Towards Linear Time Overlapping Community Detection in Social Networks

Jierui Xie and Boleslaw K. Szymanski

Rensselaer Polytechnic Institute
Troy, New York 12180, USA
{xiej2,szymansk}@cs.rpi.edu

Abstract. Membership diversity is a characteristic aspect of social networks in which a person may belong to more than one social group. For this reason, discovering *overlapping* structures is necessary for realistic social analysis. In this paper, we present a fast algorithm[1], called SLPA, for overlapping community detection in large-scale networks. SLPA spreads labels according to dynamic interaction rules. It can be applied to both *unipartite* and *bipartite* networks. It is also able to uncover overlapping *nested hierarchy*. The time complexity of SLPA scales *linearly* with the number of edges in the network. Experiments in both synthetic and real-world networks show that SLPA has an excellent performance in identifying both *node* and *community* level overlapping structures.

1 Introduction

Community or modular structure is considered to be a significant property of real-world social networks. Thus, numerous techniques have been developed for effective community detection. However, most of the work has been done on *disjoint* community detection. It has been well understood that people in a real social network are naturally characterized by *multiple* community memberships. For example, a person usually has connections to several social groups like family, friends and colleges; a researcher may be active in several areas; in the Internet, a person can simultaneously subscribe to an arbitrary number of groups.

For this reason, overlapping community detection algorithms have been investigated. These algorithms aim to discover a *cover* [7], defined as a set of clusters in which each node belongs to at least one cluster. In this paper, we propose an efficient algorithm for detecting both individual overlapping nodes and overlapping communities using the underlying network structure alone.

2 Related Work

We review the state of the art and categorize existing algorithms into five classes that reflect how communities are identified.

---

[1] Available at https://sites.google.com/site/communitydetectionslpa/

Clique Percolation: CPM [11] is based on the assumption that a community consists of fully connected subgraphs and detects overlapping communities by searching for *adjacent* cliques. CPMw [4] extends CPM for weighted networks by introducing a subgraph intensity threshold.

Local Expansion: The iterative scan algorithm (IS) [2],[6] expands small cluster cores by adding or removing nodes until a local density function cannot be improved. The quality of seeds dictates the quality of discovered communities. LFM [7] expands a community from a random node. The size and quality of the detected communities depends significantly on the resolution parameter of the fitness function. EAGLE [14] and GCE [9] start with all maximal cliques in the network as initial communities. EAGLE uses the agglomerative framework to produce a dendrogram in $O(n^2 s)$ time, where n is the number of nodes, and s is the maximal number of join operations. In GCE communities that are similar within a distance ϵ are removed. The greedy expansion takes $O(mh)$ time, where m is the number of edges, and h is the number of cliques.

Fuzzy Clustering: Zhang [16] used the spectral method to embed the graph into low dimensionality Euclidean space. Nodes are then clustered by the fuzzy c-mean algorithm. Psorakis et al. [12] proposed a model based on Bayesian non-negative matrix factorization (NMF). These algorithms need to determine the number of communities K and the use of matrix multiplication makes them inefficient. For NMF, the complexity is $O(Kn^2)$.

Link Partitioning: Partitioning links instead of nodes to discover communities has been explored, where the node partition of a link graph leads to an edge partition of the original graph. In [1], single-linkage hierarchical clustering is used to build a link dendrogram. The time complexity is $O(nk_{max}^2)$, where k_{max} is the highest degree of the n nodes.

Dynamical Algorithms: Label propagation algorithms such as [13, 5, 15] use labels to uncover communities. In COPRA [5], each node updates its belonging coefficients by *averaging* the coefficients from all its neighbors in a synchronous fashion. The time complexity is $O(vm \log(vm/n))$ per iteration, where parameter v controls the maximum number of communities with which a node can associate, m and n are the number of edges and number of nodes respectively.

3 SLPA: Speaker-listener Label Propagation Algorithm

Our algorithm is an extension of the Label Propagation Algorithm (LPA) [13]. In LPA, each node holds only a single label and iteratively updates it to its neighborhood majority label. Disjoint communities are discovered when the algorithm converges. Like [5], our algorithm accounts for *overlap* by allowing each node to possess multiple labels but it uses different dynamics with more general features.

SLPA mimics human pairwise communication behavior. In each communication step, one node is a speaker (information provider), and the other is a listener (information consumer). Unlike other algorithms, each node has a *memory* of the labels received in the past and takes its content into account to make the
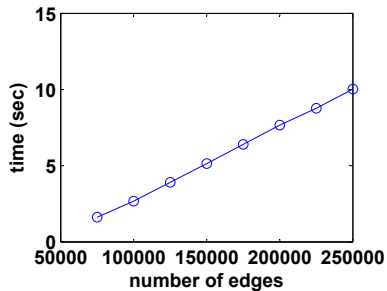
Fig. 1: The execution times of SLPA in synthetic networks with $n = 5000$ and average degree $\overline{k}$ varying from 10 to 80.

current decisions. This allows SLPA to avoid producing a number of small size communities as opposed to other algorithms. In a nutshell, SLPA consists of the following three stages:

Algorithm 1 : SLPA(T, r)

T: the user defined maximum iteration

r: post-processing threshold

1) First, the *memory* of each node is initialized with a unique label.
2) Then, the following steps are repeated until the maximum iteration T is reached:
   a. One node is selected as a listener.
   b. Each neighbor of the selected node randomly selects a label with probability proportional to the occurrence frequency of this label in its memory and sends the selected label to the listener.
   c. The listener adds the most popular label received to its memory.
3) Finally, the post-processing based on the labels in the memories and the threshold r is applied to output the communities.

Note that SLPA starts with each node being in its own community (a total of n), the algorithm explores the network and outputs the desired number of communities in the end. As such, the number of communities is not required as an input. Due to the step c, the size of memory increases by one for each node at each step. SLPA reduces to LPA when the size of memory is limited to one and the stop criterion is convergence of all labels. Empirically, SLPA produces relatively stable outputs, independent of network size or structure, when T is greater than 20. Although SLPA is non-deterministic due to the random selection and ties, it performs well on average as shown in later sections.

Post-processing and Community Detection: In SLPA, the detection of communities is performed when the stored information is post-processed. Given the memory of a node, SLPA converts it into a probability distribution of labels. Since labels represent community id's, this distribution naturally defines the

strength of association to communities to which the node belongs. To produce *crisp* communities in which the membership of a node to a given community is *binary*, i.e., either a node is in a community or not, a simple thresholding procedure is performed: if the probability of seeing a particular label during the whole process is less than a given threshold $r \in [0, 0.5]$, this label is deleted. After thresholding, connected nodes having a particular label are grouped together and form a community. If a node contains multiple labels, it belongs to more than one community and is called an *overlapping node*. A smaller value of r produces a larger number of communities. However, the effective range is typically narrow in practice. When $r \geq 0.5$, SLPA outputs disjoint communities.

Complexity: The initialization of labels requires $O(n)$, where n is the total number of nodes. The outer loop is controlled by the user defined maximum iteration T, which is a small constant which in our experiments was set to 100. The inner loop is controlled by n. Each operation of the inner loop executes one speaking rule and one listening rule. The speaking rule requires exactly $O(1)$ operation. The listening rule takes $O(\overline{k})$ on average, where $\overline{k}$ is the average node degree. In the post-processing, the thresholding operation requires $O(Tn)$ operations since each node has a memory of size T. In summary, the time complexity of the entire algorithm is $O(Tn\overline{k})$ or $O(Tm)$, linear with the total number of edges m. The execution times for synthetic networks where averaged for each $\overline{k}$ over networks with different structures, i.e., different degree and community size distributions (see Section 4.1 for details). The results shown in Fig. 1 confirm the *linear* scaling of the execution times. On a desktop with 2.80GHz CPU, SLPA took about six minutes to run over a two million nodes Amazon co-purchasing, which is ten times faster than GCE running over the same network.

4 Tests in Synthetic Networks

4.1 Methodology

To study the behavior of SLPA, we conducted extensive experiments in both synthetic and real-world networks. For synthetic random networks, we adopted the widely used LFR benchmark[2] [8], which allows heterogeneous distributions of node degrees and community sizes.

Table 1: Algorithms in the tests.

Algorithm	Complexity	Imp
CFinder [11], 2005	-	C++
LFM [7], 2009	$O(n^2)$	C++
EAGLE [14], 2009	$O(n^2 s)$	C++
CIS [6], 2009	$O(n^2)$	C++
GCE [9], 2010	$O(mh)$	C++
COPRA [5], 2010	$O(vm \log(vm/n))$	Java
NMF [12], 2010	$O(Kn^2)$	Matlab
Link [1], 2010	$O(nk_{max}^2)$	C++
SLPA, 2011	$O(Tm)$	C++

Table 2: The ranking of algorithms.

Rank	RS_{Omega}	RS_{NMI}	RS_F
1	SLPA	SLPA	SLPA
2	COPRA	GCE	CFinder
3	GCE	NMF	COPRA
4	CIS	CIS	Link
5	NMF	LFM	LFM
6	LFM	COPRA	CIS
7	CFinder	CFinder	GCE
8	Link	EAGLE	EAGLE
9	EAGLE	Link	NMF

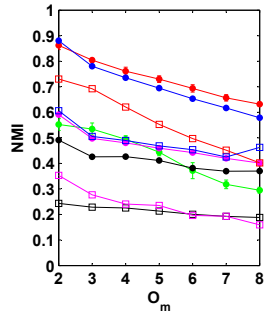[2] http://sites.google.com/site/andrealancichinetti/files

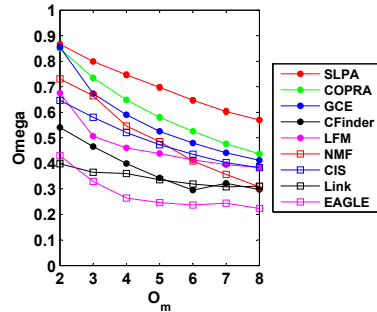Fig. 2: NMI as a function of the number of memberships O_m in LFR.

Fig. 3: Omega as a function of the number of memberships O_m in LFR.

We used networks with size $n = 5000$. The average degree is kept at $\overline{k} = 10$. The degree of overlapping is determined by two parameters. $\boldsymbol{O_n}$ defines the number of overlapping nodes and is set to 10% of all nodes. $\boldsymbol{O_m}$ defines the number of communities to which each overlapping node belongs and varies from 2 to 8 indicating the diversity of overlap. By increasing the value of O_m, we create harder detection tasks. Other parameters are as follows: node degrees and community sizes are governed by the power laws with exponents 2 and 1; the maximum degree is 50; the community size varies from 20 to 100; the expected fraction of links of a node connecting it to other communities, called the mixing parameter μ, is set to 0.3. We generated ten instance networks for each setting.

In Table 1, we compared SLPA with eight other algorithms representing different categories discussed in section 2. For algorithms with tunable parameters, the performance with the optimal parameter is reported. For CFinder, k varies from 3 to 10; for COPRA, v varies from 1 to 10; for LFM α is set to 1.0 [7]. For $Link$, the threshold varies from 0.1 to 0.9 with an interval 0.1. For SLPA, the number of iterations T is set to 100 and r varies from 0.01 to 0.1. The average performance together with error bar over ten repetitions are reported for SLPA and COPRA. For NMF, we applied a threshold varying from 0.05 to 0.5 with an interval 0.05 to convert it to a crisp clustering.

To summarize the vast amount of comparison results and provide a measure of relative performance, we proposed $RS_M(i)$, the averaged ranking for algorithm i with respect to measure M as follows:

$$RS_M(i) = \sum_{j=1} w_j \cdot rank(i, O_m^j), \tag{1}$$

where O_m^j is the number of memberships in $\{2, 3, \cdots, 8\}$, w_j is the weight, and function $rank$ returns the ranking of algorithm i for the given O_m. For simplicity, we assume equal weights over different O_m's in this paper. Sorting RS_M in increasing order gives the final ranking among algorithms.
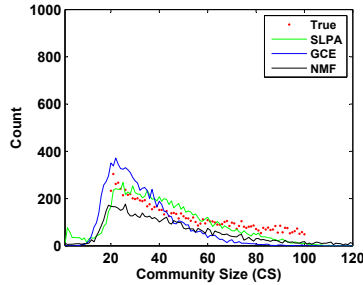
Fig. 4: The unimodal histogram of the detected community sizes for SLPA, GCE and NMF.
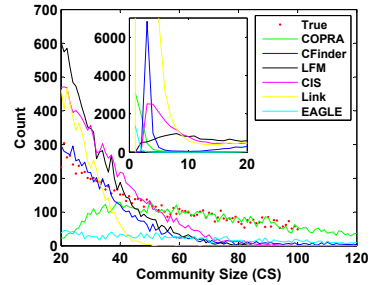

Fig. 5: The bimodal histogram of the detected community sizes for COPRA, CIS, LFM, Link, EAGLE and CFinder.

4.2 Identifying Overlapping Communities in LFR

The extended normalized mutual information (NMI) [7] and Omega Index [3] are used to quantify the quality of communities discovered by an algorithm. NMI measures the fraction of nodes in agreement in two covers, while Omega is based on pairs of nodes. Both NMI and Omega yield the values between 0 and 1. The closer this value is to 1, the better the performance is.

As shown in Fig. 2 and Fig. 3, some algorithms behave differently under different measures (the rankings of RS_{Omega} and RS_{NMI} among algorithms in Table 2 also change). As opposed to NMF and COPRA, which are especially sensitive to the measure, SLPA is remarkably stable in NMI and Omega.

Comparing the detected and known numbers of communities and distributions of community sizes (CS) helps to understand the results. On one hand, we expect the community size to follow a power law with exponent 1 and to range from 20 to 100 by design. As shown in Fig. 4, high-ranking (with high NMI) algorithms such as SLPA, GCE and NMF typically yield a *unimodal* distribution with a peak at $CS = 20$ fitting well with the ground truth distribution. In contrast, algorithms in Fig. 5 typically produce a *bimodal* distribution. The existence of an extra dominant mode for CS ranging from 1 to 5 results in a significant number of small size communities in CFinder, LFM, COPRA, Link and CIS. These observations nicely explain the ranking with respect to NMI.

4.3 Identifying Overlapping Nodes in LFR

Identifying nodes overlapping multiple communities is an essential component of measuring the quality of a detection algorithm. However, the node level evaluation was often neglected. Here we first look at the number of detected overlapping nodes O_n^d (see Fig. 6) and detected memberships O_m^d (see Fig. 7) relative to the ground truth O_n and O_m, based on the information in Fig. 2. Note that a value close to 1 indicates closeness to the ground truth, and values over 1 are possible when an algorithm detects more nodes or memberships than there are known to
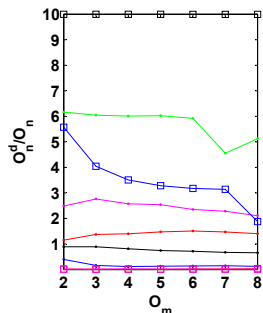
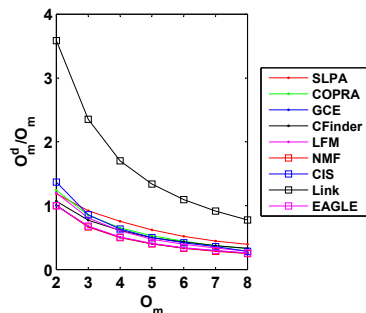Fig. 6: The number of detected overlapping nodes relative to the ground truth.

Fig. 7: The number of memberships of detected overlapping nodes relative to the ground truth.

exist. As shown, SLPA yields the numbers that are close to the ground truth in both cases.

Note that O_n^d alone is insufficient to accurately quantify the detection performance, as it contains both true and false positive. To provide precise analysis, we consider the identification of overlapping nodes as a *binary classification* problem. A node is labeled as *overlapping* as long as $O_m>1$ or $O_m^d>1$ and labeled as *non-overlapping* otherwise. Within this framework, we can use F-score as a measure of detection accuracy defined as

$$F = \frac{2 \cdot precision \cdot recall}{precision + recall}, \tag{2}$$

where *recall* is the number of correctly detected overlapping nodes divided by O_n, and *precision* is the number of correctly detected overlapping nodes divided by O_n^d. F-score reaches its best value at 1 and worst score at 0.

As shown in Fig. 8, SLPA achieves the best score on this metric. This score has a positive correlation with O_m while scores of other algorithms are negatively correlated with it. SLPA correctly uncovers a reasonable fraction of overlapping nodes even when those nodes belong to many groups (as demonstrated by the high precision and recall in Fig. 9 and Fig. 10). Other algorithms that fail to have a good balance between precision and recall result in low F-score, especially for EAGLE and Link. The high precision of EAGLE (also CFinder and GCE for $O_m = 2$) shows that clique-like assumption of communities may help to identify overlapping nodes. However, they under-detect the number of such nodes.

With the F-score ranking, GCE and NMF no longer rank in the top three algorithms, while SLPA stays there. Taking both community level performance (NMI and Omega) and node level performance (F-score) into account, we conclude that SLPA performs well in the LFR benchmarks.
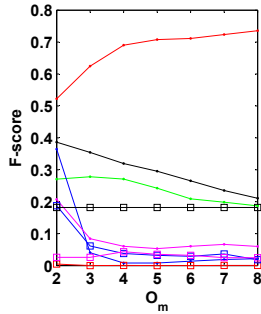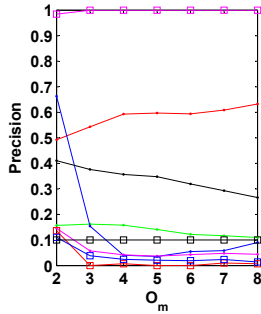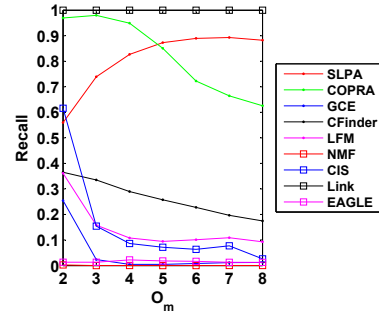
Fig. 8: The F-score. Fig. 9: The precision. Fig. 10: The recall.

Table 3: Social networks in the tests

Network	n	$\overline{k}$	Network	n	$\overline{k}$
karate (KR)	34	4.5	Email (EM)	33696	10.7
football (FB)	115	10.6	P2P	62561	2.4
lesmis (LS)	77	6.6	Epinions (EP)	75877	10.6
dolphins (DP)	62	5.1	Amazon (AM)	262111	6.8
CA-GrQc (CA)	4730	5.6	HighSchool (HS1)	69	6.3
PGP	10680	4.5	HighSchool (HS2)	612	8.0

5   Tests in Real-world Social Networks

We applied SLPA to a wide range of well-known social networks[3] as listed in Table 3. The high school friendship networks that were analyzed in a project funded by the National Institute of Child Health and Human Development, are social networks in high schools self-reported by students together with their grades, races and sexes. We used these additional attributes for verification.

5.1   Identifying Overlapping Communities in Social Networks

To quantify the performance, we used the overlapping modularity, Q_{ov}^{Ni} (with values between 0 and 1), proposed by Nicosia [10], which is an extension of Newman's modularity. A high value indicates a significant overlapping community structure relative to the null model. We removed CFinder, EAGLE and NMF from the test because of either their memory or their computation inefficiency on large networks. As an additional reference, we added the disjoint detection results with the Infomap algorithm.

   As shown in Fig. 11, in general, SLPA achieves the highest average Q_{ov}^{Ni}, followed by LFM and COPRA, even though the performance of SLPA has larger fluctuation than that in synthetic networks. Compared with COPRA, SLPA is

---

[3] `www-personal.umich.edu/~mejn/netdata/` and `snap.stanford.edu/data/`
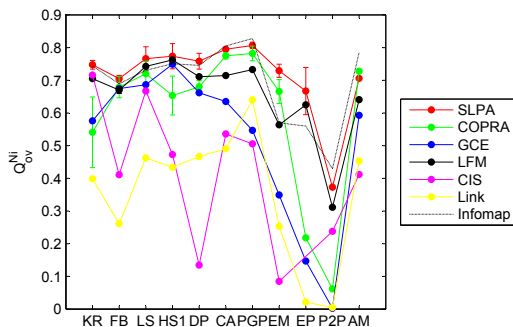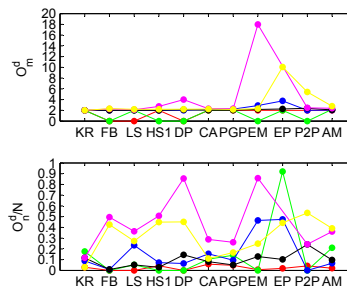
Fig. 11: Overlapping modularity Q_{ov}^{Ni}.

Fig. 12: The number of detected memberships (top) and the fraction of detected overlapping nodes (bottom).

more stable as evidenced by smaller deviation of its Q_{ov}^{Ni} score. In contrast, COPRA does not work well on highly sparse networks such as *P2P*, for which COPRA finds merely one single giant community. COPRA also fails on *Epinions* network because it claims too many overlapping nodes in view of consensus of other algorithms as seen in the bottom of Fig. 12. Such over-detection also applies to CIS and Link, resulting in low Q_{ov}^{Ni} scores for these two algorithms. The results in Fig. 12 (based on the clustering with the best Q_{ov}^{Ni}) show a common feature in the tested real-world networks, which is a relatively little agreement between results of different algorithms, i.e., the relatively small overlap in both the fraction of overlapping nodes (typically less than 30%) and the number of communities of which an overlapping node is a member (typically 2 or 3).

As known, a high modularity might not necessarily result in a *true* partitioning as it does in the disjoint community detection. We used the high school network (HS1) with known attributes to verify the output of SLPA. As shown in Fig. 13, there is a good agreement between the found and known partitions in term of student's *grades*. In SLPA, the grade 9 community is further divided into two subgroups. The larger group contains only white students, while the smaller group demonstrates *race* diversity. These two groups are connected partially via an overlapping node. It is also clear that overlapping nodes only exist on the boundaries of communities. A few overlapping nodes are assigned to three communities, while the others are assigned to two communities (i.e., their O_m is 2).

5.2 Identifying Overlapping Communities in Bipartite Networks

Discovering communities in bipartite networks is important because they provide a natural representation of many social networks. One example is the online tagging system with both users and tags. Unlike the original LPA algorithm, which performs poorly on bipartite networks, SLPA works well on this kind of networks.
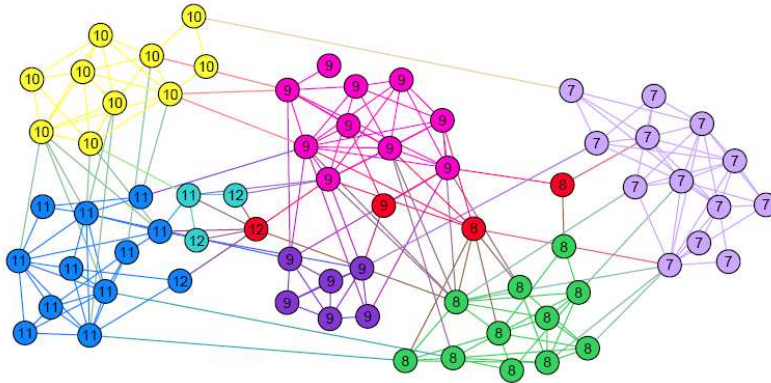
Fig. 13: High school network ($n = 69$, $\overline{k} = 6.4$). Labels are the known grades ranging from 7 to 12. Colors represent communities discovered by SLPA. The overlapping nodes are highlighted by red color.

We demonstrate this using two real-world networks[4]. One is a Facebook-like social network. One type of nodes represents users (abbr. FB-M1), while the other represents messages (abbr. FB-M2). The second network is the interlocking directorate. One type of nodes represents affiliations (abbr. IL-M1), while the other individuals (abbr. IL-M2).

We compared SLPA with COPRA in Table 4. One difference between SLPA and COPRA is that SLPA applies to the *entire* bipartite network directly, while COPRA is applied to each type of nodes *alternatively*. Q_{ov}^{Ni} is computed on the *projection* of each type of nodes. Again, we allow *overlapping* between communities. Although COPRA is slightly better (by 0.03) than SLPA on the second type of nodes for interlock network, it is much worse (by 0.11) on the first type. Moreover, COPRA fails to detect meaningful communities in the Facebook-like network, while SLPA demonstrates relatively good performance.

Table 4: The Q_{ov}^{Ni} of SLPA and COPRA for two bipartite networks.

Network	n	SLPA (std)	COPRA (std)
FB-M1	899	0.23 (0.10)	0.02 (0.07)
FB-M2	522	0.36 (0.02)	0.02 (0.07)
IL-M1	239	0.59 (0.02)	0.48 (0.02)
IL-M2	923	0.69 (0.01)	0.72 (0.01)

---

[4] Data are available at `http://toreopsahl.com/datasets/`

C1(68%)		C10(97%)		C11(86%)	C13(96%)		C16(91%)
C1-25(100%)	C1-40(83%)	C10-35(100%)	C10-36(71%)	C11-38(100%)	C13-45(100%)	C13-46(100%)	C16-32 / C16-32-39 / C16-32-39-49(75%)

Fig. 14: The nested structure in the high school network represented as a Treemap. The color represents the best explaining attribute: *blue* for *grade*; *green* for *race*; and *yellow* for *sex*. Numbers in parenthesis are the matching scores defined in the text. The size of shapes is proportional to the community size. Due to the page limit, only part of the entire treemap is shown.

5.3 Identifying Overlapping Nested Communities

In the above experiments, we applied a post-processing to remove subset communities from the raw output of stages 1 and 2 of SLPA. This may not be necessary for some applications. Here, we show that rich *nested* structure can be recovered in the high school network (HS2) with $n = 612$. The hierarchy is shown as a treemap[5] shown in Fig. 14. To evaluate the degree to which a discovered community matches the known attributes, we define a *matching score* as the *largest* fraction of matched nodes relative to the community size among three attributes (i.e., grade, race and sex). The corresponding attribute is said to best explain the community found by SLPA.

As shown, SLPA discovers a tree with a height of four. Most of the communities are distributed on the first two levels. The community name shows the full hierarchy path (connected by a dash '-') leading to this community. For example, *C1* has id 1 and is located on the first level, while *C1-25* has id 25, and it is the second level sub-community of community with id 1.

Nested structures are found across different attributes. For example, *C13* is best explained by *race*, while its two sub-communities perfectly account for *grade* and *sex* respectively. In *C1*, sub-communities explained by the same attribute account for *different* attribute *values*. For example, both *C1-25* and *C1-40* are identified by *sex*. However, the former contains only *male* students, while in the latter *female* students are the majority. Although the treemap is not capable of displaying overlaps between communities, the nested structures overlap as before.

6 Conclusions

We introduced a dynamic interaction process, SLPA as a basis for an efficient and effective *unified* overlapping community detection algorithm. SLPA allows us to analyze different kinds of community structures, such as disjoint communities, individual overlapping nodes, overlapping communities and overlapping nested hierarchy in both unipartite and bipartite topologies. Its underlying

---

[5] Treemap is used for visualization: `www.cs.umd.edu/hcil/treemap/`.

process can be easily modified to accommodate other types of networks (e.g., k-partite graphs). In the future work, we plan to apply SLPA to temporal community detection.

References

1. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. Nature 466, 761–764 (2010)
2. Baumes, J., Goldberg, M., Magdon-Ismail, M.: Efficient identification of overlapping communities. LNCS 3495, 27–36 (2005)
3. Collins, L.M., Dent, C.W.: Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. MBR 23, 231–242 (1988)
4. Farkas, I., Ábel, D., Palla, G., Vicsek, T.: Weighted network modules. New Journal of Physics 9(6), 180 (2007)
5. Gregory, S.: Finding overlapping communities in networks by label propagation. New J. Phys. 12, 103018 (2010)
6. Kelley, S.: The existence and discovery of voerlapping communities in large-scale networks. Ph.D. thesis, Rensselaer Polytechnic Institute, Troy, NY (2009)
7. Lancichinetti, A., Fortunato, S., Kertesz, J.: Detecting the overlapping and hierarchical community structure in complex networks. New J. Phys. p. 033015 (2009)
8. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev. E 78, 046110 (2008)
9. Lee, C., Reid, F., McDaid, A., Hurley, N.: Detecting highly overlapping community structure by greedy clique expansion. In: snakdd. pp. 33–42 (2010)
10. Nicosia, V., Mangioni, G., Carchiolo, V., Malgeri, M.: Extending the definition of modularity to directed graphs with overlapping communities. J. Stat. Mech. p. 03024 (2009)
11. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature pp. 814–818 (2005)
12. Psorakis, I., Roberts, S., Ebden, M., Sheldon, B.: Overlapping community detection using bayesian non-negative matrix factorization. Phys. Rev. E 83, 066114 (2011)
13. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E 76, 036106 (2007)
14. Shen, H., Cheng, X., Cai, K., Hu, M.B.: Detect overlapping and hierarchical community structure. Physica A 388, 1706 (2009)
15. Xie, J., Szymanski, B.K.: Community detection using a neighborhood strength driven label propagation algorithm. In: IEEE NSW 2011. pp. 188–195 (2011)
16. Zhang, S., Wangb, R.S., Zhang, X.S.: Identification of overlapping community structure in complex networks using fuzzy c-means clustering. Physica A 374, 483–490 (2007)