

Community Detection in Tagging-Induced Hypergraphs

Nicolas Neubauer
Neural Information Processing Group
Technische Universität Berlin
Berlin, Germany
neubauer@cs.tu-berlin.de

Klaus Obermayer
Neural Information Processing Group
Technische Universität Berlin
Berlin, Germany
oby@cs.tu-berlin.de

1. INTRODUCTION

Social bookmarking services like Delicious let users annotate their bookmarks with “tags”, freely chosen keywords to facilitate later retrieval. The success of these systems and the central storage of the generated data has led to the creation of vast networked datasets comprised of (document, user, tag)-triples. These datasets can be thought of as commented traces of human online behaviour, and as such have been subject to a significant amount of research lately, showing, e.g. systematic convergence properties[6]. This interest is further sparked by their peculiar theoretical properties: If the triples are interpreted as edges, the resulting graphs are 3-partite 3-uniform (or 3,3-) hypergraphs, i.e. generalized graphs in which each edge connects three nodes from three distinct partitions. Many established methods from complex network analysis need to be generalized in order to be properly applicable to these structures[1, 5, 10]. Here, we discuss the theoretical intricacies and practical benefits of generalizing community detection.

Community detection aims to identify groups of closely connected nodes in graphs. Identifying related nodes may be useful in itself, but furthermore helps in examining complex networks’ macrostructure by collapsing those nodes into clusters and examining the network of clusters instead. Given the complexity, noise and sheer size of social bookmarking datasets, such reductions appear crucial for distilling the latent information contained. Out of a vast space of possible approaches [4], we will examine the particularly well-established approach of modularity optimization[12] and its possible extensions for hypergraphs.

We will proceed to discuss the challenges for generalizing modularity. In addition to discussing earlier work, we will then introduce a new approach which works natively on hypergraphs. After evaluating the different approaches, we turn to the practical side and introduce an interactive visualization tool that not only illustrates the differences between different algorithms, but more generally demonstrates the richness of the structures found in social bookmarking data.

2. MODULARITY AND HYPERGRAPHS

For a graph $G = (V, E)$, let M be the number of edges $|E|$ and A the adjacency matrix. Let σ be a function assigning vertices V to communities. Then Newman’s modularity Q of a community assignment σ is defined as

$$Q = \frac{1}{2M} \sum_{(i,j) \in V \times V} [A_{ij} - P_{ij}] \delta(\sigma(i), \sigma(j))$$

where $\delta(x, y) = 1$ iff $x = y$ and 0 otherwise [12]. This rewards the clustering of nodes with above-expectation adjacency, where expected adjacency is computed relative to a null-model which often defaults to $P_{ij} = \frac{k_i k_j}{2M}$ (configuration model), where k_i is the degree of node i . Optimizing this measure is a well-established method for community detection, and various efficient methods for this optimization exist [2, 12]. However, its application on (3,3)- or, more generally, (k, k) hypergraphs meets the following challenges:

Distinct community structures Different partitions may have different community structures: Communities in one partition may not exactly correspond to another community in another partition; there may also be different numbers of communities. Therefore, k different sets of communities should be supported.

Correspondence instead of equality If we assume distinct community structures per partition, the notion of connected nodes (which, by definition of partiteness, come from different partitions) belonging to the “same” community becomes obsolete. Instead, a generalized notion of elements from different partitions belonging to *corresponding* communities is required. We also need to take care not to punish the non-adjacency of (necessarily non-adjacent) elements in the same community within a single partition.

Hyper-incidence While the previous two points have been pointed out before (e.g. [8]) for bipartite graphs (which are, after all, (2,2)-hypergraphs), dealing with hypergraphs additionally requires us to generalize to k elements the formerly binary concept of two elements being either in corresponding communities or not.

2.1 Examples

Figure 1 shows three possible scenarios of community structure in a 3,3-hypergraph. We will use them to highlight the challenges presented in the previous section. In all figures, squares represent (communities of) documents, triangles are users, and circles tags.

SIMPLE: In the simplest case (Figure 1a)), each community in one partition corresponds to exactly one community in the two other partitions.

OVERLAPPING: In Figure 1b), there is only one community of tags, but two communities in the other two partitions: Imagine a community of tags around the topic of “memory” which may be used by communities of computer scientists and psychologists to annotate distinct communities of documents. This is the simplest example of why dif-

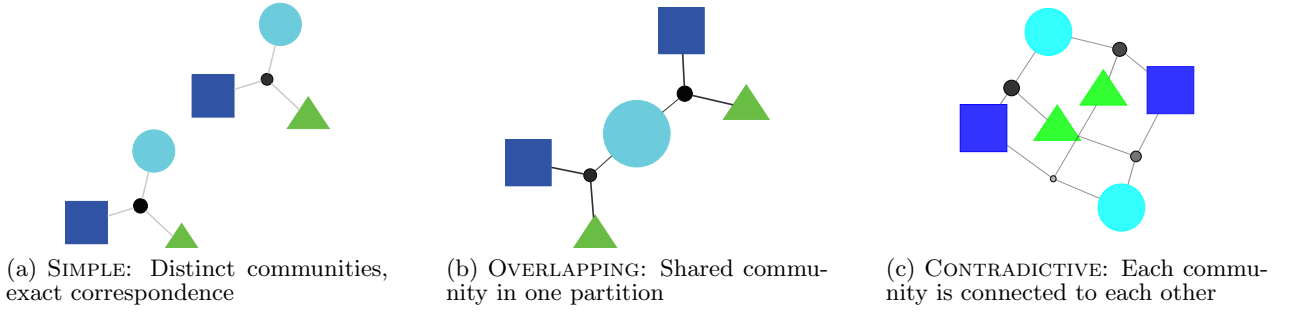


Figure 1: Possible community constellations in 3,3-hypergraphs.

ferent numbers of communities for different partitions might be necessary (challenge 1) and why it does not make sense to talk about objects from different partitions being from the “same” community (challenge 2); there is no single e.g. document community to which the “memory” tags could be said to belong.

CONTRADICTIVE: In Figure 1c), each community is connected to each other community, but in a contradicting fashion. Imagine two groups of people A and B with different tastes, two communities of tags (positive and negative), and two sets X and Y of, say, movies. This example shows a situation where people from group A tag all movies X positively, the other movies negatively, and the other group of people acts exactly in the opposite way. This highlights the extension of correspondence noted in challenge 3: The question “Do A and X correspond?” needs to be replaced by the question “Do A, X and + correspond?”

3. METHODS

We will now introduce different modularity measure which meet these requirements to varying degrees.

3.1 Unipartite Approach

The simplest approach is to reduce a (k, k) -hypergraph H to a simple graph. This can be achieved by creating a new graph $G(H)$ from H such that for each edge (d, u, t) from H , three new edges (d, u) , (d, t) , (u, t) are created in G , where multiple occurrences of the new edges are represented by an increased weight.

This approach lets us apply standard modularity detection methods, without however addressing the afore-mentioned challenges[11]. It still serves as a valuable baseline to confirm whether these challenges really need to be dealt with.

3.2 Multi-Bipartite Approach

The multi-bipartite approach[11] uses Murata’s bipartite modularity[8] Q_M given by

$$Q_M(G, \sigma) = \sum_l (e_{lm} - a_l a_m), \quad m = \operatorname{argmax}_m (e_{lm}),$$

$$e_{lm} = \frac{1}{2M} \sum_{i \in V: \sigma(i)=l} \sum_{j \in V: \sigma(j)=m} A(i, j), \quad \text{and } a_l = \sum_m e_{lm}$$

This modularity measure approaches the first two challenges for bipartite graphs: For each community l , the community m that elements from l share the most edges with is chosen as the corresponding community. The usual modularity

can then be applied, summing however over edges between corresponding instead of identical communities.

The multi-bipartite approach works by reducing the hypergraph H into several bipartite graphs, each representing one of the $\frac{k(k-1)}{2}$ binary relations. So, edges (d, u, t) become new edges (d, u) in G_{DU} , (d, t) in G_{DT} , and (u, t) in G_{UT} . We then define the overall modularity as the average bipartite modularity of the three graphs:

$$Q_{MB} = \frac{1}{3}Q_M(H_{DU}, \sigma) + \frac{1}{3}Q_M(H_{DT}, \sigma) + \frac{1}{3}Q_M(H_{UT}, \sigma)$$

This approach gives us a modularity measure for hypergraphs which considers the first two challenges. The particular hypergraphical structure addressed in challenge 3 however is still broken down into lower-order relations.

3.3 Multipartite Approaches

If we rewrite unipartite modularity as

$$Q = \sum_{i \in C} (e_{ii} - a_i^2) \\ = \sum_{(i,j) \in C \times C} (e_{ij} - a_i a_j) f(i, j), \\ f(i, j) = \delta(i, j),$$

we can see the similarity to Q_M , which can be rewritten as

$$Q_M = \sum_{i \in C} (e_{ij} - a_i a_j), j = \operatorname{argmax}_k (e_{ik}) \\ = \sum_{(i,j) \in C \times C} (e_{ij} - a_i a_j) f(i, j), \\ f(i, j) = \begin{cases} 1 & \text{if } j = \operatorname{argmax}_k (e_{ik}) \\ 0 & \text{else} \end{cases}$$

Identity in Q simply becomes one possible choice for a correspondence function f which is replaced by an argmax by Q_M . Rewriting Q_M one step further as

$$Q_M = \sum_{(i,j) \in C_1 \times C_2} (e_{ij} - a_i a_j) (f_1(i, j) + f_2(i, j))$$

where $f_1(i, j) = f(i, j)$ and $f_2(i, j) = f(j, i)$, the generalization to the multipartite¹ case becomes obvious:

$$Q_{MP} = \sum_{(i,j,k) \in C_1 \times C_2 \times C_3} (e_{ijk} - a_i a_j a_k) \left(\sum_{d=1}^3 f_d(i, j, k) \right)$$

¹Even though the notation spells out the tripartite case, we think the generalization to higher dimensions is now obvious and therefore refer to that method as “multipartite” modularity.

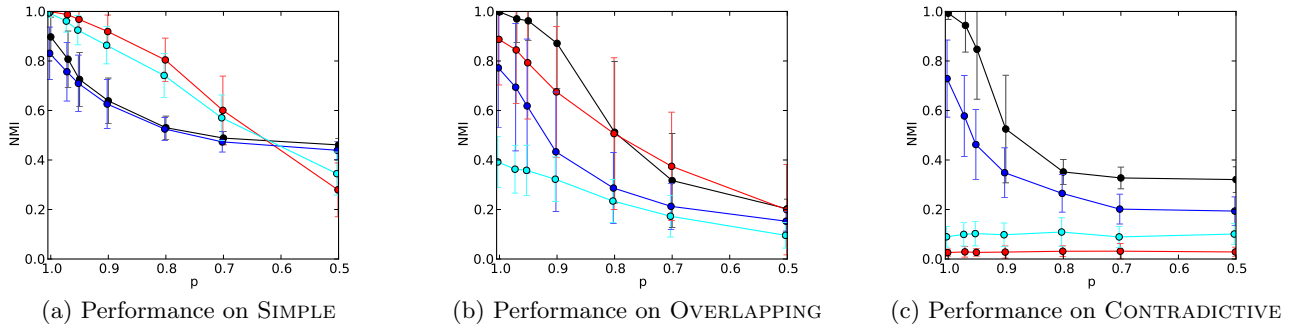


Figure 2: Reconstruction performance (y) on the three synthetic datasets, relative to noise (x), using graphs with 10 nodes per community and partition. Error bars indicate standard deviation over 100 samples. Turquoise: Unipartite Modularity, Red: Multi-Bipartite Modularity, Blue: Murata’s Tripartite Modularity (Multipartite with ArgMax), Black: Linear Multipartite Modularity

3.3.1 Murata’s Tripartite Modularity

Murata has proposed direct generalization of his approach to tripartite hypergraphs[9] which can be seen as an instance of this formulation with

$$f_1(i, j, k) = \begin{cases} 1 & \text{if } (j, k) = \operatorname{argmax}_{(j', k')} (e_{ij'k'}) \\ 0 & \text{else} \end{cases}$$

and f_2, f_3 correspondingly. While this approach considers all three challenges, we will see in the next section that optimization of this measure yields instable results. Our explanation is that the fixation on a single corresponding community/pair of communities is actually too restrictive.

3.3.2 Linear Multipartite Modularity

We therefore propose to further generalize correspondence to a real-valued measure, able to represent ambiguous relations, where a single community may correspond to several communities at the same time:

$$f_1(i, j, k) = \frac{e_{i,j,k}}{a_i}, \quad f_2(i, j, k) = \frac{e_{i,j,k}}{a_j}, \quad f_3(i, j, k) = \frac{e_{i,j,k}}{a_k}$$

If a document community i shares 4 of its 10 edges with a tag/user pair (j, k) , and 6 edges with (j', k') , the correspondence with this approach will be $(0.4, 0.6)$ instead of $(0.0, 1.0)$ as with the previous approach.

4. EVALUATION

4.1 Experiments

An established metric for evaluating community detection algorithms is measuring the reconstruction quality on known, synthetic graphs. In the unipartite case, this is often done with so-called cave-man graphs [13]: Edges primarily exist between nodes from the same community (cave), with inter-cave edges intermixed according to a noise level $1 - p$.

The diagrams in Figure 1 not only serve as examples, but also describe three possible generalizations of the caveman scenario to 3,3-hypergraphs. Since, as we noted earlier, the notion of nodes being in the “same” community is obsolete, “legal” (non-noise) edges in these cases are defined by the connections shown in the diagrams.

We test the different modularity measures by applying a greedy, bottom-up optimization similar to the one described

in [2] (or using that algorithm directly in the case of the unipartite modularity). Reconstruction quality is expressed in terms of normalized mutual information (NMI) between the found and the correct community assignments[3]:

$$\operatorname{NMI}(A, B) = \frac{2\operatorname{MI}(A, B)}{\operatorname{H}(A) + \operatorname{H}(B)},$$

where MI is mutual information and H entropy.

Please note that we only show results on a single set of parameters, even though many parameters exist (number of nodes, edge density, number of communities in the SIMPLE case). Examples are chosen between “easy” and “hard” parameters such that differences between methods become salient – other parameters do however not qualitatively change the reported findings.

4.2 Results

Figure 2 shows the results of the experiments. Community detection based on “normal” modularity on a flattened version of the hypergraph performs well in the SIMPLE case. Since partition information is lost during the actual optimization, it has however no way to assign distinct community structures to elements from different partitions. It therefore correctly finds two communities for documents and users in the OVERLAPPING case, but forces tags to be (incorrectly) split into two communities as well – misassigning half of the tags and resulting in the average of maximally 0.6. In the CONTRADICTIVE example, the method lacks both partition information and triple structure, resulting in an NMI of almost 0.

Multi-Bipartite Modularity not only excels in the SIMPLE case, but also performs well in the OVERLAPPING case: The partition information and the support for distinct community structures allow for a perfect reconstruction in many graphs. However, the information it receives in the CONTRADICTIVE case is too limited to allow for any sensible reconstruction: The bipartite connections indicate connections exist between any two communities (e.g., users from A has tagged movies from X, as have users from B), and therefore it has no way to tell apart elements from different communities. The example was constructed to require the full triple information, and we see that it in fact does.

Accordingly, the two Multipartite Modularities, working

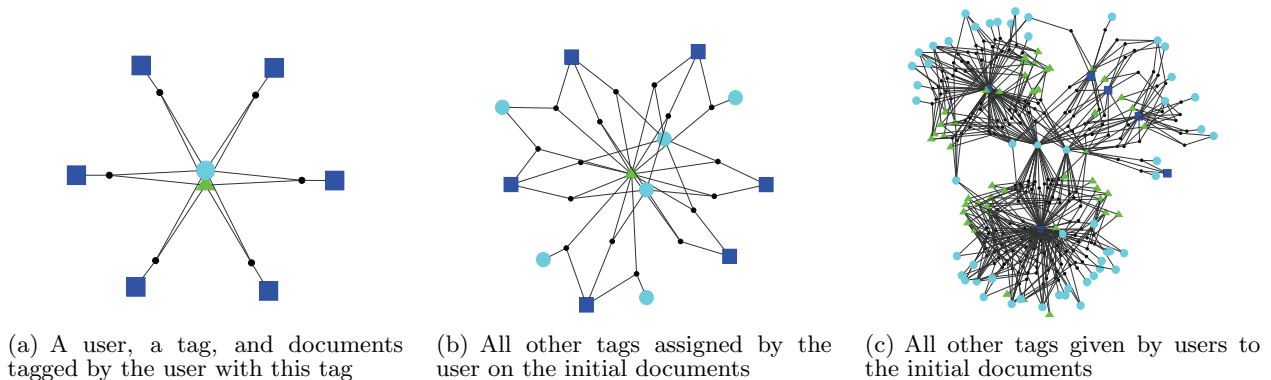


Figure 3: Real data: Expansion around a user/tag pair

on the native hypergraphical structure, are the only ones which can make any sense of the CONTRADICTIVE dataset. They perform similarly convincing on the OVERLAPPING dataset. We can see that the linear variant consequently outperforms the argmax-based variant. Interestingly, neither of the two methods performs particularly well on the SIMPLE dataset – the reasons for this intuitively surprising behaviour are outside the scope of this paper and subject of our current research.

To conclude, we could justify the challenges defined in the beginning. We have constructed a set of synthetic datasets which incrementally incorporate these challenges, and shown how methods not regarding these challenges fail on the corresponding datasets, while theoretically grounded extensions designed to deal with these challenges succeed.

5. APPLICATIONS

We will now shortly investigate possible applications of multipartite community detection. We demonstrate the tool *mpce* (Multipartite Community Exploration) which interactively visualizes results of community detection algorithms on (k, k) -hypergraphs.

mpce exploits the hierarchical clustering generated by bottom-up community detection algorithms: Starting with each node as its own community, communities are recursively joined until finally all nodes belong to a single community. Instead of just considering the optimal spot in-between as returned by community detection algorithms, *mpce* lets users undo this development in a step-wise fashion: Starting with all elements from the same partition in a single community, users can expand either community, gradually navigating to their area of interest. The interface can also be prompted to display the state of optimal modularity, or, as in the example, of minimum description length, which tends to favour clustering states with lower numbers of communities but cannot be further introduced here due to space constraints.

Figure 4 shows two clustering results in *mpce*. Gray circles indicate sets of edges existent between the incident communities. Sets of edges below a minimum size are discarded to avoid visual clutter. The size of the edge circles indicates the potential number of edges (the product of the involved communities’ sizes), whereas darkness indicates the density, i.e., the actual presence of edges.

For the example, we have used one month worth of De-

licious bookmarks taken from the dataset provided by [14], and extracted a subgraph consisting of all edges containing any of the documents tagged with “programming” by a particular user, as shown in the steps in Figure 3, but with another starting user, resulting in around 17.000 edges. We believe this is a realistic use case: Users might want to browse all their documents related to a certain tag, and consider all tags given by other users to bring additional structure to their collection.

The two graphs in Figure 4 are the results of applying the unipartite and the multi-bipartite approach on our example. We can see that the two different clustering algorithms yield qualitatively different results: The unipartite approach identifies distinct triples of document, user, and tag communities. The multi-bipartite approach does identify such triples as well, but each document and tag community is also connected to the central user community. In this specific case, we find that the central community of users seems to have tagged documents on all thematic clusters related to programming, whereas smaller user communities exist that are related only to, e.g., frontend programming technologies like JavaScript. While this results is anecdotal, it highlights the additional types of insights that can be gained by considering tripartite relationships.

What is the explanation for these differences? The unipartite approach is forced to group documents, users, and tags into shared communities, not being able to tell the difference between them. Opposed to that, the multi-bipartite approach possesses the flexibility to model edges beyond closely linked community triples. Like this, different pairs of document/tag communities can all be strongly connected to the central user cluster and yet remain unmerged.

6. CONCLUSIONS & OUTLOOK

We have motivated community detection on social bookmarking data and described the challenges related to community detection on the (k, k) -hypergraphs involved. Based on these challenges, we have introduced Multipartite Modularity and empirically validated its performance on synthetic datasets. We have then introduced an interactive exploration tool to highlight the practical effects of the choice of modularity, and, more generally, demonstrate possible applications built on information extracted from these datasets.

$(3,3)$ -hypergraphs have been mostly discussed in relation

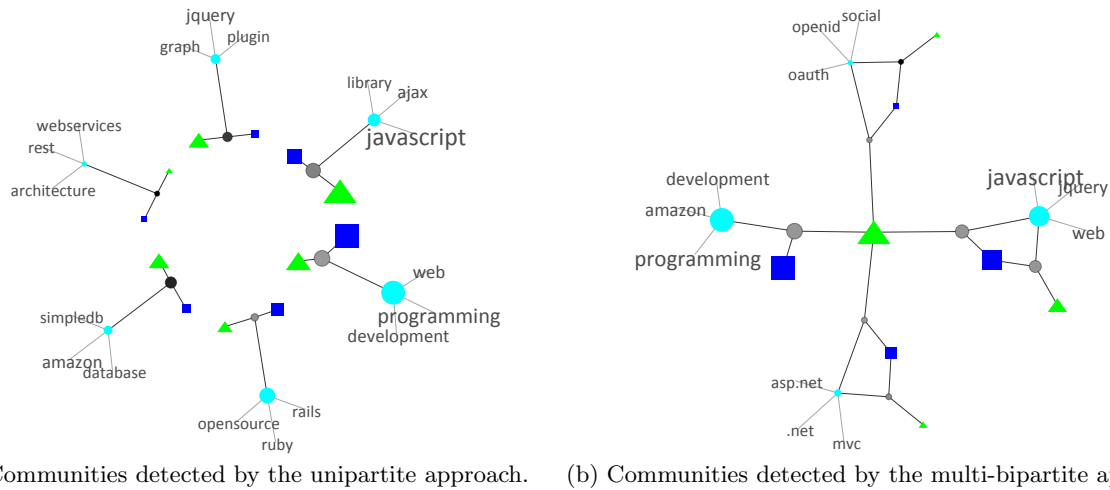


Figure 4: Visualizing two possible community detection results on the same dataset

to social bookmarking data, but in fact they are significant to a much wider range of datasets: Wherever terms are connected to entities not in a static, objective way as represented by the classic term/document matrices, but associated by users in noisy, subjective ways, the information about who associated the term becomes crucial. Therefore, the three-dimensional tensors gained by extending the term/document matrix by a user dimension, and the hypergraphs created by these tensors, appear as the basic model of social media: comments made by a particular user to a blog post/product/etc, tweets by a user involving a URL, changes made by a user to a wikipedia entry – all these cases involve tripartite structures, and we hope to have provided motivation for a dedicated treatment of these peculiar structures.

We could envision further research into at least two directions: First, it appears the last word appears has not been spoken about multipartite modularity, while many other community detection approaches are suitable for generalization to these structures as well. In [7], e.g., methods based on tensor factorization are discussed – one of the next steps should be laying a foundation for solid comparisons between different algorithms in this field. Second, with greater knowledge about the structure underlying social bookmarking and similar datasets, new applications may become possible that truly tap into the latent knowledge contained.

7. REFERENCES

- [1] C. Cattuto, C. Schmitz, A. Baldassarri, V. D. P. Servedio, V. Loreto, A. Hotho, M. Grahl, and G. Stumme. Network properties of folksonomies. *AI Communications Journal, Special Issue on "Network Analysis in Natural Sciences and Engineering"*, 20(4):245–262, 2007.
- [2] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70(6):066111, Dec 2004.
- [3] L. Danon, A. D. Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(9):P09008–09008, September 2005.
- [4] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.
- [5] G. Ghoshal, V. Zlatić, G. Caldarelli, and M. E. J. Newman. Random hypergraphs and their applications. *Phys. Rev. E*, 79:066118, 2009.
- [6] S. Golder and B. A. Huberman. Usage patterns of collaborative tagging systems. *Journal of Information Science*, 32(2):198–208, April 2006.
- [7] Y.-R. Lin, J. Sun, P. Castro, R. Konuru, H. Sundaram, and A. Kelliher. Metafac: community discovery via relational hypergraph factorization. In *KDD '09*, pages 527–536, New York, NY, USA, 2009. ACM.
- [8] T. Murata. Modularities for bipartite networks. In C. Cattuto, G. Ruffo, and F. Menczer, editors, *HT '09*, pages 245–250, New York, NY, USA, 2009. ACM.
- [9] T. Murata. Detecting communities from tripartite networks. In *WWW '10*, pages 1159–1160, New York, NY, USA, 2010. ACM.
- [10] N. Neubauer and K. Obermayer. Hyperincident connected components of tagging networks. In C. Cattuto, G. Ruffo, and F. Menczer, editors, *HT '09*, pages 229–238, New York, NY, USA, 2009. ACM.
- [11] N. Neubauer and K. Obermayer. Towards community detection in k-partite k-uniform hypergraphs. In *Workshop on Analyzing Networks and Learning with Graphs at NIPS 2009*, 2009.
- [12] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74(3):036104, Sep 2006.
- [13] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.
- [14] R. Wetzker, C. Zimmermann, and C. Bauckhage. Analyzing social bookmarking systems: A delicio.us cookbook. In *Mining Social Data (MSoDa) Workshop Proceedings, ECAI 2008*, pages 26–30, 2008.