# Modeling and Detecting Community Hierarchies

Maria Florina Balcan and Yingyu Liang

School of Computer Science
Georgia Institute of Technology
Atlanta, GA 30332

**Abstract.** Community detection has in recent years emerged as an invaluable tool for describing and quantifying interactions in networks. In this paper we propose a theoretical model that explicitly formalizes both the tight connections within each community and the hierarchical nature of the communities. We further present an efficient algorithm that provably detects all the communities in our model. Experiments demonstrate that our definition successfully models real world communities, and our algorithm compares favorably with existing approaches.

**Keywords:** community detection, hierarchical structure.

## 1 Introduction

The structure of networks has been extensively studied over the past several years in many disciplines, ranging from mathematics and computer science to sociology and biology. A significant amount of recent work in this area has focused on the development of community detection algorithms. The community structure reflects how entities in a network form meaningful groups such that interactions within the groups are more active compared to those between the groups and the outside world. The discovery of these communities is useful for understanding the structure of the underlying network, or making decisions in the network [8,9,28,29].

Generally, a community should be thought of as a subset whose members have more interactions with each other than with the remainder of the network. This intuition is captured by some recently proposed models [2,3,1,12,15]. Additionally, recent studies show that networks often exhibit hierarchical organization, in which communities can contain groups of sub-communities, and so forth over multiple scales. For example, this can be observed in ecological niches in food webs, modules in biochemical networks or groups of common interest in social websites [31,19,7]. It is also shown empirically and theoretically that hierarchical structures can simultaneously explain and quantitatively reproduce many commonly observed topological properties of networks [6,32,10]. This suggests that the hierarchical structure should also be reflected when modeling real world communities.

Although some heuristic approaches [10,21] have been proposed to detect community hierarchies, few works have formalized this hierarchical property, and there are no theoretical performance guarantees for the algorithms. Inspired by the related work in clustering [4], in this paper we define a notion of communities that both reflects the tight connections within communities and explicitly models the hierarchy of communities. In our model, each member of a community falls into a sub-community, and the sub-communities within this community have active interactions with each other while entities outside this community have fewer interactions with members inside. Given this formalization, we then propose an efficient algorithm that detects all the communities in this model, and prove that all the communities form a hierarchy. Empirical evaluations demonstrate that our formalization successfully models real world communities, and our algorithm compares favorably with existing approaches.

In the remainder of the paper, we formalize our model in Section 2, and then describe and analyze our algorithm in Section 3. We then present the results of our experiments in Section 4, and conclude our paper in Section 5.

## 2 Hierarchical Community Model

A network is typically represented as a graph $G = (V, E)$ on a set of $n = |V|$ points[1], where the edges could be undirected or directed, unweighted or weighted. The graph implicitly specifies a neighborhood structure on the points, i.e. for each point there is a ranking of all other points according to the level of possible interaction. More precisely, we assume that we have a neighborhood function $N$ which given a point $p$ and a threshold $t$ outputs a list $N_t(p)$ containing the $t$ nearest neighbors of $p$ in $V$.

The neighborhood function can be used to formalize a model of hierarchical communities. Using this neighborhood function, the tight connections within communities can be naturally rephrased as follows: for suitable $t$, most points $p$ in the community have most of the nearest neighbors $N_t(p)$ from the community while points outside have just a few nearest neighbors from the community. Besides this, we also want to formalize the hierarchical structure that sub-communities in a lower, more local level actively interacting with each other form a community in a higher, more global level. The connections between the sub-communities can also be rephrased using the language of neighborhood: a majority of points in each sub-community have most of the nearest neighbors from the sub-communities in the same community.

In the remainder of the section, we specify our model based on the neighborhood function. We begin with the following notion of compact blobs, which will serve as a building block for our model.

**Definition 1.** *A subset $A$ of points is called an $\alpha$-compact blob, if out of the $|A|$ nearest neighbors:*

- *any point $p \in A$ has at most $\alpha n$ neighbors outside $A$, i.e. $|N_{|A|}(p) \setminus A| \leq \alpha n$;*
- *any point $q \notin A$ has at most $\alpha n$ neighbors inside $A$, i.e. $|N_{|A|}(q) \cap A| \leq \alpha n$.*

[1] We distinguish the nodes in the hierarchy our algorithm builds from the points in the graph.

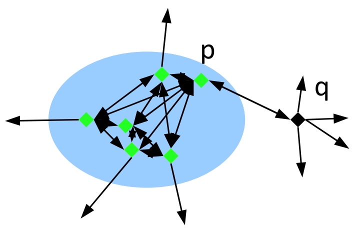

**Fig. 1.** Illustration of an $\alpha$-compact blob. An edge $(x, y)$ means that $y$ is one of $x$'s nearest neighbors.

Note that the notion of compact blobs is the same as the clusters that satisfy the $\alpha$-good neighborhood property defined in [4]. The notion captures the desired property of communities to be detected: members in the community have many more interactions with other members inside the community and have fewer interactions with those outside. However, in practice, the notion may seem somewhat restricted. First, it requires all the members in the community have most interactions with other members inside the community, which may not be the case in real life. For example, some members in the boundary may have more interactions with the outside world, i.e. they have more than $\alpha n$ neighbors from outside. Based on this consideration, we define the $(\alpha, \beta)$-stable property as follows.

**Definition 2.** *A community $C$ is $(\alpha, \beta)$-stable if*

- *any point $p \in C$ falls into a $\alpha$-compact blob $A_p \subseteq C$ of size greater than $6\alpha n$,*
- *for any point $p \in C$, at least $\beta$ fraction of points in $A_p$ have all but at most $\alpha n$ nearest neighbors from $C$ out of their $|C|$ nearest neighbors,*
- *any point $q$ outside $C$ has at most $\alpha n$ nearest neighbors from $C$ out of their $|C|$ nearest neighbors.*

Informally, the first condition means that every point falls into a sufficiently large compact blob in its community. This condition formalizes the local neighborhood structure that each member interacts actively with sufficiently many members in the community. Note that the compact blob should be large enough so that the membership of the point is clearly established, i.e. it should have size comparable to $\alpha n$, the number of connections to points outside. Here we choose a minimum size of $6\alpha n$ mainly because it guarantees that our algorithm can still identify the blob in the worst case. The second condition means that at least $\beta$ fraction of points in these compact blobs have most of their nearest neighbors from the community. This condition formalizes more global neighborhood structure about how the compact blobs interact with each other to form a community. The third condition formalizes how the community is separated from the outside.

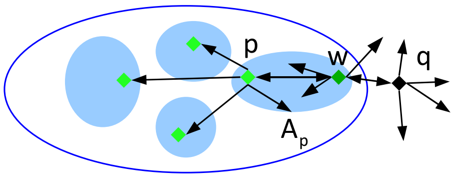

**Fig. 2.** Illustration of an $(\alpha, \beta)$-stable community. An edge $(x, y)$ means that $y$ is one of $x$'s nearest neighbors. Note that point $w$ lies on the "boundary" of the community. It falls into the compact blob $A_p$, but does not have most of its nearest neighbors from the community.

Note that we no longer require all the members in the community have most interactions inside; we only require each member interacts with sufficiently many members and a majority of members in these local groups interact actively. Also note that the definition is hierarchical in nature: sufficiently large compact blobs clearly satisfy the definition of $(\alpha, \beta)$-stable property and thus can be viewed as communities in lower levels. Furthermore, in the next section we will show that all the $(\alpha, \beta)$-stable communities form a hierarchy. We show this by presenting an algorithm and proving that each $(\alpha, \beta)$-stable community is a node in the hierarchy output by the algorithm. So our formulation explicitly models the hierarchical structure of communities observed in networks.

Next we propose a further generalization that considers possible noise in real world data. There may be some abnormal points that do not exhibit clear membership to any community, in the presence of which our definition above does not model the communities well. For example, suppose there is a point that has connections to all other points in the network, then no non-trivial subsets satisfy our definition above. We call such points bad since they do not fit into our community model above. To deal with the noise, we can naturally relax the $(\alpha, \beta)$-stable property to the $(\alpha, \beta, \nu)$-stable property defined as follows. Informally, it requires that the target community satisfies the $(\alpha, \beta)$-stable property after removing a few bad points $B$. For convenience, we call the other points in $S \setminus B$ good points.

**Definition 3.** *A community $C$ is $(\alpha, \beta, \nu)$-stable if there exist a subset of bad points $B$ of size at most $\nu n$, such that*

- *any good point $p \in G = C \setminus B$ falls into a compact blob $A_p \subseteq C$ of size greater than $6(\alpha + \nu)n$,*
- *for any point $p \in G$, at least $\beta$ fraction of points in $A_p$ have all but at most $\alpha n$ nearest neighbors from $G$ out of their $|G|$ nearest neighbors in $S \setminus B$,*
- *any good point $q$ outside $C \cup B$ has at most $\alpha n$ nearest neighbors from $G$ out of their $|G|$ nearest neighbors in $S \setminus B$.*

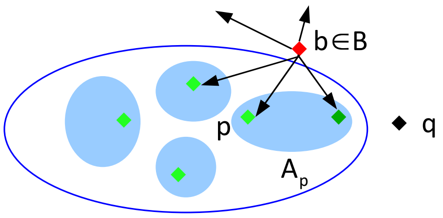

**Fig. 3.** Illustration of an $(\alpha, \beta, \nu)$-stable community. An edge $(x, y)$ means that $y$ is one of $x$'s nearest neighbors. Point $b$ is a bad point and does not exhibit clear membership to any community.

**Note 1.** The parameters $\alpha, \nu$ are defined globally, i.e. they are defined as ratios with respect to the total number of points. So a local change to some community can affect the values of these parameters for the other communities. For example, suppose we add $Kn$ new points to some community, with all the new points having neighbors only inside this special community. Since the number of points increases to $(K+1)n$, the communities outside the modified community are now $(\alpha/(1+K), \beta, \nu/(1+K))$-stable. However, the local change does not affect the identifiability of these communities. Our algorithm described in the next section can still detect these communities, given the value of $(\alpha+\nu)n$.

**Note 2.** The input of the community detection task is usually a graph representing the network, and there are different ways to lift the graph to a neighborhood function. The simplest one is to directly sort for each point $p$ all the other points $q$ according to the weights of the edges $(p, q)$ and break ties randomly (we assume without loss of generality that the weights are in $[0, 1]$ and the weight of an edge not in $E$ is regarded as 0). However, as pointed out in [3], we also have alternative approaches to convert the observed graph into a neighborhood function. More specifically, we assume the observed graph reflects some underlying unobserved set of relations, and thus we can lift the graph to an affinity system based on various beliefs about the connection between the latent relations and the observed graph, and then sort the points according to the affinity system to get the neighborhood function. For example, based on the belief that random walks on the graph can reflect the similarities between entities, we can define the affinity to be the diffusion kernel $\exp\{\lambda A\}$ where $A$ is the adjacent matrix and $\lambda$ is a parameter. Note that the results of appropriate lifting procedures can better reflect the true relationships between entities, and thus the conversion can address the challenging issue of sparsity in the observed graph.

## 3 Hierarchical Community Detection Algorithm

In the section, we propose an algorithm for detecting communities satisfying the $(\alpha, \beta, \nu)$-stable property. The goal of our algorithm is to output a set of

**Algorithm 1.** Hierarchical Community Detection Algorithm

```
Input: neighborhood function $N$ on a set of points $V$, $n = |V|$, $\alpha > 0, \nu > 0$.

Step 1  Initialize $\mathcal{C}'$ to be a set of singleton points, and $t = 6(\alpha+\nu)n + 1$.
        while $|\mathcal{C}'| > 1$ do
Step 2    Build $F_t$ on $V$ as follows.
          for any $x, y \in V$ that satisfy $|N_t(x) \cap N_t(y)| \geq t - 2(\alpha+\nu)n$ do
            Connect $x, y$ in $F_t$.
Step 3    Build $H_t$ on $\mathcal{C}'$ as follows. Let $N_F(x)$ denote the neighbors of $x$ in $F_t$.
          for any $U, W \in \mathcal{C}'$ do
            if $U, W$ are singleton subsets, i.e. $U = \{x\}, W = \{y\}$ then
              Connect $U, W$ in $H_t$, if $|N_F(x) \cap N_F(y)| > (\alpha+\nu)n$.
            else
              Set $S_t(x,y) = |N_F(x) \cap N_F(y) \cap (U \cup W)|$, $\forall x \in U, y \in W$.
              Connect $U, W$ in $H_t$, if $\mathrm{median}_{x\in U, y\in W} S_t(x,y) > \frac{|U|+|W|}{4}$.
Step 4    for any component $R$ in $H_t$ that satisfies $|\bigcup_{C\in R} C| \geq 4(\alpha+\nu)n$ do
            Update $\mathcal{C}'$ by merging subsets in $R$ into one subset.
Step 5    $t = t + 1$.
        end while

Output: Hierarchy $T$ with single points as leaves and internal nodes corresponding
to the merges performed.
```

communities such that each community satisfying the $(\alpha, \beta, \nu)$-stable property is close to one in the output. To be precise, we say that a community $C$ is $\nu$-close to another community $C'$ if $|C \setminus C'| + |C' \setminus C| \leq \nu n$. We first describe the details in Algorithm 1, and then present the analysis in Theorem 1.

Now we prove that the algorithm successfully outputs a hierarchy such that any community satisfying the $(\alpha, \beta, \nu)$-stable property with sufficiently large $\beta$ is close to one of the nodes in the hierarchy. Formally,

**Theorem 1.** *Algorithm 1 outputs a hierarchy such that any community satisfying the $(\alpha, \beta, \nu)$-stable property with $\beta \geq 5/6$ is $\nu$-close to a node in the hierarchy. The algorithm runs in time $O(n^{\omega+1})$, where $O(n^{\omega})$ is the state of the art for matrix multiplication.*

The correctness of the theorem follows from Lemma 3 and the running time follows from Lemma 4. In the following analysis, we always assume $\beta \geq 5/6$. Before presenting the analysis for the general communities in Lemma 3, we first prove a lemma for the base case of compact blobs, showing that for any compact blob, a node close to it will be formed.

**Lemma 1.** *For any good point $p$, when $t \leq |A_p|$, good points from $A_p$ will not be merged with good points outside $A_p$. At the end of the threshold $t = |A_p|$, all points in $A_p$ have been merged into a subset.*

*Proof.* We prove this by induction on $t$. The claim is clearly true initially. Now assume for induction that at the beginning of a threshold $t \leq |A_p|$, in $\mathcal{C}'$ good points from $A_p$ are not merged with good points outside $A_p$, i.e. any subset can contain good points from only one of $A_p$ and $V \setminus B \setminus A_p$. We now analyze the properties of the graphs $F_t$ and $H_t$, and show that at the end of the current threshold, the claim is still true.

First, as long as $t \leq |A_p|$, the graph $F_t$ has the following properties.

– No good point $x$ in $A_p$ is connected to a good point $y$ outside $A_p$. By the definition of compact blobs, out of the $t$ nearest neighbors, $x$ has at most $(\alpha + \nu)n$ neighbors outside $A_p$. For $y \in V \setminus B \setminus A_p$, $y$ has at most $(\alpha + \nu)n$ neighbors in $A_p$. Then $x, y$ have at most $2(\alpha + \nu)n < t - 2(\alpha + \nu)n$ common neighbors, so they are not connected.
– No bad point $z$ is connected to both a good point $x$ in $A_p$ and a good point $y$ outside $A_p$. We know that out of the $t$ nearest neighbors, $x$ has at most $(\alpha + \nu)n$ neighbors outside $A_p$. So if $z$ is connected to $x$, then $z$ must have more than $t - 3(\alpha + \nu)n$ neighbors in $A_p$ and less than $3(\alpha + \nu)n$ neighbors outside $A_p$. Since $y$ has at most $(\alpha + \nu)n$ neighbors in $A_p$, we have that $y, z$ share less than $3(\alpha + \nu)n + (\alpha + \nu)n < t - 2(\alpha + \nu)n$ neighbors, so they are not connected.

Based on the properties of $F_t$ and the inductive assumption that any subset can contain good points from only one of $A_p$ and $V \setminus B \setminus A_p$, we show that the graph $H_t$ has the following properties.

– No subset $U$ containing good points from $A_p$ is connected to a subset $W$ containing good points outside $A_p$. This is clearly true if they are singleton subsets. In the other cases, note that the fraction of bad points in $U$ or $W$ is at most $1/4$. Then the number of pairs $(x, y)$ with good points $x \in U$ and $y \in W$ is at least $\frac{3}{4}|U| \times \frac{3}{4}|W| > |U||W|/2$, i.e. more than half of the pairs $(x, y)$ with $x \in U$ and $y \in W$ are pairs of good points. This means there exist good points $x^* \in U, y^* \in W$ such that $S_t(x^*, y^*)$ is no less than $\text{median}_{x \in U, y \in W} S_t(x, y)$. By the properties of $F_t$, $x^*, y^*$ have no common neighbors. Therefore, $U$ and $W$ are not connected.
– If a subset $W$ contains only bad points, then it cannot be connected to both a subset containing good points from $A_p$ and a subset containing good points outside $A_p$. Suppose it is connected to $U$ which contains good points from $A_p$. Note that since $W$ contains only bad points, it must contain only a single point $z$. If $U = \{x\}$ is singleton, then $x, z$ share more than $(\alpha + \nu)n$ neighbors in $F_t$. Since in $F_t$, $x$ is only connected to good points from $A_p$ and bad points, $z$ and $x$ must share some common neighbors from $A_p$, then $z$ must be connected to some good points in $A_p$. In the other cases, note that the fraction of bad points in $U$ is at most $1/4$. So there exists a good point $x^* \in U$ such that $S_t(x^*, z) \geq \text{median}_{x \in U} S_t(x, z)$. Then we have $S_t(x^*, z) > (|U| + |W|)/4 > \nu n$, and thus $z$ must also be connected to some good points in $A_p$. Similarly, if $W$ is connected to a subset containing good points outside $A_p$, then the point in $W$ must connect to some good point outside $A_p$. But this is contradictory to the fact that in $F_t$ no bad point is connected to both a good point in $A_p$ and a good point outside $A_p$.

By the properties of $H_t$, no connected component contains both good points in $A_p$ and good points outside $A_p$. So at the end of this threshold $t$, the claim is still true. Then by induction, we know that when $t \leq |A_p|$, we will not merge good points from $A_p$ with good points outside $A_p$.

Next we show that at the end of the threshold $t = |A_p|$, we will merge all points in $A_p$ into a subset. First, at this threshold, all good points in $A_p$ are connected in $F_t$. Any good point in $A_p$ has at most $(\alpha + \nu)n$ neighbors outside $A_p$, so when $t = |A_p|$, any two good points $x, y$ in $A_p$ are connected, and thus they share at least $|A_p|$ common neighbors in $F_t$. Second, all subsets containing good points in $A_p$ are connected in $H_t$. If no good points in $A_p$ have been merged, then these singleton points will be connected in $H_t$ since they share at least $|A_p|$ singleton subsets as common neighbors in $F_t$. If some good points in $A_p$ have already been merged into non-singleton subsets, we can show that in $H_t$ these non-singleton subsets will be connected to each other and connected to singleton subsets containing good points from $A_p$. For any such pair of subsets $U$ and $W$, the fraction of bad points in $U$ or $W$ is at most $1/4$, so there exist good points $x^* \in U, y^* \in W$ such that $\text{median}_{x \in U, y \in W} S_t(x, y)$ is no less than $S_t(x^*, y^*)$. Since $x^*, y^*$ are connected to all good points in $A_p$ in $F_t$, $S_t(x^*, y^*)$ is no less than the number of good points in $U$ and $W$. So $\text{median}_{x \in U, y \in W} S_t(x, y) \geq S_t(x^*, y^*) > (|U| + |W|)/4$, and thus $U, W$ are connected in $H_t$. Therefore, all points in $A_p$ are merged into a subset. □

The following is a consequence of Lemma 1, which will be used in the analysis for the general communities in Lemma 3.

**Lemma 2.** *In Algorithm 1, if a subset $U$ satisfies that for any good point $p \in U$, $A_p \subseteq U$, then there exist a subset of good points $P \subseteq U$, such that $\{A_p : p \in P\}$ is a partition of $U \setminus B$.*

*Proof.* We have $U \setminus B = \cup_{p \in U \setminus B} A_p$. We only need to show that sets in $\{A_p : p \in U \setminus B\}$ are laminar, i.e. for any $p, q \in U \setminus B$, either $A_p \cap A_q = \emptyset$ or $A_p \subseteq A_q$ or $A_q \subseteq A_p$. Assume for contradiction that there exist $A_p$ and $A_q$ such that $A_p \setminus A_q \neq \emptyset, A_q \setminus A_p \neq \emptyset$ and $A_p \cap A_q \neq \emptyset$. Without loss of generality, suppose $|A_p| \leq |A_q|$. Then by Lemma 1, at the end of the threshold $t = |A_p|$, we have merged all good points in $A_p$ into a subset. Specifically, this means that we have merged $A_p \cap A_q$ with $A_p \setminus A_q$. So for $t \leq |A_q|$, we have merged good points in $A_q$ with good points outside $A_q$, which is contradictory to Lemma 1. □

By the above lemmas, for any good point $p$, the subset $A_p$ will be formed before points in it are merged with good points outside. Once these subsets are formed, we can show that subsets in the same target community will be merged together before they are merged with those from other communities, and thus the hierarchy produced has a node close to the target community. Formally, we have the following result.

**Lemma 3.** *For any community $C$ satisfying the $(\alpha, \beta, \nu)$-stable property with $\beta \geq 5/6$, $\mathcal{C}' \setminus B$ in Algorithm 1 is always laminar to $C \setminus B$, i.e. for any $C' \in \mathcal{C}'$,* *either* $(C' \setminus B) \cap (C \setminus B) = \emptyset$ *or* $(C' \setminus B) \subseteq (C \setminus B)$ *or* $(C \setminus B) \subseteq (C' \setminus B)$. *Furthermore, there is a node* $U$ *in the hierarchy produced such that* $U \setminus B = C \setminus B$.

*Proof.* we will show by induction on $t$ that: for any community $C$ satisfying the $(\alpha, \beta, \nu)$-stable property with $\beta \geq 5/6$,

– at the end of threshold $t$, $\mathcal{C}' \setminus B$ is laminar to $C \setminus B$,
– at the end of threshold $t$, for any $C$ such that $|C \setminus B| \leq t$, we have merged all points in $C \setminus B$ into a subset.

These claims are clearly true initially. Assume for induction that they are true for the threshold $t - 1$, we now show that they are also true for the threshold $t$.

We first show that the laminarity is preserved. The laminarity is broken only when we connect in $H_t$ two subsets $U, W$ such that $U$ is a strict subset of $C$ after removing the bad points, and $W$ is a subset containing good points from outside. If there is a good point $p \in U$ such that $A_p \not\subseteq U$, then by Lemma 1, they cannot be connected. So we only need to consider the other case when for any good point $p \in U, A_p \subseteq U$. For convenience, we call a point great if it is a good point in $C$, and it has less than $\alpha n$ neighbors outside $C \setminus B$ out of the $|C \setminus B|$ nearest neighbors in $V \setminus B$. We now show that $U, W$ are not connected in $H_t$. Since $U \setminus B$ is a strict subset of $C \setminus B$, by induction on the second claim, we have $t \leq |C \setminus B|$. Then great points in $U$ and points in $W$ share at most $2(\alpha + \nu)n < t - 2(\alpha + \nu)n$ common neighbors, so they are not connected in $F_t$. By Lemma 2 and the second condition of the $(\alpha, \beta, \nu)$-stable property, we know that at least 5/6 fraction of points in $U \setminus B$ are great points. Then there exist a great point $x^* \in U$ and a point $y^* \in W$ such that $S_t(x^*, y^*)$ is no less than $\text{median}_{x \in U, y \in W} S_t(x, y)$. Since in $F_t$ great points in $U$ are not connected to points in $W$, we have $S_t(x^*, y^*) \leq (|U| + |W|)/4$. So $\text{median}_{x \in U, y \in W} S_t(x, y) \leq (|U| + |W|)/4$ and $U, W$ are not connected in $H_t$. Therefore, the laminarity is preserved.

Next we show that at the end of the threshold $t = |C \setminus B|$, all points in $C \setminus B$ are merged into a subset. By Lemma 1, all good points in $C \setminus B$ are now in sufficiently large subsets. We claim that any two of these subsets $U, W$ are connected in $H_t$, and thus will be merged. Again by Lemma 2, we know at least 5/6 fraction of points in $U \setminus B$ or $W \setminus B$ are great points, and thus there exist great points $x^* \in U, y^* \in W$ such that $S_t(x^*, y^*)$ is no more than $\text{median}_{x \in U, y \in W} S_t(x, y)$. Notice that all great points in $U$ are connected to great points in $W$ in $F_t$, since they share at least $t - 2(\alpha + \nu)n$ neighbors. Then $S_t(x^*, y^*) \geq 3(|U| + |W|)/4 > (|U| + |W|)/4$, and thus $\text{median}_{x \in U, y \in W} S_t(x, y) > (|U| + |W|)/4$. Therefore, any two subsets containing good points from $C \setminus B$ are connected in $H_t$ and thus are merged.

So the two claims hold for all $t$, specially for $t = n$. Then the algorithm must stop after this threshold, and we have the lemma as desired. □

**Lemma 4.** *Algorithm 1 has a running time of* $O(n^{\omega+1})$.

*Proof.* To implement the algorithm, we introduce some data structures. For any $x \in V$, if $y$ is within the $t$ nearest neighbors of $x$, let $I_t(x, y) = 1$, otherwise

$I_t(x, y) = 0$. Initializing $I_t$ takes $O(n^2)$ time. Next we compute $CN_t(x, y)$, the number of common neighbors between $x$ and $y$. Notice that $CN_t(x, y) = \sum_{z \in V} I_t(x, z) I_t(y, z)$, so $CN_t = I_t I_t^T$. Then we can compute the adjacent matrix $F_t$ (overloading notation for the graph $F_t$) from $CN_t$. These take $O(n^\omega)$ time.
To compute the graph $H_t$, we introduce the following data structures. Let $FS_t(x, y) = 1$ if $x, y$ are singleton subsets and $F_t(x, y) = 1$, and let $FS_t(x, y) = 0$ otherwise. Let $NS_t = FS_t(FS_t)^T$, then for two singleton subsets $x, y$, $NS_t(x, y)$ is the number of singleton subsets they share as neighbors in common in $F_t$. Similarly, let $FC_t(x, y) = 1$ if $x$ and $y$ are in the same subset and $F_t(x, y) = 1$, and let $FC_t(x, y) = 0$ otherwise. Let $S_t(x, y) = NS_t(FC_t)^T + FC_t(NS_t)^T$, then for two points $x \in U, y \in W$ where $U, W$ are two non-singleton subsets, $S_t(x, y)$ is the number of points in $U \cup W$ they share as neighbors in common in $F_t$. Based on $NS_t$ and $S_t$ we can build the graph $H_t$. All these take $O(n^\omega)$ time.

When we perform merge or increase the threshold, we need to update the data structures, which takes $O(n^\omega)$ time. Since there are $O(n)$ merges and $O(n)$ thresholds, Algorithm 1 takes time $O(n^{\omega+1})$ in total. □

## 4 Experiments

In this section, we present our experimental results on evaluating our model and algorithm. While our main concern is building theoretical model for communities, empirical study is valuable in verifying the model and providing guidance for further improvement. Therefore, we applied our algorithm on both real world and synthetic data sets.

Note that the networks are represented as graphs, and we need to lift the graphs to get neighborhood functions for our algorithm. We use two lifting approaches for our experiments. The first approach is direct lifting: first, for any $x, y$ set the affinity between $x$ and $y$ to be 1 if $(x, y) \in E$ and 0 otherwise; then for each $x$, sort all the other points according to the affinities; break ties randomly to avoid bias. The second approach is diffusion lifting: first set the affinity matrix $K$ between entities to be $K = \exp\{\lambda A\}$ where $\lambda = 0.05$ and $A$ is the adjacent matrix of the graph; then for each $x$, sort all the other points according to the affinities.

For comparison, we implemented two other algorithms: the lazy random walk algorithm (LRW [34]) and the Girvan-Newman algorithm (GN [10]). The lazy random walk algorithm performs truncated random walk from a seed point in the network and outputs selected communities where the selection is guided by the walk distribution and conductance. The conductance has been widely used as a criterion for quantifying the tight connections within communities, and thus the comparison to the lazy random walk algorithm provides an evaluation on how well our model and algorithm capture this intuition. The GN algorithm repeatedly removes the edge with the maximum edge-betweenness and regards the created connected components as communities. Although no theoretical model of hierarchical communities is targeted, the algorithm builds a hierarchy during its execution. It has been shown that the algorithm performs remarkably well on modeling communities in real-world data sets [10,29]. We use the code from [5] for fast computation of edge betweenness in the algorithm.

For algorithms with parameters, we run them multiple times with different values of parameters, and report the best result. More specifically, we run our algorithm using parameters $(\alpha + \nu) = \frac{i}{5n} (i = 1, 2, \ldots, 5)$. For the lazy random walk algorithm, we enumerate the parameters $\theta_0 = 0.05i (i = 1, \ldots, 4)$ and $b = 1, 2, \ldots, \lceil \log m \rceil$. In each run, 100 seed points are generated uniformly at random, each of which leads to a community. Since not all communities are meaningful (e.g. a singleton subset or the entire set of points), communities containing less than 10 points or containing more than $n - 10$ points are removed, and the rest communities are regarded as the output communities. We then evaluate the average error of the output communities. The error for a ground-truth community $C$ with respect to a set $\mathcal{C}$ of output communities is defined as

$$\text{error}(C, \mathcal{C}) = \min_{C' \in \mathcal{C}} \frac{|C \setminus C'| + |C' \setminus C|}{n}.$$

This criterion measures how well the ground-truth communities are recovered by the algorithm. We further note that our algorithm outputs fewer communities than the other algorithms in all the conducted experiments, and thus has advantage when they achieve similar performance.

## 4.1 Evaluation on Real-World Networks

To assess the performance of the proposed method in terms of accuracy, we conduct experiments on the following real world data sets[2] : karate [36], dolphins [23], polbooks [18], and football [10].

Figure 4 shows the average error and running time of the algorithms. We observe that our algorithm with diffusion lifting achieves the best performance on 3 out of 4 data sets, and achieves performance comparable to the GN algorithm on the football data set. It recovers the ground truth communities remarkably well over all the data sets. Our algorithm with direct lifting does not achieve good results. Note that this is due to the fact that diffusion lifting reflects the true neighborhood structure more accurately than direct lifting. More precisely, when we sort neighbors for a point $p$ in direct lifting, all points not adjacent to $p$ are ranked randomly. In fact some of them can be reached by a few steps and thus should be ranked as close neighbors, while others are actually far away from the point $p$. On the other hand, diffusion lifting leads to a neighborhood function that more accurately reflects the neighborhood information. The LRW algorithm has the worst performance, though it is the fastest. Our algorithm, especially with the diffusion lifting, runs 10-100 faster than the GN algorithm. Therefore, our algorithm with suitable neighborhood functions is the most favorable for detecting real world communities.

[2] Detailed descriptions and links for download can be found on `http://www-personal.umich.edu/~mejn/netdata/`

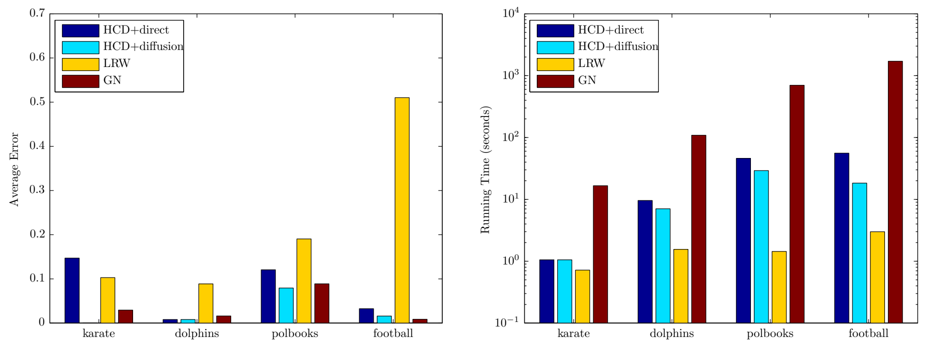

**Fig. 4.** The average error and running time using our hierarchical community detection algorithm with direct lifting (HCD+direct) or diffusion lifting neighborhood function (HCD+diffusion), Lazy Random Walk (LRW [34]) and the Girvan-Newman algorithm (GN [10]). Note that the running time is in log scale.

### 4.2 Evaluation on Synthetic Networks

**Table 1.** The parameters of the synthetic data sets for performance evaluation. $n/m$: number of nodes/edges; $k/maxk$: average/maximum degree of the nodes; $minc/maxc$: minimum/maximum size of the lower level communities; $minC/maxC$: minimum/maximum size of the higher level communities.

| Data set | $n$ | $m$ | $k$ | $maxk$ | $minc$ | $maxc$ | $minC$ | $maxC$ |
|---|---|---|---|---|---|---|---|---|
| LF50 | 50 | ≈500 | 10 | 15 | 10 | 15 | 20 | 30 |
| LF100 | 100 | ≈1500 | 15 | 20 | 15 | 20 | 30 | 40 |
| LF150 | 150 | ≈3000 | 20 | 30 | 20 | 30 | 40 | 60 |
| LF200 | 200 | ≈6000 | 30 | 40 | 30 | 40 | 60 | 80 |

Besides real-world networks, we further use the Lancichinetti-Fortunato (LF) benchmark[3] graphs [20] to evaluate the performance of the algorithms. By varying the parameters of the networks, we can analyze the behavior of the algorithms in detail. We generate four unweighted undirected benchmark networks with two level community hierarchies. The numbers of nodes are $50, 100, 150$ and $200$ respectively, and some important parameters of the networks are given in Table 1. For each type of dataset, we range the mixing parameter $\mu$ from 0.1 to 0.5 with a span of 0.1, and set the low-level mixing parameter $\mu_1 = \mu/4$ and the high-level mixing parameter $\mu_2 = \mu - \mu_1$, resulting in five networks. Generally, the higher the mixing parameter of a network is, the more difficult it is to reveal the community structure.

[3] The source code we use and details about the parameters can be found on `https://sites.google.com/site/andrealancichinetti/software`

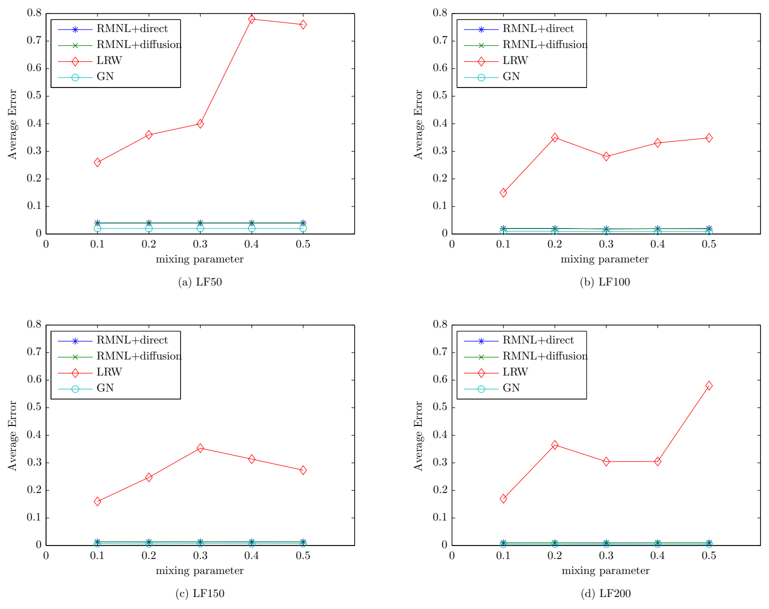

**Fig. 5.** The average error on the synthetic data sets

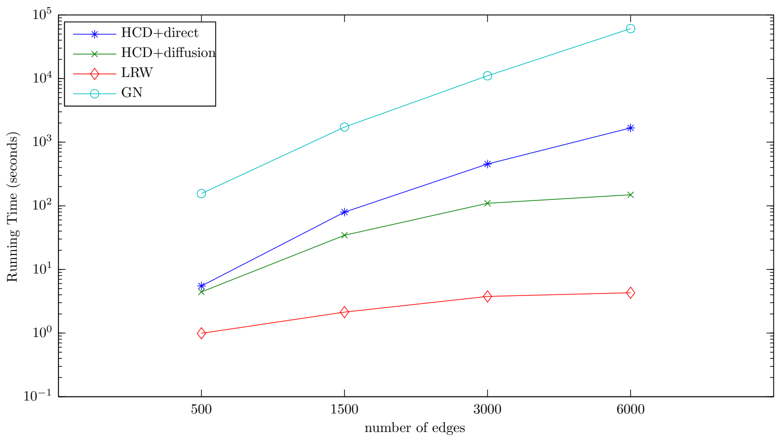

**Fig. 6.** The running time on the synthetic data sets

Figure 5 shows the average errors of the algorithms and Figure 6 shows the running time. Our algorithm with direct or diffusion lifting and the GN algorithm achieve similar results on all the benchmark networks. The errors of these algorithms are below 5%, and hardly increase with the mixing parameter. This suggests that they recover the ground truth communities remarkably well even in the hard case when the members of the communities have significant connections with the outside. In contrast, the LRW algorithm does not recover the communities well, even though it runs much faster than the other algorithms. Our algorithm runs about 50 times faster than the GN algorithm over all the data sets. These results are consistent with those observed on real world data sets, and again demonstrate the advantage of our algorithm.

## 5 Conclusion

In this paper we propose a model of communities that both reflects the tight connections within communities and explicitly models the hierarchy of communities. We present an efficient algorithm that provably detects all the communities in this model. Experiments demonstrate that our definition successfully models communities arising in the real world, and our algorithm compares favorably with existing approaches.

For future work, we plan to perform systematic empirical study of our model and algorithm using more neighborhood functions and on more real-world data sets. Another direction would be to speed up the computation of the neighborhood function and the algorithm and adapt them to large-scale scenarios.

**Acknowledgements.** This work was supported in part by NSF grant CCF-0953192, AFOSR grant FA9550-09-1-0538, and a Microsoft Research Faculty Fellowship.

## References

1. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. J. Mach. Learn. Res. 9, 1981–2014 (2008)
2. Arora, S., Ge, R., Sachdeva, S., Schoenebeck, G.: Finding overlapping communities in social networks: toward a rigorous approach. In: Proceedings of the 13th ACM Conference on Electronic Commerce (2012)
3. Balcan, M.F., Borgs, C., Braverman, M., Chayes, J., Teng, S.H.: Finding endogenously formed communities. In: SODA 2013 (2013)
4. Balcan, M.F., Gupta, P.: Robust hierarchical clustering. In: Proceedings of the Conference on Learning Theory, COLT (2010)
5. Bounova, G., de Weck, O.L.: Overview of metrics and their correlation patterns for multiple-metric topology analysis on heterogeneous graph ensembles. Phys. Rev. E 85(016117) (2012)
6. Clauset, A., Moore, C., Newman, M.E.J.: Hierarchical structure and the prediction of missing links in networks. Nature 453(7191), 98–101 (2008)
7. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E 70(6), 066111+ (2004)
8. Fortunato, S.: Community detection in graphs. Physics Reports 486(3-5), 75–174 (2010)
9. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99(12), 7821–7826 (2002)

10. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99(12) (2002)
11. Gleiser, P., Danon, L.: Community Structure in Jazz. Advances in Complex Systems 6(4), 565–573 (2003)
12. He, J., Hopcroft, J., Liang, H., Suwajanakorn, S., Wang, L.: Detecting the structure of social networks using ($\alpha$, $\beta$)-communities. In: Frieze, A., Horn, P., Prałat, P. (eds.) WAW 2011. LNCS, vol. 6732, pp. 26–37. Springer, Heidelberg (2011)
13. Ho, Q., Parikh, A.P., Xing, E.P.: A Multiscale Community Blockmodel for Network Exploration. Journal of the American Statistical Association 107(499), 916–934 (2012)
14. Huang, J., Sun, H., Han, J., Deng, H., Sun, Y., Liu, Y.: Shrink: a structural clustering algorithm for detecting hierarchical communities in networks. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, pp. 219–228. ACM (2010)
15. Kim, M., Leskovec, J.: Latent multi-group membership graph model. In: ICML (2012)
16. Kleinberg, J.: Complex networks and decentralized search algorithms. In: Proceedings of the International Congress of Mathematicians, ICM (2006)
17. Knuth, D.E.: The Stanford GraphBase: a platform for combinatorial computing. ACM, New York (1993)
18. Krebs, V.: `http://www.orgnet.com/` (unpublished)
19. Cosentino Lagomarsino, M., Jona, P., Bassetti, B., Isambert, H.: Hierarchy and feedback in the evolution of the Escherichia coli transcription network. Proceedings of the National Academy of Sciences 104(13), 5516–5520 (2007)
20. Lancichinetti, A., Fortunato, S.: Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Physical Review E 80(1), 016118 (2009)
21. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics 11(3), 033015 (2009)
22. Leskovec, J., Lang, K.J., Mahoney, M.: Empirical comparison of algorithms for network community detection. In: Proceedings of the 19th International Conference on World Wide Web, WWW 2010, pp. 631–640. ACM, New York (2010)
23. Lusseau, D., Schneider, K., Boisseau, O.J., Haase, P., Slooten, E., Dawson, S.M.: Behavioral Ecology and Sociobiology 54, 396–405 (2003)
24. McAuley, J., Leskovec, J.: Learning to discover social circles in ego networks. In: Advances in Neural Information Processing Systems 25, pp. 548–556 (2012)
25. Mishra, N., Schreiber, R., Stanton, I., Tarjan, R.E.: Clustering social networks. In: Bonato, A., Chung, F.R.K. (eds.) WAW 2007. LNCS, vol. 4863, pp. 56–67. Springer, Heidelberg (2007)
26. Mishra, N., Schreiber, R., Stanton, I., Tarjan, R.E.: Finding strongly knit clusters in social networks. Internet Mathematics 5(1), 155–174 (2008)
27. Narasimhan, G., Smid, M.: Geometric spanning networks. Cambridge University Press (2007)
28. Newman, M.E.J.: Detecting community structure in networks. The European Physical Journal B - Condensed Matter and Complex Systems 38(2), 321–330 (2004)
29. Newman, M.E.J.: Modularity and community structure in networks. Proceedings of the National Academy of Sciences 103(23), 8577–8582 (2006)
30. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proceedings of the National Academy of Sciences of the United States of America 101(9), 2658–2663 (2004)

31. Ravasz, E., Somera, A.L., Mongru, D.A., Oltvai, Z.N., Barabási, A.L.: Hierarchical Organization of Modularity in Metabolic Networks. Science 297(5586), 1551–1555 (2002)
32. Schweinberger, M., Snijders, T.A.B.: Settings in social networks: A measurement model. Sociological Methodology 33, 307–341 (2003)
33. Shen, K., Song, L., Yang, X., Zhang, W.: A hierarchical diffusion algorithm for community detection in social networks. In: 2010 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), pp. 276–283. IEEE (2010)
34. Spielman, D.A., Teng, S.-H.: Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In: Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, STOC 2004, pp. 81–90. ACM, New York (2004)
35. Yang, B., Di, J., Liu, J., Liu, D.: Hierarchical community detection with applications to real-world network analysis. Data & Knowledge Engineering (2012)
36. Zachary, W.W.: An information flow model for conflict and fission in small groups. Journal of Anthropological Research 33, 452–473 (1977)