Community detection in networks: Structural communities versus ground truth

Darko Hric, Richard K. Darst, and Santo Fortunato
Department of Biomedical Engineering and Computational Science, Aalto University School of Science, P.O. Box 12200, FI-00076, Finland
(Received 2 June 2014; revised manuscript received 5 September 2014; published 9 December 2014)

Algorithms to find communities in networks rely just on structural information and search for cohesive subsets of nodes. On the other hand, most scholars implicitly or explicitly assume that structural communities represent groups of nodes with similar (nontopological) properties or functions. This hypothesis could not be verified, so far, because of the lack of network datasets with information on the classification of the nodes. We show that traditional community detection methods fail to find the metadata groups in many large networks. Our results show that there is a marked separation between structural communities and metadata groups, in line with recent findings. That means that either our current modeling of community structure has to be substantially modified, or that metadata groups may not be recoverable from topology alone.

DOI: 10.1103/PhysRevE.90.062805 PACS number(s): 89.75.Hc

I. INTRODUCTION

Detecting communities in networks is one of the most popular topics of network science [1]. Communities are usually conceived as subgraphs of a network, with a high density of links within the subgraphs and a comparatively lower density between them. The existence of community structure indicates that the nodes of the network are not homogeneous but divided into classes, with a higher probability of connections between nodes of the same class than between nodes of different classes. This can have various reasons. In a social network, for instance, the communities could be groups of people with common interests, or acquaintanceships; in protein interaction networks, they might indicate functional modules, where proteins with the same function frequently interact in the cell, hence they share more links; in the web graph, they might be web pages dealing with similar topics, which therefore refer to each other.

One of the drivers of community detection is the possibility to identify node classes, and to infer their attributes, when they are not directly accessible via experiments or other channels. However, community detection algorithms are usually informed only by the network structure (in many cases this is all the information available). So, one postulates that structural communities coincide or are strongly correlated with the node classes, which correspond to their intrinsic features or functions. In a sense, the field has been silently assuming that structural communities *reveal* the nontopological classes. This is confirmed by the fact that community detection algorithms are typically tested on a (low) number of real networks where the classification of the nodes is available, such as, e.g., Zachary's karate club [2], Lusseau's dolphins' network [3], and the college football network [4]. This way, one implicitly tunes hypotheses and/or parameters such to get the best match between the communities detected by the method and the metadata groups of those systems.

Our goal is testing this basic hypothesis. This has finally become possible, due to the availability of several large datasets with information on the classification of the nodes (the node *metadata*). In recent work, Yang and Leskovec have studied the topological properties of metadata groups in social, information, and technological networks [5–7]. They found that they have peculiar properties, some of which are in contrast with the common picture of community structure. For instance, it seems that overlapping communities have a higher density of links in the overlapping than in the nonoverlapping parts [7], which is the opposite of what one usually thinks.

In this paper, we will compare the community structure detected by popular community detection algorithms on a collection of network datasets with the metadata node groups of the networks. Comparisons will be carried out both at the level of the whole partition, and at the level of the individual communities. We find that the match between topological and supposed "ground truth communities" (metadata groups) is not good, for all methods employed in the analysis. This questions the usefulness of (purely topological) community detection algorithms to extrapolate the hidden (nontopological) features of the nodes.

Before we proceed, it is worthwhile to clarify some nomenclature. We take *community* to represent a connected subgraph with a density of internal links which is appreciably higher than the density of external links. The term *cluster* is often used interchangeably with "community" within the physics literature, but has a more general meaning within computer science. For clarity, the sets of nodes derived from the network metadata (which are hopefully detected by methods) are known as *metadata groups*. These are not assumed to represent structural communities until proven. The term *ground truth* is used in other literature to refer to these metadata groups in order to invoke the concept of a true result to which we will attempt to match. We avoid the term here because of the reason above. The term *partition* formally refers to a complete, nonoverlapping set of communities, but in this work we loosen the definition to any set of communities. While some datasets do have strict partitions, others can have overlapping nodes (nodes in multiple groups) and nodes in no groups.

In Sec. II, we will introduce our collection of datasets and the community detection methods used in the study. Section III reports some basic structural properties of the metadata groups. Sections IV and V expose the results of the comparison between detected communities and metadata groups, both at the level of the partition as a whole (Sec. IV) and at the level of the individual groups (Sec. V). In Sec. VI, we will discuss the implications of the results.

TABLE I. Basic properties of all datasets used in this analysis. `fb100` consists of 100 unique networks of universities, so we show the ranges of the number of nodes and edges of the networks, as well as of the metadata groups of the various partitions. `amazon` consists of a hierarchical set of 11 group levels, we report the range of the number of groups. The number of groups is calculated after our indicated preprocessing (see text).

Name	No. Nodes	No. Edges	No. Groups	Description of group nature
`lfr`	1000	9839	40	Artificial network (`lfr`, 1000S, $\mu = 0.5$)
`karate`	34	78	2	Membership after the split
`football`	115	615	12	Team scheduling groups
`polbooks`	105	441	2	Political alignment
`polblogs`	1222	16782	3	Political alignment
`dpd`	35029	161313	580	Software package categories
`as-caida`	46676	262953	225	Countries
`fb100`	762–41536	16651–1465654	2–2597	Common students' traits
`pgp`	81036	190143	17824	email domains
`anobii`	136547	892377	25992	Declared group membership
`dblp`	317080	1049866	13472	Publication venues
`amazon`	366997	1231439	14–29432	Product categories
`flickr`	1715255	22613981	101192	Declared group membership
`orkut`	3072441	117185083	8730807	Declared group membership
`lj-backstrom`	4843953	43362750	292222	Declared group membership
`lj-mislove`	5189809	49151786	2183754	Declared group membership

II. DATA AND COMMUNITY DETECTION METHODS

A. Network datasets

We collected many networks with node metadata that can be used for creating different node classes to approximate communities, which we refer to as *metadata groups*. These datasets can roughly be classified in two groups: classical and big datasets. Full details on all datasets can be found in Appendix A.

The first group contains real and synthetic networks that have regularly been used for testing community detection algorithms. Zachary's karate club network (`karate`) is a classic testbed for community detection algorithms [2]: it has two natural communities, corresponding to the split of the club in two factions. So is `football`, which represents matches played between U. S. college football teams in year 2000 [4]; the metadata groups are team conferences. `polblogs` is the network of political blogs after the 2004 elections in the U. S. [8], grouped by political alignment. `polbooks` represent copurchased books on politics on Amazon bookstore around the time of 2004 elections, and grouped by political alignment [9]. We also used a state-of-the-art artificial network with built-in (topological) communities, the LFR benchmark [10], with 1000 vertices, small communities, and mixing parameter of $\mu = 0.5$ (`lfr`).

The second group contains more recent and challenging networks. The Debian package dependencies (`dpd`) are dependencies of software packages in Debian Linux distribution, grouped by crowd-sourced tag assignment. The Pretty Good Privacy network (`pgp`) contains email addresses with signatures between them, with groups represented by the email domains [11]. The Internet topology at the level of autonomous systems (`as-caida`) is collected by the CAIDA project, and is grouped by countries [12,13]. The Amazon product copurchasing network (`amazon`) has groups of product categories [14]. `anobii` is a book recommendation social network popular in Italy, where users can join groups [15,16]. The `dblp` network of coauthorships in computer science literature has publication venues as groups [17]. The Facebook university networks (`fb100`) consist of 100 separate networks of Facebook users at U. S. universities from 2005 [18]. The multiplicity was used to provide statistics. The groups are freely entered by users and are formed with different criteria (such as field of study or graduation year) provided in the node metadata. The network of Flickr users (`flickr`) consists of photo-sharing users who join user groups to share content [19]. The LiveJournal network consists of users friendships and explicit group memberships. We have two independent sources for this network, `jl-backstrom` [17] and `lj-mislove` [19], which are analyzed separately. The Orkut social network (`orkut`) consists of users and groups they join.

We present the list of datasets in Table I. We converted all networks to undirected, unweighted networks, and take their largest connected component. This is the largest weakly connected component (LWCC) of the directed graphs. Any graph members outside of this LWCC are dropped. The numbers of Table I refer to the LWCC of each network.

In general, the metadata groups in the data can be disconnected within the graph. We applied the following preprocessing steps. Each group's connected components over the network were taken as separate groups for the analysis. That means that several distinct groups may end up having the same node membership. On the other hand, community detection methods would not be able to associate disconnected groups, so it is necessary to proceed like this. Any group with less than three members is dropped, from both the metadata partition and the detected partition. The comparison is limited to the set of nodes belonging to both the metadata and the detected partition after the above preprocessing steps. Since in some cases the fraction of nodes of the system belonging to such intersection can be quite low, we report results only when it exceeds 10%.

B. Community detection methods

We have a collection of community detection methods with available codes. These methods come from a variety of different theoretical frameworks. Some of them are designed to detect overlapping communities, others can only deliver disjoint communities. Not all methods run to completion on the largest datasets in a reasonable time; such dataset and method combinations are excluded from the analysis.

`Louvain` is a greedy agglomerative method based on modularity [20]. `Infomap` [21] is based on information compression of random walks. We also used a variant `InfomapSingle` [22], which returns a single partition instead of a hierarchy. `LinkCommunities` [23] is a method that clusters edges instead of nodes. `CliquePerc` [24,25] scans for the regions spanned by a rolling clique of certain size. `Conclude` [26] uses edge centrality distances to grow communities. `COPRA` [27] uses propagation of information to classify communities (label propagation). `Demon` [28] exploits node-local neighborhoods. `Ganxis` [29] (formerly SLPA) is based on label propagation. `GreedyCliqueExp` [30] begins with small cliques as seeds and expands them optimizing a local fitness function.

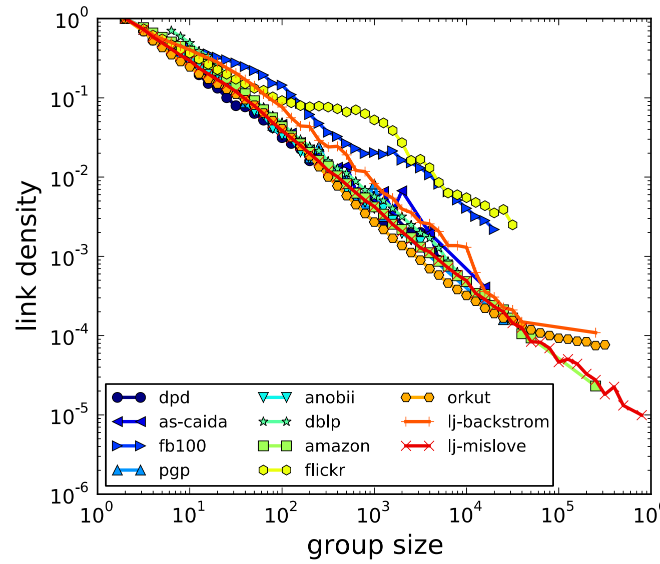

FIG. 2. (Color online) Link density versus size of the metadata groups. Each curve corresponds to a specific dataset of our collection.

III. STRUCTURAL PROPERTIES OF NODE GROUPS FROM METADATA

Here, we show some basic topological features of the metadata groups of our datasets. Figure 1 reports the distribution of the group sizes, which is skewed for all datasets. Power law fits of the tails deliver exponents around -2. This is in agreement with the behavior of the size distributions for the communities found by community detection algorithms on real networks [1].

The link density of a subgraph $\mathcal{S}$ is the ratio between the number of links joining pairs of nodes of $\mathcal{S}$ and the total maximum number of links that could be there, which is given by $n_{\mathcal{S}}(n_{\mathcal{S}}-1)/2$, $n_{\mathcal{S}}$ being the number of nodes of $\mathcal{S}$. In Fig. 2, we see the link density of the metadata groups versus their sizes. Clearly, the larger the size of the group, the lower the link density. This is because real graphs are typically sparse, so the total number of links scales linearly with the number of nodes. This holds for parts of the network too, modulo small variations, so the link density decreases approximately as a power of the number of links of the group (with exponent close to -1). Since the latter is proportional to the group size, we obtain that the link density decreases as the inverse of the group size, as we see in Fig. 2.

Finally, in Fig. 3 we report the relation between the group embeddedness and its size. The embeddedness of a group is the ratio between the internal degree of the group and the total degree. The internal degree of a group is given by the sum of the internal degrees of the group's nodes, i.e., twice

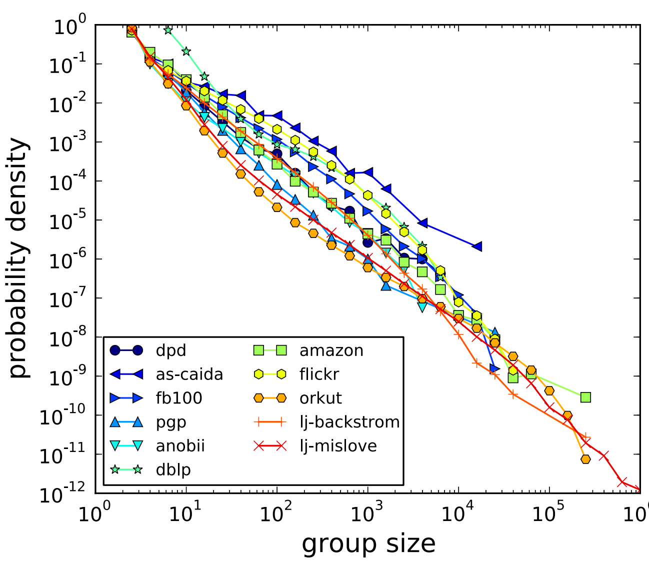

FIG. 1. (Color online) Distribution of sizes of metadata groups. Each curve corresponds to a specific dataset of our collection.

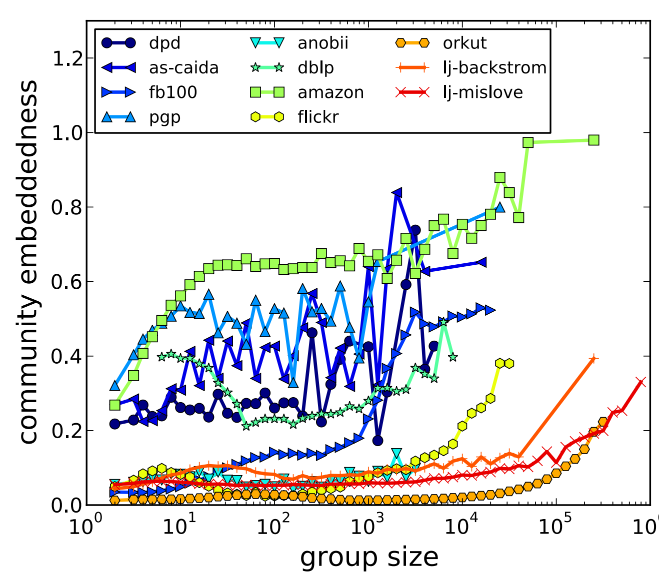

FIG. 3. (Color online) Embeddedness versus size of the metadata groups. Each curve corresponds to a specific dataset of our collection.

the number of links inside the group. The total degree of the group is the sum of the degrees of its nodes. A group is "good" if it has high embeddedness, i.e., if it is well separated from (loosely connected to) the rest of the graph. We notice that some of the datasets of our collection have groups with fairly large values of the embeddedness (e.g., Amazon), so they are fairly well separated from the other groups. For the largest datasets we have, the online social networks, embeddedness is very low (and fairly independent of group size). In this case, their detection by means of community detection algorithms is more difficult.

IV. PARTITION LEVEL ANALYSIS

The similarity of partitions can be computed in various ways (see Ref. [1]). Here, we stick to the normalized mutual information (NMI), a measure taken from information theory [31]. Since the nontopological group structure of several datasets is made of overlapping groups, we use the generalization of the NMI proposed by Lancichinetti *et al.*, that allows for the comparison of covers (i.e., of partitions into overlapping groups) [32].

Many metadata as well as detected partitions do not cover all nodes present in the network. Often, these coverages mismatch, leaving many nodes present only in one of the compared partitions. In order to circumvent this problem we decided to follow the best possible scenario (which generally increases the score), by using only the nodes present in both partitions. In some cases, the fraction of overlapping nodes was very small, so we did not calculate NMI scores if the coverage was less than 10%. This only applies to comparisons between metadata and detected partitions, for comparisons between partitions detected with different methods we used the full sets returned by the algorithms.

The overview of all the NMI scores is conveniently presented in what we call "NMI grids," like the one in Fig. 4. Each grid refers to a specific network. In addition to the NMI scores between the metadata groups structure and the one detected by each algorithm, we also show the similarity between structural partitions detected by different methods. Since some methods may deliver different hierarchical partitions, the tiles involving those methods are further subdivided.

A. pgp NMI grid analysis

As an example, we provide a detailed discussion of the pgp NMI grid of Fig. 4 (the others are shown in the Appendix). The main conclusions are consistent across all datasets, though. Hierarchical layers were ordered by their granularity, 0 being the lowest, most granular one. For some algorithms layers are partitions obtained using different parameter values (see Appendix B).

First, we compare partitions returned by different algorithms, including all returned layers (all tiles except bottom row). On the diagonal we have the mutual comparison of different layers delivered by the same algorithm. The diagonal of each tile is, of course, black, as one is comparing each layer with itself, which yields an NMI score of 1. Off-diagonal elements show similarity between different layers. Most algorithms return a group of layers which are quite similar

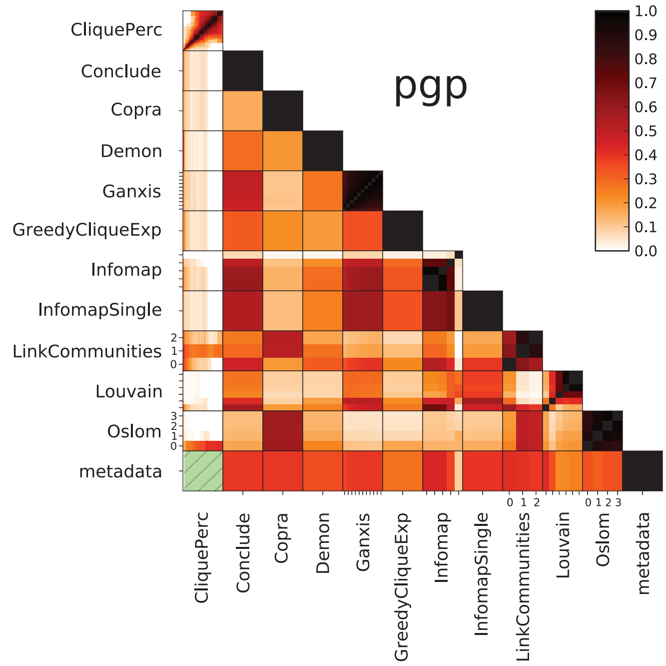

FIG. 4. (Color online) NMI grid of the pgp dataset. Each tile represents the NMI scores of the comparison of the structural partitions obtained from different algorithms and the metadata partition(s) (bottom stripe), and of the comparisons between partitions obtained by different algorithms. Each tile contains a grid within, corresponding to different partitions delivered by the algorithm (hierarchical levels or partitions obtained for given parameter choices). The color of each element of a tile indicates the NMI score, with values discarded due to low coverage marked with hatched green.

to each other (Infomap, Louvain, Oslom). Comparing the results of one algorithm versus those of other algorithms, we can see, for instance, that the highest layer of Infomap is similar to some extent only to middle layers of Louvain. The lowest layer of CliquePerc is much more similar to layers found by other algorithms than to higher layers of CliquePerc. Layers of LinkCommunities (threshold values 0.25, 0.5, 0.75) show varying behavior: the threshold value of 0.25 yields the most similar partitions to the ones obtained by the other algorithms, except for Copra and Oslom, and to some extent CliquePerc. Lower levels of Infomap, Louvain, and Oslom tend to be more similar to the layers returned by other algorithms. We can also draw conclusions about the general behavior of algorithms. For instance, Demon returns a partition that is not very similar to partitions returned by other algorithms (this is more pronounced in other datasets).

The bottom row is metadata versus detected partitions, where we can see how similar is the metadata partition to the detected ones, which is the focus of our work. The intersections of metadata partitions and higher order CliquePerc layers cover less than 10% of total nodes, so we discarded these results, indicated with hatched green in the figure. Most of the algorithms return scores that are similar, around 0.3. Ganxis layers have almost the same scores (they are also very similar among themselves) whereas Infomap and Louvain layers are very different, the lower ones scoring better.

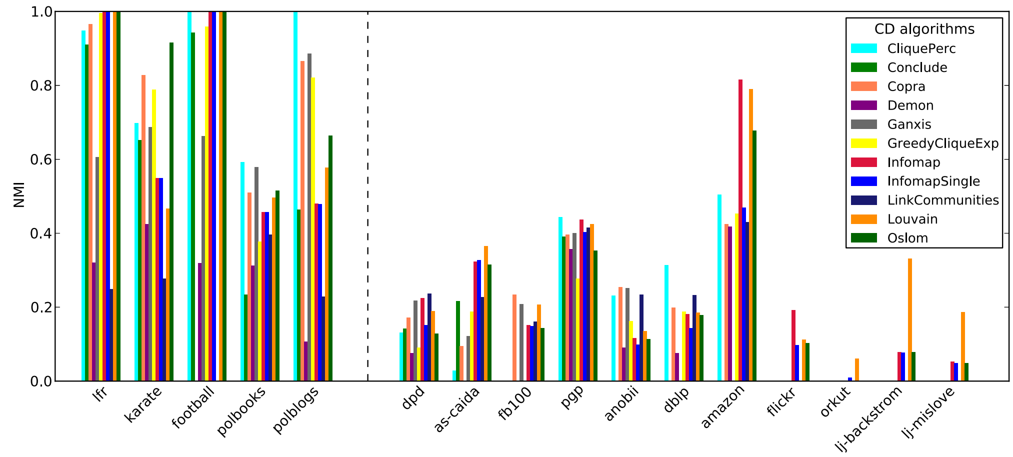

FIG. 5. (Color online) NMI scores between structural communities and metadata groups for different networks. Scores are grouped by datasets on the x axis. The height of each column is the maximal NMI score between any partition layer of the metadata partitions and any layer returned by the community detection method, considering only those comparisons where the overlap of the partitions is larger than 10% of total number of nodes.

B. Overall NMI scores

In order to compare how well different algorithms detect metadata groups, we took the best scores of each dataset-algorithm pair and present them on Fig. 5. In real-world applications one would not know what the returned layers represent and, consequently, which one of them corresponds more truthfully to the partition one would like to detect. So, the NMI scores we derive are in general higher than the ones obtained by comparing individual levels with each other.

The results can be separated into three groups. The highest recall of metadata groups is in the case of the artificial dataset `lfr`, as it is expected, since many community detection algorithms are tested on the LFR benchmark. The second group consists of small, classical datasets (`karate`, `football`, `polblogs`, `polbooks`) that are often used for testing community detection methods. These NMI scores are fairly high, but not as high as for `lfr`. The third group includes the big datasets of our collection. Here, algorithms were not very successful in finding the metadata groups. The only exception is `amazon`, for which we find a much higher score than for the others because it has many levels for the metadata groups, some of which turn out to be partially recoverable. Scores for the other networks rarely go above 0.3; for some datasets they lie even below 0.1.

A possible explanation of the result could be that the optimization process at the basis of several techniques is not successful, and that the partition delivered by those methods corresponds to a value of the measure far from the sought extreme. We reject this hypothesis though. For one thing, some of the community detection techniques we adopted are not based on optimization procedures (e.g., `CliquePerc`), still they do not seem to lead to better results. Furthermore, for `as-caida` we have computed the value of Newman-Girvan modularity Q for the metadata partition, and the ones obtained through the `Louvain` method, corresponding to the hierarchical level most similar to the metadata partition and to the level yielding the best approximation to the modularity maximum. They are 0.3839, 0.5064, and 0.5176, respectively. So, the values of Q of `Louvain`'s partitions are far higher than the one corresponding to the metadata partition. We could not repeat this test for the other datasets because the metadata partitions are overlapping (they are nonoverlapping only in the case of `as-caida`), while `Louvain` computes nonoverlapping partitions. Since there is no straightforward extension of modularity to the overlapping case, it is not possible to make meaningful comparisons of the values.

V. COMMUNITY LEVEL ANALYSIS

The previous section shows that global measures indicate that partitions returned by community detection methods do not align with partitions built from metadata, but what about specific groups? Can we detect *any* of the groups well? Are some groups reflected in the graph structure and detectable, but lost in the bulk noise of the graph? This is what we wish to investigate here.

The basis of our analysis is the Jaccard score between two groups. Let C_i represent (the set of nodes of) the known group i, and D_j represent (the set of nodes of) the detected community j. The Jaccard score between these two sets is defined as

$$J(C_i,D_j) = \frac{|C_i \cap D_j|}{|C_i \cup D_j|}, \tag{1}$$

with $|\dots|$ set cardinality, $\cap$ set intersection, and $\cup$ set union. The Jaccard score ranges from one (perfect match) to zero and roughly indicates the fraction of nodes shared between the two sets: the match quality.

The *recall score* measures how well one known group is detected. The recall score of one known group C_i is defined as the maximal Jaccard score between it and every detected community D_j:

$$R(C_i) = \max_{D_j \in \{D\}} J(C_i,D_j). \tag{2}$$

It is near one if the group is well detected and low otherwise. We can study the distribution of these scores to see how many groups can be detected at any given quality level. Recall

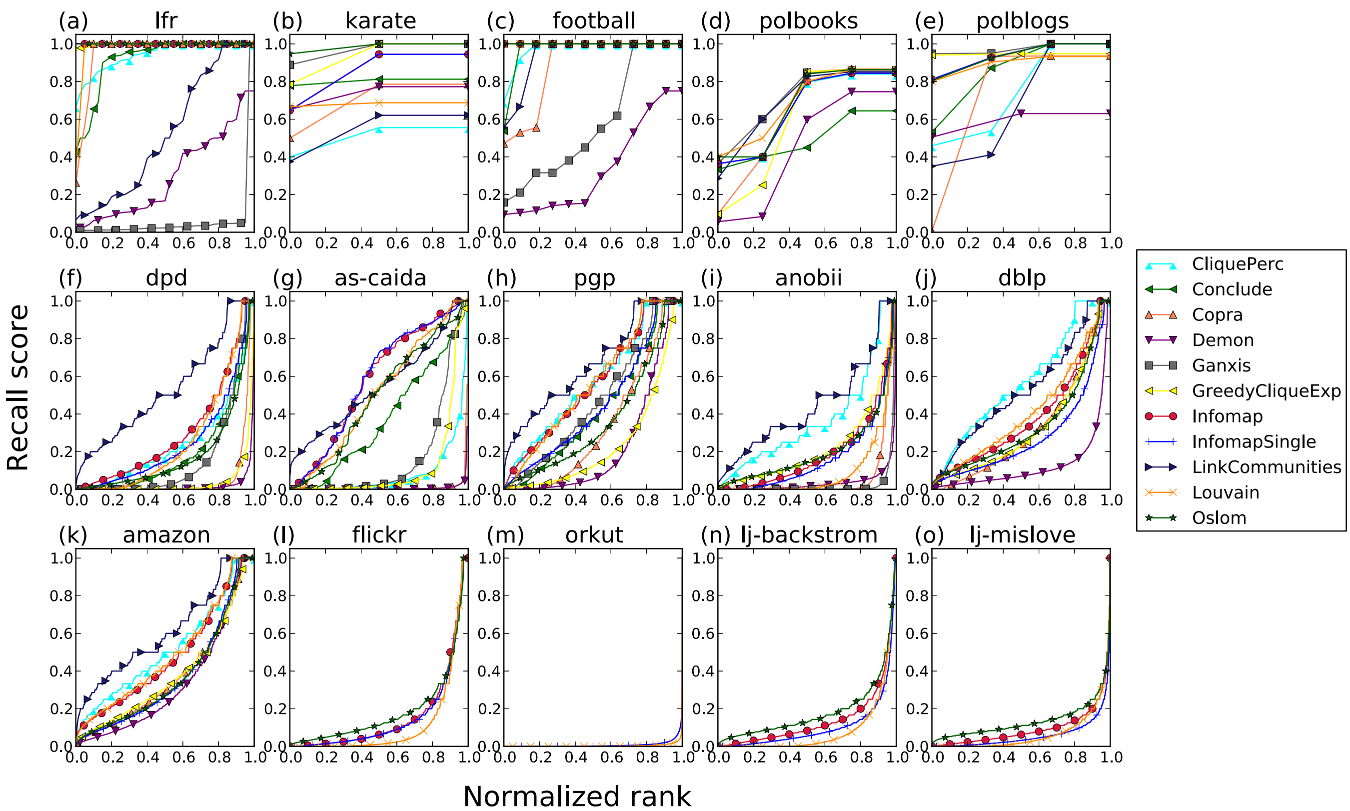

FIG. 6. (Color online) Recall of known groups plotted versus group rank (sorted by recall) for various datasets and methods. Every known group is compared with every detected community in any layer. We see that performance is usually poor (close to zero) for all networks except for some of the classic benchmarks (uppermost row of diagrams), which are typically used to test algorithms. The metadata groups of some graphs, such as `livejournal`, `orkut`, and `flicker` have but a little overlap with the detected communities.

measures the detection of known groups, and to measure the significance of detected communities, we can reverse the measure to calculate a *precision score*

$$P(D_j) = \max_{C_i \in \{C\}} J(D_j, C_i). \tag{3}$$

The precision score tells us how well one detected community corresponds to any known group.

We can now directly quantify the two conditions for good community detection: every known group must correspond to some detected community, and every detected community must represent some known group. Both of these measures are still interesting independently: a high recall but low precision indicates that the known groups are reflected in the network structurally, but there are many structural communities that are not known. We visualize the scores by means of *rank-Jaccard plots* which give an overview of the network's detection quality. We compute the recall (precision) for every known (detected) group and sort the groups in order of ascending Jaccard score. We plot recall (precision) versus the group rank, sorted by recall (precision) score so that the horizontal scale is the relative group rank, i.e., the ratio between the rank of the group and the number of groups (yielding a value between 0 and 1). Similar to our treatment of the partition-level analysis, we only plot matchings whose intersection covers more than 10% of total nodes in the graph. In our final plots, the average value of the curve (proportional to the area under it) is the average recall or precision score over all groups. The shape of the curve can tell us if all groups are detected equally well (yielding a high plateau) or if there is a large inequality in detection (a high slope). Furthermore, this allows us to compactly represent multiple layers. Each independent layer of known (detected) groups can be plotted in the same figure. We would generally look for the highest curve to know if any layer has a high recall (precision). When computing recall (precision), unless otherwise specified, as detected communities we consider the communities of all partitions delivered by a method, whereas the metadata groups are those present in all metadata partitions (if more than one partition is available in either case). This will give us the maximum possible recall (precision), which might be far higher than values coming from real applications, where one typically compares groups of the same partition (level).

In Figs. 6 and 7, we show the group recall and precision for every dataset and every community detection method. Similar to the situation with NMI, with the benchmark graph `lfr` most methods are able to recover the true communities. The other small graphs (b)–(e) also have most of the structure recoverable by most methods, as they are also used as benchmarks. However,once we get to large data, (f)–(o), we see a very different story. The vast majority of these networks have only a small number of groups detected fairly well and not many detected communities resemble any of the metadata groups. Many networks, e.g., the online social networks, have almost no metadata groups reflected in the detected communities,

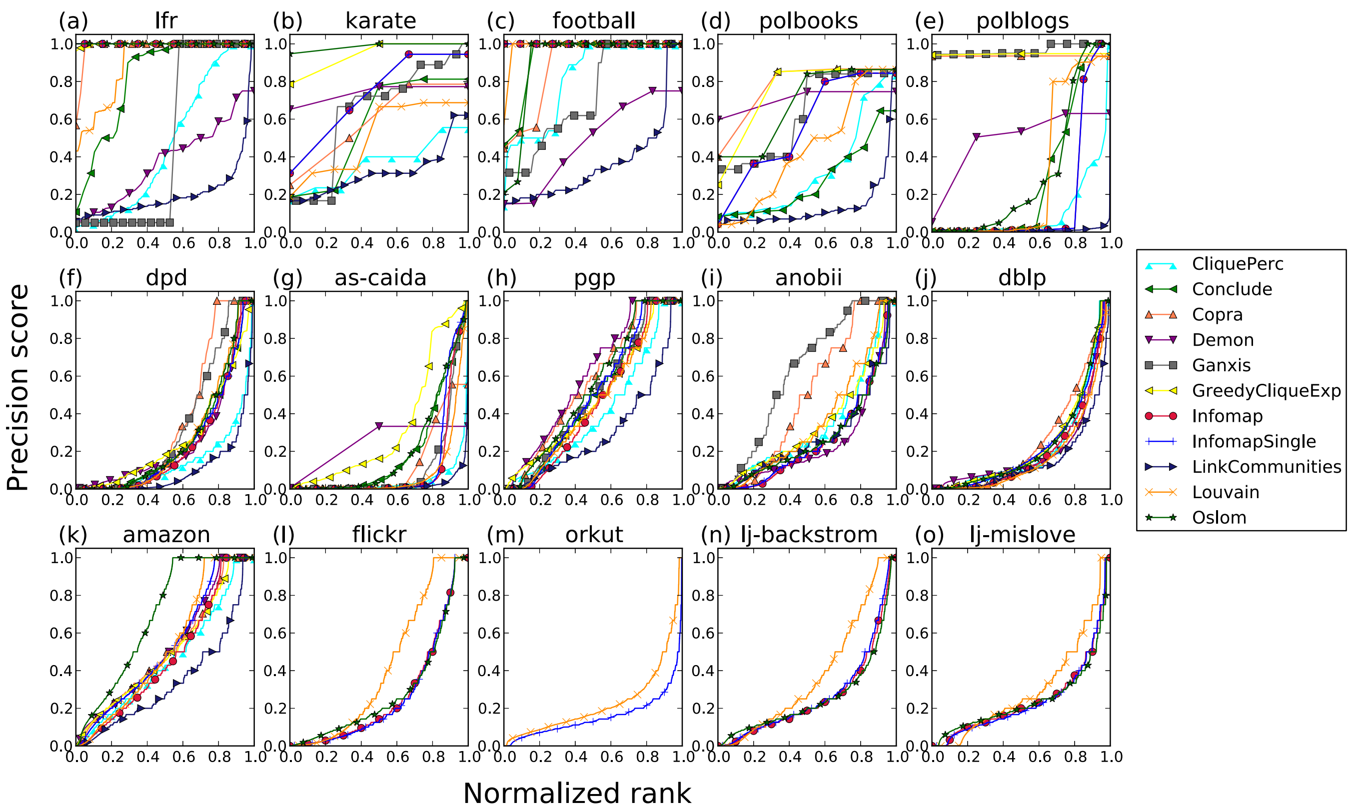

FIG. 7. (Color online) Precision of detected communities for various datasets and methods. All detected communities (in any layer) are compared to all known groups. Results are similar as for recall (Fig. 6).

and vice versa, by any method. In some networks, such as pgp and amazon, a fraction of groups are well detected. For example, amazon has 20% of groups with a maximal recall Jaccard score greater than 0.6, for any method, and is the network with best detected communities. The performance of the methods is comparable in most cases. LinkCommunities appears to give higher recall than all other methods in most instances. However, this is due to the fact that it usually detects many more communities than the other methods, so there is a higher chance to find a community that gives high overlap with the metadata groups. However, the precision of LinkCommunities is very low. On the largest graphs, Louvain and InfomapSingle have consistently worse recall than Oslom, but the latter has lower precision. In Table II,

TABLE II. Average Jaccard recall (R) and precision (P) scores for all datasets. The scores are simple averages over all groups. Horizontal lines separate the classic benchmarks from the large datasets.

	Clique P		Conclude		Copra		Demon		Ganxis		Grd CE		Info		InfoS		LinkC		Louvain		Oslom	
	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P
lfr	0.92	0.47	0.91	0.82	0.95	0.98	0.28	0.36	0.05	0.45	1.00	1.00	1.00	1.00	1.00	1.00	0.52	0.20	0.98	0.90	1.00	1.00
karate	0.48	0.34	0.80	0.50	0.64	0.51	0.71	0.71	0.94	0.63	0.89	0.89	0.80	0.63	0.80	0.63	0.50	0.30	0.68	0.48	0.97	0.97
football	0.91	0.76	0.86	0.77	0.87	0.94	0.30	0.40	0.81	0.87	0.95	0.90	0.95	0.90	0.95	0.90	0.89	0.35	0.95	0.92	0.97	0.86
polbooks	0.60	0.31	0.46	0.25	0.55	0.71	0.37	0.67	0.67	0.62	0.52	0.66	0.60	0.49	0.60	0.49	0.64	0.13	0.64	0.41	0.63	0.63
polblogs	0.67	0.10	0.80	0.24	0.63	0.94	0.57	0.43	0.97	0.96	0.94	0.94	0.91	0.15	0.91	0.15	0.59	0.01	0.87	0.29	0.91	0.29
dpd	0.24	0.15	0.19	0.26	0.05	0.35	0.02	0.24	0.16	0.33	0.04	0.27	0.30	0.24	0.24	0.24	0.53	0.10	0.26	0.27	0.18	0.27
as-caida	0.08	0.04	0.37	0.19	0.01	0.10	0.01	0.17	0.17	0.12	0.11	0.32	0.58	0.11	0.58	0.12	0.55	0.02	0.50	0.09	0.49	0.20
pgp	0.58	0.41	0.43	0.50	0.34	0.57	0.23	0.61	0.47	0.51	0.24	0.49	0.57	0.46	0.45	0.51	0.67	0.31	0.57	0.47	0.37	0.55
anobii	0.37	0.35			0.07	0.50	0.04	0.26	0.02	0.62	0.23	0.35	0.19	0.26	0.17	0.27	0.44	0.28	0.09	0.36	0.20	0.28
dblp	0.57	0.23			0.32	0.26	0.13	0.21			0.34	0.25	0.38	0.20	0.28	0.23	0.52	0.15	0.41	0.18	0.34	0.24
amazon	0.52	0.46			0.38	0.51	0.33	0.52			0.38	0.50	0.48	0.47	0.37	0.53	0.61	0.34	0.49	0.53	0.37	0.69
flickr													0.16	0.28	0.15	0.27			0.12	0.42	0.19	0.29
orkut															0.01	0.16			0.00	0.23		
lj-backstrom													0.14	0.26	0.10	0.27			0.09	0.38	0.18	0.26
lj-mislove													0.10	0.25	0.06	0.24			0.07	0.30	0.14	0.25

we report the average recall and precision for all datasets and algorithms.

In Appendix D, we further analyze recall and precision by narrowing the problem to group classes selected based on size, density or cohesiveness, or attribute types. This includes a full analysis of the fb100 dataset with its specific attributes such as student class year, field of study, or residence. We see that, in general, narrowing the focus to these specific classes of groups does not allow increased predictive power on most networks.

In this section, we have broken down the community detection problem into something more specific: instead of asking for all known groups to match all detected communities, we are asking if (a subset of) known groups are found by any detected communities, or if (a subset of) detected communities correspond to real known groups. Even if full community detection does not have high accuracy, a positive answer in either of these questions can produce a result of practical use. Instead, we see that recall and precision are highly network dependent, with most networks producing very low values for both. This even extends to social networks with user-defined social groups.

VI. CONCLUSIONS

Algorithms to find communities in networks are supposed to recover groups of nodes with the same or similar features or functions. Therefore, whenever a new algorithm is introduced, it is usually tested not only on artificial benchmark graphs, but also on real graphs with known node groups derived from some metadata. A good match between the detected partition and the attribute-based partition is considered evidence that the method is reliable. However, the correspondence between structural communities (the ones detected by an algorithm) and metadata groups (identified by the nodes' attributes) has been given for granted. In this work, we have made a systematic test of this hypothesis.

We have compared the partitions detected by several popular community detection algorithms with the partitions resulting from nontopological features of the nodes, on large real network datasets. We find that there is a substantial difference between structural communities and metadata groups. At the partition level, we find low similarity scores. Precision and recall diagrams show that detected communities have low overlap with the metadata groups, and vice versa. A more detailed analysis, in which one restricts the comparison to groups of comparable size, link density, or embeddedness, does not reveal major improvements. Overall, results depend more on the network than on the specific method adopted, none of which turns out to be particularly good on any (large) dataset. It is fair to remark that we have applied the community detection algorithms on the undirected and unweighted versions of the datasets. We have done so because few methods can handle link directions and weights, while we wanted to test a broad class of techniques. On the other hand, it is possible that by accounting for link directions and weights, the comparison between detected communities and metadata groups could improve.

Our results rely on the classification of the nodes, which may not always be reliable. However, our collection comprises a list of very diverse systems, and the message coming from all of them is the same. Clearly, we cannot exclude that there may be other datasets whose metadata groups match more closely the structural communities found by community detection algorithms. Still, even if there were such datasets, our point that metadata groups are not necessarily correlated with the communities found by standard methods, contrary to common belief, would hold.

We remark that low similarity scores between structural and metadata partitions were reported by Yang and Leskovec as well [6]. However, that was not the focus of the work, like in our case, and we have considered a larger set of methods and a broader spectrum of datasets.

What kind of implications does this finding have? We envision two possible scenarios. It may be that our conception of community structure, which is underlying the methods currently used, is not correct. Most algorithms usually focus on things like link densities within the communities, or between the communities (or both). It may be that metadata groups are not well represented by link density, for instance, or at least not by link density alone. Other features, like, e.g., degree correlations, density of loops (like, e.g., triangles), etc., might play a role. Indeed, Abrahao *et al.* have shown that structural properties of communities detected with several algorithms are in general different from those of metadata groups [33]. Therefore, our best bet would be carrying out a detailed investigation of the topological properties of the metadata groups, and trying to infer a general description from it, which could be used as starting point of the development of new algorithms. The recent discovery of dense overlaps between groups, for instance, might inform new techniques, the Affiliation Graph Model being one example of them [7].

The other possible interpretation is that metadata groups cannot be inferred from topology alone. There certainly is a correlation between structural and metadata groups, but it may be not very strong. Therefore, in order to detect metadata groups, nontopological inputs might be necessary. In the most recent literature on community detection, several such approaches have been proposed, mostly by computer scientists [34–49].

We stress, however, that structural communities are very important for the function of a network, as they can significantly affect the dynamics of processes taking place on the network, such as diffusion, synchronization, opinion formation, etc. So, detecting topological communities remains crucial. We are saying that one should not expect too much in terms of content, at least not from the algorithms currently in use. We hope that the scientific community of scholars working on community detection in networks will seriously reflect on the results of our analysis, in order to produce more reliable algorithms for applications.

ACKNOWLEDGMENTS

We thank T. Evans for providing an updated version of the college football data set. We acknowledge the computational resources provided by Aalto University Science-IT project. R.K.D. and S.F. gratefully acknowledge MULTIPLEX, Grant No. 317532 of the European Commission.

APPENDIX A: DATASET DESCRIPTIONS

Here, we will give more detailed descriptions of all datasets. A full description of each dataset can be found in the cited references. In some cases, the networks are created via complex processes in special environments, so the true meaning of links and groups may not have a simple interpretation. Nevertheless, the breadth of our data gives us a wide perspective on real-world, as opposed to artificial, networks.

`lfr` (Lancichinetti-Fortunato-Radicchi): Benchmark graph with 1000 vertices ($N = 1000$) and "small" communities (min size $= 10$, max size $= 50$), at mixing parameter $\mu = 0.5$ [10]. The other parameters (average degree 20, maximum degree 50, exponent of degree distribution -2, exponent of community size distribution -1) are standard. This graph has a clear community structure that is a standard used to optimize and test most current algorithms, and thus serves as a baseline reference for a network with known and detectable structure.

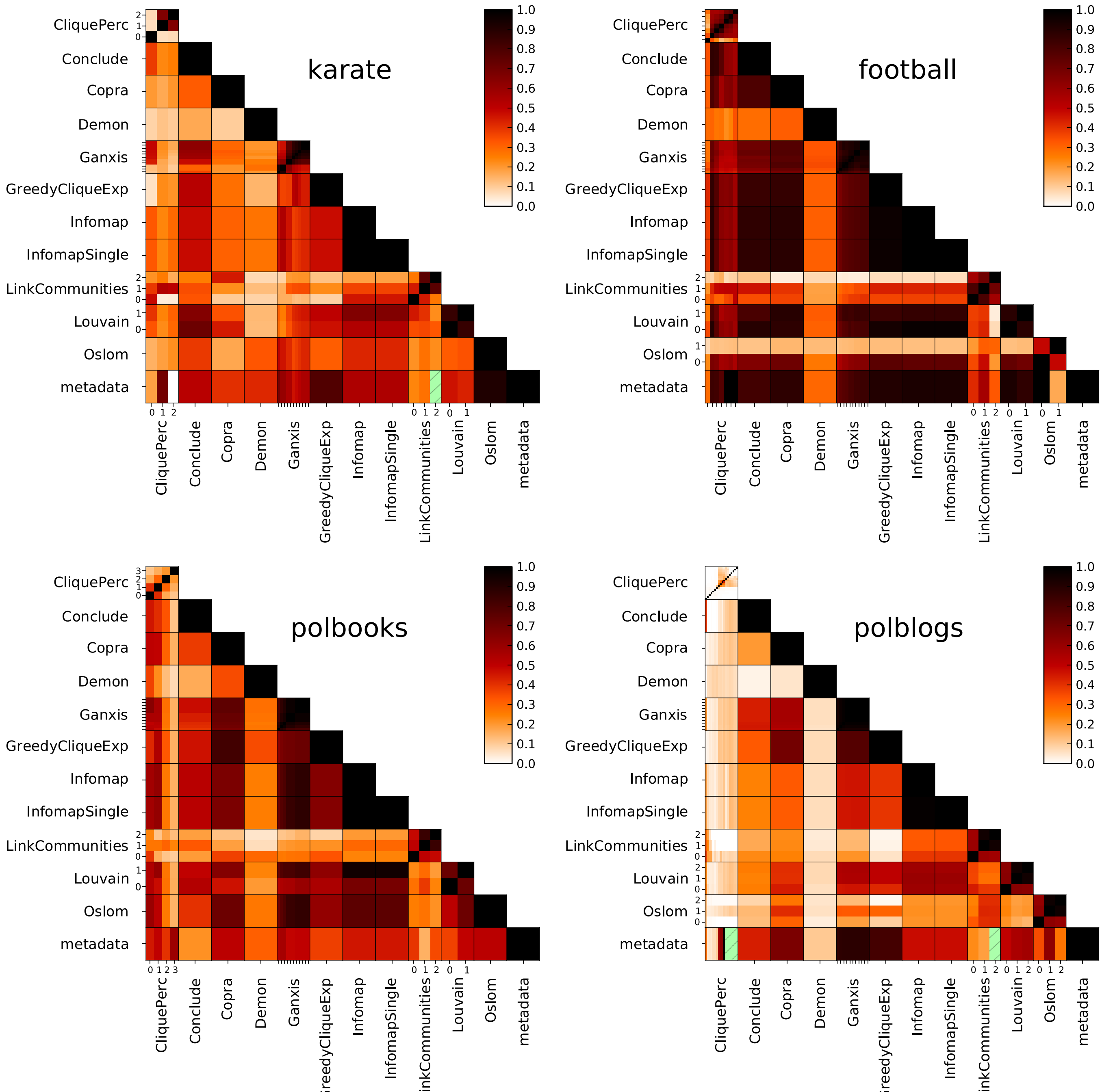

FIG. 8. (Color online) NMI grids of `karate`, `football`, `polbooks`, and `polblogs`. These datasets have a pronounced community structure, which is the reason why they are heavily used in the community detection literature. The algorithms are quite successful in detecting the metadata groups (bottom row) and the cross algorithmic stability is quite good, although still lower than expected.

The network was created with standard LFR code (see Ref. [50]).

`karate` (karate club network): A well known network of friendships in a karate club in an American University [2]. After a dispute between the coach and the treasurer, the club split in two clubs. We use the standard unweighted version, with two metadata groups defined by the membership after the split.

`football` (American college football): Network of American football games between Division IA colleges during the regular season Fall 2000 [4,51]. Edges exist if two teams played any game, and groups are conferences, scheduling groups joined by the schools for the purpose of regular season scheduling. Each season, conferences mandate and schedule a certain number of intraconference games played, and other matches are decided by negotiation between schools. The data available in Ref. [52]) contain conference assignments for year 2001, and an updated version (used in this paper) contains

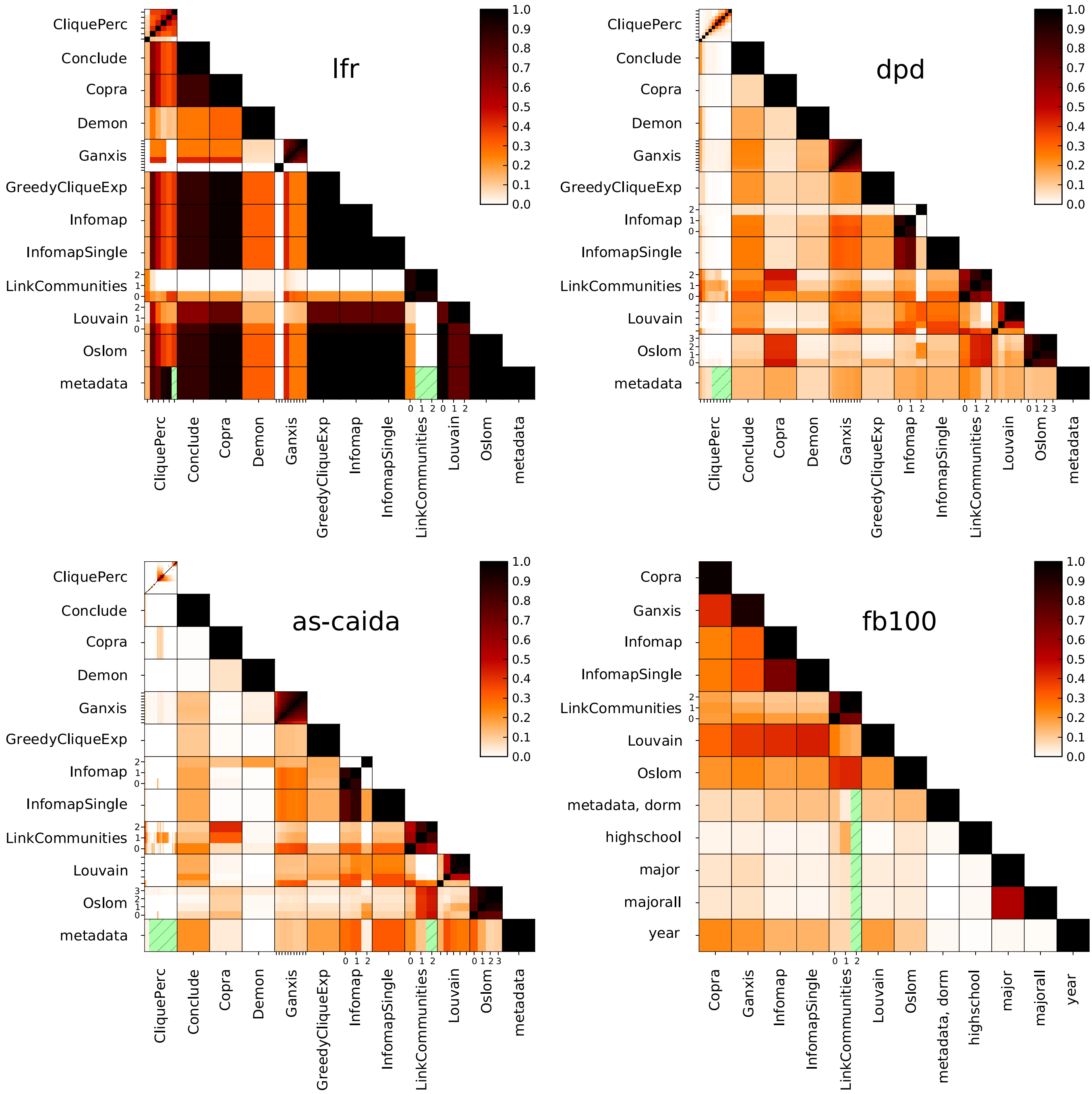

FIG. 9. (Color online) NMI grids of `lfr`, `dpd`, `pgp`, and `as-caida`. The first dataset (`lfr`) is computer generated. Some algorithms performed poorly, whereas others scored very well. Other datasets are taken from the real world, and were a bigger challenge. `fb100` is a collection of 100 data sets, so we report the averaged maximum values for each tile, except for `LinkCommunities`, where the number of layers is fixed at 3. Groups built using graduation year were detected the best.

correct conference assignments from Fall 2000, courtesy of Evans [51].

`polblogs` (political blogs): A directed network of hyperlinks between weblogs on U. S. politics, recorded in 2005 by Adamic and Glance [8]. Links are all front-page hyperlinks at the time of the crawl. Groups are “liberal” or “conservative” as assigned by either blog directories or occasional self-evaluation. The data are available in Ref. [52]).

`polbooks`: Network of books about U. S. politics from 2004 U. S. presidential election [9] taken from the online bookseller Amazon.com. Edges are Amazon recommendations on each book, indicating copurchasing by others on the site. Groups are based on political alignment of “liberal,” “neutral,” or “conservative” through human evaluation. Data can be found in Ref. [52].

`dpd`: Software dependencies within the Debian GNU/Linux operating system [53,54]. Nodes are unique software packages, such as `linux-image-2.6-amd64`,

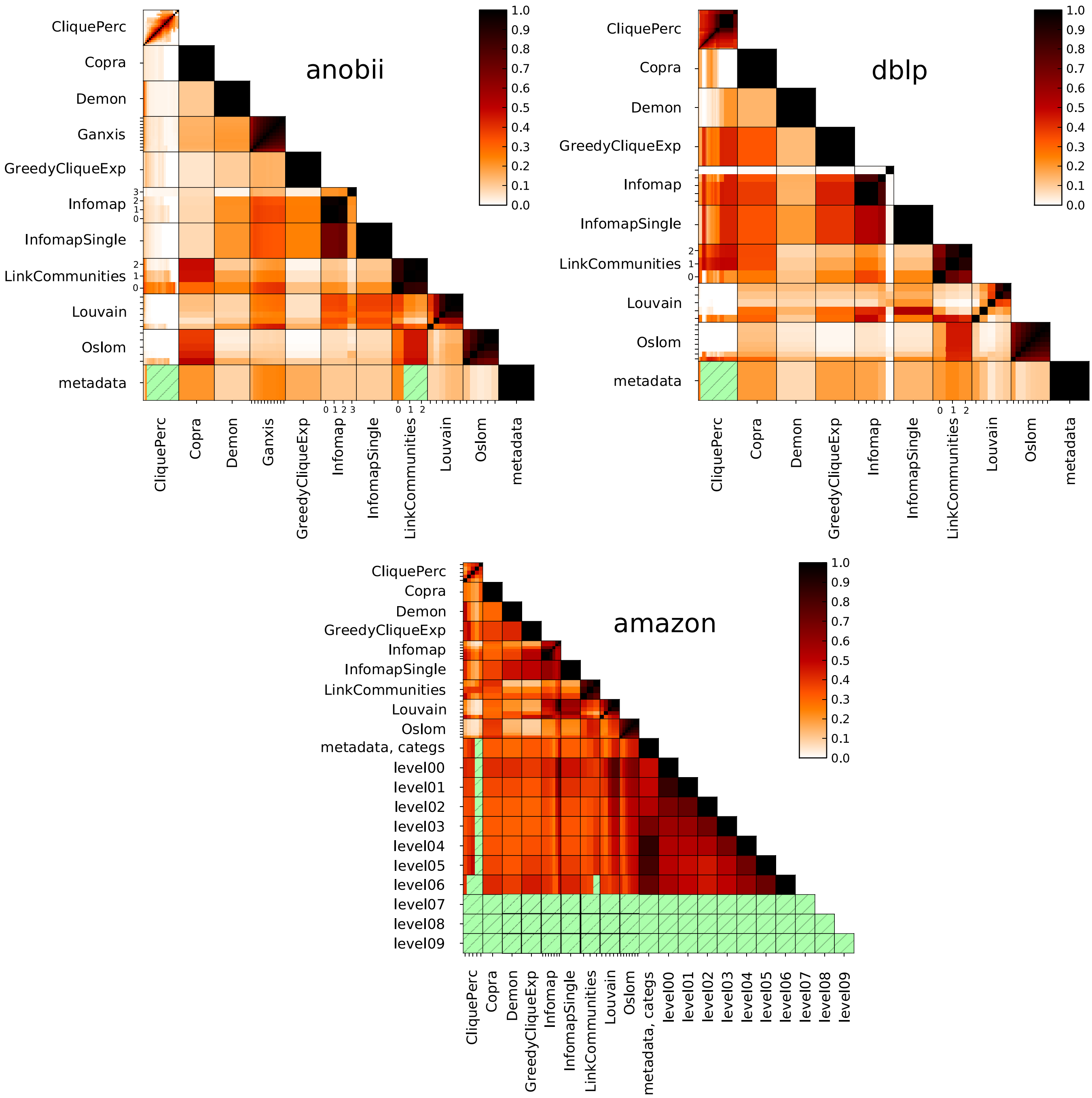

FIG. 10. (Color online) NMI grids of `anobii`, `dblp`, and `amazon`. `CliquePerc` returned many spurious layers for `anobii` and `dblp`, that were discarded due to poor coverage. More can be told for `amazon`, which contains hierarchical levels of product categories as different levels. Deeper levels were discarded, but higher ones are detected, to some degree.

libreoffice-gtk, or python-scipy. Links are the "depends," "recommends," and "suggests" relationships, which are a feature of Debian's APT package management system designed for tracking dependencies. Groups are tag memberships from the DebTags project [55], such as devel::lang:python or web::browser [56]. The network was generated from package files in Debian 7.1 Wheezy as of 2013-07-15, "main" area only. Similar files are freely available in every Debian-based OS. Tags can be found in the *_Packages files in the /var/lib/apt/ directory in an installed system or on mirrors, for example, see Ref. [57].

pgp: The "Web of trust" of PGP (Pretty Good Privacy) key signings, representing an indication of trust of the identity of one person (signee) by another (signer) [11]. A node represents one key usually, but not always, corresponding to a real person or organization. Links are signatures, which by convention are intended to only be made if the two parties are physically present, have verified each others' identities, and have verified the key fingerprints. Groups

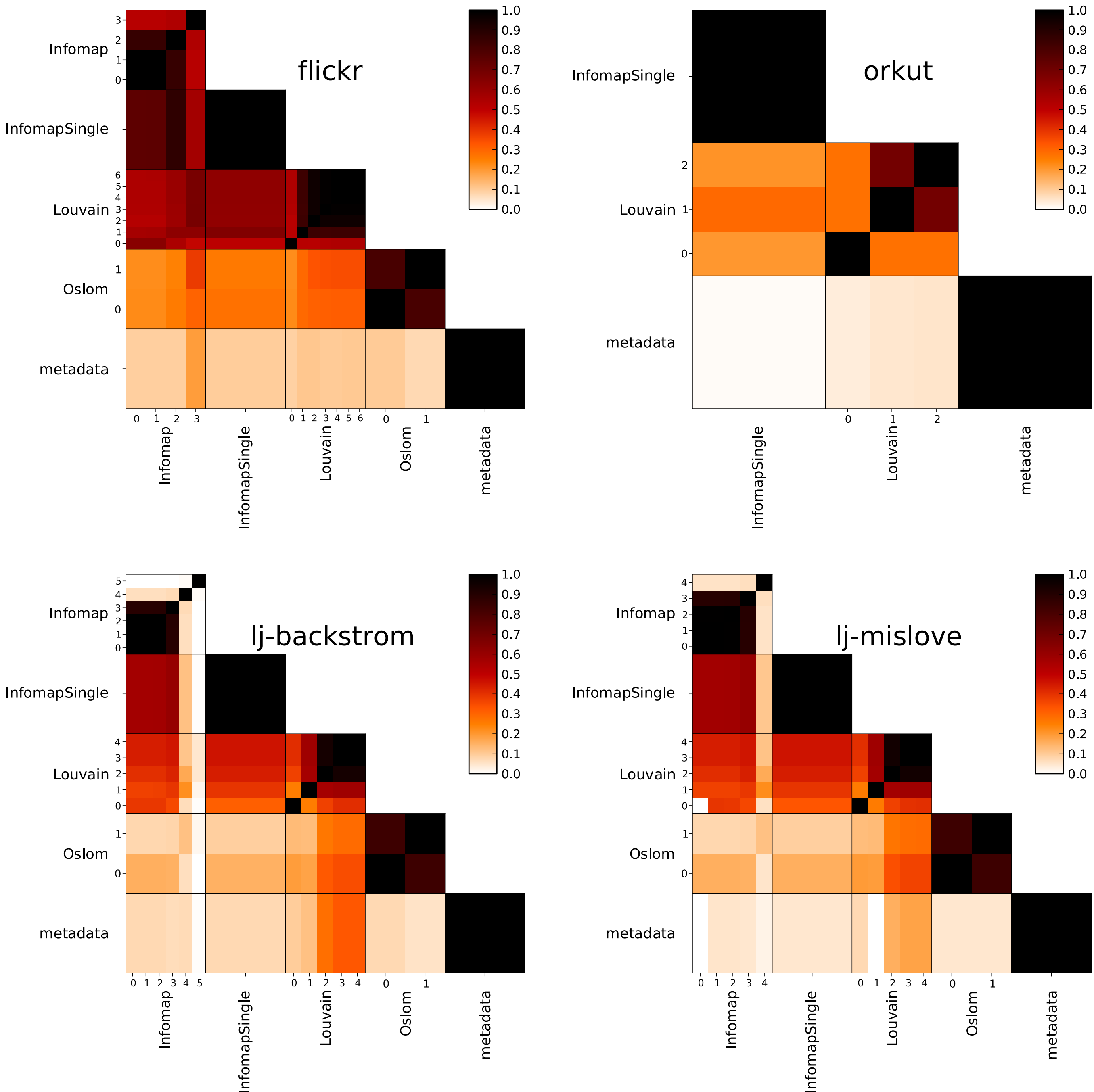

FIG. 11. (Color online) NMI grids of flickr, orkut, lj-backstrom, and lj-mislove. Due to their size, many algorithms could not be run on these data sets. The detection of the metadata partitions was poor, while the similarity of detected partitions is noticeably higher. This suggests that there is a large disparity between metadata and topological groups.

are email domain or subdomain names. The network was generated using full data downloaded from the keyserver network [58]. Signatures were not checked for cryptographic validity. Domains were broken into all subdomains, for example, the address `example@becs.aalto.fi` would be added to the three groups `becs.aalto.fi`, `aalto.fi`, and `fi`. Large webmail providers and top level domains were discarded by hand: `com`, `info`, `net`, `org`, `biz`, `name`, `pro`, `edu`, `gov`, `int`, `gmail.com`, `yahoo.com`, `mail.com`, `excite.com`, `hotmail.com`

`as-caida`: Network of the Internet at the level of autonomous systems [59]. Nodes represent autonomous systems, i.e., systems of connected routers under the control of one or more network operators with a common routing policy. Links represent observed paths of Internet Protocol traffic directly from one AS to another. Groups are countries of registration of each AS, which are by construction nonoverlapping. Data come from both the *AS Relationships Dataset* from 2013-08-01 [12] and *The IPv4 Routed /24 AS Links Dataset* from 2013-01-01 to 2013-11-25 [13]. This means that our network contains every direct link observed by these two subprojects on the Internet over a period of approximately one year. AS country assignments from all Regional Internet Registries (AFRINIC, APNIC, ARIN, LACNIC, and RIPENCC) are taken from the mirror [60] on 2013-11-25.

`amazon`: Network of product copurchases on online retailer Amazon. Nodes represent products, and edges are said to represent copurchases by other customers presented on the product page [14]. The true meaning of links is unknown and is some function of Amazon's recommendation algorithm. Data were scraped in mid-2006 and downloaded from Ref. [61]. We used copurchasing relationships as undirected edges. Product categories, such as `Books/Fiction/Fantasy` or `Books/Nonfiction` can be split into levels, which we used to make a fully hierarchical network, for example, `Books` in `layer00`, and `Books/Fiction` and `Books/Nonfiction` in `layer01`, down to `layer09`. Finally, there is one layer `categs` representing full categories, in this example `Books/Fiction/Fantasy` and `Books/Nonfiction` even though they contain a different number of "/" characters.

`anobii`: Social network of book recommendation, popular in Italy. Two types of directed relationships were taken as undirected links (friends and neighbors). Users can form and join groups. Data were provided by Aiello [15,16].

`dblp`: Network of collaboration of computer scientists. Two scientists are connected if they have coauthored at least one paper [17]. Groups are publication venues (scientific conferences). Data can be found in [5,62].

`fb100`: Facebook social networks. 100 complete (but separate) Facebook networks at United States universities in 2005. There are all friendships (undirected), as well as six pieces of node metadata: dorm (residence hall), major, second major, graduation year, former high school, and gender. These pieces of metadata were used to form separate levels of groups. Networks were originally released by Porter [18] and are available on several sites on the web. The "gender" metadata were discarded from the analysis as they form one giant network-spanning group for male and female, with isolated fringes.

`flickr`: Picture-sharing website and social network, as crawled by Mislove [19]. Nodes are users and edges exist if one user "follows" another. Groups are Flickr user groups centered around a certain type of content, such as `Nature` or `Finland`. The collectors estimate that they have a vast majority of the LWCC by comparing to a random sampling of users. 21% of users are in groups.

`lj-backstrom`: LiveJournal social network, as crawled by Backström [17]. The raw scrape from Livejournal, a now-dormant blogging service. An edge was put between users if there is any kind of relationship between them (friend or follower). Groups are based on groups which users can join.

`lj-mislove`: LiveJournal social network, as crawled by Mislove [19]. The data source and node/edge/group interpretation are the same as in `lj-backstrom`, but were independently crawled. 61% of users are in groups.

`orkut`: Orkut social network, as crawled by Mislove [19]. Nodes are users, edges are bidirectional (undirected) friendships, and groups are user-created groups. This crawl contains

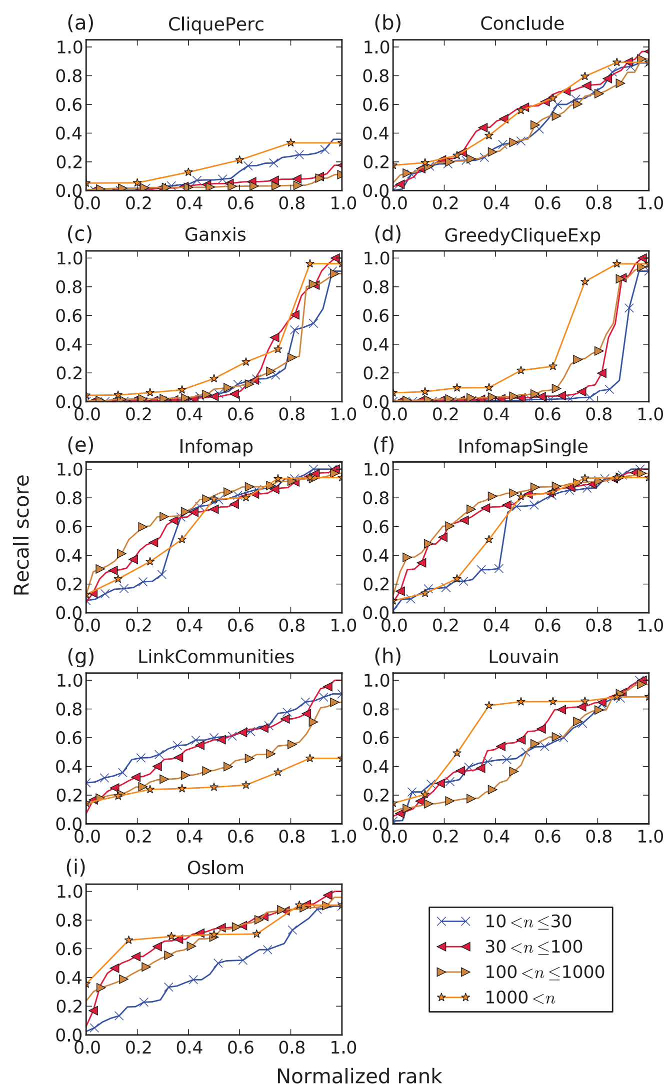

FIG. 12. (Color online) Recall of known `as-caida` groups, broken down by size n (of known groups), matched to all detected communities. We see that most methods do not have a good performance for most group sizes.

10% of Orkut's user population at the time of the crawl (according to published figures). Only 13% of users are in groups.

APPENDIX B: COMMUNITY DETECTION METHOD DESCRIPTIONS

This section contains a complete description of all community detection methods and parameters used in this work. Some methods do not scale to the largest datasets, in which case results are not presented. In analogy to the dataset preprocessing, we also remove all detected communities of size less than 3.

`Infomap` (hierarchical mode): Method based on compression of the information associated to random walks on networks [21]. Computed with code from Ref. [63] with all default settings.

`InfomapSingle` (nonhierarchical mode): Same as `Infomap` but restricted to a nonhierarchical partition [22]. Computed with the same code as `Infomap` but with the `--two-level`.

`Louvain`: Greedy, hierarchical modularity maximization algorithm [20]. For each run, it is invoked 10 times and the execution which has the maximal modularity (for each level) is taken. The updated code from Ref. [64] is used for the calculations.

`Oslom`: Order Statistics Local Optimization Method, based on community statistical significance [65]. Code from Ref. [66] is used with all default settings, in particular we run with 10 trials of the most granular level and 50 hierarchical trials of higher levels. However, the `-singlet` option is given, which causes all communities to be strictly statistically significant. Nodes not in any community are left as singletons and then removed by our postprocessing, leaving community assignments which do not cover the entire network.

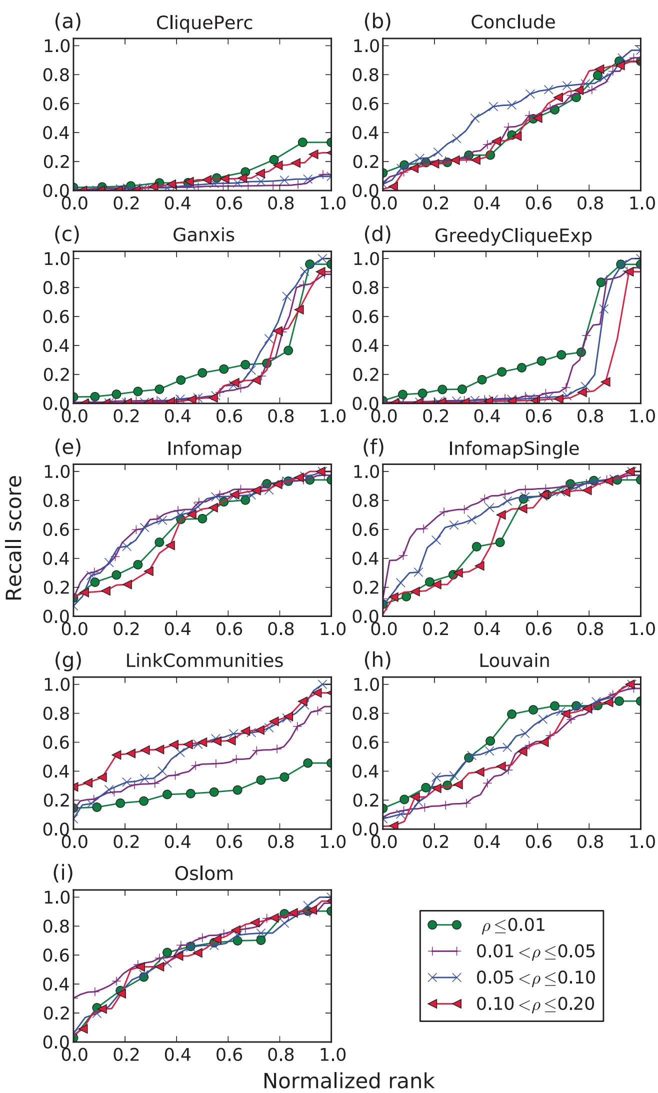

FIG. 13. (Color online) Recall of known `as-caida` groups, broken down by density, matched to all detected communities. Density is heavily correlated with inverse size, explaining the apparently higher performance on less dense groups.

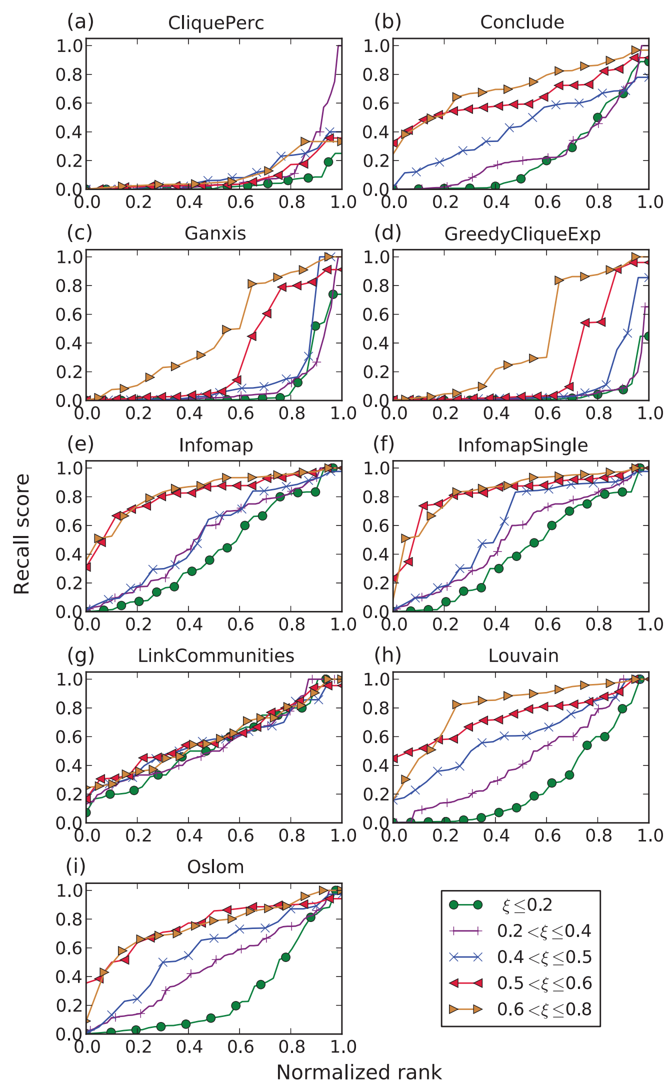

FIG. 14. (Color online) Recall of known `as-caida` groups, broken down by group embeddedness $\xi = k_{\mathrm{in}}/k_{\mathrm{tot}}$, matched to all detected communities. We see that, unlike size and density, embeddedness can very well predict the detectability of metadata groups. Higher embeddedness directly corresponds to better detectability for almost all methods.

`CliquePerc`: Clique percolation algorithm from [25]. Code from Ref. [67]. We include one layer for each clique size from $k = 3$ to $k_{\max}$ for each method. By construction, each layer does not span the entire network. Layers are numbered starting from layer 0, which is percolation of cliques with $k = 3$ (triangles), up to layer $k_{\max} - 3$, which is percolation of cliques with $k = k_{\max}$.

`Copra`: A method based on label propagation [27]. Code from Ref. [68] is used with all default parameters. In particular, this limits us to nonoverlapping communities with $v = 1$, as there is no option for automatically choosing the optimal parameter.

`Conclude`: A method using random walkers to re-weight edges, then network distances are recalculated and used to optimize weighted network modularity [26]. Code from Ref. [69] is used with all default options.

`Demon`: A method which combines node knowledge of local neighborhoods into global communities [28]. Code from Ref. [70] is used with alldefault options.

`Ganxis`: Formerly the Speaker-listener Label Propagation Algorithm (SLPA), a version of a label propagation algorithm [29]. Code version 3.0.2 from Ref. [71] with overlaps allowed, undirected mode, and one trial. We chose all other default parameters. The code by default runs with eleven thresholds $r \in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$, and all thresholds are kept (in order $0, \ldots, 10$) in our analysis.

`GreedyCliqueExp`: An algorithm, which finds cliques as seeds and then optimizes a local fitness function around those seeds [30]. Code from Ref. [72] is used with all default parameters.

`LinkCommunities`: Method partitioning links, instead of nodes, into communities [23]. Code from Ref. [73] is used with all default parameters. Instead of scanning all thresholds, we

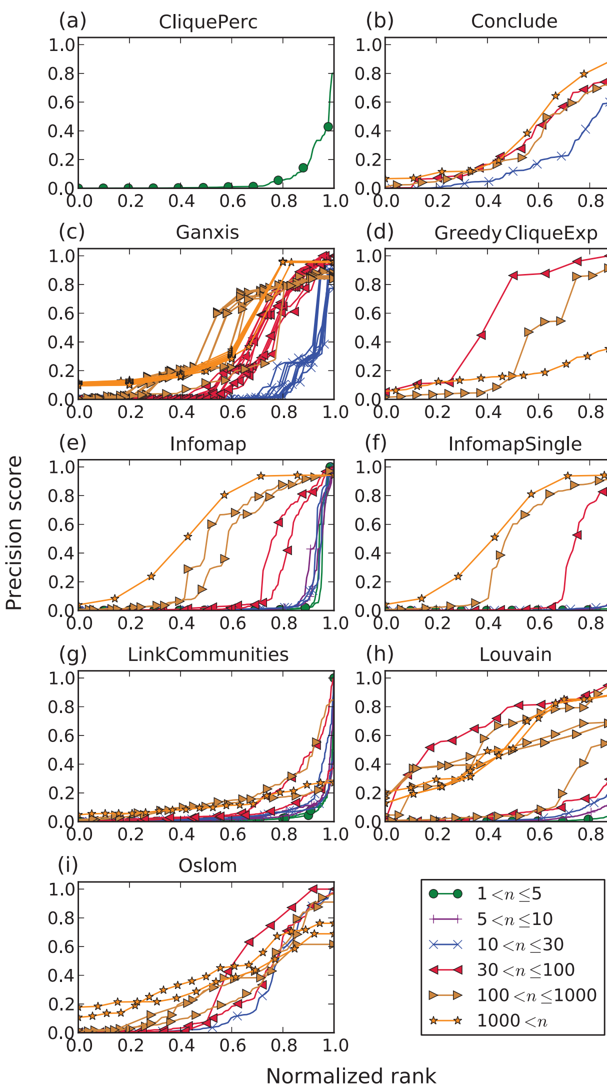

FIG. 15. (Color online) Precision of detected communities, broken down by size, compared to all known `as-caida` groups. The recall of each layer of the algorithm is plotted separately to allow us to see if any individual layers have high performance. This produces a very messy field of lines, but it is sufficient to see that there are no outliers in performance.

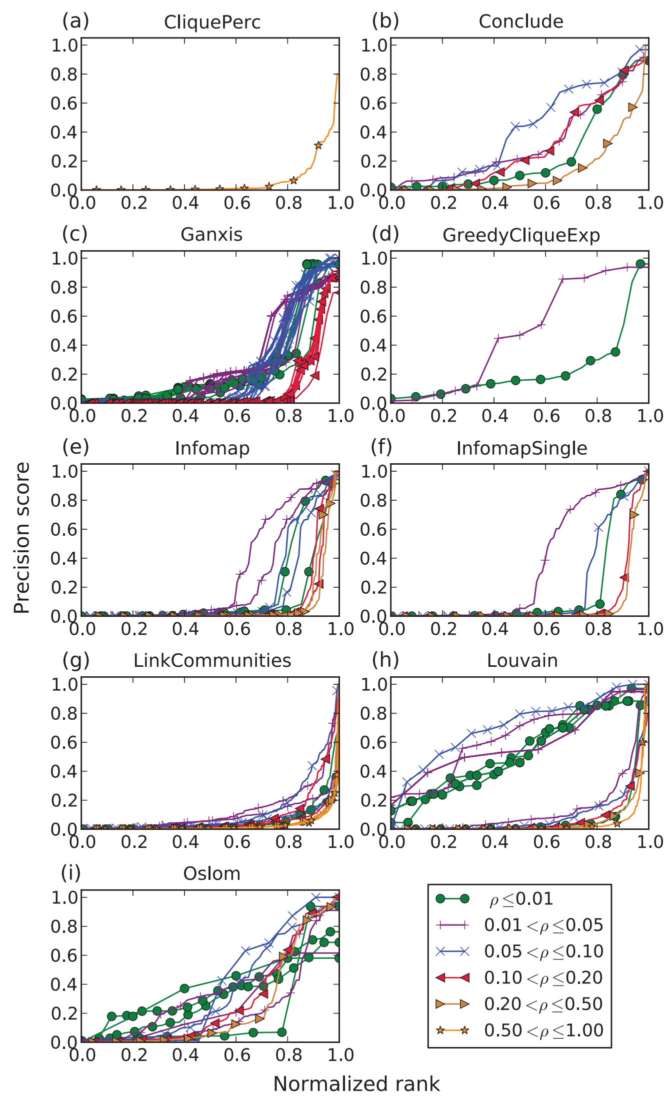

FIG. 16. (Color online) Precision of detected communities, broken down by density, compared to all known `as-caida` groups. For further information, see the caption of Fig. 15.

use three thresholds: 0.25, 0.5, and 0.75 which are identified as layers 0, 1, and 2, respectively. All default parameters are kept. Links which are not part of any community at a given threshold become *singleton links*, which become communities of size two. These communities have no significance, and thus are filtered out in our postprocessing.

APPENDIX C: NMI GRIDS

Here, we present NMI grids for all datasets (Figs. 8–11). The description is the same as for pgp in Sec. IV A.

Most of the higher order layers of `CliquePerc` (and suboptimal threshold parameter values in `LinkCommunities`), after removing singletons and doubletons, cover a very small portion of each dataset (less than 10% of the nodes), and are marked with hatched green. Larger datasets lack the results of the slowest algorithms due to computational restrictions.

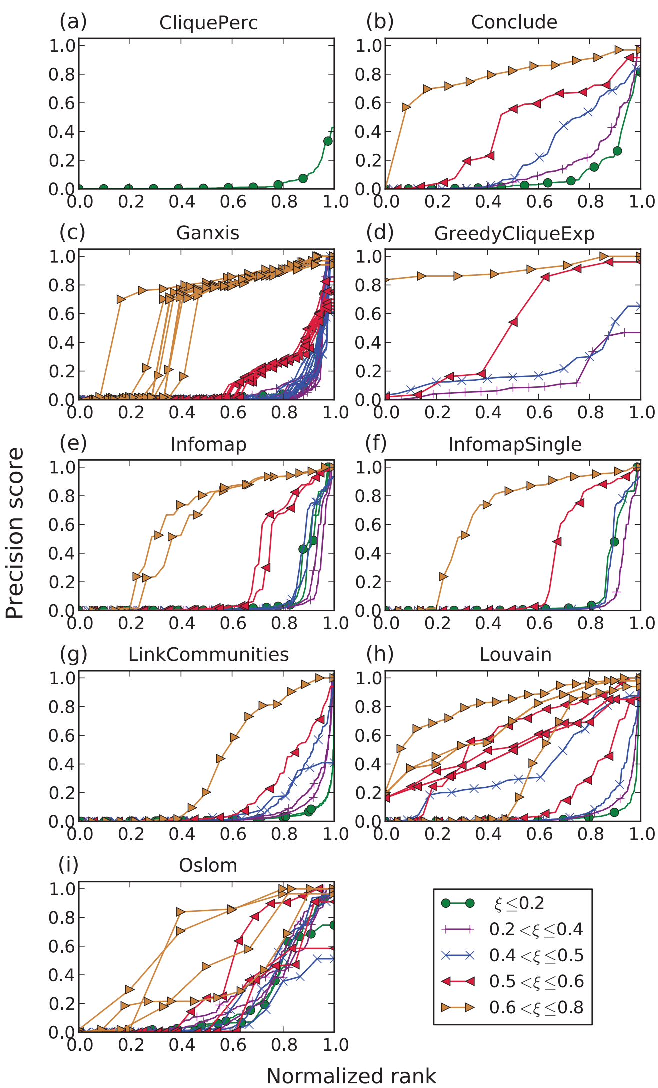

FIG. 17. (Color online) Precision of detected communities, broken down by community embeddedness, compared to all known `as-caida` groups. For further information, see the caption of Fig. 15. In contrast to Fig. 14, no particular embeddedness predicts a higher performance.

APPENDIX D: ADDITIONAL COMMUNITY-LEVEL ANALYSIS

In Sec. V, we showed that when matching community-to-community, communities detected by various algorithms often do not correspond to "true" groups, or vice versa. In this section, we will further this analysis to show that there is little opportunity for narrowing our scope to increase the predictive power of community detection methods.

We look at the properties of *group size*, *group density*, and *group embeddedness* and see if any of these are indicative of a type of group with a greater predictive power for either recall or precision. For a group of n nodes, sum of internal degrees $k_{\rm in}$, sum of total degrees of $k_{\rm tot}$, we define the density ρ as

$$\rho = \frac{k_{\rm in}}{\frac{1}{2}n(n-1)} \tag{D1}$$

and the group embeddedness ξ as

$$\xi = \frac{k_{\rm in}}{k_{\rm tot}}. \tag{D2}$$

Because some bins (parameter ranges) may have very little data, such as only one group, we only plot bins that have at least five groups and whose sum of group sizes is at least 1% of the network. Furthermore, some community

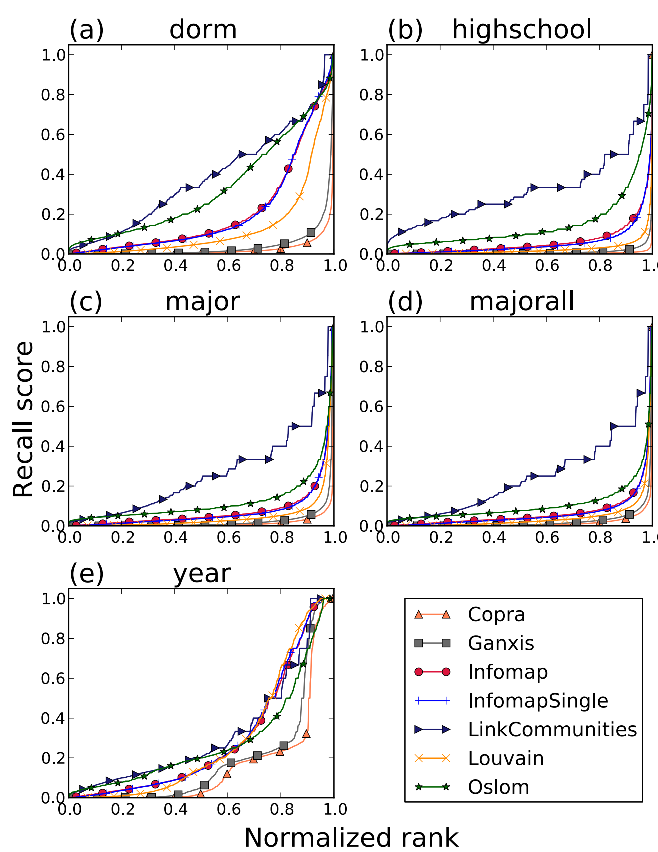

FIG. 18. (Color online) Recall of metadata groups of `fb100`. Each diagram refers to a grouping of the students based on one specific feature (e.g., their dorm, top left). We see that few groups of any type of metadata are found by any of the community detection methods.

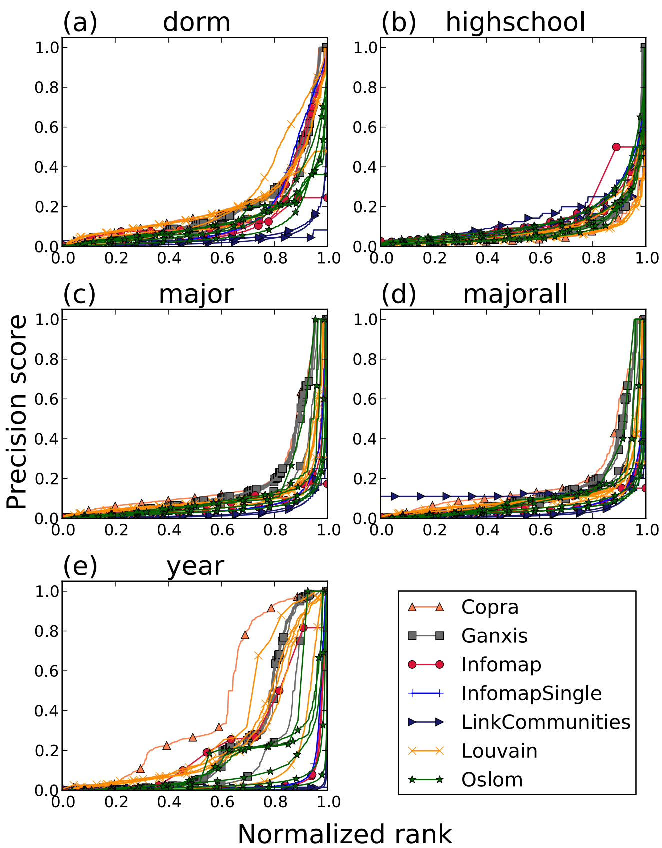

FIG. 19. (Color online) Precision of each partition level of several community detection algorithms with respect to metadata groups of `fb100`, corresponding to a single feature of the students. For those algorithms returning multiple detected levels, each level is plotted separately in order to see the performance of each detected layer individually. This produces a field of lines, but it is sufficient to see that there are no outliers in performance.

detection methods return multiple covers of the system, from different input parameters (see Appendix B). When computing recall, a known group is matched to every detected community regardless of its detected layer or size, density, or embeddedness. When computing precision, one could ask if any one particular layer would have greater predictive power than all layers taken together. To show this, we plot the precision of each detected layer separately. If one particular layer or set of parameters was very good, then we could see one line above the rest. As we will see, there are no significant outliers, so the identity of each line does not matter. This procedure is performed on the precision plots from Figs. 12 to 19.

In Figs. 12–17, we see the recall and precision of the `as-caida` dataset broken down by the group properties above. `Copra` and `Demon` did not return sufficient communities in each bin to perform a meaningful analysis, so their results are not shown. In Figs. 12 and 15, we see the recall (precision) of groups as a function of the size of the known (detected) group. We are able to see some variations in the performance. Most methods seem to do a better job in detecting large groups than small ones. A notable exception is `LinkCommunities`, which has the highest recall for the smallest metadata groups, although the precision for the smallest detected communities is not the highest. In general, the curves are quite close to each other. For some algorithms, such as `GreedyCliqueExp`, `InfomapSingle`, `LinkCommunities`, and `OSLOM`, there is a more visible spread of the curves.

If we consider density bins, Figs. 13 and 16 show a consistent pattern as that observed for Figs. 12 and 15, as link density is correlated to group size: small groups tend to have higher link density than large groups.

Finally, if one considers embeddedness (Figs. 14 and 17), both recall and precision are highest for the most embedded groups, i.e., the ones most weakly attached to the rest of the system, and systematically decreases if embeddedness decreases. This is expected, as most algorithms look for subgraphs which are loosely connected to the rest of the system, and high embeddedness means high separation.

The `fb100` data set provides us with a unique opportunity to further understand the factors which allow high community detection performance. It is a collection including the Facebook social networks at 100 universities, with different types of metadata to allow us to form groups of different types. We can see if methods can better detect groups of a certain type. The metadata include dorm (the student residence), high school (the school of each user before attending university), major (the student's field of study), majorall [the student's major(s) possibly including a second major], and year (the student's graduation year). In Figs. 18 and 19, we plot the recall (precision) of various methods with respect to groups corresponding to each of the above attributes, averaged over the 100 universities included in this dataset. None of the students' features appear to generate well recoverable groups. `LinkCommunities` appears to have a higher recall than most methods, for each grouping of the students, but it has much lower precision, due to the much bigger number of detected groups.

[1] S. Fortunato, Phys. Rep. **486**, 75 (2010).
[2] W. W. Zachary, J. Anthropol. Res. **33**, 452 (1977).
[3] D. Lusseau, Proc. R. Soc. London, Ser. B **270**, S186 (2003).
[4] M. Girvan and M. E. Newman, Proc. Natl. Acad. Sci. USA **99**, 7821 (2002).
[5] J. Yang and J. Leskovec, in *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics* (ACM, New York, NY, 2012), pp. 3:1–3:8.
[6] J. Yang and J. Leskovec, in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining* (ACM, New York, NY, 2013), pp. 587–596.
[7] J. Yang and J. Leskovec, ACM Trans. Intell. Syst. Technol. **5**, 26:1 (2014).
[8] L. A. Adamic and N. Glance, in *LinkKDD '05: Proceedings of the 3rd international Workshop on Link Discovery* (ACM, New York, NY, 2005), pp. 36–43.
[9] V. Krebs, http://www.orgnet.com/

[10] A. Lancichinetti, S. Fortunato, and F. Radicchi, Phys. Rev. E **78**, 046110 (2008).

[11] S. Garfinkel, PGP: Pretty Good Privacy, O'Reilly Media, Inc., 1995.

[12] See http://www.caida.org/data/as-relationships/

[13] See http://www.caida.org/data/active//ipv4_routed_topology_aslinks_dataset.xml

[14] J. Leskovec, L. A. Adamic, and B. A. Huberman, ACM Trans. Web **1**, 5 (2007).

[15] L. M. Aiello, M. Deplano, R. Schifanella, and G. Ruffo, in *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media* (AAAI, Palo Alto, 2012), pp. 10–17.

[16] L. M. Aiello, A. Barrat, C. Cattuto, G. Ruffo, and R. Schifanella, in *Social Computing (SocialCom), 2010 IEEE Second International Conference* (IEEE, New York, 2010), pp. 249–256.

[17] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan, in *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, NY, 2006), pp. 44–54.

[18] A. L. Traud, P. J. Mucha, and M. A. Porter, Physica A (Amsterdam) **391**, 4165 (2012).

[19] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement* (ACM, New York, 2007), pp. 29–42.

[20] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, J. Stat. Mech. (2008) P10008.

[21] M. Rosvall and C. T. Bergstrom, PLoS ONE **6**, e18209 (2011).

[22] M. Rosvall and C. T. Bergstrom, Proc. Natl. Acad. Sci. USA **105**, 1118 (2008).

[23] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, Nature (London) **466**, 761 (2010).

[24] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, Nature (London) **435**, 814 (2005).

[25] F. Reid, A. McDaid, and N. Hurley, in *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference* (IEEE, New York, 2012), pp. 274–281.

[26] P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti, J. Comput. Syst. Sci. **80**, 72 (2014).

[27] S. Gregory, New J. Phys. **12**, 103018 (2010).

[28] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi, in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, NY, 2012), pp. 615–623.

[29] J. Xie and B. K. Szymanski, in *Advances in Knowledge Discovery and Data Mining* (Springer, Berlin, 2012), pp. 25–36.

[30] C. Lee, F. Reid, A. McDaid, and N. Hurley, arXiv:1002.1827.

[31] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, J. Stat. Mech. (2005) P09008.

[32] A. Lancichinetti, S. Fortunato, and J. Kertesz, New J. Phys. **11**, 033015 (2009).

[33] B. Abrahao, S. Soundarajan, J. Hopcroft, and R. Kleinberg, in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, NY, 2012), pp. 624–632.

[34] M. Ester, R. Ge, B. J. Gao, Z. Hu, and B. Ben-Moshe, in *SDM'06*, edited by J. Ghosh, D. Lambert, D. B. Skillicorn, and J. Srivastava (SIAM, Philadelphia, 2006).

[35] Y. Liu, A. Niculescu-Mizil, and W. Gryc, in *Proceedings of the 26th Annual International Conference on Machine Learning* (ACM, New York, NY, 2009), pp. 665–672.

[36] F. Moser, R. Colak, A. Rafiey, and M. Ester, in *SDM'09* (SIAM, Philadelphia, 2009), pp. 593–604.

[37] Y. Zhou, H. Cheng, and J. X. Yu, Proc. VLDB Endow. **2**, 718 (2009).

[38] L. Tang, X. Wang, and H. Liu, in *IEEE International Conference on Data Mining (ICDM'09)* (IEEE, New York, 2009), pp. 503–512.

[39] A. Silva, W. Meira, Jr., and M. J. Zaki, in *Proceedings of the Eighth Workshop on Mining and Learning with Graphs* (ACM, New York, NY, 2010), pp. 119–126.

[40] R. Balasubramanyan and W. W. Cohen, in *SDM'11* (SIAM, Philadelphia, 2011), pp. 450–461.

[41] M. Atzmueller and F. Mitzlaff, in *Proceedings of the Twenty-Fourth International Florida Artificial Intelligence Research Society Conference, May 18-20, 2011, Palm Beach, Florida, USA*, edited by R. C. Murray and P. M. McCarthy (AAAI Press, Palo Alto, CA, 2011), pp. 459–464.

[42] L. Akoglu, H. Tong, B. Meeder, and C. Faloutsos, in *SDM'12* (SIAM, Philadelphia, 2012), pp. 439–450.

[43] F. Bonchi, A. Gionis, F. Gullo, and A. Ukkonen, in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, NY, 2012), pp. 1321–1329.

[44] Y. Sun, C. C. Aggarwal, and J. Han, Proc. VLDB Endow. **5**, 394 (2012).

[45] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng, in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data* (ACM, New York, NY, 2012), pp. 505–516.

[46] N. Barbieri, F. Bonchi, and G. Manco, in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining* (ACM, New York, NY, 2013), pp. 33–42.

[47] Y. Ruan, D. Fuhry, and S. Parthasarathy, in *Proceedings of the 22nd International Conference on World Wide Web* (International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 2013), pp. 1089–1098.

[48] J. Yang, J. McAuley, and J. Leskovec, in *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM)* (IEEE, Piscataway, NJ, 2013), pp. 1151–1156.

[49] S. Pool, F. Bonchi, and M. van Leeuwen, ACM Trans. Intell. Syst. Technol. **5**, 28:1 (2014).

[50] See https://sites.google.com/site/santofortunato/inthepress2

[51] T. Evans, http://dx.doi.org/10.6084/m9.figshare.93179

[52] http://www-personal.umich.edu/mejn/netdata/

[53] See http://debian.org/

[54] A. Barth, A. Di Carlo, R. Hertzog, L. Nussbaum, C. Schwarz, and I. Jackson, https://www.debian.org/doc/manuals/developers-reference/

[55] See https://wiki.debian.org/Debtags

[56] E. Zini, in *Proceedings of the 5th annual Debian Conference,* Helsinki, Finland, 2005, pp. 59–74, http://debtags.alioth.debian.org/paper-debtags.html

[57] See ftp://ftp.debian.org/debian/dists/wheezy/main/binary-amd64/

[58] See http://sks-keyservers.net

[59] J. Hawkinson and T. Bates, http://tools.ietf.org/html/rfc1930

[60] See ftp://ftp.ripe.net/pub/stats/

[61] See http://snap.stanford.edu/data/amazon-meta.html
[62] See http://snap.stanford.edu/data/com-DBLP.html
[63] See http://mapequation.org
[64] See https://sites.google.com/site/findcommunities/
[65] A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato, PLoS ONE **6**, e18961 (2011).
[66] See http://oslom.org/
[67] See https://github.com/aaronmcdaid/MaximalCliques
[68] See http://www.cs.bris.ac.uk/steve/networks/software/copra.html
[69] See http://www.emilio.ferrara.name/conclude/
[70] See http://www.michelecoscia.com/?page_id=42
[71] See https://sites.google.com/site/communitydetectionslpa/
[72] See https://sites.google.com/site/greedycliqueexpansion/ (version r2011-11-06).
[73] See http://barabasilab.neu.edu/projects/linkcommunities/