Clustering Hypergraphs for Discovery of Overlapping Communities in Folksonomies

Abhijnan Chakraborty* **and Saptarshi Ghosh**

1 Introduction

Online social systems have become very popular in today's Web, where millions of users form social relationships with one another and generate and share various forms of contents. Among these websites, some are specifically designed for content sharing which are known as *social tagging systems* or *folksonomies*. Here, users share various types of contents or resources—such as URLs in Delicious (www.delicious.com), images in Flickr (www.flickr.com), music files in LastFm (www.last.fm) and movie reviews in MovieLens (www.movielens.org)—and collaboratively annotate resources with descriptive keywords (known as "tags") in order to facilitate search and retrieval of interesting resources.

With the growing popularity of folksonomies, a tremendous amount of resources are being uploaded to these sites, and it has become practically impossible for users to discover on their own interesting resources and people having common interests. Hence it is important to develop algorithms for personalized search [36] as well as resource and friend recommendation [20]. One approach to these tasks is to group the entities (resources, tags, users) into communities or clusters which are typically groups of entities having more or better interactions or similarity among themselves than with entities outside the group.

*This work was completed when the author was in IIT Kharagpur.

A. Chakraborty (✉)
Microsoft Research India, Bangalore – 560001, India
e-mail: t-abcha@microsoft.com

S. Ghosh
Department of Computer Science and Engineering Indian Institute of Technology Kharagpur, Kharagpur – 721302, India
e-mail: saptarshi@cse.iitkgp.ernet.in

Folksonomies are modelled in the Complex Networks literature as *tripartite hypergraphs* [7, 36], which include user, resource and tag nodes, where a hyperedge (u, t, r) indicates that the user u has assigned the tag t to the resource r. Detecting communities from such hypergraphs is a challenging problem. On the other hand, this not only helps in efficient search and recommendation of resources or friends to users but also in the organization of the vast amount of the resources present in folksonomies into different semantic categories.

Several algorithms have been proposed for detecting communities in hypergraphs [4, 25, 27, 28, 34] (details in Sect. 2). However, most of the prior approaches do not consider an important aspect of the problem—they assign a single community to each node, whereas in reality, nodes in folksonomies frequently belong to *multiple overlapping communities*. For instance, users have multiple topics of interest, and thus they link to resources and tags of many different semantic categories. Similarly, the same resource is frequently associated with semantically different tags by users who appreciate different aspects of the resource. Considering multiple overlapping communities enables more complete knowledge about the characteristics of the users and the resources and hence can lead to better recommendations.

To the best of our knowledge, only two prior studies have addressed the problem of identifying overlapping communities in folksonomies—(i) Wang et al. [35] proposed an algorithm to detect overlapping communities considering only the user-tag relationships (i.e. the user-tag bipartite projection of the hypergraph), and (ii) Papadopoulos et al. [32] detected overlapping *tag* communities by taking a projection of the hypergraph onto the set of tags. It can be noted that these approaches work on bipartite and unipartite projections of the tripartite hypergraph, respectively.

Taking projections result in loss of some of the information contained in the original tripartite network, and it is known that qualities of the communities obtained from projected networks are not as good as those obtained from the original network [18]. Another important drawback of these algorithms is that none of them consider the resource nodes. However, it is necessary to detect overlapping communities of users, resources and tags simultaneously for personalized recommendation of resources to users. The goal of the presented "Overlapping Hypergraph Clustering" algorithm is to detect overlapping communities, utilizing the complete tripartite structure of folksonomies. It achieves this goal by using the concept of *link clustering*, which is explained next.

Though a node in a network can be associated to multiple semantic topics, a *link* (or edge, the terms are used interchangeably) is usually associated with only one semantics [1]. For instance, a user can have multiple topical interests, but each link created by the user is likely to be associated with exactly one of his interests. *Link clustering* algorithms utilize this notion to detect overlapping communities, by clustering *links* instead of the more conventional approach of clustering nodes. Although each link is placed in exactly one link cluster, this automatically associates multiple overlapping communities with the nodes since a node inherits membership of all the communities into which its links are placed.

Link clustering algorithms have recently been proposed for unipartite [1, 11] and bipartite networks [35]. However, to our knowledge, "Overlapping Hypergraph Clustering" is the first link clustering algorithm for tripartite hypergraphs. Initial versions of this algorithm were presented in [13] and [9]. In this chapter, we present the "Overlapping Hypergraph Clustering" algorithm and its analysis in more detail. We compare the performance of this algorithm with the existing algorithms by Papadopoulos et al. [32] and Wang et al. [35]. Section 3 details the working of the algorithm. Extensive experiments on synthetically generated hypergraphs (Sect. 4) as well as on real data from three popular folksonomies—Delicious, MovieLens and LastFm (Sect. 5)—show that the "Overlapping Hypergraph Clustering" algorithm outperforms both these algorithms. Section 6 concludes the chapter by summarizing the contributions and potential applications of the presented algorithm.

2 Related Work

As networks are increasingly being used to model complex systems [1, 3, 8, 14, 33], several algorithms have been proposed for finding communities in networks. Girvan and Newman proposed one of the initial algorithms for community detection [14], which iteratively removes edges based on their betweenness centrality and, thus, splits the network into disconnected communities. Later, they introduced the notion of *modularity* as a measure of the quality of community structure in a network [15]. Subsequently, several community detection algorithms based on modularity maximization have been proposed, such as techniques based on agglomerative hierarchical clustering [10]. The reader is referred to [12] for a detailed survey of community detection algorithms for graphs.

A large majority of the proposed algorithms assign unique communities to nodes. However, as stated earlier, nodes in social networks (including folksonomies) typically belong to multiple overlapping communities; for example, a user is usually a part of multiple communities of family members, colleagues, schoolmates, college mates and so on. Next we discuss some of the algorithms which have been proposed to identify overlapping communities.

Overlapping Community Detection in Graphs: One of the earliest methods to find overlapping communities was by Baumes et al. [2], which find subsets of nodes whose induced subgraph locally optimizes a given metric based on the edge density of the subgraph. As different overlapping subsets may all be locally optimal, nodes may belong to multiple communities. Clique Percolation Method (CPM) by Palla et al. [31] is possibly the most well-known overlapping community detection technique. CPM finds all k-cliques with a fixed constant k, and each community is formed by merging a maximum set of such k-cliques if they share $k-1$ nodes. One node may belong to multiple disconnected k-cliques and hence to multiple communities. Among other methods, Lancichinetti et al. [21] proposed a local clustering algorithm which optimizes a fitness function defined using the internal and external degrees of the computed clusters. By varying the parameters in the fitness function, both overlapping and hierarchical community structures can be obtained using the algorithm. Gregory [16] proposed an algorithm which works in multiple stages. First, the nodes with highest split betweenness centrality are identified (these are the nodes which may potentially belong to multiple communities) and are split into multiple nodes connected by edges. The original graph is thus transformed into a larger graph including these node sets, on which any nonoverlapping clustering technique can be applied. Finally, the communities are mapped back into the original graph. The well-known modularity metric has also been extended to the overlapping community scenario [30], using which any modularity maximization algorithms can be applied to detect overlapping communities.

Some of the recent algorithms [1, 11] adopt the methodology of *link clustering* for detecting overlapping communities: that is, they group "similar" links (edges) unlike conventional attempts to group similar nodes. Link clustering strategies build upon the idea that even though actors can belong to multiple groups, a link is mostly associated with a single category. Evans et al. [11] considered a modified random walk on the line graph of a given graph along with other diffusion processes. Ahn et al. [1] proposed to cluster links with an agglomerative hierarchical clustering technique and, thus, identify overlapping communities for nodes. The advantage of these algorithms is that while overlapping communities of nodes are indeed discovered (since a given node inherits membership of all communities that contain the edges associated with the node), these algorithms are much simpler and more efficient than the ones which directly find overlapping groups of nodes. Hence in the present study, we adopt the link clustering methodology to propose an algorithm for overlapping community detection in tripartite hypergraphs. In the next section, we discuss existing community detection algorithms for hypergraphs.

Community Detection in Hypergraphs: Several algorithms have been proposed for detecting communities in hypergraphs. Vazquez [34] proposed a Bayesian formulation of the problem of finding hypergraph communities—starting from a statistical model on hypergraphs, they use a mean field approximation to identify communities. Bulo et al. [5] proposed a game-theoretic approach to hypergraph clustering. They show that the hypergraph clustering problem can be converted into a non-cooperative multiplayer clustering game, where the notion of a cluster is equivalent to a classical game-theoretic equilibrium concept. Zhou et al. [37] generalized spectral clustering techniques to hypergraphs.

Approaches based on tensor decomposition [19] have also been proposed—for instance, Lin et al. [25] proposed an efficient multi-tensor factorization method for detecting hypergraph communities. Neubauer et al. [28] proposed a modularity maximization technique to extract communities from hypergraphs. The original k-partite hypergraph is decomposed into $k(k+1)/2$ bipartite projections. The algorithm tries to optimize a joint modularity measure, which is based on the average bipartite modularity in the individual bipartite graphs, in a brute-force, greedy bottom-up fashion. Later, Murata defined a tripartite modularity metric and proposed an algorithm to detect communities from hypergraphs using tripartite modularity maximization principle [27]. Other possible approaches include mapping each hyperedge into a multidimensional space and then applying standard clustering algorithms (e.g. the ROCK algorithm [17]) to the point space.

Overlapping Community Detection in Folksonomies: All the hypergraph community detection algorithms stated above assign a single community to each node. To our knowledge, only two studies have addressed the problem of overlapping community detection in folksonomies.

Wang et al. [35] proposed an edge clustering methodology to detect overlapping communities that uses only user-tag subscription information. In effect, they consider the projection of the given tripartite hypergraph onto a user-tag bipartite graph. Their algorithm is a k-mean variant which maximizes intra-cluster similarity. The network is considered in an edge-centric view where, for determining each centroid, only a small set of edges are compared against each other. Though this is a computationally fast algorithm, it requires the number of communities as an input which is difficult to predict in case of real-world folksonomies.

Papadopoulos et al. [32] proposed an algorithm to detect overlapping communities of *tags*. This algorithm extracts a resource-tag association graph from the tripartite hypergraph, transforms it to a tag co-occurrence network and then finds overlapping tag communities. The proposed scheme initially searches for core sets (densely connected groups) in the tag co-occurrence network and then successively expands the identified cores by maximizing a local subgraph quality measure.

Both the above approaches actually work on unipartite or bipartite projections of the given tripartite hypergraph. Although taking projections reduces the complexity associated with hypergraphs, it loses a significant part of the information contained in the original tripartite network. As an example, let us suppose that two users u_1 and u_2 annotate a common resource r with two different tags t_1 and t_2. In the user-tag bipartite projection (as considered in [35]), u_1 will be linked with t_1 and u_2 will be linked with t_2. However, the information that both these annotations were applied on the same resource will be lost. Since the tags applied on the same resource are likely to be semantically related, this information could have been useful for community discovery. Moreover, Guimera et al. [18] have shown that the quality of communities obtained from projected networks is usually worse than those obtained from the original network. "Overlapping Hypergraph Clustering" algorithm, which is detailed in the next section, does not lose this information as it detects overlapping communities in folksonomies considering the complete tripartite hypergraph structure.

3 Overlapping Hypergraph Clustering Algorithm

This section details the "Overlapping Hypergraph Clustering" algorithm for detecting overlapping communities in tripartite hypergraphs. "Overlapping Hypergraph

Clustering" is abbreviated to "OHC", and both these names are used interchangeably in the remaining part of the chapter.

As discussed earlier, a folksonomy is modelled as a tripartite hypergraph, more specifically a 3-uniform tripartite hypergraph, denoted as $G = (V, E)$, where V is the set of nodes and E is the set of hyperedges. V is composed of three partitions (i.e. three types of vertices) V^X, V^Y and V^Z, and each hyperedge in E connects triples of nodes (a, b, c) where $a \in V^X, b \in V^Y, c \in V^Z$.

3.1 Basic Idea of the Algorithm

For a given hypergraph G, the algorithm converts G to the *weighted line graph* $G^{'}$ which is a *unipartite graph* in which the hyperedges in G are the nodes. Two nodes e_1 and e_2 in $G^{'}$ are connected by an edge if the hyperedges e_1 and e_2 are "*similar*" in G, and the weight of the edge (e_1, e_2) in $G^{'}$ is the similarity between the two hyperedges. Section 3.2 details different ways to compute the similarity between the hyperedges.

Once the *weighted line graph* $G^{'}$ is constructed from the given tripartite hypergraph G, any conventional (nonoverlapping) community detection algorithm for simple graphs can be used to cluster the nodes in $G^{'}$ (i.e. the hyperedges in G).[2] Section 3.3 discusses about the various options and the selection of the community detection algorithm for the line graph.

As we get the node communities in $G^{'}$, each hyperedge in G gets placed into a single *link community*. This automatically assigns multiple overlapping communities to the *nodes* in G since a node inherits membership of all those communities into which the hyperedges connected with this node are placed.

3.2 Calculating Similarity Between Hyperedges

Similarity between a pair of hyperedges can be computed in various ways. For instance, hyperedges can be expressed as feature vectors, on which vector-based similarity measures can be used. Another way of measuring similarity is by considering the neighbourhood of the end vertices of hyperedges.

Expressing Hyperedges as Vectors: For a hypergraph having n nodes, one can express each hyperedge as a vector of size n, where an element of the vector represents the amount of participation of a particular node in that hyperedge. For

[2]Overlapping community detection algorithms can also be used for the line graph. But since a link (hyperedge) is usually associated with one particular semantics, we consider only nonoverlapping community detection algorithms.

example, the i-th entry in the vector representation for a particular hyperedge e will be 0 if there is no path from node i to any of the end nodes of e; otherwise, the i-th entry will be the inverse of the product of degrees of the intermediate vertices in the shortest path from node i to any of the end nodes of e. Different entries in the vector can be easily generated by formulating this as a random walk problem which will take care of the situations like presence of multiple shortest paths. These vectors express the closeness as well as the importance of a particular node to a particular hyperedge. With such representation, standard vector-similarity metrics such as cosine similarity or Pearson correlation can be used to find similarity between hyperedges.

Considering Vertex Neighbourhoods: Similarity between the hyperedges can be measured by utilizing the neighbour set of their end vertices. We measure the similarity between only those hyperedges which are *adjacent* in the hypergraph G (i.e. which have at least one node in common). Nonadjacent hyperedges are considered to have zero similarity.

It can be noted that the adjacency of two hyperedges can be defined in two ways: (i) they have at least one node in common and (ii) they have exactly two nodes in common. Although the second definition is a special case of the first definition, the choice will have some impact on the overall performance of the algorithm. The line graph $G^{'}$ will be sparser in case (ii) than in case (i). So, $G^{'}$ in case (ii) will contain more disconnected components. Detecting communities from this sparser $G^{'}$ will be more difficult. Also, in case of real folksonomies, the condition of having two nodes common is too rigid. Therefore, we have considered only the first definition of adjacency.

Two popular metrics are considered for measuring the similarity among the hyperedges: (i) *matching similarity*, which is the size of the overlap between the neighbour sets of the end points, and (ii) *Jaccard similarity* which is the size of overlap normalized by the size of the union of the neighbour sets of the end points. Jaccard similarity value can range from 0 to 1.

Choice of a Similarity Metric: Vector-based similarity metrics are global metrics whose computation requires the knowledge of the entire hypergraph; hence, they are inefficient for use on the large real folksonomies. On the other hand, neighbourhood-based metrics can be efficiently computed locally for a pair of hyperedges. Also, experiments on synthetically generated hypergraphs (details in Sect. 4.3) show that Jaccard similarity gives the best performance compared to other similarity metrics. Further, a metric similar to it was found to perform well in detecting overlapping communities in unipartite graphs [1]. Hence, we selected Jaccard similarity as the similarity metric for OHC algorithm.

The Jaccard similarity between two hyperedges $e_1 = (a, b, c)$ and $e_2 = (p, q, r)$ (assuming the common node is $a = p$) is computed using Algorithm 5, where $N^X(i)$, $N^Y(i)$ and $N^Z(i)$ denote the set of neighbours of node i of type V^X, V^Y and V^Z, respectively. Note that if $i \in V^X$, then $N^X(i) = \phi$ since the nodes in the same partition are not linked.

Algorithm 5 Compute similarity between two hyperedges

```
Input: hyperedges $e_1 = (a, b, c)$ and $e_2 = (p, q, r)$; $a, p \in V^X$; $b, q \in V^Y$;
  $c, r \in V^Z$
Output: $sim$, Similarity between $e_1$ and $e_2$

  if $a \neq p$ AND $b \neq q$ AND $c \neq r$ then
    $sim \leftarrow 0$ /* Hyperedges are non-adjacent */
  else
    /* w.l.o.g., let $a = p$; Any of the other pairs may be common as well */
    $S_1 \leftarrow N^X(b) \bigcup N^X(c),\ S_2 \leftarrow N^Y(c),\ S_3 \leftarrow N^Z(b)$
    $S_1^{'} \leftarrow N^X(q) \bigcup N^X(r),\ S_2^{'} \leftarrow N^Y(r),\ S_3^{'} \leftarrow N^Z(q)$

    $sim \leftarrow \frac{|S_1 \bigcap S_1^{'}| + |S_2 \bigcap S_2^{'}| + |S_3 \bigcap S_3^{'}|}{|S_1 \bigcup S_1^{'}| + |S_2 \bigcup S_2^{'}| + |S_3 \bigcup S_3^{'}|}$
  end if
  return $sim$
```

3.3 Detecting Communities in Line Graph

With the similarity measure in Algorithm 5, OHC converts the hypergraph to its corresponding line graph where any unipartite community detection algorithm can be used. We experimented with different community detection algorithms to find the best candidate.

Hierarchical Clustering: We can use single-linkage hierarchical clustering to construct a dendrogram. We start with each node in the line graph as an individual cluster, then, at each step, the two most similar clusters are merged. This procedure continues until all nodes belong to a single cluster, and cutting this dendrogram at some suitable level gives the final clusters of nodes. The optimal level for the cut can be decided based on the *partition density* metric [1].

Fast Modularity Optimization: Clauset et al. [10] proposed a fast and greedy approach to implement the modularity maximization technique proposed by Newman [29]. Starting from a set of isolated nodes in the graph, the links (which are present in the original graph) are iteratively added to produce the largest possible increase in the modularity at each step. Time complexity of this algorithm is $O(n \cdot \log^2 n)$ where n is the number of nodes in the graph.

Louvain Method: Blondel et al. [3] proposed a multistep technique (named "Louvain algorithm") where communities are first detected by locally optimizing modularity in each node's neighbourhood; in the next step, a weighted graph is formed where the communities detected are super-nodes. These two steps are iterated until modularity (which is always computed in the original graph) does not increase any further. This algorithm is of complexity $O(m)$ where m is the number of edges in the original graph.

Infomap: Rosvall et al. [33] showed that finding the best cluster structure of a graph is equivalent to optimizing *minimum description length* of a random walk taking place on the graph. This optimization can be carried out with greedy search and simulated annealing. Time complexity of this algorithm (named as "Infomap" by the authors) is also $O(m)$.

Choice of a Community Detection Algorithm: We compared the performances of all the above community detection algorithms using synthetic hypergraphs (details in Sect. 4.3). The Infomap algorithm is found to perform better than the other algorithms. Lancichinetti et al. [22] also showed that for community detection in large graphs, Infomap can identify communities more accurately as compared to several other algorithms. Further, the relatively low computational complexity of Infomap allows its use on line graphs of large real folksonomies. Therefore, Infomap is used as the community detection algorithm.

3.4 Computational Complexity of OHC Algorithm

Now that the choice of the similarity metric for hyperedges and the unipartite community detection algorithm has been made, we analyse the computational complexity of OHC algorithm. Let the number of nodes in the hypergraph be n and average node degree be d, which implies that the number of hyperedges will be $\frac{n \cdot d}{3}$. Each hyperedge will, on average, be adjacent to $3 \cdot (d-1)$ other hyperedges. So, the *line graph* will have $\frac{n \cdot d}{3}$ nodes and $\frac{n \cdot d}{3} \times 3 \cdot (d-1) = n \cdot d \cdot (d-1) = O(n \cdot d^2)$ edges. Since complexity of Infomap algorithm is linear in the size of the graph and similarity calculation in the hypergraph also takes $O(n \cdot d^2)$ time, the time complexity of OHC is $O(n \cdot d^2)$.

It is to be noted that the real-world folksonomies are known to be sparse, having small average degree d. So, essentially the complexity of OHC becomes $O(n)$ which makes this algorithm scalable to large real-world folksonomies. The performance of OHC is evaluated in the next section.

4 Experiments and Evaluation

In this section, we evaluate the performance of the presented OHC algorithm. We first compare different choices of similarity metrics as well as community detection algorithms for line graph to be used in OHC (as discussed in Sects. 3.2 and 3.3, respectively). Then, we compare the performance of OHC algorithm with the algorithms by Wang et al. [35] and Papadopoulos et al. [32], which are henceforth referred to as "CL" and "HGC", respectively.[3]

Since evaluation of clustering is difficult without the knowledge of the 'ground truth' regarding the community memberships of the nodes, we have used *synthetically generated hypergraphs* with a known community structure for evaluation of the algorithms. We now discuss the generation of synthetic hypergraphs and the metric used to evaluate the algorithms, followed by the results of experiments on synthetic hypergraphs.

4.1 Generation of Synthetic Hypergraphs

Synthetic hypergraphs are generated using a modified version of the method used in [35]. The generator algorithm takes the following as input—(i) number of nodes in a partition (all three partitions are assumed to contain equal number of nodes), (ii) number of communities C, (iii) fraction γ of all nodes which belong to multiple communities and (iv) hyperedge density β (i.e. the fraction of total number of possible hyperedges that actually exist in the hypergraph).

Initially, the nodes in each partition are evenly distributed among each community under consideration (e.g. $|V^X|/C$ nodes in the partite set V^X are assigned to each of the C communities). Subsequently, γ fraction of nodes is selected at random from each of V^X, V^Y and V^Z, and each selected node is assigned to some randomly chosen communities apart from the one it has already been assigned to. Nodes assigned to the same community are then randomly selected, one from each partition, and interconnected with hyperedges. The number of hyperedges is decided based on the specified density β.

Users in the real-world folksonomies often tag a few resources related to the topics that are different from their topics of primary interest, according to their transient interests at different times. Though such tagging is typically much fewer than those related to the primary interests of users, it can adversely affect the performance of algorithms that assign a single community to nodes. To investigate how the algorithms perform in presence of such links, a second set of synthetic hypergraphs is generated, where 1% of the generated hyperedges interconnect randomly selected nodes from *different* communities. We shall refer to these hyperedges as "scattered" hyperedges.

The above assignment of communities to nodes constitutes the "ground truth". After hypergraphs are generated, information about the communities is hidden and then communities are detected from the hypergraph by the three different community detection algorithms. The community structure detected by each algorithm is compared with the ground truth using the metric "normalized mutual information (NMI)" which is explained below.

[3]We acknowledge the authors of [32, 35] for providing us with the implementations of their algorithms.

4.2 Normalized Mutual Information

Normalized mutual information (NMI) is an information-theoretic measure of similarity between two partitionings of a set of elements, which can be used to compare two community structures for the same graph (as identified by different algorithms). The traditional definition of NMI does *not* consider the case of a node being present in multiple communities. Hence, Lancichinetti et al. [23] proposed an alternative definition of NMI considering overlapping communities. According to [23], given two community structures/node partitions X and Y, NMI is defined as

$$NMI(X,Y) = 1 - \frac{1}{2}\Big(H(X|Y)_{norm} + H(Y|X)_{norm}\Big)$$

where

$$H(X|Y)_{norm} = \frac{1}{N_X}\sum_i \frac{\min_{j\in\{1,2,\ldots,N_X\}} H(X_i|Y_j)}{H(X_i)}$$

$$H(Y|X)_{norm} = \frac{1}{N_Y}\sum_i \frac{\min_{j\in\{1,2,\ldots,N_Y\}} H(Y_i|X_j)}{H(Y_i)}$$

Here, $H(X)$ and $H(Y)$ are entropies of X and Y. $H(Y|X)$ and $H(X|Y)$ are conditional entropies and N_X and N_Y are number of clusters in X and Y, respectively.

The NMI is computed in two steps. First, the pairs of clusters that are closest to each other are found from two clusterings. Second, the mutual information between those pairs of clusters is averaged. The NMI value is in the range [0, 1]; the higher the NMI value, the more similar are the two community structures (refer to [23] for details).

4.3 Design Choices for OHC

To decide a similarity metric and a community detection method (as required by OHC), we generated synthetic hypergraphs having various hyperedge densities β = 0.1, 0.2, …, 1.0. In each of these hypergraphs, 10% of the nodes in each partition belonged to multiple communities (i.e. γ = 0.1).

First, we compare the performances of the different similarity metrics. Infomap is used as the community detection method in line graph for all cases. The NMI values (considering the true and detected community structures) are shown in Fig. 1a. We can see that the Jaccard similarity metric consistently gives the best result.

Once Jaccard similarity has been chosen as the desired similarity metric, we compare different community detection methods which can be applied on line graph (discussed in Sect. 3.3). Figure 1b shows the comparison of NMI values—across all hyperedge densities, Infomap algorithm is found to perform better than the other algorithms.

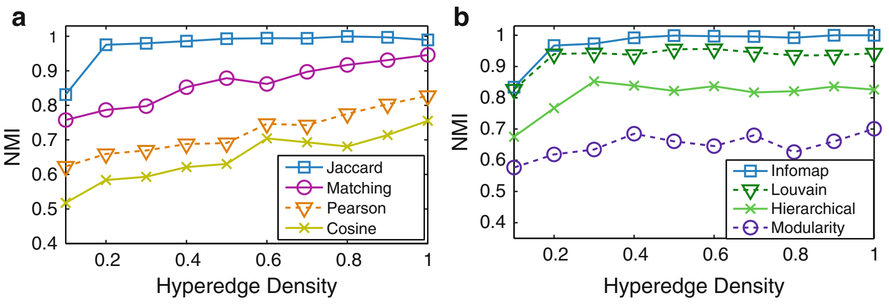

Fig. 1 Comparison of the NMI values with varying hyperedge density for (**a**) different similarity metrics and (**b**) different community detection algorithms

4.4 Comparing OHC with the Other Algorithms

The CL and HGC algorithms produce only the user and the tag communities, respectively. Hence, while calculating the NMI value for these algorithms, we have used the community memberships of only the user (respectively, the tag) nodes according to the ground truth. On the other hand, the proposed OHC algorithm gives composite communities containing all three types of nodes. Hence, to evaluate the performance of OHC, we have considered the community memberships of all three types of nodes.

For all the following experiments, $|V^X| = |V^Y| = |V^Z| = 200$ and number of communities $C = 20$. For each result, random hypergraphs were generated 50 times using the same set of parameter values, and the average performances over all those 50 runs are reported.

Performance w.r.t. Number of Hyperedges: To study how the number of hyperedges affects the performance of the clustering algorithms, we generated synthetic hypergraphs having various hyperedge densities $\beta = 0.1, 0.2, \ldots, 1.0$. In each of these hypergraphs, 10% of nodes in each partition belonged to multiple communities (i.e. $\gamma = 0.1$). The NMI values for the three algorithms are shown in Fig. 2a. Across all the hyperedge densities, OHC performs significantly better than HGC and CL algorithms. A possible explanation for this is that the proposed OHC algorithm utilizes the complete tripartite structure of the hypergraph, whereas both CL and HGC algorithms work on unweighted projections which is known to result in loss of a significant part of the information contained in the original tripartite network [18].

Note that even for very low hyperedge densities, when detecting community structures is difficult, the proposed OHC algorithm performs very well, resulting in NMI scores above 0.8. This makes OHC suitable for real-world folksonomies where the density of hyperedges is typically low.

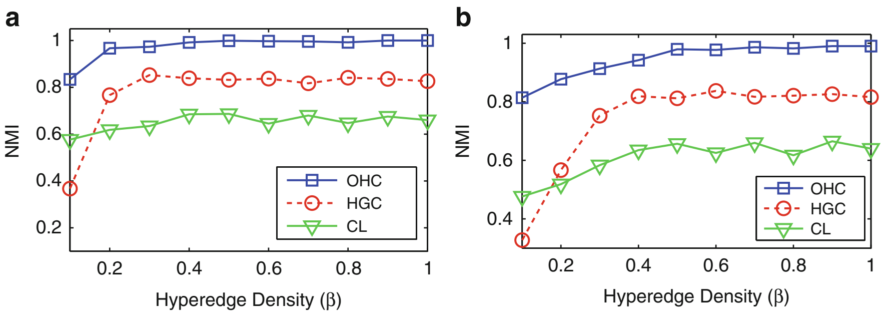

Fig. 2 Comparison among OHC, CL and HGC algorithms—variation of the NMI values with varying hyperedge density (**a**) without scattered hyperedges (**b**) in presence of 1% scattered hyperedges

Performance in Presence of Scattered Hyperedges: We have also experimented with synthetic hypergraphs having 1% of total hyperedges as "scattered". Figure 2b shows the result—as the presence of scattered hyperedges disturbs the community structure, the performance of all three algorithms degrades as expected. However, the performance of OHC is still better than HGC and CL algorithms—the NMI scores for OHC remain above 0.8 which signifies its effectiveness in detecting community structure even in the presence of noisy or scattered hyperedges.

Performance w.r.t. Fraction of Nodes in Multiple Communities: A node, which belongs to multiple communities, creates links to nodes in all those communities. Hence, from the perspective of a particular community, the links created by this member node to nodes in other communities reduce the exclusivity of this particular community. Therefore, as the number of nodes in multiple communities increases, the community structure becomes more difficult to identify. We now study how this affects the performance of the algorithms.

We generated synthetic hypergraphs by varying the fraction of nodes in multiple communities $\gamma = 0.1, 0.2, \ldots, 1.0$ while keeping hyperedge density β constant at 0.2. This low value of hyperedge density was chosen to measure the effectiveness of the algorithms in sparse environment (as in real folksonomies). Figure 3a shows that OHC performs consistently better than HGC and CL algorithms. As the community structure becomes more and more complex, information loss as a result of projections becomes increasingly more crucial. Hence, the performance of HGC and CL algorithms degrades sharply with the increase in γ, while the performance of OHC algorithm shows relatively much greater stability.

Performance w.r.t. Size of Real Community: We also observed how the performances of different algorithms are affected by the size of each real community. Hypergraphs having 200 nodes in each partition were generated while changing the number of real communities. Here, hyperedge density is fixed at 0.2 and 10% of total nodes belong to multiple communities. The results are shown in Fig. 3b. When

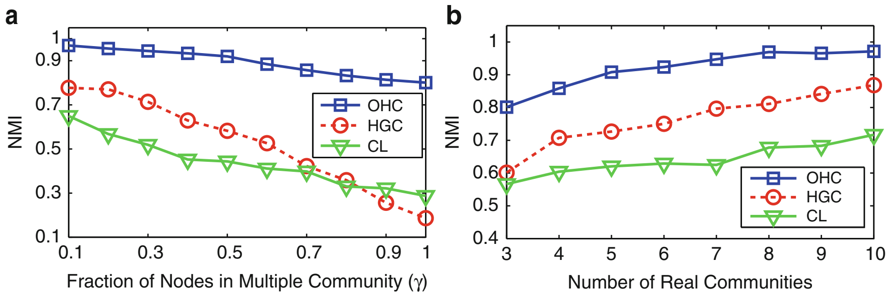

Fig. 3 Another comparison among OHC (proposed), CL and HGC algorithms—variation of the NMI values keeping hyperedge density constant at 0.2 and changing (**a**) fraction of nodes in multiple communities and (**b**) number of "real" communities

number of nodes in one community is large, random assignment of hyperedges during the generation of synthetic hypergraphs may create smaller communities inside one large community. Community detection algorithms find these smaller communities rather than the large encompassing community. For this reason, as the number of real communities increases, the size of each community decreases, enabling better NMI performance. Here also, the performance of OHC is superior than CL and HGC algorithms.

The above experiments clearly validate the motivation that considering the complete tripartite structure of hypergraphs can result in better identification of the community structure, as compared to considering projections (as done in prior studies). In the next section, we use OHC to study the community structure of real-world folksonomies.

5 Experiments on Real Folksonomies

In this section, we apply OHC algorithm to gain insights into the community structures prevalent in the real-world folksonomies. For this, we use publicly available datasets [6] consisting of snapshots of the popular folksonomies Delicious, LastFm and MovieLens. The statistics of these datasets are summarized in Table 1.

5.1 Using OHC to Detect Overlapping Communities

For all three datasets, OHC successfully groups semantically related resources as well as the tags and the users tagging these resources. As an illustration, Table 2 shows the resources and the tags placed in some example communities for each of

Table 1 Statistics of real folksonomy datasets

Dataset	# users	# resources	# tags	# hyperedges
Delicious	1,867	69,226	53,388	437,593
LastFm	1,892	17,632	11,946	186,479
MovieLens	2,113	10,197	13,222	47,957

Table 2 Examples of communities detected by proposed OHC algorithm. The algorithm successfully clusters nodes related to a common semantic theme (Column 2). Nodes related to multiple themes (boldfaced and italicized) are placed in overlapping communities

Community	Theme	Example of member nodes
LastFm artists	Hard rock	***Van Halen, Deep Purple, Aerosmith***, Alice Cooper, Guns N' Roses, Scorpions, White Lion, Bad Company, Bon Jovi, Hardline
(resources)	Heavy metal	***Van Halen, Deep Purple, Aerosmith***, Iron Maiden, Motorhead, Black Sabbath, Metallica, Twisted Sister, Crazy Lixx
LastFm tags	Metal	***Blues rock, psychedelic rock, rap metal, nu metal***, metal, progressive metal, speed metal, metalcore, viking metal, power metal
	Rock	***Blues rock, psychedelic rock, rap metal, nu metal***, progressive rock, soft rock, gothic rock, punk rock, hard rock, pop rock
MovieLens movies	Superhero	***The Incredibles, Shrek, Shrek 2, The Incredible Hulk***, Batman Begins, Batman Returns, Spider-Man, Superman, X-Men
(resources)	Animation	***The Incredibles, Shrek, Shrek 2, The Incredible Hulk***, Kung fu Panda, Beowulf, Ratatouille, Finding Nemo, Toy Story
MovieLens tags	Criticism	***Violent, brutal***, too violent, waste of celluloid, disturbing, junk, tragically stupid, lousy script, waste of money, confusing plot
	Violence	***Violent, brutal***, violence, murder, fatality, civil war, great villain, dark, Spanish civil war, serial killer, Vietnam war, world war II
Delicious tags	Web 2.0	Social networking, social web, php, drupal, xml, cms, webdesign, css3, Twitter, Skype, Ruby, Facebook, Snippets, Wikipedia, blog

the three datasets. It is evident that the resources and the tags that are placed in the same community are often related to a common semantic theme.

A closer look at Table 2 reveals that the algorithm also correctly identifies nodes that are related to multiple overlapping communities (themes). For instance, the band *Van Halen* is placed in two different communities detected from LastFm. The

Wikipedia article about Van Halen[4] justifies this placement pointing their genre as both "hard rock" and "heavy metal".

A *nonoverlapping* community detection algorithm would have placed this node in either of the two communities (assume "hard rock"). Thus, community-based recommendation systems (which recommend resources to users based on common memberships in communities) would have overlooked the fact that this resource is a candidate for recommendation to users who are interested in "heavy metal" as well. OHC algorithm places the resource in both communities, thus raising the chances of this resource being recommended to users interested in "hard rock" as well as "heavy metal".

5.2 Evaluation of Communities Detected

The principal difficulty in evaluating the communities detected in case of real folksonomies is the absence of the "ground truth" regarding the community memberships of nodes in folksonomies, since their huge size makes it impossible for human experts to evaluate the quality of identified communities. Hence, we use the following two methods for evaluation:

(1) We use the graph-based metric *conductance*, which has been shown to correctly conform with the intuitive notion of communities and is extensively used for evaluating the quality of communities in online social networks [24].

(2) In case of the folksonomies which allow users to form a social network among themselves, we can assume that users having similar interests are likely to be linked in the social network or at least to have a common social neighbourhood (a property known as *homophily* [26]). We utilize this notion to evaluate the user communities detected by CL algorithm and the user nodes in the communities identified by OHC algorithm.

Comparison of Conductance Value

As conductance is defined only for unipartite networks, we compare the tag communities detected by HGC with the tag nodes in the communities identified by OHC algorithm. Conductance values range from 0 to 1 where a *lower* value signifies *better* community structure [24].

Figure 4 shows the cumulative distribution of the conductance values of the detected tag communities by the two algorithms. Across all three datasets, OHC produces more communities having lower conductance values, which implies that OHC can find communities of better quality than obtained by HGC algorithm. The reason for this superior performance is that OHC groups semantically related nodes

[4]http://en.wikipedia.org/wiki/Van_Halen

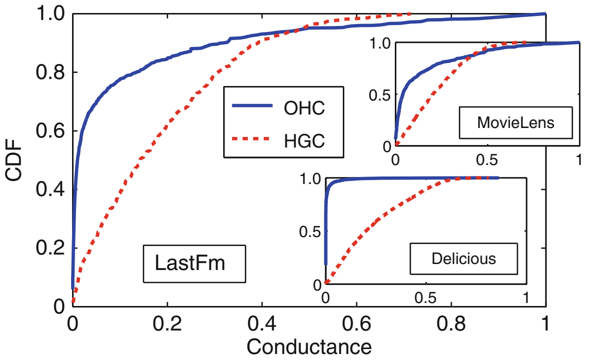

Fig. 4 Comparison between OHC (proposed) and HGC algorithms—cumulative distribution of conductance values of tag communities obtained from LastFm (main plot), Delicious and MovieLens (both inset)

into relatively smaller cohesive communities instead of creating a few number of generalized large communities. For examples of semantically related communities identified by OHC, refer to Table 2.

Comparing Detected User Communities with Social Network:
In case of the folksonomies, which allow users to form a social network,[5] there are two types of relationships among users—explicit social connections (in the social network) and implicit connections through their tagging behaviour (e.g. tagging the same resource). A community detection algorithm for hypergraphs utilizes the implicit relationships to identify the community structure, and we propose to evaluate the detected community structure using the explicit connections that the users themselves create in the social network. For instance, if a large fraction of the users who are socially linked (or share a common social neighbourhood in the social network) are placed in the same community (by an algorithm), the detected community structure can be said to group together users having common interests.

To compare the community structure identified by two algorithms, we consider the user pairs who are within a certain distance from each other in the social network (where distance 1 implies directly connected friends) and compute the fraction of such user pairs who has been placed in a common community by each algorithm. Figure 5a shows the result for the proposed OHC algorithm and the CL algorithm, for the LastFm dataset. Across all distances, OHC places a larger number of user pairs who share a common social neighbourhood, in a common community, as compared to the CL algorithm. Also, as the distance between two users in the social network increases, both algorithms put a smaller fraction of such user pairs in the same community.

[5]LastFm has an undirected social network, while in Delicious, a user can be a "fan" of another user, but this fan relationship may or may not be reciprocated. We assumed two users are linked

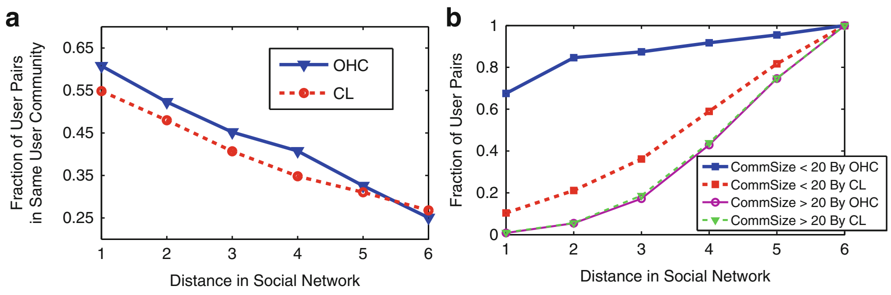

Fig. 5 Comparing the community structures detected by OHC (proposed) and CL algorithms with the social networks in (**a**) LastFm and (**b**) Delicious

We also investigate the reverse question—among the users who are placed in a common community (by a community detection algorithm), what fraction of these users is actually connected in the social network (or share a common social neighbourhood)? While investigating this question, it is to be noted that "quality" of large communities detected by community detection algorithms is known to be lower than smaller communities [24]. Hence, it is meaningful to answer this question for detected communities taking their size into consideration. Figure 5b shows the fraction of user pairs that is placed in a common community by the OHC and CL algorithms, which are within a certain distance in the social network, for the Delicious dataset. For detected user communities of size lesser than 20, about 70% of the user pairs that are placed in a common community by OHC are actually friends in the social network, whereas the corresponding value for the CL algorithm is much lesser. However, for larger detected communities (having more than 20 users), the fraction of user pairs who share a common social neighbourhood is much lower and almost identical for both algorithms.

The above results clearly show that the presented OHC algorithm can detect much better community structure in real folksonomies, as compared to the existing CL and HGC algorithms. The fact that a very large fraction of the user pairs that are placed in a common community by OHC are actually friends shows that OHC can be used to identify potential friends directly from the hypergraph structure.

6 Conclusion

In this chapter, we presented the "Overlapping Hypergraph Clustering" (OHC) algorithm which detects overlapping communities considering the full tripartite hypergraph structure of folksonomies. Through extensive experiments on synthetic

if they belong to a *mutual* fan relationship. In the LastFm and Delicious datasets, there are 12,717 and 7,668 bidirectional links, respectively, in the social network among users.

as well as real folksonomy networks, we showed that OHC algorithm outperforms existing algorithms that consider projections of hypergraphs.

In large folksonomies, it is difficult for an individual user to find other like-minded users as well as resources of her interest. OHC algorithm successfully groups nodes into multiple communities where each community represents a topic of interest. Based on these interests, like-minded users as well as resources can be identified. Thus, this algorithm can be effectively used in recommending interesting resources and friends to users in folksonomies. Our future work will be to build such a recommendation system taking advantage of the effectiveness of the algorithm.

References

[1] Y.-Y. Ahn, J.P. Bagrow, S. Lehmann, Link communities reveal multiscale complexity in networks. Nature **466**(7307), 761–764 (2010)

[2] J. Baumes, M.K. Goldberg, M.S. Krishnamoorthy, M.M. Ismail, N. Preston, Finding communities by clustering a graph into overlapping subgraphs, in *Proc. IADIS Conference on Applied Computing*, pp. 97–104, 2005

[3] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks. J. Stat. Mech. Theor Exp. **2008**(10), (2008)

[4] M. Brinkmeier, J. Werner, S. Recknagel, Communities in graphs and hypergraphs, in *Proc. ACM Conference on Information and Knowledge Management (CIKM)*, 2007

[5] S.R. Bulo, M. Pelillo, A game-theoretic approach to hypergraph clustering. Adv. Neural Inform. Process. Syst. **22**, 1571–1579 (2009)

[6] I. Cantador, P. Brusilovsky, T. Kuflik, 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011), in *Proc. ACM Conference on Recommender Systems (RecSys)*, 2011

[7] C. Cattuto, C. Schmitz, A. Baldassarri, V.D.P. Servedio, V. Loreto, A. Hotho, M. Grahl, G. Stumme, Network properties of folksonomies. AI Comm. **20**(4), 245–262 (2007)

[8] A. Chakraborty, Credibility measurement of users in e-learning forums, in *Proc. National Convention of Computer Engineers*, February 2012

[9] A. Chakraborty, S. Ghosh, N. Ganguly, Detecting overlapping communities in folksonomies, in *Proc. ACM Hypertext Conference*, June 2012

[10] A. Clauset, M.E.J. Newman, C. Moore, Finding community structure in very large networks. Phys. Rev. E **70**, 066111 (2004)

[11] T.S. Evans, R. Lambiotte, Line graphs, link partitions, and overlapping communities. Phys. Rev. E **80**, 016105 (2009)

[12] S. Fortunato, Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)

[13] S. Ghosh, P. Kane, N. Ganguly, Identifying overlapping communities in folksonomies or tripartite hypergraphs, in *Proc. ACM Conference on World Wide Web (WWW) companion volume*, pp. 39–40, Mar 2011

[14] M. Girvan, M.E.J. Newman, Community structure in social and biological networks. Proc. Natl. Acad. Sci. **99**(12), 7821–7826 (2002)

[15] M. Girvan, M.E.J. Newman, Finding and evaluating community structure in networks. Phys. Rev. E 69 (2004)

[16] S. Gregory, Finding overlapping communities using disjoint community detection algorithms, in *Complex Networks*, vol. 207 of *Studies in Computational Intelligence* (Springer, Berlin, 2009), pp. 47–61

[17] S. Guha, R. Rastogi, K. Shim, ROCK: a robust clustering algorithm for categorical attributes. Inform. Syst. **25**(5), 345–366 (2000)

[18] R. Guimera, M. Sales-Pardo, L.A.N. Amaral, Module identification in bipartite and directed networks. Phys. Rev. E **76**, 036102 (2007)

[19] T.G. Kolda, B.W. Bader, Tensor decompositions and applications. SIAM Rev. **51**(3), 455–500 (2009)

[20] I. Konstas, V. Stathopoulos, J.M. Jose, On social networks and collaborative recommendation, in *Proc. ACM SIGIR Conference*, pp. 195–202, 2009

[21] A. Lancichinetti, S. Fortunato, Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Phys. Rev. E **80**(1), 9 (2009)

[22] A. Lancichinetti, S. Fortunato, Community detection algorithms: a comparative analysis. Phys. Rev. E **80**, 056117 (2009)

[23] A. Lancichinetti, S. Fortunato, J. Kertesz, Detecting the overlapping and hierarchical community structure in complex networks. New J. Phys. **11**, 033015 (2009)

[24] J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Statistical properties of community structure in large social and information networks, in *Proc. ACM Conference on World Wide Web (WWW)*, 2008

[25] Y.-R. Lin, J. Sun, P. Castro, R. Konuru, H. Sundaram, A. Kelliher, Metafac: community discovery via relational hypergraph factorization, in *Proc. ACM SIGKDD Conference*, pp. 527–536, 2009

[26] M. McPherson, L. Smith-Lovin, J. Cook, Birds of a feather: Homophily in social networks. Ann. Rev. Sociol. **27**, 415–444 (2001)

[27] T. Murata, Detecting communities from social tagging networks based on tripartite modularity, in *Proc. Workshop on Link Analysis in Heterogeneous Information Networks*, July 2011

[28] N. Neubauer, K. Obermayer, Towards community detection in k-partite k-uniform hypergraphs, in *Proc. Workshop on Analyzing Networks and Learning with Graphs*, pp. 1–9, 2009

[29] M.E.J. Newman, Fast algorithm for detecting community structure in networks. Phys. Rev. E **69**, 066133 (2004)

[30] V. Nicosia, G. Mangioni, V. Carchiolo, M. Malgeri, Extending the definition of modularity to directed graphs with overlapping communities. J. Stat. Mech. Theor Exp. **3**, 03024 (2008)

[31] G. Palla, I. Derenyi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society. Nature **435**, 814–818 (2005)

[32] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, A graph-based clustering scheme for identifying related tags in folksonomies, in *Proc. Conference on Data Warehousing and Knowledge Discovery (DaWaK)*, pp. 65–76, 2010

[33] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure. Proc. Natl. Acad. Sci. **105**, 1118–1123 (2008)

[34] A. Vazquez, Finding hypergraph communities: a Bayesian approach and variational solution. J. Stat. Mech. Theor Exp. **2009**, P07006 (2009)

[35] X. Wang, L. Tang, H. Gao, H. Liu, Discovering overlapping groups in social media, in *Proc. IEEE Conference on Data Mining (ICDM)*, pp. 569–578, 2010

[36] S. Xu, S. Bao, B. Fei, Z. Su, Y. Yu, Exploring folksonomy for personalized search, in *Proc. ACM SIGIR Conference*, pp. 155–162, 2008

[37] D. Zhou, J. Huang, B. Scholkopf, Learning with hypergraphs: clustering, classification, and embedding, in *Proc. Advances in Neural Information Processing Systems*, 2006