Finding Overlapping Communities in Social Networks: Toward a Rigorous Approach

Sanjeev Arora* Rong Ge† Sushant Sachdeva‡ Grant Schoenebeck§

November 15, 2011

Abstract

A *community* in a social network is usually understood to be a group of nodes more densely connected with each other than with the rest of the network. This is an important concept in most domains where networks arise: social, technological, biological, etc. For many years algorithms for finding communites implicitly assumed communities are nonoverlapping (leading to use of clustering-based approaches) but there is increasing interest in finding overlapping communities. A barrier to finding communities is that the solution concept is often defined in terms of an NP-complete problem such as Clique or Hierarchical Clustering.

This paper seeks to initiate a rigorous approach to the problem of finding overlapping communities, where "rigorous" means that we clearly state the following: (a) the object sought by our algorithm (b) the assumptions about the underlying network (c) the (worst-case) running time.

Our assumptions about the network lie between worst-case and average-case. An average-case analysis would require a precise probabilistic model of the network, on which there is currently no consensus. However, some plausible assumptions about network parameters can be gleaned from a long body of work in the sociology community spanning five decades focusing on the study of individual communities and *ego-centric networks* (in graph theoretic terms, this is the subgraph induced on a node's neighborhood). Thus our assumptions are somewhat "local" in nature. Nevertheless they suffice to permit a rigorous analysis of running time of algorithms that recover global structure.

Our algorithms use random sampling similar to that in property testing and algorithms for dense graphs. We note however that our networks are not necessarily dense graphs, not even in local neighborhoods.

Our algorithms explore a local-global relationship between ego-centric and socio-centric networks that we hope will provide a fruitful framework for future work both in computer science and sociology.

*Princeton University, Computer Science Department and Center for Computational Intractability arora@cs.princeton.edu. Supported by National Science Foundation.

†Princeton University, Computer Science Department and Center for Computational Intractability rongge@cs.princeton.edu.Supported by National Science Foundation.

‡Princeton University, Computer Science Department and Center for Computational Intractability sachdeva@cs.princeton.edu. Supported by National Science Foundation.

§Princeton University, Computer Science Department and Center for Computational Intractability gschoene@cs.princeton.edu. This research was supported by the Simons Foundation Postdoctoral Fellowship.

1 Introduction

Community structure is an important characteristic of social networks and has long been studied in sociology. The classic paper of Luce and Perry in 1949 —which introduced the term "Clique" to graph theory —described a community as subsets of individuals every pair of whom are acquainted. The text of Scott [32] equates communities with objects such as *cliques* or other *dense subgraphs*. Another seminal 1974 paper, Breiger [7] develops a theory of communities in terms of *affiliation networks*, which in graph theoretic terms consist of using a bipartite graph with people on one side and and communities on the other. In sociology today, the answer to many natural and important questions depends on a better understanding of community structure. Can you travel from one node to another random node using only a few "strong ties" [19]? Do networks contain "wide" bridges [10]? How much do communities overlap?

The problem of identifying communities arose independently in other fields such as internet search, study of the web graph, the problem of clustering network nodes (in networks of biological interactions, citations, etc.). In his recent comprehensive survey of algorithmic approaches, Fortunato [31] divides them into two camps based upon whether or not the algorithm assumes —implicitly or explicitly— that communities are disjoint. Assuming disjointness implicitly lead to a view of a community as a *nonexpanding node set*: it contains many edges but has relatively few edges leaving it [1]. This viewpoint suggests many approaches that have been tried: graph partitioning, hierarchical clustering, spectral clustering, simulated annealing, modularity, betweenness, etc. Gibson et al. [16] discovered interesting communities via hubs and authorities; Hopcroft et al. [22] used agglomerative clustering on the the Citeseer database and exhibited interesting communities that persist over time.

However, recently Leskovec et al [26] presented an extensive study of many of the above methods on larger datasets, and question whether they uncover meaningful structure at larger scales. Leskovec et al. often detect a large "core" in the network that is difficult to break into communities. One possible interpretation is that if there are communities in the core, they must overlap.

Thus there is growing interest in finding communities that are allowed to overlap, as they do in most real-life social networks. When communities overlap, each community will not in general be a nonexpanding set. (Consequently, clustering-based approaches may not work.) For instance, imagine that the network contains many communities that are equal-sized cliques with bounded pairwise intersections, and every person belongs to four communities. Then each community/clique will have in general as much as three times as many edges going out of it as are contained in it.

Approaches for finding overlapping communities involve either heuristic clique-finding, or local-search procedures that maintain overlapping clusters and improving them via a series of heuristics. Sometimes a probabilistic generative model is assumed and a max-likelihood fit is attempted via EM and other ideas. (The very recent survey of Xie et al. [37] evaluates dozens of competing heuristics introduced in the last two years alone.) However, Fortunato states at the end of his 100-page survey:

..research on graph clustering has not yet given a satisfactory solution of the problem and leaves us with a number of important open issues. The field has grown in a rather chaotic way, without a precise direction or guidelines...What the field lacks the most is a theoretical framework...everybody has his or her own idea of what a community is.

[1] The Girvan-Newman [17] algorithm does not explicitly define communities as nonexpanding sets. Instead it defines the *betweenness* of a node u as the fraction of nodes v, w whose shortest path passes through u. It iteratively removes nodes of low betweenness to isolate communities.

Modeling communities. Of course, it is entirely possible that Fortunato's questions have no clean answer, or at least one that spans all types of networks of interest. Quite possibly, a community in a network of gene-gene interactions is an inherently different object than one in the graph of facebook friendships. Furthermore, a clear definition of the problem does not in itself guarantee a simple algorithm—e.g., if the definition involves cliques.

A related issue is development of models for community formation/growth whose mathematical analysis yields predictions testable on real-life networks. An inspiration here is the large body of Barabasi-Albert [5] style models which make predictions about degree distributions, graph distance etc. One concrete attempt to model community formation is Lattanzi and Sivakumar's [25] *affiliation networks* model that is inspired by sociology work. In this dynamic model the communities are cliques (or dense subgraphs), and the mode of network growth is adding of either new individuals (i.e., nodes) or new communities (i.e., cliques) to an affiliation network. New individual partially copy the community memberships of existing individuals, and new communities are offshoots (subsets) of existing communities. Additional generative models with community structure appear in [28, 1, 6, 24].

What Fortunato's questions point to, though, is a seeming chicken-and-egg situation. Developing models requires reliable data about community structure in real networks. Conversely, finding reliable data about community structure requires some implicit model, since without such a model the algorithm is consigned to solving worst-case instances of NP-hard problems like Clique, Dense Subgraph, and Small-set Expansion. (This issue clearly does not arise for simpler graph properties like node degrees.)

This paper. We seek a more rigorous approach to the problem of finding communities, where "rigorous" means that we clearly state the following: (a) the object sought by our algorithm (b) the assumptions about the underlying network (c) the (worst-case) running time. We try to break out of the chicken-and-egg situation as follows. Instead of proposing a generative model *per se*, we list fairly minimal assumptions about the network that are based on theoretical and empirical work in sociology. Moreover, these assumptions are "local" in nature and they largely depend upon objects well-studied in sociology, namely ego-centric networks and individual communities. We think these assumptions will be satisfied by many plausible generative models (including Latanzi-Sivakumar, according to our simulations [30]). Thus it is interesting that these suffice to recover the communities.

Since our formalization of (a) and (b) draws on sociology we expect our approach to apply more to, say, the Facebook graph than biological networks. Furthermore, since our algorithms involve random sampling in node neighborhoods, they may mesh well with a dominant approach for network study in sociology, namely, *ego-centric analysis* [34]. An *ego-centric* network [32] consists of a person (the ego) and his ties (called "alters"). Ego-centric networks and their structure have been extensively studied in sociology (often via questionnaires and field-study) [34] as a way to gain insight into the entire network [33, 8, 9, 13]. They give a view of how people develop and manage their social network resources [27, 33]. Data about such networks is easy to collect, even in the field, far from computers [21]. The are even included on the biyearly General Social Survey, a central resource in sociology .

Sociological Foundations for our Assumptions. Sociologists have observed that ego-centric networks can be clustered into a few communities, from which they infer that individuals participate in only a small number of communities [35, 20, 8, 9]. Furthermore, they have observed that a large portion of a person's ties fit into communities [35, 20]. The celebrated theory of Feld [12]

gives a theoretical understanding for this fact based upon "foci." He defines a *focus* as "a social, psychological, legal, or physical entity around which joint activities are organized (e.g., workplaces, voluntary organizations, hangouts, families, etc.)" and his theory says that they are responsible for creating many ties in the network.

Mathematically, one could say that each individual participates in up to d communities, and these communities explain γ fraction of his/her ties. This information already may greatly help the algorithm, whose running time may depend upon d and γ.

What is a community? As mentioned, in sociology communities are thought of as either cliques or as dense subgraphs which are "relatively tightly connected" together compared with the rest of the network (see chapter 6 of [32] for an introduction and survey of how to sociology models communities). Sometimes variants are considered, e.g., Alba [2] considers cliques of the tth power of the graph (i.e., edges correspond to having distance $\leq t$ in the original graph). Jackson's text [23] allows communities to be dense graphs and describes various models for how edges are generated within a community: e.g., $G(n,p)$ or the *expected degree model*.

Furthermore, not all dense graphs would pass muster as communities [32, 15]. For example, a union of two disjoint cliques of size t is a fairly dense graph of size $2t$ but the latter is not community. We will assume that a "community" is a dense subgraph with edges inside it generated according to the expected degree model. (While this assumption simplifies the exposition, in Section 4.1 we will observe that this assumption can be relaxed to deal with other families of dense graphs.) However, we leave it as future work to extend our notions to more hierarchical notions of communities such as in [36].

Another principle often used in sociology is *maximality*: we should not be able to add nodes to the community and get the same structure [15]. Otherwise these nodes should be considered part of the community. (This was the basis for the maximal clique problem introduced in Luce and Perry's 1949 paper; see also the text of Scott[32].) Thus nodes within the community are (in some way) better attached to the community than nodes outside of the community.

1.1 Formal Assumptions and Statement of Results

Our assumptions are grounded in the above observations. The network is a graph of size n. Each edge (u,v) has a probability $e_{u,v}$ with which it is picked. This can even be 1, so we allow adversarial edges. Each community C is an *arbitrary* subset of nodes (unknown to the algorithm), and each node is in at most d communities, so that the communities are allowed to overlap. We think of d as constant or small (though several of our algorithms run in polynomial time even for $d = 2^{\sqrt{\log n}}$). Any edge (u,v) where u and v share a community C is a *community edge*. The remaining edges we call *ambient edges*.

Assumption 1) *Community edges are chosen according to the expected degree model.*

Each node u in C has an *affinity* p_u that lies in $[0,1]$. (Node u has a different affinity for each community it belongs to.) Two nodes u, v in C are connected by an edge with probability $p_u p_v$ (this is called the *expected degree model* in the standard text Jackson [23]). Notice that the model is sufficiently flexible to include other well-studied cases: the subcase $p_u = 1$ for all $u \in C$ corresponds to "C is a clique," and the subcase $p_u = \sqrt{\alpha}$ for all $u \in C$ corresponds to "C is a dense subgraph generated according to the random graph model $G(k,\alpha)$." We will usually assume p_u is lowerbounded by some constant, so that the community is always a fairly dense graph. However in Section 5 we show conditions under which our algorithms can handle even sparser communities.

For nodes u, v that belong to more than one community, the probability that they are connected is at least the maximum of $p_u p_v$ for all the communities they are in.

While Assumption 1 seems to be tending towards a "generative model," we use this formulation

primarily for ease of presentation. We remark in Section 4.1 that our algorithms work so long as the edges are "well-distributed."

Assumption 2) *Maximality assumption with gap ε (also called "Gap Assumption").*

Nodes outside the community C are less strongly connected to it than community nodes are. For example, if one posits that $p_u = \sqrt{\alpha}$ for $u \in C$ —i.e., each u has edges to about α community nodes—then our assumption would say that each $w \notin C$ has edges to *less than* a $\alpha - \varepsilon$ fraction of nodes in C. This seems reasonable since otherwise one should consider whether w should belong to C as well. Of course, such assumptions about *maximality* are standard, though the gap ε in this context is new. We are able to relax the Gap Assumption able in certain instances (see Section 4.2), though the results are no longer as clean and crisp.

Assumption 3) *Community membership accounts for a significant portion of each node's edges, say a constant fraction $\gamma > 0$.*

Surprisingly, Assumptions 1-3 suffice to let us efficiently recover the communities even though we made *no other assumptions* about the ambient edges, and allow arbitrary affinities.

Informal Theorem 1 (See Theorem 5)*If all node affinities are lowerbounded by $\sqrt{\alpha}$ then the communities can be recovered in $n^{C \log kd}$ time, where $C = C_{\alpha,\gamma,\varepsilon}$ and k is size of the largest community.*

Unfortunately, this running time is only "quasipolynomial" instead of polynomial, since $\log k$ could be as high as $\log n$. We can get polynomial and even near-linear time algorithms for more restricted versions of the problem. The paper contains many such theorems and the following is representative.

Informal Theorem 2 (See Section 2) *For every constants $\alpha, \delta > 0$, if all affinities are $\sqrt{\alpha}$ (note: this includes cliques as a special case), and all communities sizes are within $1/\delta$ factor of each other, then the communities can be recovered in time $O(nkd^{C \log d})$ where $C = C_{\alpha,\delta,\gamma,\varepsilon}$. Moreover, if affinities are only guaranteed to be at least $\sqrt{\alpha}$, the communities can be recovered in time $O(nk^{2\log(10/\varepsilon)/\alpha+1} d^{C \log d})$.*

Our approach makes heavy use of the Gap Assumption and uses random sampling coupled with some exhaustive enumeration of small cliques in the subgraph induced on the sample. Similar ideas are well-known in property testing [18] and the related field of approximation algorithms for NP-hard problems in dense graphs [3, 14]. Note however that our graphs are not dense. If $d = O(1)$ then the induced graph in the neighborhood of each node is dense, though we do allow d to be large as $2^{\sqrt{\log n}}$ in many settings. Even when $d = O(1)$ we do not know how to use, for example, the weak regularity lemma [14] (a standard tool in those other fields) to recover the communities.

The use of local sampling in the neighborhood of a single node gives our algorithms the feel of *ego-centric* analysis in sociology. However, we examine communities and dense subgraphs, algorithm (partially) explores a two-hop neighborhood from a starting node, which is a more generalized ego-centric analysis, and is necessary because no one node has ties to the entire community. We also adapt our ideas to the case where communities are not dense graphs (as is plausible in really large networks). Though our results are preliminary, that algorithm explores a two-hop neighborhood from a starting node.

Paper Organization. Section 2 presents algorithms for the case when all community sizes are within an $O(1)$ factor of each other. These algorithms are quite efficient and are a good introduction to our techniques. Section 3 allows communities to have vastly different sizes, derives the most general result (Theorem 5, our Informal Theorem 1), and then studies how to derive more efficient algorithms for specialized cases or under additional assumptions.

Section 4 shows how in some cases some of the Assumptions 1-3 can be relaxed. Here there are many open problems, which are also discussed in Section 6. Section 5 shows (under some stronger

assumptions) how to handle the case where each community is a sparse random graph.

1.2 Related Work

The above setting seems superficially similar to other planted graph problems that were successfully treated using SVD (singular value decomposition) (see McSherry [28] and others). This similarity is however illusory because, first, the non-community edges in our model are not necessarily randomly distributed, and more seriously, because the SVD techniques are known for finding *vertex partitions* whereas here due to overlap between communities we need to find *edge partitions*.

Eppstein, Löffler, and Strash [11] show how to provably find all maximal cliques in time that is exponential in the "degeneracy" of the network, which is bounded by the maximum degree in the worst case. Our network model, however, allowed graphs with arbitrary degeneracy and also work for concepts of community more general than a clique.

Mishra et al. [29] also study overlapping communities in social networks. They show a simple and elegant algorithm for detecting overlapping communities in a certain parameter space. Their algorithm works best for communities where the overlap is not too large. In our parameter space, to detect a community C with density α and gap ε, they require that some $v \in C$ has fewer than $(\alpha + \varepsilon - 1)|C|$ neighbors outside of C. This is a strong restriction of the amount of overlapping, and is impossible for small ε when α is bounded away from 1.

Related independent work: Balcan et al. [4] have independently studied the problem of infering overlapping communities. They have a very different starting point in terms of an explanatory model of how network ties are formed via a preference/ranking function among the individuals. Surprisingly, they ultimately arrive at very similar set of "minimal" assumptions and algorithms for infering the communities. Perhaps this convergence is some kind of validation of both approaches.

2 When Communities have Similar Sizes

This section will give very efficient algorithms to find all communities in the graph when the following assumption holds:

Assumption: Each community has size between δk and k where $\delta > 0$ is some constant and k is arbitrary but known to the algorithm.

We continue to make the three assumptions made in Section 1.1, and the parameters $n, \gamma, d, \varepsilon$ are as defined there. To emphasize, the communities can be arbitrary sets in the graph so long as the gap assumption is not violated and each node is in at most d communities. Furthermore, the placement of "ambient" (i.e., non-community) edges in the graph can be adversarial as long as it does not violate the assumptions.

The running time is exponential in $1/\delta$, so one would not use these algorithms if communities have radically different sizes; that case is handled in Section 3.

2.1 Warmup: Communities are Cliques

In this section, "community" is understood to be synonymous with "clique" which corresponds to all affinity values $p_u = 1$.

The algorithm focuses on the neighborhood $\Gamma(v)$ of a node v, and takes a random sample of nodes S from it. Then it uses brute force (or any other suitable heuristics) to find cliques of size about $\log d$ in the graph induced on S, and tries to extend them to communities. (To use sociology terms, here *egocentric* or node-based analysis leads to provably correct *socio-centric* or societal

analysis.) The running time is linear in $n \cdot k$, albeit with a big "constant" factor term dependent upon $d, \delta, \varepsilon, \gamma$.

Theorem 1 *Given a graph satisfying the assumptions in this section, the* CLIQUE-COMMUNITY-FIND-ALGORITHM *outputs each community with probability at least* $2/3$ *in time* [2] $O(nk/\delta\gamma)2^{\tilde{O}(\log^2 d)}$.

The intuition behind why sampling works is that for any node v the neighborhood $\Gamma(v)$ has size at most kd/γ as communities have size at most k . Each community C containing v lies within $\Gamma(v)$ and has size at least δk, which is at least $\delta\gamma/d$ fraction of $\Gamma(v)$. Thus a random sample of $\Gamma(v)$ will have many representatives from S. The only subtlety is to watch out for "false positives": sets that are not cliques but may present themselves as such during sampling.

CLIQUE-COMMUNITY-FIND-ALGORITHM

1: Pick $\frac{9n}{\delta k}$ "starting nodes" uniformly at random and do the following:
2: For each starting node v randomly sample $S \subseteq \Gamma(v)$ by including each node with probability $p = \log(12d/\varepsilon\delta\gamma)/\delta\varepsilon k$. Proceed only if the sample has size at most three times the expectation $\frac{deg(v)\log(12d/\varepsilon\delta\gamma)}{\delta\varepsilon k} \leq \frac{d\log(12d/\varepsilon\delta\gamma)}{\gamma\delta\varepsilon}$.
3: **for all** cliques U of size at most $2pk$ in the induced graph $G(S)$ on S **do**
4: Let V' be the set of nodes in $\Gamma(v)$ which are connected to all nodes in U, and let $G(V')$ be the induced subgraph on V'
5: Let U' be the set of nodes in $G(V')$ whose degree in this subgraph is at least $(1-\varepsilon/2)|V'|$. Output U' if it is a clique of size at least δk, and for all $v \notin U'$, v is connected to at most $1-\varepsilon$ fraction of U'.
6: **end for**

PROOF:(Theorem 1) For any community C the probability that a randomly chosen starting node v belongs to it is at least $\delta k/n$. Thus the expected number of times we pick a starting node from C is at least 9 and so by a Markov bound such a node is selected with probability $1/9$. We show that in each such trial the probability that community C is output is at least $7/9$.

So let $v \in C$. Simple calculation based upon Chernoff bounds shows that each of the following sequence of three statements holds with high probability. (i) the subsampling gives a sample S of size at most thrice the expectation. (ii) $1/\varepsilon\log(12d/\varepsilon\gamma\delta) \leq |S \cap C| \leq 2pk$. Note that $S \cap C$ is a clique of $G(S)$. Now consider what happens when the for-loop of step 3 tries $U = S \cap C$. Since every node in C has an edge to every node in U, the set V' will contain C. (iii) $|V'| \leq |C| + \varepsilon|C|/4$. This follows since by the Gap Assumption that each $u \in \Gamma(v) \setminus C$ has edges to at most $(1-\varepsilon)$ fraction of nodes in C, and thus the probability that it has edges to each node in the random subset $S \cap C$ is less than ε. Also, the size of $\Gamma(v)$ is at most kd/γ, so in expectation the number of nodes in $V' \setminus C$ is only $(1-\varepsilon)^{|S\cap C|} \cdot kd/\gamma \leq \varepsilon|C|/12$.

However, if $|V' \setminus C| \leq \varepsilon|C|/4$, then we can identify nodes in C just by their degree in $G(V')$! Each $w \in C$ has degree is at least $|C| \geq (1-\varepsilon/2)|V'|$; whereas each $w \notin C$ has at most $|C|(1-\varepsilon)+|V' \setminus C| \leq (1-3\varepsilon/4)|C| < (1-\varepsilon/2)|V'|$. Thus the algorithm returns exactly C. □

A practical note: In step 3 we enumerate over all cliques of a certain size. With a slight parameter modification we can show it suffices to enumerate over all maximal cliques of this size, for which in practice one may be able to use existing heuristic algorithms and reduce running time.

Namely, in the proof we pick $p = \frac{\log(24d\log(d/\varepsilon\delta\gamma)/\varepsilon\delta\gamma)}{\delta\varepsilon k}$, and still take U to be $S \cap C$. Then Chernoff bound and union bound show that the probability that there is a node in $S \setminus C$ that connects to every node in $S \cap C$ is small. So in this case $S \cap C$ is a maximal clique in $G(S)$.

[2] In this paper $\tilde{O}$ hides polynomial terms of the parameters δ, γ, ε and also β, α if they are relevant.

2.2 When Communities are Dense Subgraphs

In the previous section, we equated "community" with "clique", as has been done in many previous works. This assumes that everybody knows everybody else in a "community"—clearly a strong assumption as networks get larger (or even in smaller networks where data about adjacencies is incomplete).

In this section, we model a community as a dense subgraph $G(k, \alpha)$ which corresponds to all affinity values $p_u = \sqrt{\alpha}$. All sets have the same affinity $\sqrt{\alpha}$; though this will be relaxed later. If nodes u, v are in more than one community, their affinity is still the same and $e_{u,v} = \alpha$.

The description of the algorithm will use the following notion, which every community necessarily satisfies.

Definition 2 *For $\alpha, \varepsilon > 0$ an $(\alpha, \alpha - \varepsilon)$-set is a subset of nodes such that 1) every node in the set has edges to at least α fraction of nodes in the set; and 2) every outside node has edges to at most $\alpha - \varepsilon$ fraction of nodes in the set.*

Theorem 3 *Given a graph satisfying the conditions of this section with $k \gg \log n$, the Dense-Community-Find-Algorithm outputs each community in $\mathcal{C}$ with probability at least $1 - \exp(-\Omega(\alpha^2\varepsilon^2\delta k))$ over the randomness of G and $2/3$ over the randomness of the algorithm in time $O(n \cdot (k/\gamma\alpha\delta) \cdot 2^{\tilde{O}(\log^2 d)})$.*

Proof: We consider the following algorithm:

Dense-Community-Find-Algorithm

1: Randomly choose $100n/\delta k$ nodes as starting nodes, for each starting node v repeat the following

2: Let $G(\Gamma(v))$ be the induced subgraph of v and her neighbors.

3: Subsample this subgraph to expected size by including each node with probability $p = \frac{2\log(30d/\alpha\varepsilon\delta\gamma)}{\alpha^2\delta\varepsilon^2 k}$. Fail this round if this graph has size more than three times the expectation.

4: **for all** sets U of nodes in the subsampled graph of size at most $2pk$ **do**

5: Let V' be the set of nodes in $\Gamma(v)$ which are connected to at least $\alpha - \varepsilon/2$ fraction of all nodes in U, let $G(V')$ be the induced subgraph on V'

6: Let U' be the set of nodes in $G(V')$ such that their degree is at least $(\alpha - \varepsilon/2)|V'|$

7: Let U'' be the set of nodes that has more than $\alpha - \varepsilon/2$ fraction of edges in to U'. Keep U'' if it is a $(\alpha - \varepsilon/8, \alpha - 7\varepsilon/8)$ set

8: **end for**

To analyze the algorithm, we first assume the graph G is "well formed". The graph G is well formed if 1) the number of edges from any node v to any community C is within $1 \pm \varepsilon/8$ of expectation. 2) For any node u, and node $v \in C$ in community C, the expected size of $\Gamma(u) \cap \Gamma(v) \cap C$ is within $1 \pm \varepsilon/8$ of expectation. In particular, we know if $u \in C$, $|\Gamma(u) \cap \Gamma(v) \cap C| \geq (\alpha - \varepsilon/8)|\Gamma(v) \cap C|$; if $u \notin C$, $|\Gamma(u) \cap \Gamma(v) \cap C| \leq (\alpha - 7\varepsilon/8)|\Gamma(v) \cap C|$.

Since all the requirements of well-formedness are ε far from their expectation, by Chernoff bound it is easy to show that the graph G is well-formed with probability at least $1 - \exp(-\Omega(\alpha^2\varepsilon^2\delta k))$.

Conditioned on $v \in C$ being selected, by Chernoff bounds we show the following statements hold with high probability: (i) the subsampling gives a sample of less than 3 times the expectation. (ii) If we choose U to be the intersection of $\Gamma(v) \cap C$ and subsampled nodes, the number of edges from most nodes in $\Gamma(v)$ to U will be close to expectation. (iii) The symmetric difference of V' and $\Gamma(v) \cap C$ is at most $\varepsilon|\Gamma(v) \cap C|/10$, because for any node in $\Gamma(v)$ to be in the symmetric difference its number of edges to U will have to be $\varepsilon/2$ away from expectation. Chernoff bound shows the probability is at most $\alpha\varepsilon\delta\gamma/30d$, and $|\Gamma(v)| \leq kd/\gamma$.

Since the symmetric difference is so small, and G is well formed, there will be a gap in degree for nodes in and outside C. For any $u \in |\Gamma(v) \cap C|$ the number of edges into V' is at least $(\alpha - \varepsilon/8 - \varepsilon/10)|\Gamma(v) \cap C|$; for any $u \notin C$, the number of edges into V' is at most $(\alpha - 7\varepsilon/8 + \varepsilon/10)|\Gamma(v) \cap C|$. Hence setting threshold at $\alpha - \varepsilon/2$ suffices to distinguish the two cases. The set U' is indeed a subset of $\Gamma(v) \cap C$ of size at least $(1 - \varepsilon/10)$ fraction.

Finally, since the graph is well formed, any node $u \in C$ must have at least $\alpha - \varepsilon/8 - \varepsilon/10$ fraction of edges to U', and any node $u \notin C$ must have at most $\alpha - 7\varepsilon/8 + \varepsilon/10$ fraction of edges to U', again a threshold of $\alpha - \varepsilon/2$ is enough to distinguish the two cases and $U'' = C$.

The running time depend on the size of the subsampled nodes, which is of order $O(p \cdot kd/\gamma) = O(\frac{2d\log(30d/\alpha\varepsilon\delta\gamma)}{\alpha^2\delta\varepsilon^2\gamma})$. Thus the running time is $O(n(k/\alpha\gamma\delta) \cdot O(\frac{2d\log(30d/\alpha\varepsilon\delta\gamma)}{\alpha^2\delta\varepsilon^2\gamma})^{2pk}) = O(n \cdot (k/\alpha\gamma\delta) \cdot 2^{\tilde{O}(\log^2 d)})$. □

2.2.1 Allowing Different Affinities

In previous subsection we required edge probabilities $e_{u,v}$ to be exactly α if u, v belong to the same community, and this probability does not rise even when they belong in more than one community. In real life these requirements may be too stringent. Here we define a new model Dense-Similar-Size which relaxes these two requirements. In this model, the Dense-Community-Find-Algorithm may fail, and we give a new algorithm that, unfortunately, is less efficient.

Dense-Similar-Size $(n, k, d, \alpha, \delta, \varepsilon, \gamma)$ **Assumptions.**

Communities satisfy Assumptions 1-3 from Section 1.1 as well as the following:

⋆ Each community $C \in \mathcal{C}$ has size between δk and k and is generated according to Assumption 1 with affinities $p_u \geq \sqrt{\alpha}$.

⋆ If u, v are in more than one community then edge has probability $e_{u,v}$ at least as large as the maximum requirement $(p_u p_v)$ of all communities that they lie in.

Theorem 4 *Given a graph G and a set of communities $\mathcal{C}$ consistent with Dense-Similar-Size Model with parameters $(n, k, d, \alpha, \delta, \varepsilon, \gamma)$ where $d \geq 2$ and $k \gg \log n$, the* Robust-Dense-Community-Find algorithm *below outputs each community in $\mathcal{C}$ with probability at least $1 - \exp(-\Omega(\alpha^2\varepsilon^2\delta k))$ over the randomness of G and probability at least $2/3$ over the randomness of the algorithm in time $O(n \cdot (k/\alpha\delta\gamma)^{2\log(10/\varepsilon)/\alpha+1} 2^{\tilde{O}(\log^2 d)})$.*

Proof: The previous algorithm may fail because in this model $\Gamma(v) \cap C$ is no longer a uniform subset of C and can be biased. Thus for a vertex u the fraction of edges into the set $\Gamma(v) \cap C$ may be quite different from the fraction of edges into C. The idea of the algorithm is that for any community C, there is always a set S such that $\Gamma(S)$ contains a large $(\geq 1 - \varepsilon/10)$ fraction of C. A uniform sample on $\Gamma(S) \cap C$ will be similar enough to a uniform sample on C, and the number of edges into sample will be close to the expectation. This allows us to get a set that is very close to $\Gamma(S) \cap C$ and then extend it similarly as before.

Robust-Dense-Community-Find

1: Let $T = 2\log(10/\varepsilon)/\alpha$.
2: Randomly choose $100n/\delta k$ starting nodes, for each starting node v repeat the following
3: **for all** sets of nodes $S \subseteq \Gamma(v)$ of size T **do**
4: Let $G(\Gamma(S))$ be the induced subgraph of S and their neighbors.

Subsample this subgraph by including each node with probability $p = O(\frac{\log(120Td/\varepsilon\delta\gamma)}{\alpha\delta\varepsilon^2 k})$. Fail this round if this graph has size more than three times the expectation.
for all sets U of nodes in the subsampled graph of size at most $2pk$ **do**
Let V' be the set of nodes in $\Gamma(S)$ which are connected to at least $\alpha - \varepsilon/2$ fraction of all nodes in U, let $G(V')$ be the induced subgraph on V'
Let U' be the set of nodes in $G(V')$ such that their degree is at least $(\alpha - \varepsilon/2)|V'|$
Let U'' be the set of nodes that has more than $\alpha - \varepsilon/2$ fraction of edges in to U'. Keep U'' if it is a $(\alpha - \varepsilon/8, \alpha - 7\varepsilon/8)$ set
end for
end for

We call the graph G *well formed* if the degree of each node and the number of edges from any node to any community is within $1 \pm \varepsilon/8$ multiplicative factor of their expectations, also for any $u, v \in C$ the size of their intersection in C $|\Gamma(u) \cap \Gamma(v) \cap C|$ is within $1 \pm \varepsilon/8$ of the expectation. By concentration bounds and union bound, the probability that G is well formed is at least $1 - \exp(-\Omega(\alpha^2\varepsilon^2\delta k))$. We shall assume G is well formed in the discussions below.

For any community C, when some $v \in C$ is the starting node, let S be a random subset of T nodes in $C \cap \Gamma(v)$. Since the size $|\Gamma(u) \cap \Gamma(v) \cap C|$ is concentrated for any $u, v \in C$ the probability that none of these T nodes are adjacent to u is at most $(1 - \alpha + \varepsilon/8)^T < \varepsilon/10$. Thus the expected size of $\Gamma(S) \cap C$ is at least $(1 - \varepsilon/10)|C|$.

We fix a set S such that $\Gamma(S) \cap C$ contains at least a $1 - \varepsilon/10$ fraction of C, and show that C is found with good probability. With high probability the subsampling step returns a sample of size less than 3 times the expectation. After sampling, fix U to be the intersection of subsampled nodes and the community C. Then this U is a uniform sample of the set $\Gamma(S) \cap C$. For any node $v \in C$, the expected number of edges from v to U is at least $(\alpha - \varepsilon/10 - \varepsilon/8)|U|$; for any node $v \notin C$, the expected number of edges from v to U is at most $(\alpha - 7\varepsilon/8)|U|$. By Chernoff bound these values are $\varepsilon/4$ away from expectation (and thus the node is in the symmetric difference of V' and $C \cap \Gamma(S)$) with probability less than $\varepsilon\gamma\delta/120Td$. The size of $\Gamma(S)$ is at most $2Tkd/\gamma$. With high probability the symmetric difference (of V' and $\Gamma(S) \cap C$)has size smaller than $\varepsilon|\Gamma(S) \cap C|/20$.

Now since V' is really close to $C \cap \Gamma(S)$, it is easy to check that for all vertices $u \in C \cap V'$, the degree in V' is larger than $(\alpha - \varepsilon/2)|V'|$; for all $u \in V' \backslash C$ the degree in V' is smaller. Thus setting a threshold at $\alpha - \varepsilon/2$ suffices to distinguish these two cases, it follows that $U' = V' \cap C$. Now U' is a large subset of C, all vertices $u \in C$ will have more than $(\alpha - \varepsilon/2)|U'|$ edges to C while all vertices $u \not\in C$ have less edges. Setting the threshold at $\alpha - \varepsilon/2$ is again sufficient to distinguish the two cases and $U'' = C$.

Finally, the running time of the algorithm depend on the size of the subsampled nodes, which is at most $2Tkd/\gamma \cdot p = O(\frac{Td\log(120Td/\varepsilon\delta\gamma)}{\alpha\delta\varepsilon^2\gamma})$. Thus the algorithm runs in time $n(kd/\gamma\varepsilon\delta)^{T+1}O(\frac{Td\log(120Td/\varepsilon\delta\gamma)}{\alpha\delta\varepsilon^2\gamma})^{2pk} = O(n \cdot (kd/\gamma\varepsilon\delta)^{2\log(10/\varepsilon)/\alpha+1} \cdot 2^{\tilde{O}(\log^2 d)})$. □

Notice that although the algorithm works for only a fixed value of α, if the communities have different densities we can also apply the algorithm with different α parameters to find all communities.

3 When Communities may have Very Different Sizes

When communities have very different sizes, the parameter δ for our models in Section 2 can be too small and the algorithms are not efficient. In this section we show we can relax the similar size requirement using a quasi-polynomial time algorithm. We can also find cliques of different sizes in

polynomial time with some additional assumptions.

3.1 Quasi-polynomial Time Algorithm for Communities of Different Sizes

When we have quasi-polynomial time, we can find all communities that have at least constant density just using assumptions 1, 2 and 3. We only make sure that the minimum density we want to find is a constant α_{min}, that is, each community satisfies Assumption 1 with smallest $p_u \geq \sqrt{\alpha_{min}}$.

Theorem 5 *Given a graph G satisfying the assumptions above with parameters $(n, k, d, \alpha_{min}, \delta, \varepsilon, \gamma)$, if all communities are $(\alpha_C - \varepsilon/8, \alpha_C - 7\varepsilon/8)$ sets (which happens with high probability when the size of the communities are not too small) the* Any-Size-Dense-Community-Find *algorithm will output all communities in $\mathcal{C}$ in time $n^{\frac{100\log(kd/\gamma)}{\alpha_{min}\varepsilon^2}+3}$.*

Proof: When trying to apply previous ideas to this model, the difficulty is that communities have very different sizes and sampling will not find small communities. To solve the problem we just enumerate over all sets S of size $T = \frac{100\log(kd/\gamma)}{\alpha_{min}\varepsilon^2}$, think of all these points are chosen uniformly at random from a certain community C. This S will serve as the sampled points, and since it is large we can apply union bound to show we will make no error when extending it to a community.

Any-Size-Dense-Community-Find

1: Let $T = \frac{100\log(kd/\gamma)}{\alpha_{min}\varepsilon^2}$
2: **for** $\alpha = 1$ downto α_{min} step $-\varepsilon/4$ **do**
3: **for all** sets of nodes S of size T **do**
4: let U be the set of all nodes that has more than $\alpha - \varepsilon/4$ fraction of edges to S.
5: keep U if it is a $(\alpha, \alpha - \varepsilon/2)$ set.
6: **end for**
7: **end for**

For each community C with density α_C, there must be a value of α in the loop (line 2) where $\alpha_C \geq \alpha + \varepsilon/8$ and $\alpha_C - \varepsilon < \alpha - \varepsilon/2 - \varepsilon/8$ (because the stepsize is $\varepsilon/4$). Assume this is the case, and let S be a uniformly random set of size T in C. For any node v, if $v \in C$, then the expected number of edges to the set S is more than α fraction; if $v \notin C$ the expected number of edges to the set S is less than $\alpha - \varepsilon/2$ fraction. The probability that the number of edges are $\varepsilon/4$ fraction away from expectation is at most $\exp(-(\varepsilon/4)^2 T\alpha_{min}) \leq (kd/\gamma)^{-2}$. We only need to apply union bound on the nodes of C and their neighbors, so the size is much smaller than $(kd/\gamma)^2$. By union bound the probability that the algorithm successfully find C is not 0. Since we are trying all possible sets S the algorithm will always find all the communities. □

Although the algorithm is for dense subgraphs, if run it with $\alpha_{min} = 1$, it will find all clique communities of any size.

3.2 Polynomial Time Algorithm for Cliques of Different Sizes

Now we try to improve the quasipolynomial time in Theorem 5 in the subcase when communities are cliques of different sizes. The idea will be to reduce the amount of sampling and exhaustive enumeration. To prove this works we need to make assumptions beyond $1, 2, 3$. The difficulty is that the solution can be highly nonunique and degenerate if communities are allowed to be too "similar." For example, suppose node w is not in a community C but is contained in other communities with large subsets of C. Should we now consider w to be part of C, since it does have edges to all (or most) of C? Our network model assumes such cases do not arise.

Assumption 4) Communities are fairly distinct. For each node u in community C, at least a constant factor, say β, of C does not lie in any other community containing u. This is in accord with the intuitive view of how communities arise: the interconnection structure provides utility to its members above and beyond what existed before [12].

The next assumption is technical and perhaps was being assumed by the reader all along. Surprisingly, we didn't need it until now.

Assumption 5) Completeness Assumption. Any set that satisfies all the assumptions of a community in the model is a community. (Also called "Duck Assumption": "*If it looks like a duck, quacks like a duck, and walks like a duck, it's a duck.*") This ensures the adversary can't satisfy Assumption 4 by just pretending that a certain set is not a community even though it looks like one.

Finally we want to strengthen Assumption 3 so that smaller communities are distinguishable in principle from the noise introduced by the ambient edges:

Assumption 3') Even the smallest community accounts for a significant portion γ/d of the edges incident to any member node.

Theorem 6 *Given a graph that satisfies all assumptions in this section with parameters* $(n, m, k, d, \beta, \varepsilon, \gamma)$ *where that* $k \geq 3$*, the* Any-Size-Clique-Community-Find-Algorithm *will output all communities in* $\mathcal{C}$ *with probability at least* $1 - n^{-5}$ *in time* $O(n \log n \cdot (kd/\gamma)^{\log(2/\varepsilon)/\beta+1} \cdot 2^{\tilde{O}(\log^2 d)})$.

Proof: The main algorithmic difficulty will be that $\Gamma(v)$ for any node v may contain cliques of many different sizes. A subsample of $\Gamma(v)$ would be likely to hit large cliques quite often, but not the smaller cliques. To solve this problem we try to find large cliques first. After cliques of size greater than k are found, we can henceforth ignore their edges, and proceed to find smaller cliques.

Another problem is that after removing all edges in the large cliques, the remaining neighborhood of v (called $\Gamma^-(v)$ in the algorithm) may not contain all nodes of a remaining clique. To solve the problem the algorithm uses a set S of size T. We should think of S as a random set in the community C, then by Assumption 4 and concentration bounds we know a large fraction of nodes in C are in $\Gamma^-(S)$.

Any-Size-Clique-Community-Find-Algorithm

Let $l = k$.
while $l \geq m$ **do**
  Randomly choose $100n \log n/l$ starting nodes , repeat the following for each starting node v
  Let $G(\Gamma(v))$ be the induced subgraph of v and her neighbors.
  Let $T = \log(2/\varepsilon)/\beta$ and enumerate over all subsets of nodes of size T: $S \subseteq \Gamma(v)$ such that $|S| = T$. (We are hoping that $S \subseteq C$ where C is a community of size between $l/2$ and l).
  for all choices of S **do**
    Let $\Gamma^-(S)$ be the set of nodes that are connected with some node in S using an edge that does not belong to any of the communities already found by the algorithm. Consider $G(\Gamma^-(S))$, the induced graph of the T nodes in S and their "out of community" neighbors. We denote the size of this new induced graph L.
    Subsample this subgraph by including each node with probability $p = \frac{4\log(30Td/\varepsilon\gamma)}{\varepsilon l}$ Fail this round if this graph has size more than three times the expectation.
    for all Cliques U in the subsampled graph of size at most $2pl$ **do**
      Let V' be the set of nodes in $\Gamma(v)$ which are connected to all nodes in U, and let $G(V')$ be the induced subgraph on V'.
      Let U' be the set of nodes in $G(V')$ whose degree in this subgraph is at least $(1-\varepsilon/4)|V'|$. Greedily extend U' to a maximal clique U''. Output U'' if it is a clique and for all $v \notin U''$,

v is connected to at most $1-\varepsilon$ fraction of U''.

12: **end for**

13: **end for**

14: Let $l = l/2$

15: **end while**

We show that if the algorithm correctly finds all cliques of size larger than l, then an iteration of the WHILE loop at line 2 will correctly find all communities with size between $l/2$ and l with probability $1 - n^{-10}$. The theorem then follows from union bound.

Fix a community C of size between $l/2$ and l, and assume a node $v \in C$ has already been chosen at step 3. Let S be a random subset of $\Gamma(v)$ of size T. By Assumption 4 we know even after ignoring all edges of larger size, the number of remaining edges from any $v \in C$ to C is at least a β fraction, thus $|\Gamma^-(v) \cap C| \geq \beta|C|$. A random set of size T intersects any set of size $\beta|C|$ with probability $1 - \varepsilon/2$, therefore in expectation $\Gamma^-(S)$ contains a $1 - \varepsilon/2$ fraction of C. Since we are enumerating over all sets S we can now assume S is such that $\Gamma^-(S)$ contains at least $1 - \varepsilon/2$ fraction of C.

Now similar to Theorem 1 it is easy to check that the following statements hold with high probability (i) the subsampling gives a sample of size at most thrice the expectation. (ii) For any node $u \notin C$ there is a set of size at least $\varepsilon|C|/2$ in $\Gamma^-(S) \cap C$ that is not connected to u. Suppose we take U to be the intersection of community $C \cap \Gamma^-(S)$ and subsampled nodes, then we have (iii) $|V'| \leq |\Gamma^-(S) \cap C| + \varepsilon|C|/15$. This is because the size of $\Gamma^-(S) \cap C$ is bounded by Tld/γ and each of the nodes outside C has only $\exp(-p\varepsilon|C|/2)$ probability of being in V'.

The last event implies V' is really close to $\Gamma^-(S) \cap C$. Now in graph $G(V')$ each $u \in V' \backslash |C|$ has degree $(1 - \varepsilon + \varepsilon/15)|C|$; each $u \in C$ has degree at least $(1 - \varepsilon/2)C$. This gap enables the algorithm to use a threshold of $(1 - \varepsilon/4)|V'|$ to distinguish whether u is in the community C or not. The set U' will be equal to $\Gamma^-(S) \cap C$.

Finally by Gap Assumption we know during the greedy extension of step 11, we can only include nodes in C and in fact will include all nodes in C. Therefore $U'' = C$ and the community C is found with high probability.

The running time of the algorithm is dominated by the round when $l = k$. At that round on the size of the subsampled set is at most $O(p \cdot Tkd/\gamma) = O(\frac{Td \log(30Td/\varepsilon\gamma)}{\varepsilon\gamma})$, we want to find a set of size $2pk$. Thus the running time is
$O(n \log n (kd/\gamma)^{T+1} O(\frac{Td \log(30Td/\varepsilon\gamma)}{\varepsilon\gamma})^{2pk} = O(n \log n (kd/\gamma)^{\log(2/\varepsilon)/\beta+1} 2^{\tilde{O}(\log^2 d)})$.

□

We leave it as an open problem to identify reasonable set of assumptions that allow polynomial time when communities are dense subgraphs. The problem is that the "duck assumption" is not well defined: we know what a clique looks like, but it's hard to tell whether a subgraph looks like a community generated according to Assumption 1 when there are overlapping communities and ambient edges. We could try to make a stronger duck assumption by assuming every large $(\alpha, \alpha-\varepsilon)$- set is a community, and then a similar algorithm will be able to find all dense communities in polynomial time. But this is not as reasonable as our other assumptions: consider two $(\alpha, \alpha-\varepsilon)$-sets C_1 and C_2 of size $2k/3$ and their intersection has size $k/3$, then it's quite likely that their union $C_1 \cup C_2$ is a $(\alpha, \alpha - \varepsilon)$ set but we don't consider this set as a community.

4 Relaxing the Assumptions

4.1 Relaxing Assumption 1

Assumption 1 states that each community's edges are generated according to a expected degree model. In this section we note that the algorithms and proofs of Theorems 4 and Theorem 5 actually apply to a more general setting.

We first note that we can substantially relax the Dense-Similar-Size Model by replacing Assumption 1 with the following two requirements:

Concentration: the number of edges from any node u to any community C is concentrated around the expectation, that is, $\Pr[|\Gamma(u) \cap C| \notin [(1 \pm \varepsilon)\,\mathbb{E}[|\Gamma(u) \cap C|]]] \leq \exp(-\varepsilon^2\,\mathbb{E}[|\Gamma(u) \cap C|])$, and the degree of each node is concentrated similarly.

(α, ε)-Regularity: for all $u, v \in C$, $\Pr[|\Gamma(u) \cap \Gamma(v) \cap C| \leq [(1-\varepsilon)\,\mathbb{E}[\alpha|\Gamma(v) \cap C|]]] \leq \exp(-\varepsilon^2\,\mathbb{E}[\alpha|\Gamma(v) \cap C|])$

These properties do not require full dependence, but only limited independence, which could be satisfied, for example, by the configuration model [23] which generates a multigraph with (nearly) any particular preassigned degree distribution. This definition could also accommodate additional structure that introduces dependencies among the edges as long as there is still sufficient independence to satisfy Concentration and Regularity. Consider, for instance, the disjoint union of two equal-sized cliques with a random bipartite graph of density β between them. This is α-regular for any $\beta > \frac{\alpha}{4-2\alpha}$. Thus communities can be much more clumpy than in the expected degree model.

Remark 1 *Theorem 4 still holds with the same proof after replacing Assumption 1 with Concentration and (α, ε)-regularity.*

Remark 2 *Theorem 5 still holds with the same proof after replacing Assumption 1 with Concentration.*

4.2 Relaxing Assumption 2–the Gap Assumption

Though plausible, the Gap Assumption may not exactly hold in a real-life graph since there will always be nodes that fall in the "gap." Our algorithm needs to still return sensible answers. Now we argue that our algorithms in Theorem 1, and Theorem 3 produce sensible answers even when this happens. We use the Clique-Community-Find-Algorithm (Theorem 1) to illustrate.

Of course in this setting we cannot hope to return the exact communities. Instead the algorithm will return some C' that contains more than a $1-\varepsilon$ fraction of C and has density more than $1-\varepsilon$.

Theorem 7 *If G is a graph that satisfies Assumptions 1 and 3 and, each each community is a clique that has size between δk and k where $\delta > 0$ is some constant and k is arbitrary but known to the algorithm, then the Clique-Community-Find-Algorithm can be adapted so that for any community C, the algorithm finds a set C' such that $|C' \cap C| \geq (1-\varepsilon)|C|$ and for each $v \in C'$, the number of edges to C' is at least $(1-\varepsilon)|C'|$.*

Proof: The idea is to run the Clique-Community-Find-Algorithm as before. Once we get V' (recall that when G satisfies model A this set V' contains C and has only $\frac{\varepsilon}{4}|C|$ nodes outside C), we know with high probability V' consists of 3 parts: the community C itself, some set of nodes that have more than $1-\varepsilon$ edges to C, and some set of nodes that have no more than $1-\varepsilon$ edges to C. In these 3 parts, the first part is what we want, the third part is very small by the proof of Theorem 1, and so only the second part worries us. Among these three parts, the nodes in

C should tend to have the largest degrees, so we will use the degree of these nodes to identify them. For any node v in V', we call $|\Gamma(v) \cap V'|/|V'|$ the *density* of v. The idea will be to run Clique-Community-Find-Algorithm with some parameter ε' that is somewhat smaller than ε to get V'. Then repeatedly remove nodes from V' that have density less than $1 - \varepsilon/2$ until density of all remaining nodes is more than $1 - \varepsilon$. We will show by a simple calculation that not many nodes in C will be removed. The details are as follows:

1. Run Clique-Community-Find-Algorithm with $\varepsilon' = \varepsilon^2/6\log(d/\delta\gamma)$.
2. Once the algorithm has produced the set V', repeatedly remove all nodes of density less than $(1 - \varepsilon/2)$ until for every $v \in V'$, the density is at least $(1 - \varepsilon)$.

Let $V' = C \cup H \cup W$, where C is the community, H are the nodes that are connected to more than $1 - \varepsilon'$ fraction of C and W are the nodes that are connected to at most $1 - \varepsilon'$ fraction of C. By the proof of Theorem 1 $|W| \leq \varepsilon'|C|$, thus it is very small and can be ignored in the computation below.

The size of V' is at most kd/γ, because each node in it is adjacent to the node v in the algorithm. We shall show that (i) each iteration removes at most $\varepsilon/\log(d/\delta\gamma)$ fraction of nodes in C, and (ii) if in any iteration removes less than half of the nodes, each node in the remaining graph will be connected to at least $1-\varepsilon$ fraction of nodes. Claim (ii) implies that we will have at most $\log(d/\delta\gamma)$ iterations and then Claim (i) implies that at most ε fraction of nodes in C are removed.

For (i), notice that as long as less than half of C is removed, the fraction of edges from any node in H to the remaining part of C is at least $1 - 2\varepsilon'$. Thus the average density of nodes in C is at least $1 - 3\varepsilon'$. By Markov's inequality the fraction of nodes that have density less than $1 - \varepsilon/2$ is at most $3\varepsilon'/(\varepsilon/2) = \varepsilon/\log(d/\delta\gamma)$.

For (ii), notice that all remaining nodes had density $1 - \varepsilon/2$, if less than half of the nodes are removed, their density should still be larger than $1 - \varepsilon$.

Notice that since the ε' used in Clique-Community-Find-Algorithm is now $\varepsilon^2/6\log(d/\delta\gamma)$, the running time of the algorithm will be $O(nk/\delta\gamma 2^{\tilde{O}(\log^3 d)})$. □

Similar ideas can be used to relax the gap assumption in Theorem 3. The main difficulty of applying the same argument is that the edges not from community membership can be adversarially chosen. However in real life graphs this is not likely to happen: if two people do not share any community the probability that they know each other should be lower. If for all edges (u, v) we have the probability $e_{u,v} \leq \alpha$, then for any community C the following algorithm can always find some set C' with density at least $\alpha - \varepsilon$ that contains at least a $1 - \varepsilon$ fraction of C:

1. Run Robust-Dense-Community-Find algorithm with $\varepsilon' = \varepsilon^2/10\log(d/\delta\gamma)$.
2. Once the algorithm has produced the set V', repeatedly remove all nodes of density less than $\alpha - \varepsilon/2$ until the density of every node is at least $(\alpha - \varepsilon)$

The proof is very similar to Theorem 7. First we focus only on the expected degree of nodes. Since $e_{u,v} \leq \alpha$ we can normalize these probabilities by multiplying $1/\alpha$. Now $e_{u,v} = 1$ for nodes in clique, $e_{u,v} \leq 1$ for all pairs u, v. Thus same argument as Theorem 7 shows that the algorithm works if we are given the true values of $e_{u,v}$. Then we argue that the algorithm should also work even if we are just given a random graph G, because the algorithm only uses the degree of nodes in various sets and they all concentrate around their expectation. There are some techinalities here when many nodes have expected degree very close to the threshold we are setting, which can be resolved if we choose a random threshold between $\alpha - \varepsilon/2$ and $\alpha - 0.4\varepsilon$.

5 Sparser Communities

In previous sections we have been talking about communities as dense graphs. This is natural when the community considered is small and people inside are closely related, such as people in the same year and department in a university. However this may not be true for larger communities: if we consider all students in a large university, or even all computer scientists, then it is unlikely that every person knows a constant fraction of other people in the community. In this section we show how our ideas can be applied to communities that are not so dense.

Consider a simple model ("Sparse") for a community where the affinities $p_u = \Omega(k^{-1/4})$. That is two people in the same community of size k know each other only with probability $\Omega(1/\sqrt{k})$. It follows We assume the network satisfies Assumption 1 from Section 1.1 as well as the following: **(1.)** Each community has size k, for any two nodes u, v in the same community the probability $e_{u,v} = B/\sqrt{k}$ where B is a constant larger than 10.**(2.)** There are no ambient edges. **(3)** The intersection of any two communities $C \cap C'$ has size at most $k/20d^2$.

Notice that we do not need to require the gap assumption nor the duck assumption here because they are both implied by property 3. and the fact that every edge is in some community.

Instead of giving an algorithm to find communities in Sparse Model, we show that it can be transformed to a graph G' such that G', $\mathcal{C}$ are generated by Dense-Similar-Size Model. Then we can directly apply the algorithm for Dense-Similar-Size Model to find the communities.

Theorem 8 *Let a graph G and a set of communities $\mathcal{C}$ be consistent with the Sparse Model. Construct a graph G' on the same set of nodes, where u, v has an edge in G' if and only if they have at least $B^2/2$ length-2 paths in G. Then the pair $(G', \mathcal{C})$ are consistent with Dense-Similar-Size Model with parameters $(n, k, d, \alpha, \delta, \varepsilon, \gamma) = (n, k, d, 0.9, 1, 0.6, 1/3d)$.*

Proof: We rely on the relaxation of the Dense-Similar-Size model in Section 4.1 using the concentration and (α, ε)-regularity.

For concentration, focus on one community C, notice that once we fix all the edges adjacent to u, the probability that v has more than $B^2/2$ length-2 paths in G are independent for different v's in the same community. This is because the number of length-2 paths is completely determined by the number of edges from v to $\Gamma_G(u)$, and these are disjoint sets for different v's. Moreover, by symmetry the probability only depends on the number of edges adjacent to u in community C. Thus once we fix the degree of u inside C in graph G, all the edges (u, v) where $v \in C$ are independent and they satisfy Chernoff bound. The degree of u itself is also concentrated.

For (α, ε)-regularity, we would like to show that for any $u, v \in C$, the size of their intersection within C $|\Gamma(u) \cap \Gamma(v) \cap C|$ is also concentrated, just consider randomly choosing $\Gamma(u)$ and $\Gamma(v)$, with high probability both their sizes and the size of their intersection are close to the expectation. In this case whether some node w has many length-2 paths to u or v is also independent (because the relevant edges are disjoint for different w). Chernoff bounds implies the concentration.

For the probability of edges α, the expected number of length-2 paths between u and v in the same community C is at least $|C| \cdot (B/\sqrt{k})^2 = B^2$, by Chernoff bound the probability that the number drops down to half of expectation is smaller than 0.1 when $B > 10$.

For Assumption 3 in Section 1.1, for each node v, if it is in $d' \leq d$ communities, by the calculation above the expected number of community edges in G' is at least $kd' \cdot 0.9 \cdot 0.9 > 0.8kd'$ (here the first 0.9 is the probability of an edge within the community, the second 0.9 is because the communities may overlap, however by property 5 the overlap must be small). The expected number of length-2 paths starting from v in G is at most $kd' \cdot (B/\sqrt{k} \times kd \cdot (B/\sqrt{k}) = kd'dB^2$, thus the expected number of edges of v in G' must be smaller than $2/B^2$ fraction, which is $2kd'd$. Since $0.8kd'/2kd'd > 1/3d$ the number of ambient edges is small.

Finally, we would like to show the gap assumption: if $u \notin C$, the expected number of edges from u to C in G' is small . To do that we only need to show the expected number of length-2 paths from u to C in G is small. We divide the length-2 paths from u to C into two cases: those that enters C at the first step and those that enters at the second step. For the first type, the expected size of $\Gamma(u) \cap C$ is only $k/20d^2 \cdot d \cdot B/\sqrt{k} = B\sqrt{k}/20d$, thus the number of length-2 paths from u to C where the first step is inside C is at most $B\sqrt{k}/20d \cdot k \cdot B/\sqrt{k} = B^2k/20d$. For the second type, in the second step each node only has at most $k/10d^2 \cdot d \cdot B/\sqrt{k} = B\sqrt{k}/20d$ expected edges to C, thus the total expected number of length-2 paths is at most $dk \cdot B/\sqrt{k} \cdot B\sqrt{k}/20d = B^2k/20$. Combining the two cases, we know the expected number of length-2 paths from u to C is at most $B^2k/20 + B^2k/20d \leq B^2k/10$, thus the expected number of edges in G' is at most $B^2k/10 \cdot 2/B^2 = 0.2 \leq \alpha - \varepsilon$.

□

6 Conclusion

We introduced a framework for rigorously thinking about community structure that allows (a) overlapping communities (b) includes well-studied notions such as cliques and dense subgraphs as subcases and (c) yet allows efficient algorithms for recovering the communities. Our assumptions lie between worst-case and average-case, are based on a long live of research, and we suspect they hold in many generative models. Our sampling-based techniques infer global structure (socio-centric analysis) from the neighborhood of vertices (ego-centric analysis). This local versus global framework, familiar in computer science, may be useful in other settings in sociology, especially because ego-centric networks are empirically observed to be dense and thus amenable to our techniques. We think our techniques should meld well with existing heuristics and plan to do a performance study on real-world data, and also to test the validity of our assumptions.

Weakening our assumptions is another promising direction, and we made a start in Section 4. The Gap Assumption (Assumption 2) makes intuitive sense but probably cannot be guaranteed for all network nodes (eg, there will be an occasional node that knows more community members than some particular member, yet is not a community member). Our use of the *expected degree model* for the intracommunity edges (Assumption 1) can be weakened somewhat, but it still is a *static* model. Arguably community evolution is a dynamic process that results in a more intricate, and possibly hierarchical structure. Researchers have started considering two-step models. For example the first step could generate an initial graph according to our assumptions and in the second step each node connects to each neighbor of a neighbor with some small probability. Making these models amenable to efficient community-detection is a good open problem.

In the complete version of this paper we also plan to include more general models that allow clumps or simple hierarchies of components, each of which fits the expected degree model [36].

Acknowledgements

We would like to thank Nikhil Srivastava for contributions in the early part of this work that helped this project find its final direction. Thanks to Balcan et al. for giving us a manuscript of their independent work [4]. We also would like to thank Bernie Hogan for useful consultations about the sociology literature.

[1] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008. 3

[2] R. D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3:3–113, 1973. 4

[3] S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of -hard problems. *Journal of Computer and System Sciences*, 58:193–210, 1995. 5

[4] M.-F. Balcan, C. Borgs, M. Braverman, J. Chayes, and S.-H. Teng. I like her more than you: Self-determined communities. Manuscript, Fall 2011. 6, 17

[5] A. Barabasi and R. Albert. Cemergence of scaling in random networks. *Science*, (286):509–512, 1999. 3

[6] C. Borgs, J. Chayes, J. Ding, and B. Lucier. The hitchhiker's guide to affiliation networks: A game-theoretic approach. In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS 2011)*, 2011. 3

[7] R. L. Breiger. The duality of persons and groups. *Social Forces*, 53(2):181–190, 1974. 2

[8] R. S. Burt. *Structural Holes*, volume 137. Harvard University Press, 1992. 3

[9] R. S. Burt. Structural holes and good ideas. *American Journal of Sociology*, 110(2):349–399, 2004. 3

[10] D. Centola and M. Macy. Complex contagions and the weakness of long ties. *American Journal of Sociology*, 113(3):702–734, 2007. 2

[11] D. Eppstein, M. Löffler, and D. Strash. Listing all maximal cliques in sparse graphs in near-optimal time. *Algorithms and Computation*, 6506:403–414, 2010. 6

[12] S. L. . Feld. The focused organization of social ties. *The American Journal of Sociology*, (5):1015–1035, March 1981. 3, 12

[13] C. Fischer. *To Dwell Among Friends*. University of California Press, 1982. 3

[14] A. Frieze and R. Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999. 5

[15] A. Friggeri, G. Chelius, and E. Fleury. Triangles to capture social cohesion. *Social Science*, 2011. 4

[16] D. Gibson, J. M. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamical systems. *VLDB J.*, 8(3-4):222–236, 2000. 2

[17] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002. 2

[18] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998. 5

[19] M. S. Granovetter. The strength of weak ties: A network theory revisited. *Sociological Theory*, 1(1983):201–233, 1983. 2

[20] B. Hogan. Pinwheel layout to highlight community structure. *Visualization Symposium*, March 2010. 3

[21] B. Hogan, J. A. Carrasco, and B. Wellman. Visualizing personal networks: Working with participant-aided sociograms. *Field Methods*, 19(2):116–144, 2007. 3

[22] J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Natural communities in large linked networks. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 541–546, New York, NY, USA, 2003. ACM. 2

[23] M. O. Jackson. *Social and Economic Networks*. Princeton University Press, 2008. 4, 14

[24] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. *Proceedings 41st Annual Symposium on Foundations of Computer Science*, 41:57–65, 2000. 3

[25] S. Lattanzi and D. Sivakumar. Affiliation networks. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC '09, pages 427–434, 2009. 3

[26] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2008. 2

[27] A. Marin and B. Wellman. Social network analysis: An introduction. Book forth-coming, chapter available on-line. 3

[28] F. McSherry. Spectral partitioning of random graphs. In *FOCS*, pages 529–537, 2001. 3, 6

[29] N. Mishra, R. Schreiber, I. Stanton, and R. E. Tarjan. Clustering social networks. *Social Networks*, 4863:56–67, 2007. 6

[30] M. Rabinovich. Undergraduate Independent Work, Fall 2011. 3

[31] Santo and Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174, 2010. 2

[32] J. Scott. *Social Network Analysis: And handbook*. Sage Publications Lt;, 2 edition, 2000. 2, 3, 4

[33] B. Wellman. The community question: The intimate networks of east yorkers. *American Journal of Sociology*, 84(5):1201–1231, 1979. 3

[34] B. Wellman. The network is personal: Introduction to a special issue of social networks. *Social Networks*, 29(3):349 – 356, 2007. Special Section: Personal Networks. 3

[35] B. Wellman, B. Hogan, K. Berg, J. Boase, J.-A. Carrasco, R. Ct, J. Kayahara, T. L. M. Kennedy, and P. Tran. Connected lives: The project 1. *interactions*, pages 1–50, 2005. 3

[36] H. C. White, S. A. Boorman, and R. L. Breiger. Social structure from multiple networks. i. blockmodels of roles and positions. *American Journal of Sociology*, 81(4):730–780, 1976. 4, 17

[37] J. Xie, S. Kelley, and B. K. Szymanski. Overlapping Community Detection in Networks: the State of the Art and Comparative Study. *ArXiv e-prints*, Oct. 2011. 2