CommPlex: Community in MultiPlexes - Definition and a Suite of Algorithms for Analysis

Abhishek Santra[1](✉), Sanjukta Bhowmick[2], and Sharma Chakravarthy[1]

[1] CSE Department and IT Lab, University of Texas at Arlington, Arlington, TX, USA
abhishek.santra@uta.edu, sharmac@cse.uta.edu

[2] Department of Computer Science, University of North Texas, Denton, TX, USA
sanjukta.bhowmick@unt.edu

Abstract. Multiplexes (also termed Multilayer Networks or networks of networks) are useful for modeling datasets with *multiple entity types, and relationships among them*. The notion of a community is well-defined for simple graphs (or a monoplex/network) and is widely used for aggregate analysis on graphs. Several simple graph algorithms (e.g., Infomap, Louvain) for computing a community and algorithms for computing other metrics (e.g., centrality, substructure, etc.) exist as well. Although multilayer networks (MLNs) are used for modeling, the concept of a community and algorithms for its computation are lacking. Ideally, an MLN community definition should be comparable to the simple graph definition and be a generalization. As MLNs have structure in terms of layers, including inter-layer edges, it is important to define a community that includes its structure and semantics. The resulting community should also be an MLN. The focus of this paper is on heterogeneous MLN (or HeMLN), which is a type of MLN with explicitly defined inter-layer edges.

In this paper, we introduce a community definition for HeMLNs that is structure-preserving and is also consistent with the traditional definition. Layer semantics are also preserved for drill-down and visualization. First, we define a community for any k connected layers of a HeMLN (termed *k-community* (1-community is the same as the traditional community on a simple graph or a layer of HeMLN.)) using binary composition. Then, we propose an algorithm for its computation using the concept of bipartite graphs. Further, we show how weight metrics can be customized to include the semantics of participating community characteristics. Our definition: i) leverages extant simple graph community computation algorithms, ii) composes partial results from different layers for computing HeMLN communities (i.e., uses the decoupling approach), iii) is customizable using weight metrics based on participating communities, and iv) is computationally efficient. We have experimentally validated the community concept (definition and computation) on several real-world and synthetic datasets.

Keywords: Community Definition and Detection · Heterogeneous Multilayer Networks · Decoupling Approach · Structure and Semantic Preservation

M. H. Lees et al. (Eds.): ICCS 2025, LNCS 15904, pp. 3–18, 2025.
https://doi.org/10.1007/978-3-031-97629-2_1

1 Motivation

As data sets become more complex in terms of the number and types of entities and relationships, approaches for their modeling and *analysis* also warrant extensions or new alternatives to match the dataset complexity. With the advent of social networks and large data sets, we have already seen a surge in the use of graphs for modeling, along with a renewed interest in concepts, such as community, substructures, and centrality (e.g., hubs) being used for analysis.

Informally, Multilayer Networks (or MLNs) are *layers of networks* where each layer is a simple graph capturing the relationship semantics between two entity instances (either of the same or different type) using an edge. Entities from different layers can also be connected. If each MLN layer has a *common subset of entities of a single type*, it is termed a homogeneous MLN (or HoMLN.) For HoMLN, intra-layer edges are shown explicitly and inter-layer edges are considered implicit (and hence not shown.) For example, US cities are linked based on a direct flight between them operated by a *specific airline* (Fig. 1 (a)). On the other hand, if the *entity types are different for each layer*, then relationships between entities across layers are shown using explicit inter-layer edges. This distinguishes a heterogeneous MLN (or HeMLN) from the previous one. For example, relationships among actors (connected based on co-acting), directors (connected if they direct movies of similar genres), and movies (related by pre-defined average rating ranges) are modeled through a heterogeneous MLN (Fig. 1 (b)). The inter-layer edges represent the relationship across layers, such

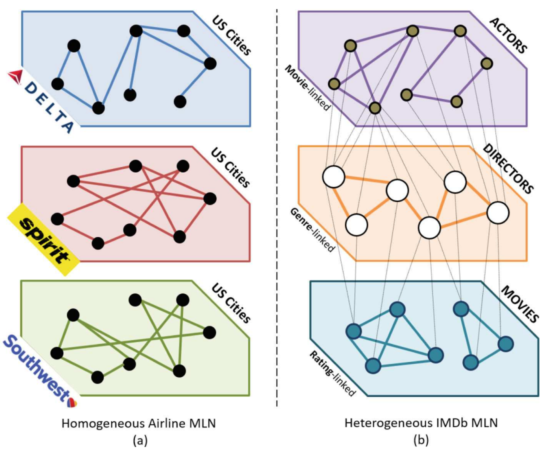

Fig. 1. Homogeneous and Heterogeneous MLNs

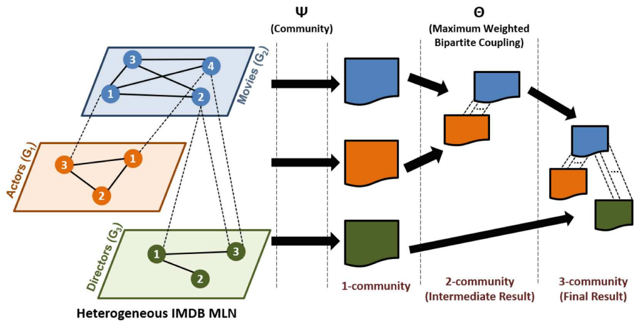

Fig. 2. Decoupling Approach: Compute 3-community ((G_2 $\Theta_{2,1}$ G_1) $\Theta_{2,3}$ G_3) ω_e

as directs-movie, directs-actor, and acts-in-movie (not illustrated). Our focus, in this paper, is on HeMLNs.

For aggregate computations on MLNs, a novel decoupling approach has been proposed in [18, 22, 23]. Figure 2 shows the **decoupling approach**. Three layers and their inter-layer connections are shown. HeMLN community computation is accomplished by combining communities from two layers of a HeMLN using a binary *composition function* (Θ) and is extended to k layers by composing the result with additional layers one at a time. Figure 2 also shows how a 3-layer HeMLN community is expressed for computation. Composing partial results from individual (or previously computed) layers is central to the efficiency of the approach as elaborated in Sect. 7. This approach also preserves the structure of the MLN and its semantics for drill-down and visualization.

The contributions of this paper are shown below with related work in Sect. 2 and conclusions in Sect. 8.

- Definition and some properties of k-community for a HeMLN (Sect. 3),
- Composition function for k-community computation (Sect. 4),
- A new bipartite match algorithm for composition (Sect. 5),
- Experimental analysis to establish the validity of the proposed approach along with performance analysis (Sects. 6 and 7)

2 Related Work

As this paper focuses on the HeMLN community definition and its efficient detection, we present relevant work on simple graphs and HeMLNs. The advantages of modeling using MLNs are discussed in [4, 14, 15, 20, 23].

Community detection on a simple graph involves identifying groups of vertices that are more connected to each other than to other vertices in the network. Most of the work in the literature considers **single networks or simple graphs** where this objective is translated to optimizing network measures such as modularity [3], conductance [16] or map equation [6]. As the combinatorial optimization of community detection is NP-complete [7], many competitive approximation algorithms and deep learning based methods have been developed (see reviews in [11, 13, 25].) Algorithms for community detection have been developed for different types of input graphs, including directed, edge-weighted, and dynamic networks. However, to the best of our knowledge, there is no community definition for HeMLNs, let alone its computation that preserves structure along with node and edge labels for drill-down (semantics).

The majority of the work on analyzing HeMLN (reviewed in [5, 24, 26]) focuses on developing meta-path based techniques for determining clustering, similarity of objects, classification of objects, predicting the missing links, ranking/co-ranking, and recommendations.

The type-independent [8] and projection-based [2] approaches used for ground truth (GT) for HeMLNs use the existing community definition and *do not preserve the structure or semantics of the community.* Both approaches, in slightly different ways, conflate layers into a simple graph keeping *all* nodes and edges (including inter-layer edges) sans their types and labels. This has been shown to result in information loss [15]. Most of the community detection work in MLNs has focused on homogeneous MLNs, where the common set of nodes is present in each layer ([12,14,21]). However, the presence of different sets of entities in each layer and the presence of intra- and inter-layer edges make the structure-preserving definition more challenging for HeMLNs and also warrants an alternate technique (the decoupling approach.) A few existing works have proposed techniques for generating clusters of entities [10,17], but they have only considered the *inter-layer* links and *not* the networks themselves.

This paper fills the gap by providing a clear **new** formal definition of community for HeMLNs that is *structure- and semantics-preserving.* This definition can be shown to be similar to the traditional modularity definition for communities. A distinct advantage of the definition and the use of the decoupling approach is that it leverages existing community detection algorithms (and several of them are currently available.) Infomap and Louvain are more popular than others. This paper also established the efficiency of the decoupling approach for HeMLN community detection.

3 Community Definition for a HeMLN

3.1 Multilayer Networks: Notations Used in the Paper

We start with a formal multilayer network definition that covers both homogeneous and heterogeneous networks.

Table 1. Notations used in this paper

$G_i(V_i, E_i)$	Simple Graph for layer i
$X_{i,j}(V_i, V_j, L_{i,j})$	Bipartite graph of layers i and j
$MLN(G,X)$	Multilayer Network of layer graphs (set G) and Bipartite graphs (set X)
Ψ	Analysis function for G_i (community)
$\Theta_{i,j}$	**Proposed** Maximum Weighted Bipartite Coupling (**MWBC**) function
$CBG_{i,j}$	Community Bipartite Graph for G_i and G_j
U_i	Meta nodes of layer i 1-community
$L'_{i,j}$	Meta edges between U_i and U_j
c_i^m	m^{th} community of G_i
$v_i^{c^m}$, $e_i^{c^m}$	Vertices and Edges in community c_i^m
H_i^m	Hubs in c_i^m
$H_{i,j}^{m,n}$	Hubs in c_i^m connected to c_j^n
$x_{i,j}$	{Expanded (meta edge $< c_i^m, c_j^n >$)}
0 and ϕ	null community id and empty $x_{i,j}$
ω_e, ω_d, ω_h	Weight metrics for meta edges (see Sect. 5)

Formally, a **multilayer network**, $MLN(G,X)$, is defined by two sets of graphs: (i). The set $G = \{G_1, G_2, \ldots, G_N\}$ contains graphs of N individual layers $L = \{L_1, L_2, \ldots, L_N\}$ each of which is a simple graph, where $G_i(V_i, E_i)$ is defined by a set of vertices, V_i and a set of edges, E_i. An edge $e(v,u) \in E_i$, connects vertices v and u, where $v,u \in V_i$ and (ii). A set $X = \{X_{1,2}, X_{1,3}, \ldots, X_{M-1,M}\}$ of bipartite graphs. Each bipartite graph $X_{i,j}(V_i, V_j, L_{i,j})$ is defined by two sets of vertices V_i and V_j, and a set of edges (also called links or inter-layer edges) $L_{i,j}$, such that for every link $l(a,b) \in L_{i,j}$, $a \in V_i$ and $b \in V_j$, where V_i (V_j) is the vertex set of graph G_i (G_j.) For a HeMLN (the focus of this paper), X is explicitly specified. Without loss of generality, we assume unique numbers for nodes across layers and disjoint sets of nodes across layers.

Our definition of community for a HeMLN uses communities from each layer and inter-layer edge connection strength between communities across layers expressed as a weight. One of the weights (number of inter-layer edges) is compatible with the simple graph definition of a community. In addition, coupling alternatives can be formulated for the same Θ to provide multiple (or a ***family*** of) community definitions that can be used for different analysis objectives as needed. Finally, the framework is extensible in that it allows one to propose additional parameters (or weights) to customize for a specific dataset or a set of analysis objectives. Furthermore, it also preserves the structure and semantics due to composition using the decoupling approach which is also shown to be computationally efficient (see Sect. 7). Table 1 lists notations used in the paper for quick reference.

3.2 Definition of HeMLN Community

A **Community Bipartite Graph** or $\boldsymbol{CBG}_{i,j}(U_i, U_j, L'_{i,j})$ is defined between two disjoint and independent communities U_i and U_j. Each element of U_i (U_j) is a community from G_i (G_j) that is represented as a single meta node. $L'_{i,j}$ is the set of meta edges between the nodes of U_i and U_j (or bipartite graph edges.) For any two meta nodes, a *single edge* is used for $L'_{i,j}$, if there is *at least one inter-layer edge* between any pair of nodes from the corresponding communities (acting as meta nodes in CBG) in layers G_i and G_j. The strength (or weight) component of the meta edges is elaborated in Sect. 5.

For a layer graph, a **1-community** is the same as the traditional communities identified using any of the community detection algorithms. A **HeMLN community for 2 layers** (termed **2-community**) is formally defined using the community bipartite graph $CBG_{i,j}(U_i, U_j, L'_{i,j})$ corresponding to layers G_i and G_j. A 2-community is a set of tuples each with a pair of elements $< c_i^m, c_j^n >$, where $c_i^m \in U_i$ and $c_j^n \in U_j$, that satisfy the *Maximum Weighted Bipartite Coupling (MWBC)* (composition function Θ) for the bipartite graph of U_i and U_j, along with the set of inter-layer edges between them (denoted by $x_{i,j}$.) The idea is to obtain the group(s) of nodes that are tightly coupled within and across layers. The pairing is done from left-to-right (as it is *not commutative*) and a single community from the left layer can pair with *zero or more communities* from the right layer. That is, *one-to-many or many-to-one pairings* are possible, unlike traditional bipartite matching.

A **HeMLN community of *k* connected layers**, termed ***k*- *community*** is defined as the application of *2-community* definition recursively to compute k-community. The 2-community definition can be applied to t1-community and t2-community to generate a (t1+t2)-community. The base case corresponds to applying the definition of 2-community for 2 layers t1 and t2. This applies to any expression with precedence. For sinplicity, we discuss the left-to-right computation of k-community.

For a left-to-right computation, the base case is applied to the first 2 layers. For each recursive step, there are two cases for the 2-community under consideration:

i) the U_i is from a layer G_i *already in the t-community* and the U_j is from a *new layer* G_j. This bipartite graph match is said to **extend** a t-community (t $< k$) to a *(t+1)-community*, or ii) **both** U_i and U_j from layers G_i and G_j, respectively are *already in the t-community*. This bipartite graph match is said to **update** a t-community (t $< k$), **not extend** it.

For both cases i) and ii) above, two outcomes are possible. A meta node from U_i either, a) matches one or more meta nodes in U_j resulting in one (or many) **consistent match**, or b) does not match a meta node in U_j resulting in a **no match**. However, for case ii) a third possibility exists which can be characterized as c) matches a node in U_j that is not consistent with a previous match, termed **inconsistent match**. Since both communities have already been matched, a previous match exists (either consistent or no match). If the current match is not the same, then it is an **inconsistent** match.

Table 2. Cases and outcomes for MWBC (Algo. 1)

(G_{left}, G_{right}) **outcome**	**Effect on tuple** t
case (i) - one processed and one new layer	
a) consistent match	**Copy & Extend** t with paired community id **and** $x_{i,j}$
b) no match	**Copy & Extend** t with 0 and ϕ
case (ii) - both are processed layers	
a) consistent match	**Copy & Update** t **only with** x
b) no match	**Copy & Update** t **only with** ϕ
c) inconsistent match	**Copy & Update** t **only with** ϕ

Structure preservation is accomplished by retaining, for each tuple of t-community, either a matching community id (or 0 if no match) and $x_{i,j}$ (or ϕ for the empty set) representing inter-layer edges corresponding to the meta edge between the meta nodes (termed **expanded(meta edge).)** The *extend* and *update* carried out for each of the outcomes on the representation is listed in Table 2. Note that due to multiple pairing of nodes during any composition, the number of tuples (or t-communities) may increase. Copying is used to deal with multiple pairings. In general, each element of a k-community can be total or partial. A **partial k-community element** has ***at least one*** ϕ or 0 as part of the tuple. Otherwise, it is a **total k-community element**. Any k-community that is **total** reflects a stronger coupling as it includes all inter-layer edges for those communities (as is the case of M-A-D-M in Fig. 6 (b) in Sect. 7.)

3.3 Characteristics of k-Community

To clearly understand, a HeMLN can be viewed as a simple graph (termed HeMLN-graph) with each *HeMLN layer being a node* and the inter-layer edges between layers denoted by a weighted edge between corresponding nodes. Then, a k-community can be specified over any connected subgraph of this HeMLN-graph. Case i) above corresponds to a k-community of *an acyclic* subgraph, and case ii) to a k-community of a *cyclic* subgraph of the HeMLN-graph. For both, the number of compositions will be determined by the number of edges in the connected subgraph and can be more than the number of layers (or nodes). Also, for both cases, MWBC results in one of the 3 outcomes: a *consistent match*, *no match*, or an *inconsistent match (only for case (ii))* as shown in Table 2.

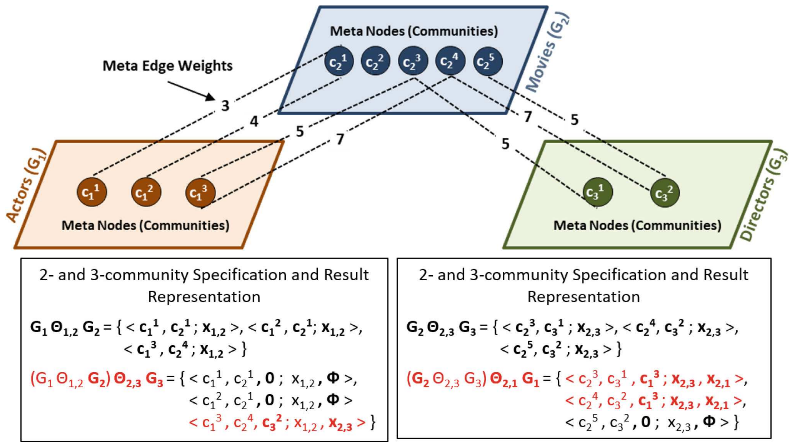
Meta Edge Weights
Meta Nodes (Communities)
Movies (G_2)
Actors (G_1)
Directors (G_3)
2- and 3-community Specification and Result Representation
$G_1 \Theta_{1,2} G_2 = \{ < c_1^1, c_2^1; x_{1,2} >, < c_1^2, c_2^1; x_{1,2} >, < c_1^3, c_2^4; x_{1,2} > \}$
$(G_1 \Theta_{1,2} G_2) \Theta_{2,3} G_3 = \{ < c_1^1, c_2^1, 0; x_{1,2}, \Phi >, < c_1^2, c_2^1, 0; x_{1,2}, \Phi > < c_1^3, c_2^4, c_3^2; x_{1,2}, x_{2,3} > \}$
2- and 3-community Specification and Result Representation
$G_2 \Theta_{2,3} G_3 = \{ < c_2^3, c_3^1; x_{2,3} >, < c_2^4, c_3^2; x_{2,3} >, < c_2^5, c_3^2; x_{2,3} > \}$
$(G_2 \Theta_{2,3} G_3) \Theta_{2,1} G_1 = \{ < c_2^3, c_3^1, c_1^3; x_{2,3}, x_{2,1} >, < c_2^4, c_3^2, c_1^3; x_{2,3}, x_{2,1} >, < c_2^5, c_3^2, 0; x_{2,3}, \Phi > \}$

Fig. 3. Illustration of order dependence on a k-community

The above definition when applied to a specification (such as the one shown in Fig. 3 generates *progressively strong coupling of communities between layers* (due to left-to-right precedence of Θ) using MWBC. *Thus, our definition of a k-community is characterized by dense connectivity within the layer (community definition) and strong coupling across layers (comparable to community definition captured by MWBC semantics.)* Hence, we believe, that this definition of k-community matches or comes close to the original modularity intuition [19] of a community[1] for a simple graph (see Table 3). By refining the edge weight using participating community characteristics, a family of community definitions is supported that can be customized.

For the evaluation purpose, we used the IMDb (layer details are shown in Table 4 of Sect. 7) and DBLP HeMLNs. For IMDb, we have used the Actor and Director layers with their inter-layer edges. For DBLP, we have used the Author and Paper layers with their inter-layer edges. For composition, we have used the metric ω_e that takes into account the number of inter-layer edges between the layer-wise communities while performing the match-

Table 3. Community modularity comparison: Type-independent vs. Proposed definitions

HeMLN	Type-Independent	Decoupling
IMDb	0.77696	0.643508
DBLP	0.694208	0.694208

[1] Modularity is a measure of the structure of a network or a graph which measures the strength of division of a network into modules (also called groups, clusters or communities). Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules. Similarly, in our definition, we pair the communities between two layers based on the inter-layer edge strength connected to that pair (of all pairs), and hence the pairs produced have dense connections within and across two layers. Here the modularity of the HeMLN is based on the dense coupling between the dense communities of layers as compared to all possible inter-layer couplings.

ing. This metric is closest to the traditional definition as type-independent aggregation does not consider any other layer-wise community characteristics. Table 3 shows the modularity values [2] for the *decoupling approach and the type-independent approach.* As can be seen they are identical for DBLP and very close for IMDb. Hence, our community definition, apart from preserving the structure and semantics, generates communities whose quality, based on modularity *is comparable to* the type-independent communities.

Need for a New Pairing Algorithm. In a traditional bipartite graph (used for dating, hiring, etc.), each node is atomic. The goal is to find the maximum number of matches (bipartite edges) such that no two matches share the same node. Hence, a node from one set is paired with *at most* one node from the other set. To accommodate multiple edges, weights are needed without changing the pairing semantics [9].

In contrast, each node of our bipartite graph is a meta node (non-atomic and corresponds to a community) and the bipartite edge is also a meta edge (set of edges between two communities). Each meta node, in our case, is a community representing a group of entities (layer nodes) with additional characteristics. Each meta edge needs to, at the least, capture the number of edges in that meta edge (i.e., inter-layer edges.) The number of edges between the meta nodes is one of the edge weights (ω_e) proposed, which corresponds to the traditional intuition behind a community.

Since edge weights play a significant role in the matching and are also used as a mechanism for determining the strength of the coupling of communities across layers, edge weights can be leveraged to include participating community characteristics. In addition to ω_e, it is possible to bring in participating community characteristics to capture additional aspects of coupling. This can be done by defining different edge weights to capture different characteristics of the participating communities. We have used hub participation from communities and the density of participating communities as weight alternatives.

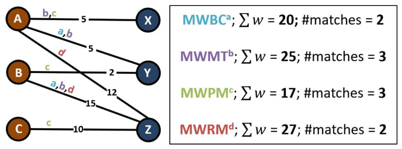
A
B
C
X
Y
Z
b,c
5
a,b
5
d
c
2
a,b,d
12
15
c
10
MWBC^a; $\sum w$ = 20; #matches = 2
MWMT^b; $\sum w$ = 25; #matches = 3
MWPM^c; $\sum w$ = 17; #matches = 3
MWRM^d; $\sum w$ = 27; #matches = 2

Fig. 4. Illustration of Traditional and Relaxed Pairings on a weighted bipartite graph

For pairing nodes of the bipartite graph, since traditional approaches are not suited for our coupling, we propose an edge weight-based coupling that reflects the semantics of the community. Each node from the first set is paired with *zero or more nodes* from the second set solely based on the outgoing edge weights of that node. This is repeated for each node from the first set. *Most importantly, unlike current alternatives in the MLN community literature, there is no information loss or distortion or hiding the effect of different entity types or relationships in our definition.*

[2] A modularity value greater than 0.5 is considered acceptable. Modularity value close to 1 indicates strong community structure, whereas a value close to 0 indicates that the community structure is not better than random.

Figure 4 provides an example of a bipartite graph to illustrate multiple types of pairings appropriate for MLN communities. MWM (Maximum Weight Matching); MWMT (MWM with Ties); MWPM (Maximum Weight Perfect Match); MWRM (Maximum Weight with Relaxed Matching).

4 HeMLN k-Community Computation

Algorithm 1 accepts a linearized specification of a k-community and computes the result as described earlier. The output is a *set* whose *elements are tuples corresponding to distinct, single HeMLN k-community* for that specification. Figure 3 shows 2- and 3-community example results computed using this algorithm.

Algorithm 1. HeMLN k-community Detection Algorithm

```
Require: -
    INPUT: HeMLN, ($G_{n1}$ $\Theta_{n1,n2}$ $G_{n2}$ ... $\Theta_{ni,nk}$ $G_{nk}$), and a weight metric ($\omega$).
    OUTPUT: Set of distinct k-community tuples
 1: Initialize: k=2, $U_i$ = $\phi$, $U_j$ = $\phi$, result$'$ = $\emptyset$
    result ← MWBC($G_{n1}$,$G_{n2}$, HeMLN, $\omega$)
    left, right ← left and right subscripts of second $\Theta$
 2: while left ≠ null && right ≠ null do
 3:     $U_i$ ← subset of 1-community($G_{left}$,result )
 4:     $U_j$ ← subset of 1-community($G_{right}$,result )
 5:     MP ← MWBC($U_i$, $U_j$, HeMLN, $\omega$) //a set of comm pairs < $c^p_{left}$,$c^q_{right}$ >
 6:     for each tuple $t \in$ result do
 7:         kflag = false
 8:         if both $c^x_{left}$ and $c^y_{right}$ are part of $t$ and ∈ MP [case ii (processed layer): consistent match] then
 9:             Update a copy of $t$ with ($x_{left,\ right}$) and append to result$'$
10:         else if $c^x_{left}$ is part of $t$ and ∈ MP and $G_{right}$ layer has been processed [case ii (processed layer): no and inconsistent match] then
11:             Update a copy of $t$ with $\phi$ and append to result$'$
12:         else if $c^x_{left}$ is part of $t$ and for each $c^x_{left}$ ∈ MP [case i (new layer): consistent match] then
13:             copy and Extend $t$ with paired $c^y_{right}$ ∈ MP and $x_{left,\ right}$ and append to result$'$;
                kflag = true
14:         else if $c^x_{left}$ is part of $t$ and ∉ MP [case i (new layer): no match] then
15:             copy and Extend $t$ with 0 (community id) and $\phi$ and append to result$'$; kflag = true
16:         end if
17:     end for
        left, right = next left, right subscripts of $\Theta$ or null
        if kflag k = k + 1; result = result$'$; result$'$ = $\emptyset$
18: end while
```

The bipartite graph for the base case and each iteration is constructed for the participating layers (either one is new or both are from the t-community for some t) and MWBC algorithm is applied. The result obtained is used to either extend or update the tuples of the t-community and add new tuples as well. All cases are described in Table 2.

The algorithm iterates **(lines 2 to 18)** until there are no more compositions to be applied. Line 5 computes the first 2-community. **Lines 6 to 17** apply the results of the MWBC (**line 5**) to generate tuples of the k-community using the

Table 2.Careistakeninthecompositiontomakesureeitherthetupleisupdated orextendedbykeepingaflagandcheckingitafter**line17**.Theorderofchecking inside the for loop (**lines6 to 17**) is important to generate the correct k-communitytuples.

5 Customizing the Bipartite Graph

Withoutincludingthecharacteristicsofmetanodesandedgesforthematch,we cannotarguethatthepairingobtainedrepresentsanalysisbasedonparticipating community characteristics. Hence, it is important to identify how qualitative communitycharacteristicscanbemappedquantitativelytoaweightmetric(that is,theweightofthemetaedgeinthecommunitybipartitegraph)toinfluencethe bipartitematching.Outofthethreedevelopedweightmetricsbasedon(number of inter-community edges (ω_e), density (ω_d), and hub participation (ω_h)), we detailonlyoneweightmetricduetospaceconstraintsbelow.

Hub Participation (ω_h): For many analyses, we are interested in knowing whether highly influential nodes within a community also interact across the community.Thiscanbetranslatedtothe*participationofinfluentialnodeswithin andacrosseachparticipatingcommunity* foranalysis.Thisismodeledbyusing the notion of **hub** [3] **participation** within a community and their interaction across layers. In this paper, we have used degree centrality for this metric to connotehigherinfluence.Theratioofparticipatinghubsfromeachcommunity andtheedgefractionismultipliedtocompute ω_h.Formally,

Forevery $(u_i^m, u_k^n) \in L'_{i,k}$,where u_i^m and u_k^n arethemetanodesdenotingthe communities, c_i^m and c_k^n intheCBG,respectively,theweight,

$$\omega_h(u_i^m, u_k^n) = \frac{|H_{i,k}^{m,n}|}{|H_i^m|} * \frac{|x_{i,k}|}{|v_i^{c^m}| * |v_k^{c^n}|} * \frac{|H_{k,i}^{n,m}|}{|H_k^n|},$$

where, $x_{i,k} = \{(a,b)\ a \in v_i^{c^m}, b \in v_k^{c^n},\ \textit{and}\ (a,b) \in L_{i,j}\}$; H_i^m *and* H_k^n *are setofhubsin* c_i^m *and* c_k^n*,respectively;* $H_{i,k}^{m,n}$ *isthesetofhubsfrom* c_i^m *thatare connectedto* c_k^n; $H_{k,i}^{n,m}$ *isthesetofhubsfrom* c_k^n *thatareconnectedto* c_i^m.

6 Expressing Analysis Objectives on HeMLNs

Wedemonstratehowanalysisobjectivescanbeexpressedask-communitycomputationsonHeMLNs.Also,dependingontheanalysis,appropriateweightmetrics can be chosen. Due to space constraints, we are not discussing the results forotherreal-world(likeDBLP)andsyntheticdatasets.

[3] Highcentralitynodes(orhubs)havebeendefinedbasedondifferentmetrics,such asdegreecentrality(vertexdegree),closenesscentrality(meandistanceofthevertexfromothervertices),betweennesscentrality(fractionofshortestpathspassing throughthevertex),andeigenvectorcentrality.

IMDb DataSet [1]**:** The IMDb dataset captures movies, TV episodes, actors, directors, and other related information, such as rating. Some IMDb detailed analysis objectives are listed below,

(A1) Based on the similarity of genres, for each director group which are the actor groups whose *majority of the most versatile members interact*?
2-community: D $\varTheta_{A,D}$ A; ω_h

(A2) For the *most popular* actor groups, for each movie rating class, find the director groups with which they have *maximum interaction* and who also direct movies with similar ratings.
Cyclic 3-community: M $\varTheta_{M,A}$ A $\varTheta_{A,D}$ D $\varTheta_{D,M}$ M; ω_e

To address the IMDb analysis requirements, three layers for the IMDb data set are generated. Layer A and Layer D connect actors and directors who act in or direct *similar genres frequently* (intra-layer edges), respectively. Layer M connects movies within the same rating range. The inter-layer edges depict *acts-in-a-movie* ($L_{A,M}$), *directs-movie* ($L_{D,M}$) *and directs-actor* ($L_{A,D}$) relationships. There are multiple ways of quantifying the similarity of actors (directors) based on the movie genres they have worked in. A vector was generated with the number of movies for each genre he/she has acted in (directed). To take into account the *frequency of genres* in the similarity calculation, two actors (directors) are connected if the Pearsons' Correlation between their corresponding genre vectors is ≥ 0.9[4]. Moreover, 10 movie rating ranges are considered - [0–1), [1–2), ..., [9–10].

For a specific analysis, the characteristics of the communities connected in the bipartite graph need to be used as meta edge weight to get the desired coupling.

For example, *most popular* in (A2) is interpreted as the higher number of edges between the participating communities. In contrast, *versatility* is mapped to the participation of hub nodes in each group as in (A1).

To compute a k-community, k needs to be identified. (A1) requires 2-community. analysis: IMdb Hespsmadm requires a cyclic 3-community using inter-layer relationships between all layers in a particular order. Note that the analysis objectives have been chosen carefully to cover the weights discussed in the paper. The limitation on the number of analysis objectives is purely due to space constraints.

7 HeMLN Community Analysis on Real-World Data Sets

We would like to point out that the choice of data sets and sizes was mainly to demonstrate the versatility of analysis using the k-community detection and its efficiency as well as drill-down capability based on structure- and -semantics

[4] The choice of the coefficient is not arbitrary as it reflects relationship quality. The choice of this value can be based on how actors (directors) are weighted against the genres. We have chosen 0.9 for connecting actors (directors) in their top genres.

preservation. We are not trying to demonstrate scalability in this paper. Also, instead of presenting communities, we present a few important drill-down results to showcase the structure and semantics preservation, and the general applicability of our approach.

7.1 Experimental Setup and Data Sets

Due to the lack of real-world HeMLNs, we generated HeMLNs from data collected/crawled from some well-known real-world domains. For IMDb HeMLN, we extracted, for the top 500 actors, the movies they have worked in (7500+ movies with 4500+ directors). The actor set was repopulated with the co-actors from these movies, giving a total of 9000+ actors. For this set of actors, directors, and movies, the HeMLN with 3 layers described in Sect. 6 was built. Widely used Louvain method [3] was used to detect layer-wise 1-communities [5]. The k-community detection algorithm 1 was implemented in Python version 3.6 and was executed on a quad-core 8^{th} generation Intel i5 processor Windows 10 machine with 8 GB RAM.

Table 4. IMDB HeMLN Statistics

	Actor	Director	Movie
#Nodes	9,485	4,510	7,951
#Edges	996,527	250,845	8,777,618
#Communities (Size>1)	63	61	9
Average Community Size	148.5	73	883.4

	#Inter-layer Edges
Actor-Director	32,033
Actor-Movie	31,422
Director-Movie	8,581

Individual Layer Statistics: Table 4 shows the IMDb HeMLN statistics. 63 Actor (A) and 61 Director (D) communities based on similar genres are generated. Out of the 10 ranges (communities) in the movie (M) layer, most of the movies were rated in the range [6–7), while the least popular rating was [1–2). No movie had a rating in the range [0–1).

7.2 Analysis Results and Discussion

(A1)
Analysis Results: 34 D-A (Director-Actor) similar genre-based community pairs were obtained, where *the majority of the most versatile members interact.* Intuitively, a group of directors that prominently makes movies in some genre

[5] Louvain numbers all communities from 1 and we only consider communities having *at least two members* for this paper. The numbering used in the paper has the layer name followed by the Louvain-generated community ID (e.g. A91).

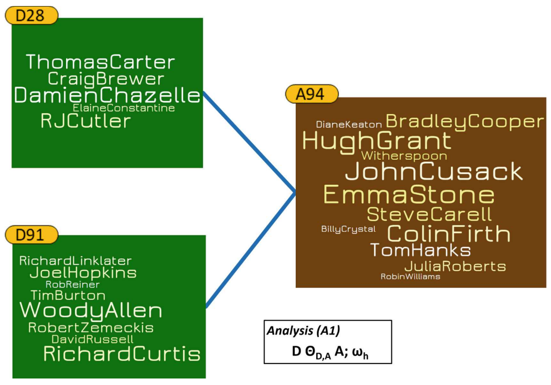

Fig. 5. Sample (A1) Result for *Romance, Comedy, Drama*

(say, Drama, Action, Romance, ...) must pair up with the group(s) of actors who primarily act in similar kinds of movies. Moreover, a *director group may work with multiple actor groups and vice-versa.* For example, in Fig. 5, the sample result shows that the director groups, D28 and D91, with academy award winners like **Damien Chazelle and Woody Allen, respectively, pair up with the actor group with members like Diane Keaton, Emma Stone, and Hugh Grant**. Members from these groups are primarily known for movies from the **Romance, Comedy, and Drama** genre.

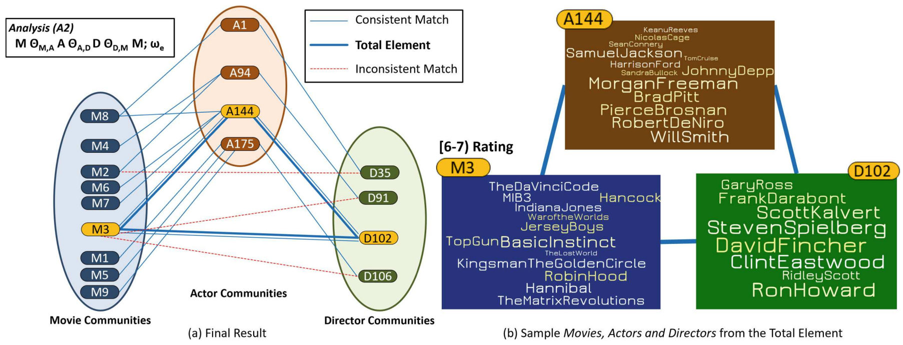

Fig. 6. (A2) Result: **1 Total, 9 Partial Elements**. (Color figure online)

(A2) **Analysis Results:** Here, the *most popular actor groups for each movie rating class are further coupled with directors.* These *director groups are coupled again with movies to check whether the director groups also have similar ratings.* Results of each successive pairing (there are 3) are shown in Fig. 6 (a) using the same color notation. The coupling of movie and actor communities (first composition) results in 10 consistent matches. When the base case is extended to the director layer (second composition) using all director communities and the matched 4 actor communities, we get 4 consistent matches. The final composition to complete the cycle uses 4 director communities and 9 movie communities as left and right sets of community bipartite graph, respectively. **Only one consistent match is obtained to generate the total element (M3-A144-D102-M3) for the cyclic 3-community** (bold blue triangle.) The resulting total element is from the **Action, Drama genre** as can be seen from the sample members shown in Fig. 6 (b). It is interesting to see 3 inconsistent matches (red broken lines) between the communities, which clearly indicate that all couplings are not satisfied by these pairs. These result in 9 partial elements. **The inconsistent matches also highlight the importance of mapping an analysis objective to a k-community specification for computation.** If a different order had been chosen (viz., director and actor layer as the base case), the result could have included the inconsistent matches.

7.3 Efficiency of Decoupling Approach

The goal of the decoupling approach was to preserve the structure as well as improve the efficiency of k-community detection. We illustrate that with the largest k-community we have computed which uses 3 iterations (including the base case.) Fig. 7 shows the execution time for the one-time and iterative costs discussed earlier for (A2). The difference in one-time 1-community cost for the 3 layers follows their density shown in Table 4. We can also see how the iterative cost is insignificant as compared to the one-time cost (by an order of magnitude.) Iteration cost includes creating the bipartite graph, computing ω_e for meta edges, and MWBC cost. **The cost of all iterations together (0.515 sec) is still almost *an order of magnitude less than the largest one-time cost* (5.21 sec for Movie layer.)** We have used this case as this subsumes all other cases. The **additional incremental cost for computing a k-community is extremely small validating the efficiency of decoupled approach**.

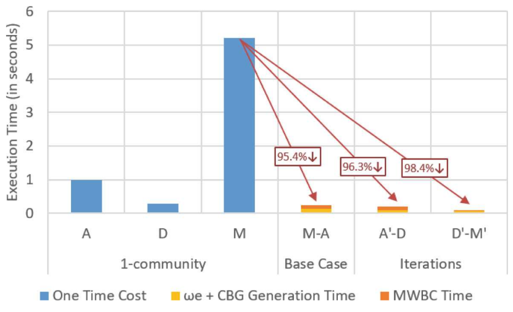

Fig. 7. Performance Results for cyclic 3-community in (A2)

8 Conclusions

In this paper, we have provided a community definition for HeMLNs that is consistent with the traditional definition and is structure preserving. This definition can be applied to an arbitrary number of layers of a HeMLN. *In fact, with* ω *as a customizable parameter, this supports a family of definitions that are customizable to analysis needs.* We proposed a new bipartite match-based composition function (MWBC algorithm) for the decoupling approach. We have compared our results with the traditional ground truths using modularity to show their compatibility. Finally, we used the proposed approach to demonstrate its analysis versatility using the IMDb data set. In the future, we plan to extend this work to weighted MLNs.

Acknowledgment:. This work was supported by NSF awards CCF-1955798, and CCF-1956373.

References

1. The internet movie database. ftp://ftp.fu-berlin.de/pub/misc/movies/database/
2. Berenstein, A., Magarinos, M.P., Chernomoretz, A., Aguero, F.: A multilayer network approach for guiding drug repositioning in neglected diseases. PLOS (2016)
3. Blondel, V.D., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of community hierarchies in large networks. CoRR arXiv:abs/0803.0476 (2008)
4. Boccaletti, S., et al.: The structure and dynamics of multilayer networks. Phys. Rep. **544**(1), 1–122 (2014)
5. Boden, B., Günnemann, S., Hoffmann, H., Seidl, T.: Mining coherent subgraphs in multi-layer graphs with edge labels. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1258–1266 (2012)
6. Bohlin, L., Edler, D., Lancichinei, A., Rosvall, M.: Community detection and visualization of networks with the map equation framework (2014). http://www.mapequation.org/assets/publications/mapequationtutorial.pdf
7. Brandes, U., Gaertler, M., Wagner, D.: Experiments on graph clustering algorithms. In: Di Battista, G., Zwick, U. (eds.) ESA 2003. LNCS, vol. 2832, pp. 568–579. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39658-1_52
8. Domenico, M.D., Nicosia, V., Arenas, A., Latora, V.: Layer aggregation and reducibility of multilayer interconnected networks. CoRR arXiv:abs/1405.0425 (2014)
9. Edmonds, J.: Maximum matching and a polyhedron with 0, 1-vertices. J. Res. Natl. Bureau Stand. B **69**(125–130), 55–56 (1965)
10. Fang, Y., Yang, Y., Zhang, W., Lin, X., Cao, X.: Effective and efficient community search over large heterogeneous information networks. Proc. VLDB Endow. **13**(6), 854–867 (2020)
11. Fortunato, S., Castellano, C.: Community structure in graphs. In: Encyclopedia of Complexity and Systems Science, pp. 1141–1163 (2009)
12. Huang, X., Chen, D., Ren, T., Wang, D.: A survey of community detection methods in multilayer networks. Data Min. Knowl. Disc. **35**, 1–45 (2021)
13. Jin, D., et al.: A survey of community detection approaches: from statistical modeling to deep learning. IEEE Trans. Knowl. Data Eng. **35**(2), 1149–1170 (2021)
14. Kim, J., Lee, J.: Community detection in multi-layer graphs: a survey. SIGMOD Rec. **44**(3), 37–48 (2015)
15. Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J.P., Moreno, Y., Porter, M.A.: Multilayer networks. CoRR arXiv:abs/1309.7233 (2013)
16. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Community structure in large n/ws: natural cluster sizes and absence of large well-defined clusters (2008)
17. Magnani, M., Hanteer, O., Interdonato, R., Rossi, L., Tagarelli, A.: Community detection in multiplex networks. ACM Comput. Surv. (CSUR) **54**(3), 1–35 (2021)
18. Mukunda, K., Roy, A., Santra, A., Chakravarthy, S.: Stress centrality in heterogeneous multilayer networks: heuristics-based detection. In: IEEE 9th International Conference on Big Data Computing Service and Applications, pp. 103–110 (2023)

19. Newman, M.E.: Modularity and community structure in networks. Proc. Natl. Acad. Sci. **103**(23), 8577–8582 (2006)
20. Santra, A., Bhowmick, S.: Holistic analysis of multi-source, multi-feature data: modeling and computation challenges. In: Big Data Analytics - Fifth International Conference, BDA 2017 (2017)
21. Santra, A., Bhowmick, S., Chakravarthy, S.: Efficient community re-creation in multilayer networks using Boolean operations. In: International Conference on Computational Science, ICCS 2017 (2017)
22. Santra, A., Bhowmick, S., Chakravarthy, S.: Hubify: efficient estimation of central entities across multiplex layer compositions. In: IEEE International Conference on Data Mining Workshops, ICDM Workshops 2017 (2017)
23. Santra, A., Irany, F.A., Madduri, K., Chakravarthy, S., Bhowmick, S.: Efficient community detection in multilayer networks using boolean compositions. Front. Big Data **6** (2023)
24. Shi, C., Li, Y., Zhang, J., Sun, Y., Philip, S.Y.: A survey of heterogeneous information network analysis. IEEE Trans. Knowl. Data Eng. **29**(1), 17–37 (2017)
25. Su, X., et al.: A comprehensive survey on community detection with deep learning. IEEE Trans. Neural Netw. Learn. Syst. (2022)
26. Sun, Y., Han, J.: Mining heterogeneous information networks: a structural analysis approach. ACM SIGKDD Explor. Newsl. **14**(2), 20–28 (2013)