Based on their findings, one of the significant drawbacks of Modularity is its sheer dependence on the (existing) links, meaning that, Modularity only focuses on discovering communities in which the number of interactive edges (links) is as many as possible. At the same time, it does not pay any attention to the missing links. This is while just as existing links can play an essential role in analyzing networks, so do the absent links. Therefore, new extensions of Modularity emerged subsequently, one of the most successful of which is the already mentioned: Max-Min Modularity [14]. Nevertheless, as discussed, Max-Min Modularity itself suffers from a critical issue: the nonexistence of a systematic way for proposing the so-called relation matrix, which is required to express the relationship between non-adjacent nodes. In this respect, Ferdowsi and Khanteymoori [16] succeeded in offering an analytical procedure to address this deficiency, generalize the conventional Max-Min Modularity, and provide an efficient local search-based algorithm to discover high-quality communities by considering both existing and missing links.

On the other side, in the past few years, most research has surged toward deep learning methods due to their power to achieve unprecedented results. These techniques aim at embedding nodes into a low-dimensional, dense vector space [27], [28]. However, unfortunately, a vast number of these methods lack the strength of encountering attributed graphs in which nodes are equipped with a set of features, and this is while we are now most surrounded by attributed networks everywhere [19]. It is worth mentioning that a few deep-learning methods have been recently proposed that consider attributed network embedding [29], [30], though most of them employ a matrix factorization manner, which endures some critical boundaries. More precisely, the representation capability of a matrix factorization-based approach is found to be more inadequate than a neural network-based method [31]. Besides, one could also argue that the majority of these proposals only rely on supervised graph algorithms, and therefore, usually fail to perform the community detection task, which can be categorized as an unsupervised assignment in graph problems [32], [33]. And the rests, which are designed for unsupervised learnings, still cannot find a promising loss function that can lead to fine communities for various given graph instances [34], [19].

II. Model Sketching

In this section, we first recite the so-called Modularity Maximization problem in terms of an integer programming formulation, enabling us to discover communities with respect to the topological aspects of a given graph. Afterward, we explain the Graph Neural Network message passing method that facilitates us to devise a procedure for determining an accurate relation matrix for the Max-Min Modularity problem. This achievement then hopefully leads to a proper integer formulation for our advanced Max-Min Modularity problem that can be used to efficiently capture communities with respect to simultaneously taking both topological and attributes aspects into consideration.

A. Topology extraction

Let the binary variable x_{ij} indicate if nodes i and j belong to the same community or not; the value of x_{ij} is zero if nodes i and j belong to the same community, and one otherwise. Let $I_{all} = \{(i,j) \in V^2 \mid i < j\}$; and $q_{ij} = a_{i,j} - \frac{d_i d_j}{2m}$, for each $(i,j) \in I_{all}$. As described in [25], the Modularity Maximization problem can be formulated in terms of the following integer linear program:

$$\max \quad \frac{1}{m} \sum_{(i,j)\in I_{all}} q_{ij}(1 - x_{ij}) \qquad \text{(IP-M)}$$

$$x_{ij} + x_{jk} - x_{ik} \geq 0 \qquad \forall i < j < k \qquad (3)$$

$$x_{ij} - x_{jk} + x_{ik} \geq 0 \qquad \forall i < j < k \qquad (4)$$

$$-x_{ij} + x_{jk} + x_{ik} \geq 0 \qquad \forall i < j < k \qquad (5)$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in I_{all} \qquad (6)$$

Constraints (3)-(5) guarantee that if i and j are in the same community and j and k are in the same community, then so are i and k. We refer to the relaxation of (IP-M), obtained by replacing the constraints $x_{ij} \in \{0,1\}$ by $x_{ij} \in [0,1]$, as (LP-M).

As discussed in [23] and [35], solving IP-M can soundly provide us with communities consisting of highly interactive edges. It is, however, incontrovertible that maximizing Modularity cannot help us tackle the attributed graphs. Consequently, to address this deficiency, our goal is to design an advanced model for the Max-Min Modularity problem so that nodal attributes are also effectively involved in the community mining process. In order to do that, we first provide a way to exploit the information diffused via nodes' features.

B. Attribute extraction

In this section, which establishes the core part of this research, we utilize the *Graph Neural Network (GNN) message-passing* framework to provide a systematic and accurate way of defining a *relation matrix* that perfectly depicts the similarity between nodes by analyzing the information spread by the attributes. More precisely, we aim to feed the nodal attributes to a GNN message passing to create single representation

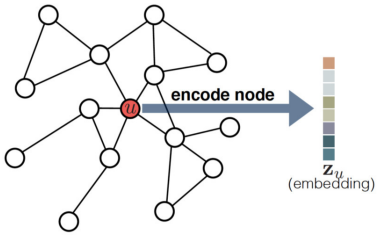


Fig. 2: A scheme of the encoder approach. The encoder maps the node u to a low-dimensional embedding z_u.

vectors, each of which captures a node's structure as well as the feature information.

To motivate our discussion, we initially raise the notion of *node embedding*, which seeks to encode nodes as low-dimensional vectors that summarize their structural graph position and the information of their local graph neighborhood. In other words, node embedding aims to project vertices into a latent space, where geometric relations in this latent space correspond to relationships (e.g., edge existence or similarity) in the original graph or network. In this fashion, an encoder can be referred to as a function that maps each node $u \in V$ to vector embedding $z_u \in \mathbb{R}^d$ (i.e., z_u corresponds to the embedding for node $u \in V$) (See Fig. 2).

We now turn our attention toward one of the substantial encoder models: *Graph Neural Network (GNN)*, a broad framework for defining deep neural networks on graph data. The elucidative characteristic of a GNN is that it adopts a form of neural message passing in which vector messages are exchanged between nodes and updated using neural networks [36]. In each message-passing iteration of a GNN, a hidden embedding $h_u^{(k)}$ corresponding to each node $u \in V$ is updated according to information aggregated from u's graph neighborhood $\mathcal{N}(u)$. Informally speaking, to each node $u \in V$ one can correspond a so-called *computational graph* that accumulates the information propagated from u's neighbors, and in turn, the messages coming from these neighbors are based on information aggregated from their respective neighborhoods, and so on. Fig. 3 exemplifies a two layers computational graph. From the mathematical perspective, GNN message-passing procedure can be expressed as follows. Suppose that each node u is associated with a d-dimensional attribute vector that we represent with $X_u \in \mathbb{R}^d$. Then, the $k-$th embedding layer $h_u^{(k)}$, corresponding to node $u \in V$, can be obtained by the following recursive formula:

$$h_u^{(k)} = \sigma\big(W^{(k)} \sum_{v \in \mathcal{N}(u)} h_v^{(k-1)} + b^{(k)}\big), \tag{7}$$

where $h_u^{(0)} = X_u$ and $W_{d\times d}$ is a trainable matrix, which weights the nodes' attributes. Moreover, let σ denotes an elementwise non-linearity (e.g., a *tanh* or *Relu*). Furthermore, $b^{(k)} \in \mathbb{R}^d$ is the bias term, which can be often omitted for the sake of simplicity, but including it could be important to obtain high-quality performance. As can be inferred from (7), first, the messages incoming from the neighbors are summed;

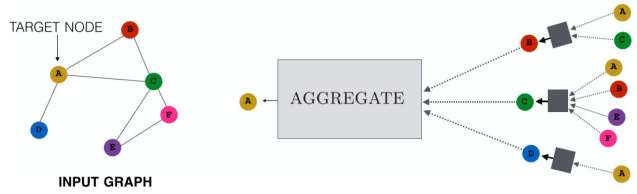


Fig. 3: Two layers computational graph corresponding to node A. The model aggregates messages from A's local graph neighbors, and in turn, the messages coming from these neighbors are based on information aggregated from their respective neighborhoods.

then, the neighborhood information with the node's previous embedding is combined using a linear combination; finally, an elementwise non-linearity is applied.

Now, in order to transform the node-level equation (7) into something we can implement, we can come up with the following graph-level definition of the model:

$$H^{(k)} = \sigma\big(AH^{(k-1)}W^{(k)}\big), \tag{8}$$

where $H^{(k)} \in \mathbb{R}^{|V|\times d}$ is the matrix of node representations at layer k in the GNN, with each node corresponding to a row in the matrix.

However, despite the straightforward intuition behind the update procedure (8), considerably notable is its two limitations. The first one is the non-consideration of the attributes of each node itself. This is crucial since the effectiveness of GNN becomes severely limited, as the information coming from the node's neighbors cannot be differentiated from the information from the node itself. This restriction originates form the fact that self-loops are not taken into account in the adjacency matrix A. The problem, however, can be easily resolved by substituting A with $A + I$, where $I \in \mathbb{R}^{n\times n}$ is the identity matrix that lets the self-loops also be involved. Another issue that may raise concern regarding (8) is the non-normality of features, which can indeed lead to numerical instabilities. However, fortunately, to also prevent this shortcoming, it seems convincing to apply the symmetric normalization as it has turned out to drive powerful dynamics [37]. For doing this, it is sufficient to replace $A + I$ with the normalization term $D^{\frac{-1}{2}}(A + I)D^{\frac{1}{2}}$, where $D \in \mathbb{R}^{n\times n}$ is the degree matrix. As a result, in the case of a basic GNN, we can end up with the following graph level definition of the model:

$$H^{(k)} = \sigma\big(D^{\frac{-1}{2}}(A + I)D^{\frac{1}{2}}H^{(k-1)}W^{(k)}\big) \tag{9}$$

It is worth pointing out that since our goal is to employ the feed-forward message passing exclusively and not to implement the back-propagation procedure, training the weight matrix W is not specifically part of our requirements. Instead, the only thing that matters to us is finding a final vector representation for each node by which we can sufficiently reflect the similarities among nodes with respect to the information that originates/propagates from attributes. For this reason, we can

safely omit W from (9) and obtain the following embedding matrix H.

$$H^{(k)} = \sigma\big(D^{\frac{-1}{2}}(A+I)D^{\frac{1}{2}}H^{(k-1)}\big) \tag{10}$$

One important insight that could be gained by (10) is that GNN feed-forward message passing is capable of effectively encoding neighborhood information in such a way that after performing the updating procedure for a number of layers, similar nodes in the graph will tend to have analogous final embedding representation [38]. This is primarily due to the fact that each node inherits the information (attributes) from its neighborhood.

Accordingly, any techniques for computing the distance between each pair of final vectors representation could naturally lead to obtaining the similarities between the corresponding vertices with respect to their attributes. In this work, we employ the well-known *z-Normalized Euclidean Distance* to measure the similarities between nodes. In this fashion, the distance between two vectors is defined as the Euclidean distance between the normal form of the two vectors, where the normalized form associated with each sequence is obtained by transforming the vector so it has a mean distribution $\mu = 0$ and standard deviation $\sigma = 1$. More explicitly, given two nodes i and j, with the corresponding vector embeddings $H_i^{(k)} \in \mathbb{R}^d$ and $H_j^{(k)} \in \mathbb{R}^d$, at layer k, we define $x^*_{ij} = \frac{||\widehat{H_i^{(k)}} - \widehat{H_j^{(k)}}||_2}{2\sqrt{d}}$ to be their z-score normalized Euclidean distance, where $||\cdot||_2$ is the Euclidean norm, $\widehat{H_i^{(k)}} = \frac{H_i^{(k)} - \mu_{H_i^{(k)}}}{\sigma_{\mathbf{H_i^{(k)}}}}$, $\mu_{H_i^{(k)}}$ is the distribution mean, and $\sigma_{H_i^{(k)}}$ is the standard deviation of the vector $H_i^{(k)}$. It is not difficult to check that for any given $i, j \in V$, we have $x^*_{ij} \in [0,1]$ [39] and that x^* preserves the triangle inequality. Subsequently, x^* forms a metric space, which we denote by *Embedding Distance (ED)*, on graph G. Clearly, the larger the obtained ED between two nodes, the less similarity between them.

C. Advanced Max-Min Modularity Model

As already mentioned, the larger x^*_{ij} is, the less likely it is that i and j are similar, that is, the less correlated they are, and therefore, the more likely they are to be in distinct communities. This intuition and also the fact that the Modularity Maximization problem can be nicely expressed for weighted graphs [40] persuade us to propose an advanced Max-Min Modularity model by recharacterizing the relation matrix and so the complemented weighted graph $G' = (V, E')$ using ED. Accordingly, we define the relation matrix $A' = (a'_{i,j})$ and G' ($(a'_{i,j})$ represents the weight of the edge between nodes i and j in G') as follows:

$$a'_{i,j} = \begin{cases} x^*_{ij} & \text{if } a_{i,j} = 0 \text{ and } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Given a relation matrix $A' = (a'_{i,j})$, and noticing that in the induced metric ED, Constraints (3)-(5) guarantee the *triangle inequality*, for any $i, j, k \in V$, the advanced Max-Min Modularity Maximization problem can be formulated as the following IP:

$$\max \quad \sum_{(i,j)\in I_{all}} (\frac{q_{ij}}{m} - \frac{q'_{ij}}{m'})(1-x_{ij}) \qquad \text{(IP-MM)}$$
$$(3), (4), (5)$$
$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in I_{all},$$

where $q'_{ij} = a'_{i,j} - \frac{d'_i d'_j}{2m'}$, for each $(i,j) \in I_{all}$; $d'_i = \sum_{j=1}^{n} a'_{i,j}$, and $m' = \sum_{(i,j)\in I_{all}} a'_{i,j}$. We refer to the relaxation of IP-MM, obtained by replacing the constraints $x_{ij} \in \{0,1\}$ by $x_{ij} \in [0,1]$, as (LP-MM).

Considerably important is the fact that maximizing IP-MM coincides with simultaneously maximizing the modularity over the original given graph G and minimizing the modularity over the complemented graph G', determined by the proposed message-passing approach. Whereas the first component makes sure to return communities, each consisting of densely connected nodes, the latter attempts to extract the communities containing vertices with the most similar attributes possible.

III. Solution Approach

Efficiently solving (IP-MM) consists of two main parts: 1) optimally solving (LP-MM) and 2) accurately rounding the obtained fractional solutions to the integer ones.

For the first task, we use the row/column generation algorithm proposed in [16] that perfectly works for (LP-MM) as well since the two models are identical apart from using different relation matrices. A summary of the applied row/-column generation technique is as follows. First, consider the following sub-problem (LPs-MM($\mathcal{I}$)) of (LP-MM) consisting of all pairs in $I = \{(i,j) \in I_{all} \mid c_{ij} > 0\}$ and some pairs in $I' = \{(i,j) \in I_{all} \mid c_{ij} \leq 0\}$:

$$\max \sum_{(i,j)\in I} c_{ij}(1-x_{ij}) + \sum_{(i,j)\in\mathcal{I}\subset I'} c_{ij}(1-x_{ij}) \qquad \text{(LPs-MM}(\mathcal{I}))$$
$$x_{ij} + x_{jk} - x_{ik} \geq 0 \qquad \forall (i,j),(j,k),(i,k) \in I \cup \mathcal{I}, c_{ij} \geq 0 \vee c_{jk} \geq 0 \tag{12}$$
$$x_{ij} - x_{jk} + x_{ik} \geq 0 \qquad \forall (i,j),(j,k),(i,k) \in I \cup \mathcal{I}, c_{ij} \geq 0 \vee c_{ik} \geq 0 \tag{13}$$
$$-x_{ij} + x_{jk} + x_{ik} \geq 0 \quad \forall (i,j),(j,k),(i,k) \in I \cup \mathcal{I}, c_{jk} \geq 0 \vee c_{ik} \geq 0 \tag{14}$$
$$x_{ij} \in [0,1] \qquad \forall\, i<j,\ (i,j),(j,k),(i,k) \in I \cup \mathcal{I} \tag{15}$$

Be advised that (LPs-MM($\emptyset$)) generates the smallest formulation while (LPs-MM(I')) is equivalent to (LP-MM) itself. Moreover, point out that (LPs-MM($\mathcal{I}$)) delivers an upper bound of the optimal value of (LP-MM) that never gets worse by adding variables [16]. Above all, however, as Theorem 3.1 in [16] depicts, if an optimal solution $\bar{x^*} = (\bar{x^*_{ij}})_{(i,j)\in I\cup\mathcal{I}}$ to (LPs-MM($\mathcal{I}$)) satisfies the following condition $(*)$, then $(x^*_{ij})_{(i,j)\in I_{all}}$ is an optimal solution to (LP-MM), where

$$x^*_{ij} = \begin{cases} \bar{x^*_{ij}} & ;\ (i,j) \in I \cup \mathcal{I} \\ 1 & ;\ \text{otherwise} \end{cases} \tag{16}$$

and

$$(*) \begin{cases} \bar{x^*_{ij}} + \bar{x^*_{jk}} \geq 1;\ (i,j),(j,k) \in I \cup \mathcal{I}, c_{ij} \geq 0 \vee c_{jk} \geq 0, (i,k) \in I' - \mathcal{I} \\ \bar{x^*_{ij}} + \bar{x^*_{ik}} \geq 1;\ (i,j),(i,k) \in I \cup \mathcal{I}, c_{ij} \geq 0 \vee c_{ik} \geq 0, (j,k) \in I' - \mathcal{I} \\ \bar{x^*_{jk}} + \bar{x^*_{ik}} \geq 1;\ (j,k),(i,k) \in I \cup \mathcal{I}, c_{jk} \geq 0 \vee c_{ik} \geq 0, (i,j) \in I' - \mathcal{I} \end{cases}$$

This fact leads to the following efficient row/column generation procedure for optimally solving (LP-MM):

- Start solving (LPs-MM($\mathcal{I}$)) with $\mathcal{I} = \emptyset$ and adding those $\bar{x^*} \in I' - \mathcal{I}$ that violate inequalities in $(*)$ in each iteration, until an optimal solution to (LPs-MM($\mathcal{I}$)) satisfies $(*)$.
- In each repeat, employ a row generation technique for solving (LPs-MM($\mathcal{I}$)):
  1) Obtain an optimal solution $\bar{x^*}$ by solving (LPs-MM($\mathcal{I}$)) with no constraints.
  2) verify whether all constraints of (LPs-MM($\mathcal{I}$)) are satisfied by $\bar{x^*}$. If not, add the violated ones and solve (LPs-MM($\mathcal{I}$)).
  3) Update $\bar{x^*}$ and repeat step (2).
- By having the optimal solution $\bar{x^*}$ to (LPs-MM($\mathcal{I}$)), determine the optimal solution x^* to (LP-MM) by Equation (16).

To accomplish the second task, we can again apply the rounding algorithm proposed in [16], which can be shortly expressed as follows. Point out that the fractional optimal solution to (LP-MM) provides us with a metric space, where the distance between every pair of nodes is at most one. Let us call this metric *LP* distance. It is apparent that in an integral solution, the distance between each pair of nodes is either zero (implying that these two nodes belong to the same community) or one (inferring that these two nodes belong to separate communities). The rounding task is to push (some of) the nodes so as to determine a final valid *configuration* of the points in which the distance between every pair of nodes is either zero or one. Obviously, such a valid configuration correlates with a feasible integral solution $\hat{x}$ to (IP-MM): $\hat{x}_{ij} = 0$, for points i and j which are co-located; and $\hat{x}_{ij} = 1$, for those with the distance one from each other. The main idea of the local search-based rounding procedure is to explore such promising configurations (using the LP information) and find a configuration leading to a solution (partitioning) whose max-min modularity value (2) is (locally) optimal.

Assume that $\bar{x}$ is the optimal fractional solution to (LP-MM), obtained by the mentioned row/column generation technique. As explained above, to compute an integral solution to (IP-MM), we need to determine a set of final locations at distance one from each other (we refer to these final locations as community centers) and move the nodes to these centers so as to obtain a valid configuration. In fact, to compute an integral solution, we only need to determine a set of distinct centers, since we can then simply move each point to the closest center (with respect to the LP distance) and obtain a corresponding integral solution: for each pair i and j, $\hat{x}_{ij} = 0$ if i and j are co-located; and 1 otherwise. Therefore, the only task of the utilized local search algorithm is to determine a good set of final centers.

More precisely, the local search algorithm works as follows: Randomly pick a subset of V as initial centers. As discussed above, move other vertices to these centers to form a solution (partitioning) and then compute the max-min modularity value of the resulting partitioning; see (2). The local search technique tries to improve the max-min modularity value by adding and/or deleting a center to/from the set of centers at a time. The local search movement that yields the most significant improvement in the max-min modularity value is selected at each iteration. The algorithm terminates when no improving local search move exists.

TABLE I: Networks under study

ID	Network
1	CiteSeer [41]
2	Arnetminer [42]
3	Caltech36 [43]
4	Reed98 [43]
5	Facebook348 [38]

IV. Computational results

This section presents a comprehensive performance evaluation for the proposed method using five well-known real-world networks listed in Table I. Ground truth (i.e., the optimal community structures) is available and known for each of these networks, and therefore, one can facilely measure the quality of a community detection algorithm by estimating the similarities between the communities obtained by the algorithm and the ground truth. For doing this, we use the well-known performance metrics *Adjusted Rand Index* and *Normalized Mutual Information*, explained in the following subsection.

A. Performance metrics

Suppose that for a given graph G, $\mathcal{C}(\mathcal{A}) = \{C_1, \ldots, C_k\}$ and $\mathcal{C}' = \{C'_1, \ldots, C'_{k'}\}$ be respectively a set of communities obtained by an algorithm $\mathcal{A}$ and the ground truth.

Although NMI [44] is a well-known clustering comparison metric, it can perfectly evaluate the similarity between the optimal communities and those discovered by an algorithm. The NMI value corresponding to the algorithm $\mathcal{A}$ can be written as

$$NMI = \frac{-2\sum_{x=1}^{|\mathcal{C}|}\sum_{y=1}^{|\mathcal{C}'|}\frac{|C_x \cap C'_y|}{n} log(\frac{n|C_x \cap C'_y|)}{|C_x||C'_y|}}{\sum_{x=1}^{|\mathcal{C}|}\frac{C_x}{n} log(\frac{C_x}{n}) + \sum_{y=1}^{|\mathcal{C}'|}\frac{C'_y}{n} log(\frac{C'_y}{n})} \quad (17)$$

In the case where the detected communities are identical to the ground truth, the NMI takes its maximum value one, while in the case where the two sets totally disagree, the NMI score is zero. Generally, the more the NMI, the better community structures have been found.

Adjusted rand index (ARI) [45], associated with algorithm $\mathcal{A}$, measures the similarity between $\mathcal{C}(\mathcal{A})$ and $\mathcal{C}'$ as follows

$$ARI = \frac{2(a \times d - b \times c)}{(a+b) \times (b+d) \times (a+c) \times (c+d)} \quad (18)$$

where a, b, c and d are respectively the number of vertex pairs that are in the same community in both $\mathcal{C}(\mathcal{A})$ and $\mathcal{C}'$, in the

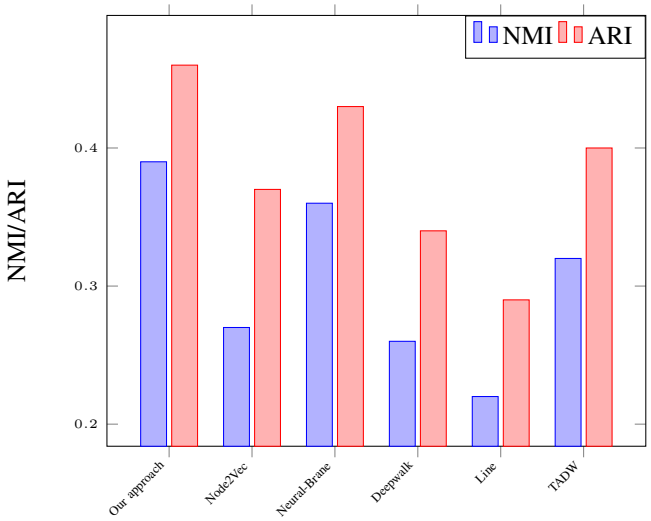


Fig. 4: Average normalized ARI and NMI performance rank of different algorithms applying to the real-world networks, presented in Table I. The average NMI and ARI values obtained by our method are respectively equal to 0.39 and 0.46.

same community in $\mathcal{C}(\mathcal{A})$ but not in $\mathcal{C}'$, in the same community in $\mathcal{C}'$ but not in $\mathcal{C}(\mathcal{A})$, and in different communities in both $\mathcal{C}(\mathcal{A})$ and $\mathcal{C}'$. Like the NMI measure, the value of ARI varies between 0 and 1, and the higher its value is, the more similarity is between the communities obtained by algorithm $\mathcal{A}$ and the grand truth of G.

B. Experiments

In what follows, we provide a process for comparing the performance of our proposed algorithm against five powerful rival algorithms: *Node2Vec* [28], *Neural-Brane* [19], *DeepWalk* [27], *Line* [46], and *Text-Associated DeepWalk (TADW)* [30]. These are all state-of-the-arts for integrating both network topology and nodal attributes for graph representation learning.

All algorithms are implemented with C++, and CPLEX optimizer 12.9 is used for solving linear programming.

Fig. 4 provides a comprehensive comparison by evaluating communities in terms of the average ARI and NMI ranks over all datasets. It is apparent that the proposed method significantly outperforms other algorithms in the sense that the communities it discovered are much more similar to the ground truths than those obtained by the other methods.

Although the results obtained in Fig. 4 perfectly illustrate the proposed method's superiority and reliability against some other state-of-the-art algorithms, it could still be worth investigating the role of the applied GNN feed-forwards message passing procedure in improving the communities. For doing this, we compute the value of NMI associated with each of the networks' obtained final communities in terms of different hidden layer numbers k. In this regard, $k = 0$ refers to the case when the embedding layer is not applied, and only Modularity is employed to identify communities with respect to the graph's topological structure. Fig. 5 shows the results.

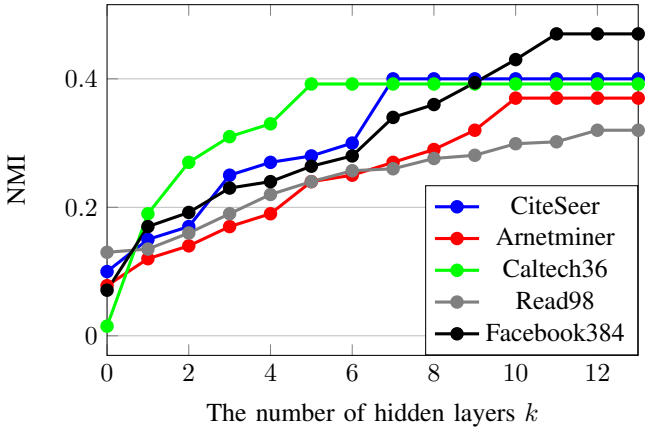


Fig. 5: The computed NMI corresponding to the final communities of each of the networks, provided in Table I, with respect to different values of k.

It is apparent from the results that, first of all, maximizing the Modularity alone (i.e., only relying on topological aspects and ignoring the nodal attributes) cannot lead to promising results. Furthermore, considerably notable is the approving effect of increasing the number of hidden layers. The more the value of k, the more accurate the embedding vector representation becomes due to the more information diffused through the neighboring vertices. However, interestingly enough, from a certain point on, increasing k does not influence the quality of communities, and this is primarily due to the fact that after a certain number of updates, each of the embedding vectors starts to converge.

V. Conclusion

In this work, we built a community discovery method on the basis of the mathematical programming formalism and the graph neural network feed-forward message passing manner. We managed to propose a systematic way to generate an authentic relation matrix for the Max-Min Modularity problem centered on an efficient node embedding technique, which enabled us to model the standard integer formulation for the Max-Min Modularity Maximization problem. A successful row/column generation technique and a local search-based rounding algorithm facilitated us in solving the model accurately and capturing the communities of a given attributed network. Furthermore, the proposed computational experiments showed that our results highly resemble the optimal solutions and that our algorithm outperforms the previous well-known algorithms.

References

[1] L. Jiang, L. Shi, L. Liu, J. Yao, and M. A. Yousuf, "User interest community detection on social media using collaborative filtering," *Wireless Networks*, pp. 1–7, 2019.

[2] A. Ferdowsi, M. Dehghan Chenary, and A. Khanteymoori, "Tscda: a dynamic two-stage community discovery approach," *Social Network Analysis and Mining*, vol. 12, no. 1, pp. 1–14, 2022.

[3] Y. Atay, I. Koc, I. Babaoglu, and H. Kodaz, "Community detection from biological and social networks: A comparative analysis of metaheuristic algorithms," *Applied Soft Computing*, vol. 50, pp. 194–211, 2017.

[4] D. Krioukov, M. Kitsak, R. S. Sinkovits, D. Rideout, D. Meyer, and M. Boguñá, "Network cosmology," *Scientific reports*, vol. 2, p. 793, 2012.

[5] S. Aparicio, J. Villazón-Terrazas, and G. Álvarez, "A model for scale-free networks: application to twitter," *Entropy*, vol. 17, no. 8, pp. 5848–5867, 2015.

[6] A. Ferdowsi, M. DehghanChenary, F. Jolai, and R. Tavakkoli-Moghaddam, "Toward unraveling multi-objective optimization problems: A hybrid approach for solving a novel facility location problem."

[7] N. Tremblay and P. Borgnat, "Graph wavelets for multiscale community mining," *IEEE Transactions on Signal Processing*, vol. 62, no. 20, pp. 5227–5239, 2014.

[8] O. A. Linares, G. M. Botelho, F. A. Rodrigues, and J. B. Neto, "Segmentation of large images based on super-pixels and community detection in graphs," *IET Image Processing*, vol. 11, no. 12, pp. 1219–1228, 2017.

[9] L. M. Freitas and M. G. Carneiro, "Community detection to invariant pattern clustering in images," in *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, 2019, pp. 610–615.

[10] F. D. Malliaros and M. Vazirgiannis, "Clustering and community detection in directed networks: A survey," *Physics Reports*, vol. 533, no. 4, pp. 95–142, 2013.

[11] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the national academy of sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[12] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *Proceedings of the national academy of sciences*, vol. 104, no. 1, pp. 36–41, 2007.

[13] T. Chakraborty, A. Dalmia, A. Mukherjee, and N. Ganguly, "Metrics for community analysis: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 4, pp. 1–37, 2017.

[14] J. Chen, O. R. Zaïane, and R. Goebel, "Detecting communities in social networks using max-min modularity," in *Proceedings of the 2009 SIAM international conference on data mining*. SIAM, 2009, pp. 978–989.

[15] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity clustering," *IEEE transactions on knowledge and data engineering*, vol. 20, no. 2, pp. 172–188, 2007.

[16] A. Ferdowsi and A. Khanteymoori, "Discovering communities in networks: A linear programming approach using max-min modularity," in *2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*. IEEE, 2021, pp. 329–335.

[17] A. R. Barghi, A. Ferdowsi, and A. Abhari, "Musical preferences prediction by classification algorithm," in *Proceedings of the Communications and Networking Symposium*, 2018, pp. 1–12.

[18] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[19] V. S. Dave, B. Zhang, P.-Y. Chen, and M. A. Hasan, "Neural-brane: Neural bayesian personalized ranking for attributed network embedding," *Data Science and Engineering*, vol. 4, no. 2, pp. 119–131, 2019.

[20] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, "Learning community embedding with community detection and node embedding on graphs," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 377–386.

[21] J. J. Choong, X. Liu, and T. Murata, "Learning community structure with variational autoencoder," in *2018 IEEE international conference on data mining (ICDM)*. IEEE, 2018, pp. 69–78.

[22] S. Souravlas, A. Sifaleras, M. Tsintogianni, and S. Katsavounis, "A classification of community detection methods in social networks: a survey," *International Journal of General Systems*, vol. 50, no. 1, pp. 63–91, 2021.

[23] G. Agarwal and D. Kempe, "Modularity-maximizing graph communities via mathematical programming," *The European Physical Journal B*, vol. 66, no. 3, pp. 409–418, 2008.

[24] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti, "Column generation algorithms for exact modularity maximization in networks," *Physical Review E*, vol. 82, no. 4, p. 046112, 2010.

[25] T. N. Dinh and M. T. Thai, "Finding community structure with performance guarantees in complex networks," *arXiv preprint arXiv:1108.4034*, 2011.

[26] A. Miyauchi and Y. Miyamoto, "Computing an upper bound of modularity," *The European Physical Journal B*, vol. 86, no. 7, p. 302, 2013.

[27] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.

[28] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[29] X. Huang, J. Li, and X. Hu, "Accelerated attributed network embedding," in *Proceedings of the 2017 SIAM international conference on data mining*. SIAM, 2017, pp. 633–641.

[30] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[31] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D. Jin *et al.*, "A comprehensive survey on community detection with deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[32] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "User profile preserving social network embedding," in *IJCAI International Joint Conference on Artificial Intelligence*, 2017.

[33] D. Jin, Z. Yu, P. Jiao, S. Pan, D. He, J. Wu, P. Yu, and W. Zhang, "A survey of community detection approaches: From statistical modeling to deep learning," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[34] K. G. Dizaji, F. Zheng, N. Sadoughi, Y. Yang, C. Deng, and H. Huang, "Unsupervised deep generative adversarial hashing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3664–3673.

[35] Q. Yuan and B. Liu, "Community detection via an efficient nonconvex optimization approach based on modularity," *Computational Statistics & Data Analysis*, vol. 157, p. 107163, 2021.

[36] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International conference on machine learning*. PMLR, 2017, pp. 1263–1272.

[37] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[38] O. Shchur and S. Günnemann, "Overlapping community detection with graph neural networks," *arXiv preprint arXiv:1909.12201*, 2019.

[39] D. D. Paepe, D. N. Avendano, and S. V. Hoecke, "Implications of z-normalization in the matrix profile," in *International Conference on Pattern Recognition Applications and Methods*. Springer, 2019, pp. 95–118.

[40] M. E. Newman, "Analysis of weighted networks," *Physical review E*, vol. 70, no. 5, p. 056131, 2004.

[41] C. L. Giles, K. D. Bollacker, and S. Lawrence, "Citeseer: An automatic citation indexing system," in *Proceedings of the third ACM conference on Digital libraries*, 1998, pp. 89–98.

[42] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 990–998.

[43] A. L. Traud, P. J. Mucha, and M. A. Porter, "Social structure of facebook networks," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 16, pp. 4165–4180, 2012.

[44] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2005, no. 09, p. P09008, 2005.

[45] K. Y. Yip, D. W. Cheung, and M. K. Ng, "Harp: A practical projected clustering algorithm," *IEEE Transactions on knowledge and data engineering*, vol. 16, no. 11, pp. 1387–1397, 2004.

[46] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 1067–1077.